

InTouch® HMI Visualization Guide

Invensys Systems, Inc.

Revision A
Last Revision: 7/25/07



Copyright

© 2007 Invensys Systems, Inc. All Rights Reserved.

All rights reserved. No part of this documentation shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Invensys Systems, Inc. No copyright or patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this documentation, the publisher and the author assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein.

The information in this documentation is subject to change without notice and does not represent a commitment on the part of Invensys Systems, Inc. The software described in this documentation is furnished under a license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of these agreements.

Invensys Systems, Inc.
26561 Rancho Parkway South
Lake Forest, CA 92630 U.S.A.
(949) 727-3200

<http://www.wonderware.com>

For comments or suggestions about the product documentation, send an e-mail message to productdocs@wonderware.com.

Trademarks

All terms mentioned in this documentation that are known to be trademarks or service marks have been appropriately capitalized. Invensys Systems, Inc. cannot attest to the accuracy of this information. Use of a term in this documentation should not be regarded as affecting the validity of any trademark or service mark.

Alarm Logger, ActiveFactory, ArcestrA, Avantis, DBDump, DBLoad, DT Analyst, FactoryFocus, FactoryOffice, FactorySuite, FactorySuite A², InBatch, InControl, IndustrialRAD, IndustrialSQL Server, InTouch, MaintenanceSuite, MuniSuite, QI Analyst, SCADAAlarm, SCADASuite, SuiteLink, SuiteVoyager, WindowMaker, WindowViewer, Wonderware, and Wonderware Logger are trademarks of Invensys plc, its subsidiaries and affiliates. All other brands may be trademarks of their respective owners.

Contents

Welcome.....	7
Documentation Conventions.....	7
Technical Support	8
Chapter 1 WindowMaker: Your Development Environment	9
Setting Your WindowMaker Preferences	10
Using the Screen Grid and Ruler	14
Snapping Objects to the Grid.....	14
Using The Ruler	15
Panning and Zooming	15
Using the Thumbnail Window to Pan and Zoom.....	16
Using the Mouse Wheel to Zoom and Pan	17
Pan and Zoom Limitations.....	17
Managing Toolbars.....	18
Using the Application Explorer	19
Navigating in the Application Explorer	20
Adding Applications to the Application Explorer	20
Using Color Palettes.....	21
Opening the Color Palette.....	21
Creating Custom Colors.....	22
Importing and Exporting Custom Colors.....	23
Setting Font Defaults.....	24
Using Full Screen Mode.....	24

Windows Dialog Box Features.....	25
Mouse Short Cuts.....	26
Moving Objects with the Arrow Keys.....	27
Viewing License Information.....	28
Chapter 2 Application Windows	29
Creating Application Windows.....	30
Importing and Exporting Windows.....	32
Setting Windows to Appear at Run Time	32
Modifying Application Windows.....	32
Opening, Saving, and Closing Windows	33
Duplicating Windows	34
Deleting Windows	34
Chapter 3 WindowMaker Objects	35
Simple Objects	36
Creating Lines and Shapes.....	36
Creating Buttons	37
Creating Polylines and Polygons.....	37
Creating Text.....	37
Complex Objects	38
Cells and Symbols	39
Grouping Objects to Cells	40
Grouping Objects to Symbols.....	40
Common Manipulations.....	41
Selecting Objects	41
Moving Objects	42
Aligning Objects	44
Layering Objects.....	45
Controlling Horizontal and Vertical Spacing	45
Flipping Objects and Cells.....	46
Resizing Objects	47
Rotating Objects	47
Changing Text Appearance	48
Changing Lines and Outlines.....	49
Changing Fill.....	50
Deleting Objects	50
Undoing Changes	51

Special Manipulations for All Objects	51
Cutting, Copying, and Pasting Objects	52
Cutting, Copying, and Pasting Object Links	52
Duplicating Objects	53
Special Manipulations for Special Objects	54
Reshaping Polyline and Polygon Objects	54
Working with Bitmap Containers	55
Defining Bitmap Transparency	56
Changing the Radius of a Rounded Rectangle.....	57
Substituting Object Text.....	57
Chapter 4 Animating Objects.....	59
Two Types of Animation Links	60
Data Display Animations.....	60
Creating Value Displays	60
Creating Movement.....	63
Creating Rotation	65
Animating Sizes.....	66
Animating Colors.....	69
Animating Fill Levels.....	73
Making Objects Blink.....	75
Enabling Visibility.....	76
Disabling Objects.....	77
Configuring ToolTips.....	78
Positioning a Touch-Sensitive Window	80
Data Entry Animations.....	83
Enabling Discrete Input.....	84
Enabling Analog Input.....	85
Enabling String Input.....	86
Enabling Sliders	87
Enabling Push Buttons	89
Opening and Closing Windows	91
Configuring On-Screen Keyboards	92
Common Animation Tasks.....	97
Selecting Tags or Attributes	97
Creating Keyboard Shortcuts	102
Changing Tagname References	103
Converting Placeholder Tags.....	104

Chapter 5 Wizards.....	105
Working with Wizards	106
Types of Wizards	106
Adding Wizards to the Toolbar.....	107
Pasting Wizard Instances	107
Configuring Wizards	108
Performing Standard Operations on Wizards	108
Installing and Removing Wizards.....	108
Trend Objects	110
Windows Controls Wizards.....	111
Creating and Configuring Windows Controls.....	112
Creating a Text Box	113
Creating a List Box	114
Creating a Combo Box	115
Creating a Check Box.....	116
Creating a Radio Button Group.....	117
Scripting Windows Controls	119
Getting or Setting the Value of a Control	119
Enabling or Disabling a Control for User Input.....	121
Hiding Windows Controls Dynamically	122
Adding and Deleting Items in Combo Boxes	123
Loading and Saving List Items From or To a File	126
Finding Items In a Combo Box or List.....	129
Working with Item Indexes in a Combo Box or List ..	130
Counting List Box or Combo Box Items.....	133
Getting or Setting the Value of a List Item	134
Getting the Name of a List Item	136
Loading the Contents of a Text Box.....	137
Checking If a Text Box is Read-Only	139
Getting or Setting the Label of a Check Box	140
Understanding Windows Controls Error Messages ...	141
Chapter 6 ActiveX Controls.....	143
Using ActiveX Controls.....	144
Configuring ActiveX Controls.....	146
Naming ActiveX Controls	147
Standard Operations on ActiveX Controls.....	147
Installing and Removing ActiveX Controls.....	148
Index	151

Welcome

This documentation describes how to use InTouch WindowMaker to create HMI application screens, called windows. You can use a variety of graphical objects in the windows, ranging from simple lines to complex symbols. At run time, your application is animated according to links you set up between the graphical objects and your manufacturing data.

You can view this document online or you can print it, in part or whole, by using the print feature in Adobe Acrobat Reader.

This guide assumes you know how to use Microsoft Windows, including navigating menus, moving from application to application, and drawing objects on the screen. If you need help with these tasks, see the Microsoft online help.

Documentation Conventions

This documentation uses the following conventions:

Convention	Used for
Initial Capitals	Paths and file names.
Bold	Menus, commands, dialog box names, and dialog box options.
Monospace	Code samples and display text.

Technical Support

Wonderware Technical Support offers a variety of support options to answer any questions on Wonderware products and their implementation.

Before you contact Technical Support, refer to the relevant section(s) in this documentation for a possible solution to the problem. If you need to contact technical support for help, have the following information ready:

- The type and version of the operating system you are using.
- Details of how to recreate the problem.
- The exact wording of the error messages you saw.
- Any relevant output listing from the Log Viewer or any other diagnostic applications.
- Details of what you did to try to solve the problem(s) and your results.
- If known, the Wonderware Technical Support case number assigned to your problem, if this is an ongoing problem.

Chapter 1

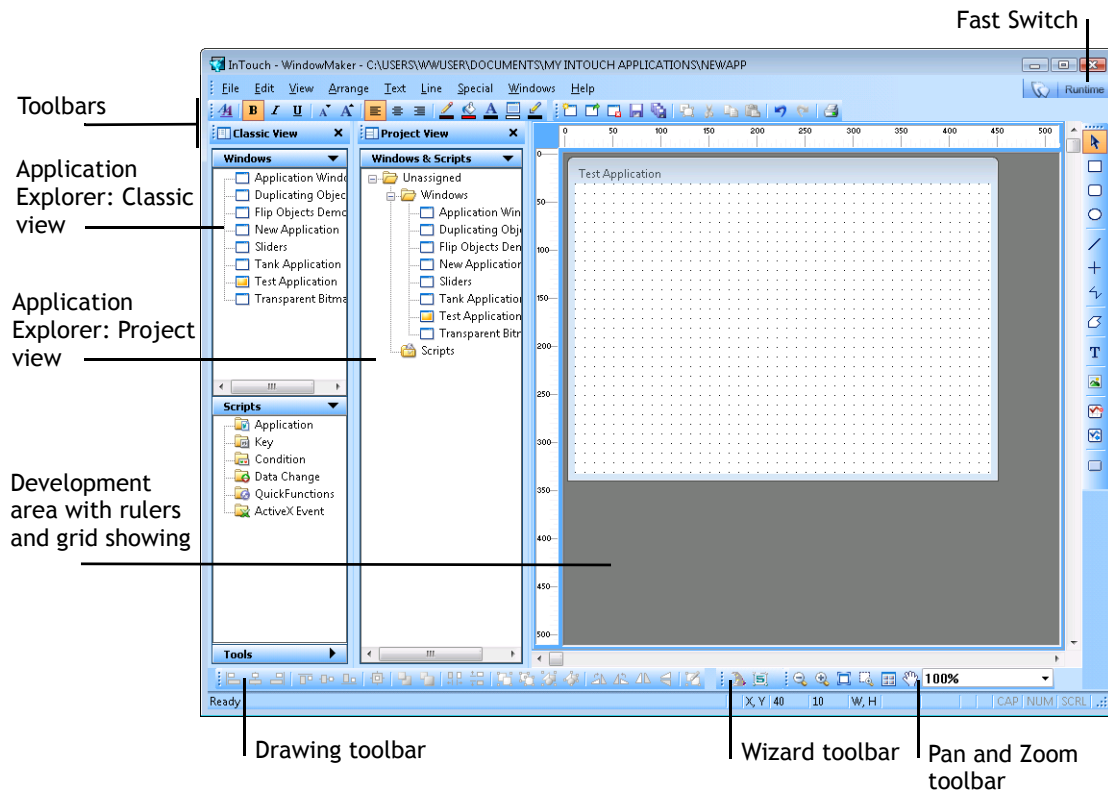
WindowMaker: Your Development Environment

WindowMaker is the development environment you use to create InTouch applications.

The main parts of the development environment are:

- Toolbars and status bar
- Classic view, which shows the windows and scripts.
- Project view, which shows the windows and scripts in a different format.
- Development area
- Fast switch button that opens WindowViewer

The following graphic shows the environment:



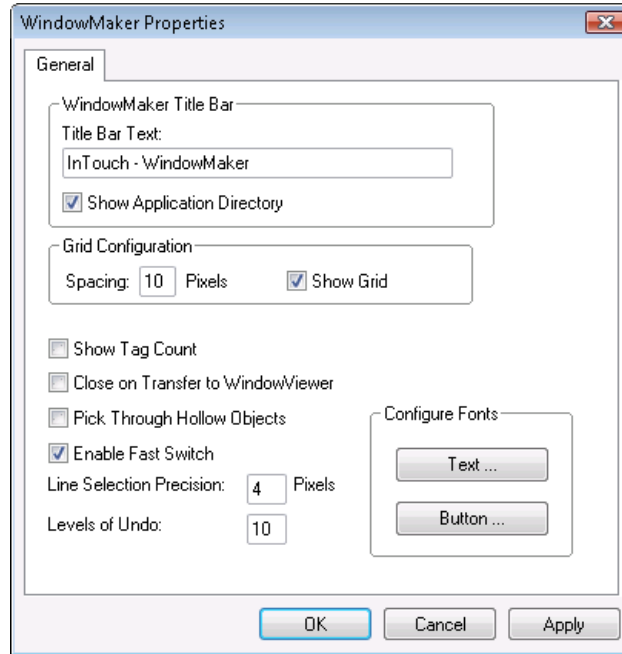
Setting Your WindowMaker Preferences

Using the WindowMaker Properties dialog box, you can configure preferences and options affecting the behavior of WindowMaker. You can:

- Change the title bar text.
- Display the grid or turn off the grid.
- Change the spacing between the pixels on the grid.
- Show the tag count.
- Change the default fonts for text and buttons.
- Set the precision for line selection.
- Set an option to close WindowMaker when switching to WindowViewer.
- Set an option to pick through hollow objects.
- Enable fast switch from WindowMaker to WindowViewer.
- Set the number of undo levels.

To set the properties for WindowMaker

- 1 On the **Special** menu, point to **Configure** and then click **WindowMaker**. The **WindowMaker Properties** dialog box appears.



- 2 In the **WindowMaker Title Bar** area, configure the appearance of the title bar. Do any of the following:
 - In the **Title Bar Text** box, type the text to appear in the title bar during design time.
 - Select the **Show Application Directory** check box to include the path to the application folder in the title bar.
- 3 In the **Grid Configuration** area, configure the background grid. Do any of the following:
 - In the **Spacing** box, type the number of pixels between the grid coordinates.
 - Select the **Show Grid** check box to show the grid.

4 Configure miscellaneous window properties. Do any of the following:

- Select the **Show Tag Count** check box to show the number of tagnames in your Tagname Dictionary in the menu bar. If you have a lot of tags, showing the tag count can impact the Tagname Dictionary performance.

This is useful if you are creating an application with a limited Tagname Dictionary size. The tagname count does not include remote tagname references or system tags. Click **Update Use Counts** on the **Special** menu to find out your remote tagname reference usage.

- Select the **Close on Transfer to WindowViewer** check box to close WindowMaker automatically when you start WindowViewer.

The purpose of this option is to conserve limited memory. If memory is not an issue and you are moving often between WindowViewer and WindowMaker, do not select this option.

When you select **Close on Transfer to WindowViewer**, the reciprocal command, **Close WindowViewer**, on the **General Properties** tab in the **WindowViewer Properties** dialog box is also selected.

- Select the **Pick Through Hollow Objects** check box to select objects that are behind hollow objects.

This allows you to do things like select an object within a frame without having to send the frame to the back.

- Select the **Enable Fast Switch** check box to use the “fast switch” to toggle between WindowMaker and WindowViewer.

The fast switch is the word **Runtime** that appears in the upper right corner of WindowMaker. In WindowViewer, it is the word **Development**.

When you use the fast switch, WindowMaker automatically saves all changes made to all open windows before switching to WindowViewer.

- In the **Line Selection Precision** box, type the number of pixels your cursor can be away from a line and still be able to select it.

In most cases, the default setting of 4 works well.

- In the **Levels of Undo** box, type the number of **Undo** and **Redo** levels to maintain.

You can have up to 25 levels. If you type zero, the undo/redo function is turned off.

One level represents one action. The **Undo** and **Redo** stacks are empty when you create a new window or open an existing window. Both stacks are emptied when you close a window.

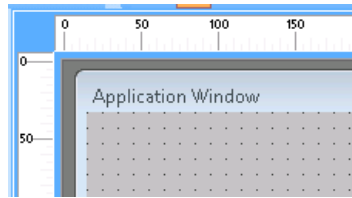
- 5 In the **Configure Fonts** area, click either **Text** or **Button** to set the default text font or button font. Select the font default, and then click **OK**.

You can override these defaults in any window by using the **Font** toolbar.

- 6 Click **OK**.
- 7 Restart WindowMaker to apply any changes you made.

Using the Screen Grid and Ruler

You can show a grid and ruler to help you arrange and align objects.



Snapping Objects to the Grid

You can make your objects snap to predefined grid points by selecting **Snap to Grid**.

By default, the grid is set to 10 pixels and to be visible when you start WindowMaker. You configure the pixel interval for the grid in the **WindowMaker Properties** dialog box.

To see the grid, you must select **Show Grid** on the **WindowMaker Properties** dialog box and select **Snap to Grid** on the **Arrange** menu.

To configure the grid

- 1 On the **Special** menu, point to **Configure** then click **WindowMaker**. The **WindowMaker Properties** dialog box appears.
- 2 In the **Spacing** box, type the number of pixels to space between coordinates.
- 3 Select the **Show Grid** check box if you want to see the grid when **Snap to Grid** is selected.

If you do not select **Show Grid**, no grid is visible in your windows when you select **Snap to Grid**.

Using The Ruler

Use the rulers for precision alignment of the objects in your windows. The rulers appear across the top and along one side of your development environment window.

To show or hide the rulers

- 1 On the **View** menu, click **Ruler**.
- 2 Repeat the step to hide the ruler.

Panning and Zooming

You can zoom in and out get a better look at the elements that you are editing to ensure objects line up exactly or are positioned correctly.

The Pan and Zoom toolbar appears by default at the bottom right of the screen. It can be floated or docked in other locations, like the other toolbars.

You can:

- Zoom in and out from 100% to 500%.
- Zoom to a specific area with the rubber band tool.
- Zoom the window to a specific percentage.
- Click and drag to pan the window.
- Return to the normal default view.

To show or hide the Pan and Zoom toolbar

- ◆ On the **View** menu, click **Pan and Zoom**.

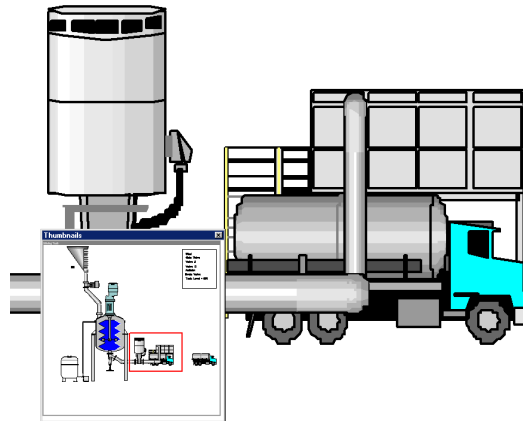
Using the Thumbnail Window to Pan and Zoom

When you zoom in to a detail of your application window, the thumbnails window shows you the relationship of the detail to the whole application.

The **Thumbnails** window gives you both an overview and a detail of the development area.

Show or hide the **Thumbnails** window by clicking the **Thumbnails** button.

A red rectangle is the boundary of the zoomed area within a window.



- Drag the red rectangle to show a different part of the window.
- Click on a different area of the window to move the rectangle to that area.
- Resize the rectangle to change the zoom level of the display area.

The thumbnail view shows a white rectangle for objects that you cannot zoom in on, such as an ActiveX control.

Using the Mouse Wheel to Zoom and Pan

If your mouse has a wheel, you can press the CTRL key and roll the wheel to change the zoom level of your image.

- As you roll the mouse wheel, each click changes the zoom level by 20%.



- You can also place the cursor in the InTouch window and press the mouse wheel to navigate within the window. When you press the mouse wheel, an icon with four arrows appears. Move the mouse to navigate in the window.

Pan and Zoom Limitations

Panning and zooming do not apply to the following controls:

- ActiveX controls
- The Distributed Alarm Object
- The 16 Pen Trend
- SPC Pro objects
- Text boxes
- Check boxes
- List boxes
- Combo boxes
- Radio Group objects

If one of these controls is in the visible area when the view is zoomed more than 100%, a rectangle with the name of the control appears in the area occupied by the control.

Managing Toolbars

You can show or hide any of the toolbars from the **View** menu.

You can move any toolbar from its default docked position to any other location within the development window. Floating toolbars have title bars and they allow you to change their size.

When you start WindowMaker, all toolbars are showing.

You can float or dock a toolbar by dragging it. When you show a docked toolbar that was hidden, it reappears in its last docked location in the window.

To show or hide a toolbar

- ◆ On the **View** menu, click the toolbar name.

To change the size of a floating toolbar

- 1 Move the cursor over any edge of the toolbar. The cursor changes to a double-ended arrow.
- 2 Drag the edge of the toolbar to move and resize the toolbar.

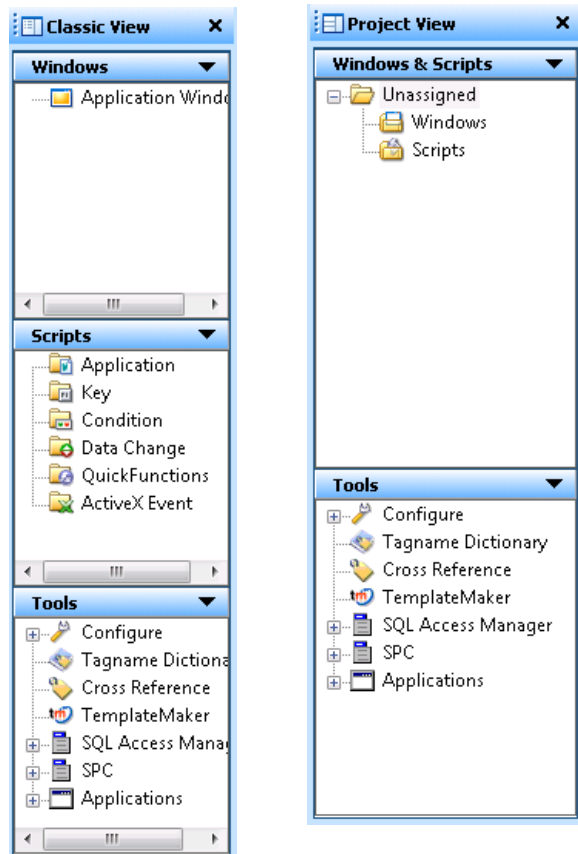
As you move the cursor, a box appears to indicate the size the toolbar when you release the mouse button.

To hide all toolbars at once

- ◆ On the **View** menu, click **Hide All**.

Using the Application Explorer

The Application Explorer has two views: **Classic View** and **Project View**. These views show you both your InTouch application windows and the tools available to you. Like other toolbars, they can be opened or closed, docked or floating.



These views give you access to all application windows, scripts, configuration menus, the Tagname Dictionary, and wizards.

Do not add WindowViewer (view.exe) to the **Application Explorer**. The proper way to start WindowViewer is by clicking **WindowViewer** on the **File** menu, or by clicking the **Runtime** fast switch in the toolbar.

Navigating in the Application Explorer

You can expand or collapse the folders in either of the Application Explorer toolbars.

The **Applications** view shows other installed applications.

To expand or collapse the Application Explorer folders

- 1 Double-click a folder or icon to expand and show the group members.
- 2 Double-click on a member to open that member.

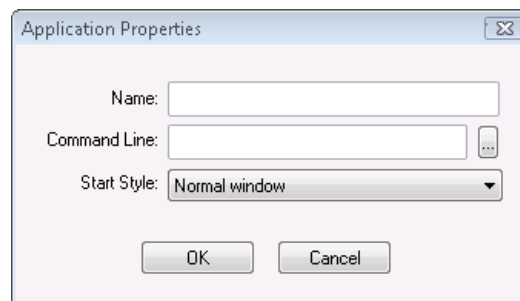
Adding Applications to the Application Explorer

The WindowMaker Application Explorer can start other applications from within WindowMaker. For example, you can run an I/O Server and configure it at the same time that you are developing your application. You can start third-party programs such as Windows Notepad, Wordpad, Microsoft Excel, Microsoft Word, Microsoft Paint and so on.

You can also configure the Application Explorer to open a specific file such as a document or spreadsheet.

To add a new application to Application Explorer

- 1 In the **Tools** pane of the **Classic** view, right-click **Applications**, and then click **New**. The **Application Properties** window appears.



- 2 In the **Name** box, type the name of the application.
- 3 In the **Command Line** box, enter the full path for the application. Click the ellipsis button to browse for the application.

You can add command line parameters for the application in the **Command Line** box.

- 4 In the **Start Style** list, click how you want the application to appear when it starts up.
- 5 Click **OK**. The application is added to the **Application Explorer** under Applications. You can now run the application at any time from WindowMaker.

Using Color Palettes

You can use color palettes to apply color to static and dynamic properties of lines, rectangles, round rectangles, ellipses, polylines, polygons and text. You can select the background color for your windows and the transparent color for bitmaps that allow you to view objects behind bitmaps.

The palette offers you a wide range of color selections, up to 16.7 million colors. The available colors may be limited by your video card capability.

You can also:

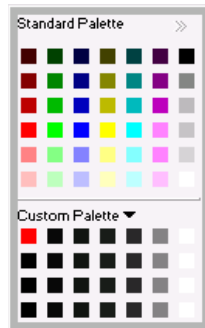
- Define and add custom colors.
- Import custom color palettes created in other Windows applications.
- Export your palettes to other Windows applications.

Opening the Color Palette

The color palette appears whenever you click a colored square in a dialog box or you click one of the color tools to apply line, fill, or text color to a selected object.

To open the color palette

- 1 Click a colored square in a dialog box. The **Standard Palette** appears.



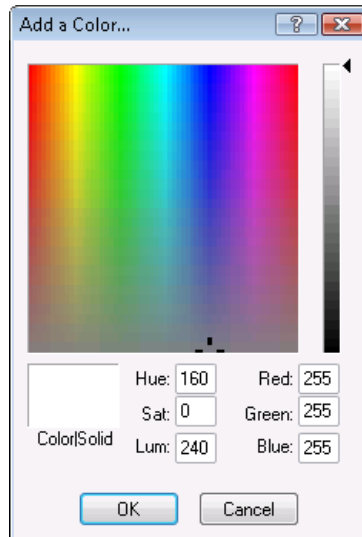
- 2 Click the right arrows to access the classic colors.
- 3 Click the color to use. The color palette closes and the color you selected is applied.

Creating Custom Colors

You can create a palette of custom colors.

To create a custom color

- 1 Open the color palette.
- 2 In the **Custom Palette** area, right-click one of the blank squares and then click **Edit Custom Color**. The **Add a Color** dialog box appears.



- 3 Do any of the following:
 - Click anywhere in the color display and use the slider to adjust the color. To specify that you want 100% of the color with no white or black, press ALT + O.
 - Type values in the **Red**, **Green** and **Blue** boxes to define a color. You can experiment with these values by observing the color matrix. Notice the values for hue, saturation and luminosity also change.
 - Type values in the **Hue**, **Sat**, and **Lum** boxes to define a color. As you change any of these values, the red, green, and blue scales change to match.

Hue is a discrete color value, where 0 is red, 60 is yellow, 120 is green, 180 is cyan, 200 is magenta, and 240 is blue.

Saturation is the amount of color in a specified hue, up to a maximum of 240.

Luminosity is the brightness of a color.

- 4 View the resulting color in the **Color | Solid** box.
If your monitor is set to display 256 colors, the **Color | Solid** box might show two colors. The right side shows how the selected color appears as a solid color. The left side shows the dithered color, or the approximation of the specific color using two of the 256 colors.
- 5 Click **OK**.
You can also create custom colors using the eye dropper tool.

Note Use this feature for creating transparent bitmaps.

To select a custom color with the eye dropper tool

- 1 Open the **Color Palette**.
- 2 Right-click one of the blank squares in the **Custom Palette** section at the bottom of the color palette.
- 3 Click the eye dropper tool and then click the color that you want to add.

Importing and Exporting Custom Colors

If you defined a custom color palette, you can export it from an InTouch application and then import it into another InTouch application.

To import a custom palette

- 1 Open the color palette.
- 2 Click the **Custom Palette** down arrow.
- 3 Click **Load Palette**. The standard Windows **Open** dialog box appears.
- 4 Locate and select the .pal palette file with the color definitions you want.
- 5 Click **Open**. The colors contained in the palette file are loaded into the **Custom Palette**.

To export a custom palette

- 1 Open the color palette.
- 2 Click the **Custom Palette** down arrow.
- 3 Click **Export Palette**. The standard Windows **Save As** dialog box appears.
- 4 Specify the name for the palette file and click **Save**.

Setting Font Defaults

You can set font defaults for text objects and button objects that have text.

Tip You can override these defaults in any window by using the toolbar to customize window or button text.

To set the font defaults

- 1 On the **Special** menu, point to **Configure** and then click **WindowMaker**. The **WindowMaker Properties** dialog box appears.
- 2 In the **Configure Fonts** area, click either **Text** or **Button** to set the default text font or button font. Select the font default, and then click **OK**.

You can override these defaults in any window by using the **Fonts** tool in the **Format** toolbar.

- 3 Click **OK**.

Using Full Screen Mode

Full screen mode hides all program elements except open windows and floating toolbars.

To toggle full screen mode on or off

- ◆ On the **View** toolbar, click the **Full Screen** button to switch from normal to full screen mode.

The **View** toolbar changes to the **Restore** toolbar automatically, and floats on top.

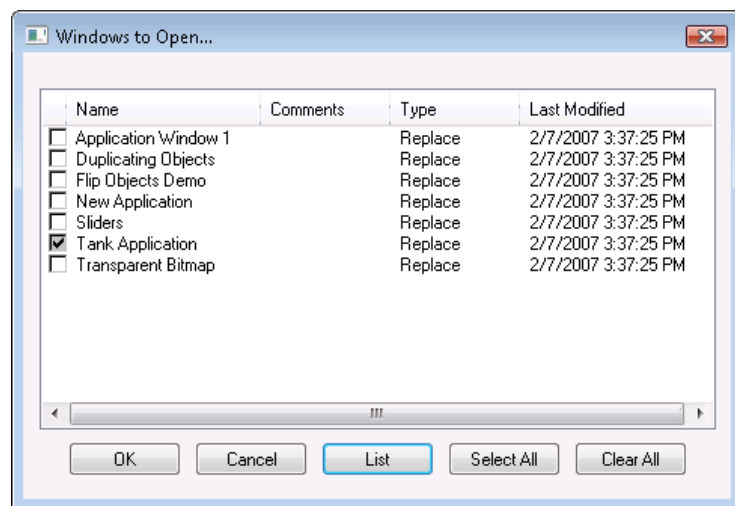
Windows Dialog Box Features

When you are opening, saving, closing, deleting or duplicating windows using the **File** menu, the names of all the windows that are valid candidates for the selected command appear in a list.

Tip Click **Details** to change from the list view to the details view. The details view includes any **Comments** you made for the window in the **Window Properties** dialog box.

To view the windows and details in a multi-column format

- 1 Click the **Details** button. A dialog box appears, showing the list of windows.



- 2 Select a window, click on the window a second time to clear it.
 - To select all windows, click **Select All**.
 - To clear all windows, click **Clear All**.
 - To open selected window(s), click **OK**.

Mouse Short Cuts

Use the following short cuts to open dialog boxes and do other common tasks.

To access menu commands for items in WindowMaker

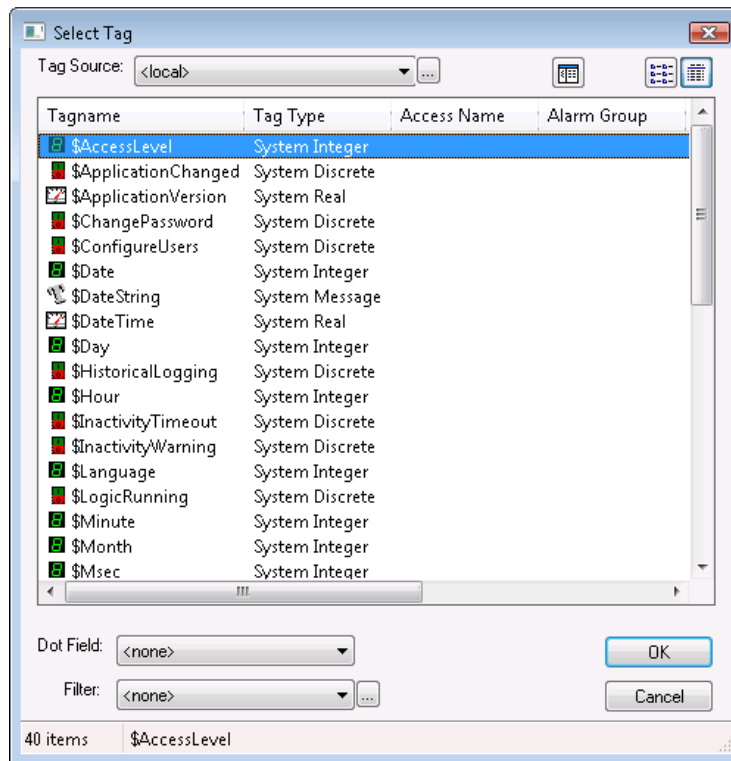
- ◆ Right-click on the item. Items include graphic objects, folder names in the views, and so on.

To open the Animation Links dialog box

- ◆ Double-click an object or symbol.

To open the Select Tag dialog box (Tag Browser)

- ◆ Double-click a blank expression input field within a link definition dialog box. The **Select Tag** dialog box appears.



To access the tag dotfields

- ◆ In any Tagname or Expression input box, type a tagname plus a period, then double-click to the right of the period. You can also type just a period and double-click to the right of it. The **Choose field name** dialog box appears showing all tagname dotfields.

To open a tagname definition in the Tagname Dictionary

- ◆ Double-click the tagname.

Moving Objects with the Arrow Keys

In WindowMaker, you can use the **arrow keys** to move a selected object or group of objects.

When moving objects with the arrow keys, how far an object moves depends upon whether or not the grid is showing.

When the grid is showing, how many pixels an object moves depends upon the grid spacing, which is set on the WindowMaker properties dialog. The default setting is ten pixels between grid points.

When the grid is showing:

- Pressing an arrow key moves the object one grid point.
- Pressing SHIFT + an arrow key moves the object two grid points.
- Pressing CTRL + an arrow key moves the object four grid points.

When the grid is not showing:

- Pressing an arrow key moves the object one pixel.
- Pressing SHIFT + an arrow key moves the object ten pixels.
- Pressing CTRL + an arrow key moves the object 50 pixels.

Viewing License Information

You can see information about your InTouch software such as:

- The version number.
- The serial number.
- If applicable, the license expiration date.

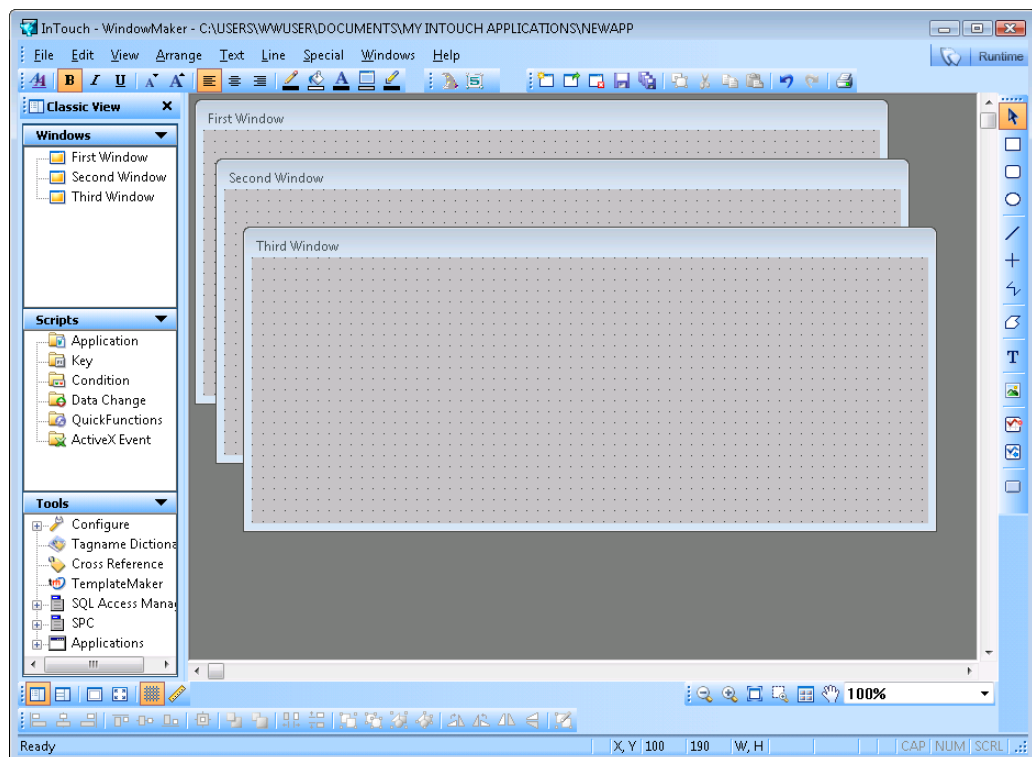
To view license information

- 1 On the **Help** menu, click **About**. The **About InTouch WindowMaker** dialog box appears.
- 2 Click **View License** to open the License Manager to see information about your license or manage the license.
- 3 Click **View License Agreement** to read the license agreement or view patent information. You must have an Internet connection to view the patent information.

Chapter 2

Application Windows

An application window is a container for one or more graphics that model your production processes. For example, you can have a window that shows equipment in a unit. Another window can show a grid of alarm information related to that unit.



You can create any number of windows and you can define window properties such as background color, screen position, window title, and so on.

Creating Application Windows

When you create a new application window, the default settings reflect those of the previously created or of the currently active window.

Window names can have up to 32 characters and can include any character on the keyboard except quotation marks. When you create a window, you are only required to provide a window name. All other items are optional.

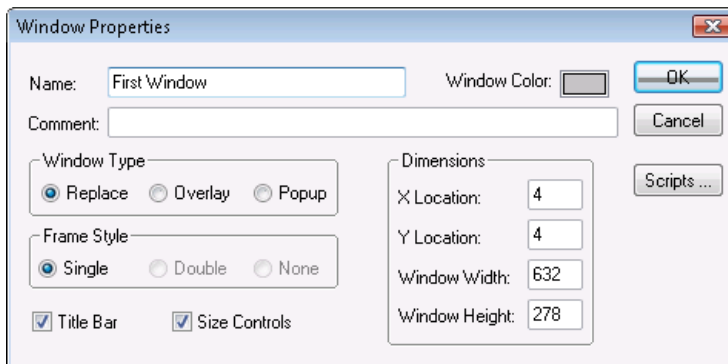
You can include a comment for a window, but it is for information purposes only. The comment appears in file listings but is not used by the application.

By default, the window dimension values are set to the dimensions of the previously created window. These values are also automatically modified if you manually change the window size by dragging the window border.

If you create an application on a system with the **Windows Classic** theme applied and then run the application on a system with the **Windows XP** theme applied, the InTouch windows may appear cut off at the bottom by a few pixels. This is because the Windows **Start** menu bar is taller in the **Windows XP** theme.

To create a new window

- 1 On the **File** menu, click **New Window**. The **Window Properties** dialog box appears.



- 2 Configure the basic window properties. Do the following:
 - In the **Name** box, type a unique name that identifies the window.
 - In the **Comment** box, type any comments you want associated with the window. The comment must be 50 characters or fewer.
 - Click the **Window Color** box to select the background color for the window.

- 3 In the **Window Type** area, configure how the window opens at run time.
 - Click **Replace** for the window to automatically close any windows it intersects with when it appears on the screen.
 - Click **Overlay** for the window to appear on top of currently open windows. It can be larger than the window(s) it is overlaying. When an overlay window closes, any windows behind it reappear. Clicking on any visible portion of a window behind an overlay window brings that window to the foreground as the active window.
 - Click **Popup** for the window to always stay on top of all other windows. Popup windows usually require a response from the user to be removed.
- 4 In the **Frame Style** area, configure the border around the window.
 - Click **Single** for a three-dimensional bordered window which can have a title bar and size controls. Select the **Title Bar** check box to include the title bar.
 - Click **Double** for a three-dimensional bordered window with no title bar and cannot be sized without size controls.
 - Click **None** for a window with no border or title bar and that cannot be sized without size controls.
- 5 Select the **Size Controls** check box to allow users to resize the window at run time.
- 6 In the **Dimensions** area, specify the window location and dimensions. Do the following:
 - In the **X Location** box, type the number of pixels between the left edge of the design area and the left edge of the window being defined.
 - In the **Y Location** box, type the number of pixels between the top edge of the design area and the top edge of the window being defined.
 - In the **Window Width** and **Window Height** boxes, type the window width and height in pixels.
- 7 Click **OK**.

Importing and Exporting Windows

You can import and export windows and their contents from one InTouch application to another. For more information, see Chapter 4, Exporting and Importing Tag Definitions, Windows, and Scripts, in the *InTouch® HMI Application Management and Extension Guide*.

Setting Windows to Appear at Run Time

“Home” windows are windows that appear in WindowViewer when the user starts WindowViewer directly, either from an icon or a menu command.

Home windows do not appear if you use the **Runtime** fast switch to start WindowViewer.

You can show home windows at any time during run time by using the ShowHome() function in a script.

To set home windows

- 1 On the **Special** menu, point to **Configure** and click **WindowViewer**. The **WindowViewer Properties** dialog box appears.
- 2 Click the **Home Windows** tab.
- 3 Select the window or windows to open when WindowViewer starts.
- 4 Click **OK**.

Modifying Application Windows

When developing your application, you can modify the properties of windows at any time.

To modify properties of an application window

- 1 Right click on the window name in the **Classic** or **Project** view, and click **Properties**. The **Window Properties** dialog box appears.
- 2 Make your changes. For more information about the window options, see "Creating Application Windows" on page 30.
- 3 Click **OK**.

Opening, Saving, and Closing Windows

While developing your application, you can open as many windows as your computer memory supports.

Opening a Window

To open a window

- 1 On the **File** menu, click **Open Window**. The **Windows to Open** dialog box appears listing the names of all windows in your application.
- 2 Do either of the following:
 - To open a single window, double-click the window name.
 - To open multiple windows, select the check boxes for the windows to open and then click **OK**.

When you save a window, all graphics, QuickScripts, properties, and so on associated with the window are also saved.

Saving a Window

To save a window

- 1 On the **File** menu, click **Save Window**. The **Windows to Save** dialog box appears, listing the names of all windows.
- 2 Select the windows that need to be saved.
- 3 Click **OK**.

When you close a window that has been modified, you are prompted to save your changes.

Closing a Window

To close a window

- 1 On the **File** menu, click **Close Window**. The **Windows to Close** dialog box appears listing the names of all currently open windows.
- 2 Select the check box next to the window name.
- 3 Click **OK**.

Duplicating Windows

When you have very similar processes to simulate and control, you may want to duplicate a window and then customize it for a secondary process or unit.

You can duplicate windows with all graphics, QuickScripts, properties, and so on, associated with the window.

Before you start, the window that you want to duplicate must be open and saved at least one time. You can only duplicate one window at a time.

To duplicate a window

- 1 On the **File** menu, click **Save Window As**. The **Window to save under new name** dialog box appears, listing the names of all currently open windows.
- 2 Select the check box next to the window name.
- 3 In the **New Name** box, type a name for the new window.
- 4 Click **OK**.

Deleting Windows

To conserve computer storage space, or if the list of windows in the Application Explorer becomes too long to manage, you can delete unused windows.

Caution Make sure you delete the correct window. You cannot restore a deleted window with **Undo**.

To delete a window

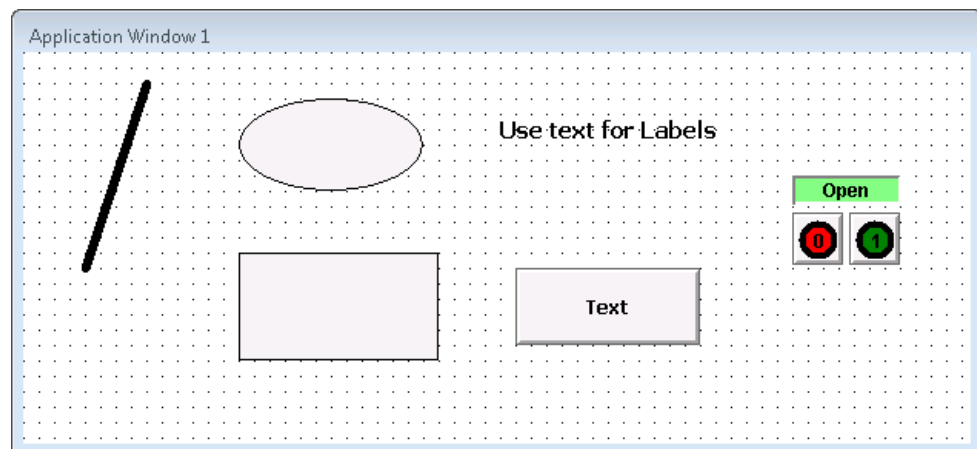
- 1 On the **File** menu, click **Delete Window**. A dialog box appears listing the names of all windows.
- 2 Select the window name you want to delete and click **OK**. When the message appears, click **Yes**.
- 3 Click **OK**.

Chapter 3

WindowMaker Objects

Graphical objects are the key parts of the human machine interface (HMI) applications you build.

As you build your applications, you create simple objects, combine simple objects to make more complex objects, and use some pre-defined complex objects.



If you are using InTouch version 10 or later and are creating new graphics, ArcestrA Symbols may be more suitable for your application. For more information, see Chapter 1, About InTouch and ArcestrA Integration, in the *InTouch® HMI and ArcestrA Integration Guide*.

Simple Objects

You can create the following types of simple objects:






- Lines
- Shapes
- Text
- Buttons

Each simple object has attributes that control how it appears:

- Line color and weight
- Fill color
- Height
- Width
- Orientation

Creating Lines and Shapes

The following table describes how to do basic drawing tasks. The drawing buttons are located on the Drawing toolbar.

To draw this	Click this	Button
Line	Line button	
Horizontal or vertical line	H/V Line button	
Rectangle	Rectangle button	
Rectangle with rounded corners	Rounded Rectangle button	
	Note To adjust the radius of the rounded rectangle corners, see Changing the Radius of a Rounded Rectangle on page 57 .	
Circle or an ellipse	Ellipse button. Press and hold the SHIFT key to draw a circle.	

Creating Buttons

You can use buttons to create points of interaction with your application. The process is very similar to creating simple drawing objects.

For information about creating polygons, see [Creating Polylines and Polygons](#) on page 37.

To create a button



- 1 On the **Drawing** toolbar, click the **Button** button.
- 2 Click and drag to place and size the button.
- 3 Edit the default button text. Do the following:
 - a Right-click the button and click **Substitute Strings**.
 - b In the **New String** box, type the text for the button.
 - c Click **OK**.

Creating Polylines and Polygons

Drawing polylines is slightly different than drawing lines.

To create polyline or polygon



- 1 On the **Drawing** toolbar, click the **Polyline** or **Polygon** button.
- 2 Click on the application window to set the first point.
- 3 Click on the application window again to set more points to define your polyline or polygon.
- 4 Double click the last point.

Creating Text

You can use text to label visual items in your application.

When you create text, the text formatting settings match those set in the **WindowMaker Properties** dialog box. You can change the appearance of selected text. For more information, see [Changing Text Appearance](#) on page 48.

When you type multiple lines of text, they become objects which can be moved and edited independently. You can also combine text objects into a symbol and edit them as a group.

To create text



- 1 On the **Drawing** toolbar, click the **Text** button.
- 2 Click the place for the text to start.
- 3 Type your text and press **ENTER**. A new line of text appears.

Complex Objects

Complex objects provide more functionality than simple objects. The following are types of complex objects.

Complex Object	Description
Cell	A group of two or more objects, including symbols or other cells, that are joined together to form a single unit. You can use cells to create virtual devices such as slide controllers. Cells are useful for creating multiple devices to be associated with different tags.
Symbol	A group of simple objects, such as lines, shapes, and text, that joined together and treated as a single object. Any attribute change applied to a symbol affects all the component objects of a symbol. Symbols cannot contain bitmaps, buttons, cells, wizards, or trends.
SmartSymbol	An InTouch cell that has been converted into a reusable graphic template. You can place one or more instances of a SmartSymbol template in your application windows. Any change to a template propagates to the instances. For more information, see Chapter 1, About SmartSymbols, in the <i>InTouch® HMI SmartSymbols Guide</i> .
ArchestrA Symbol	A highly versatile graphic created using the Symbol Editor in the ArchestrA Integrated Development Environment (IDE). For more information, see Chapter 1, About InTouch and ArchestrA Integration, in the <i>InTouch® HMI and ArchestrA Integration Guide</i> .
Bitmap Container	Objects that allow you to import images such as photographs, drawings, and screen shots. You can rotate a bitmap and you can give it a transparent background. For more information, see Working with Bitmap Containers on page 55.

Complex Object	Description
Trend Objects	Charts of real-time or historical data value changes of multiple tags over time. For more information, see Trend Objects on page 110.
Wizard	Pre-built object that you only need to select, place, and configure for your application. For more information, see Wizards on page 105.
ActiveX Control	Software component that runs within your application. WindowMaker supports both Wonderware and third-party ActiveX controls. For more information, see Using ActiveX Controls on page 144.

Cells and Symbols

You can combine multiple objects into two different types of single units: cells and symbols. A cell can contain any object. A symbol can only contain simple objects. Symbols cannot contain cells.

To find out if a specific object is a cell or a symbol, double-click the object.

- A cell opens, either the **Substitute Tagnames** dialog box, or if the cell doesn't contain tagnames, the Substitute Names warning message appears.
- A symbol or simple graphic object opens the **Animation Links Selection** dialog box.

About Cells

Use a cell to combine and maintain a fixed spatial relationship among multiple elements. You can also use a cell to move multiple elements around and align with other graphical elements.

To change the elements of a cell, you must break the cell, change the elements, and then combine the elements into a cell again.

You can animate elements of a cell but you cannot animate a cell. You also cannot resize a cell.

About Symbols

You can animate a symbol and simple objects. You can also use a symbol to animate parts of a complex graphic.

You cannot make a symbol if more than one of the selected objects has links.

If you combine two symbols into a new symbol, the original symbol structure is lost. If you break the new symbol, it is broken into the individual components of each original symbol. The two original symbols are lost.

Grouping Objects to Cells

You can combine symbols, bitmaps, trends, buttons, wizards and other cells into a cell. If you include a symbol in a cell, all animation links associated with that symbol remain intact.

After you use a cell in a SmartSymbol, if you break the SmartSymbol, you cannot resize the cell.

To create a cell

- 1 Select the objects that you want to include.
- 2 On the **Arrange** menu, click **Make Cell**.

To break a cell

- 1 Select the cell.
- 2 On the **Arrange** menu, click **Break Cell**.

Grouping Objects to Symbols

Symbols cannot contain bitmaps, buttons, cells, wizards or trends. If one of the selected objects has animation links attached to it, the links are attached to the new symbol.

To create a symbol

- 1 Select the objects that you want to include.
- 2 On the **Arrange** menu, click **Make Symbol**.

To break a symbol

- 1 Select the symbol.
- 2 On the **Arrange** menu, click **Break Symbol**.

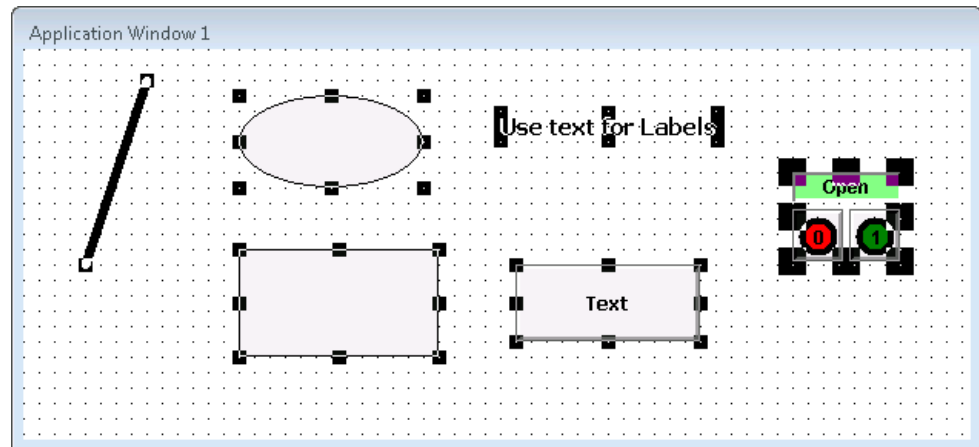
Common Manipulations

Right-click an object to see a menu showing the valid commands or actions you can apply to that object. You can:

- Select objects
- Align objects
- Layer objects
- Flip objects
- Resize objects
- Change font
- Change fill
- Control horizontal and vertical spacing
- Move objects
- Arrange objects
- Undo changes
- Flip symbols
- Rotate objects
- Change line or outline
- Delete objects

Selecting Objects

You must select an object before you can modify it. When you select an object, handles appear on the perimeter of the object. You can use these handles to resize and/or reshape the object.



To select all objects in the active window

- ◆ On the **Edit** menu, click **Select All**, or press **F2**.

To select an object



- ◆ Click the **Select Mode** button and click the object you want to select.

To un-select an object

- ◆ Click a blank area of the window.

To select multiple objects

- ◆ Click the **Select Mode** button, select the first object, then SHIFT+click the other objects.

To select a group of objects

- ◆ Click the **Select Mode** button and drag a box around the objects. All the objects that are completely within the rectangle are selected.

To un-select a specific object or objects from a group of objects

- ◆ SHIFT+click the object.

Moving Objects

You can move objects by:

- Dragging them.
- Using the arrow keys on your keyboard.
- Typing window coordinates in boxes in the status bar.

When you are moving an object, notice how the coordinates in the status bar change.

To move an object by dragging

- ◆ Select the object and drag it.

When moving objects with the arrow keys, how far an object moves depends upon whether or not the grid is showing.

When the grid is showing, how many pixels an object moves depends upon the grid spacing, which is set on the WindowMaker properties dialog. The default setting is ten pixels between grid points.

When the grid is showing,

- Pressing an arrow key moves the object one grid point.
- Pressing SHIFT + an arrow key moves the object two grid points.
- Pressing CTRL + an arrow key moves the object four grid points.

When the grid is not showing,

- Pressing an arrow key moves the object one pixel.
- Pressing SHIFT + an arrow key moves the object ten pixels.
- Pressing CTRL + an arrow key moves the object 50 pixels.

To move an object with the Arrow keys

- ◆ Select an object and
 - press an arrow key.
 - SHIFT + press an arrow key.
 - CTRL + press an arrow key.

To move an object with the status bar








- 1 Select the object.
- 2 Type the X and Y coordinates in the status bar.
- 3 Press Enter.

X, Y	188	121
------	-----	-----

Aligning Objects

You can align objects by their left or right edges, centers, center points, tops, middles, or bottoms.

Using the menu commands or the buttons, you can align in several ways.

Select	Or click	To do this
Align Left		Align left edges of the objects with the left edge of the object farthest to the left.
Align Center		Align objects with the vertical center line of the group.
Align Right		Align right edges of objects with the right edge of the object that is farthest to the right.
Align Tops		Align top edges with the top edge of the highest object.
Align Middle		Align horizontal centers with the middle of the group.
Align Bottom		Align bottom edges with the bottom edge of the lowest object.
Align Center points		Align center points with the center point of the group.

To align objects

- 1 Select multiple objects.
- 2 On the **Arrange** menu, point to **Align** and click the appropriate align command.

Layering Objects

You can position objects in front of or behind others.

To position an object behind another object

- 1 Select the objects(s).
- 2 On the **Arrange** menu, click **Send to Back**.



To position an object in front of another object

- 1 Select the objects(s).
- 2 Do one of the following:
 - On the **Arrange** toolbar, click **Bring to Front**.
 - On the **Arrange** menu, click **Bring to Front**.
 - Press SHIFT+F9.



Controlling Horizontal and Vertical Spacing

You can space objects horizontally between the left most selected object and the right most selected object.

You can also control the vertical spacing between the top most selected object and the bottom most selected object.

To space objects horizontally or vertically

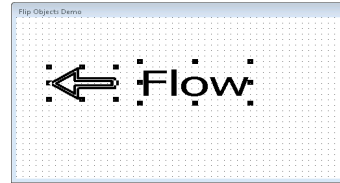
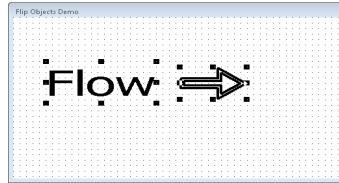
- 1 Select the objects.
- 2 On the **Arrange** Toolbar, click **Space Horizontally** or click **Space Vertically**.



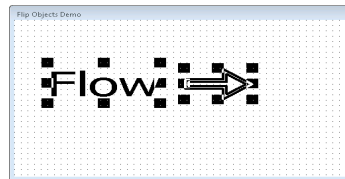
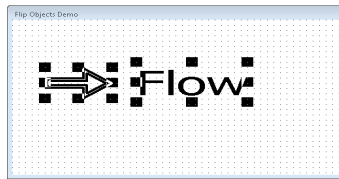
Flipping Objects and Cells

You can flip most objects horizontally or vertically. You can flip objects singly or in groups.

When you flip an object, you transform it to its mirror image. You cannot flip text.



When you flip cells, they are not mirrored. Only the position of the cell in the group of objects is mirrored.



Compare the location of the cells before and after they are flipped. The position is flipped, not the contents.

To flip an object or a cell

- 1 Select the object(s).
- 2 On the **Arrange** toolbar, click **Flip Horizontal** or click **Flip Vertical**.



Resizing Objects

You can resize an object using two methods. You can drag it or specify an exact width and height.

If snap to grid is turned on, the object snaps to the grid during the proportional resizing operation. The result is a slight deviation in the ratio between vertical to horizontal size. To avoid this deviation, turn off the snap to grid.

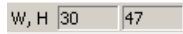
To resize an object by dragging

- 1 Select the object and then position the point of the arrow cursor in the center of a handle.
- 2 Drag the handle to resize the object.

To resize an object by proportional resizing

- ◆ Select the object and SHIFT+drag.

To resize an object by dimensions



- 1 Select the object.
- 2 Type the width and height dimensions in the **W, H** boxes on the status bar.

Rotating Objects

You can rotate most objects including symbols, text, and bitmaps. You cannot rotate cells.

You can rotate objects clockwise or counter clockwise 360 degrees in 90 degree increments.

Rotating objects in WindowMaker has nothing to do with dynamically rotating objects at run time or in WindowViewer. You rotate objects in WindowViewer by linking them to an orientation animation.

To rotate an object



- 1 Select the object(s).
- 2 On the **Arrange** toolbar, click **Rotate Clockwise** or click **Rotate Counter Clockwise**.











Changing Text Appearance

You can set the appearance of fonts before you create text by setting defaults in advance or you can change the appearance of fonts after you create them. For information about setting font defaults, see [Setting Font Defaults](#) on page 24.

The text justification attribute settings are important for text objects showing dynamic values. The justification determines how fields of varying length appear at run time.

For example, if you are showing a numeric value at the end of a text string that is centered or is right justified, the entire text string, including the value, is centered or justified each time there is a change in the number of digits shown.

Using the menu commands or the buttons, you can configure text in several ways.

To	Click	Button
Change the font, style, color, or text size	Font	 
Make the text bold	Bold	
Make the text italic	Italic	
Make the text underlined	Underline	
Reduce or enlarge font size	Reduce Font or Enlarge Font	 
Change the justification	Left Justified, Centered, or Right Justified	  

To configure the text appearance


- 1 Select the text object.
- 2 On the **Text** menu, click the appropriate text command.

Changing Lines and Outlines


You can change the color and either the pattern or width of lines or outlines around outlined objects. Outlined objects include filled shapes such as ellipses, rectangles, and polygons, as well as bitmaps and other imported images.

Wider lines take longer to draw in run time. Dashed and dotted lines can only be one pixel wide.

To set the line appearance default settings

- 1 Click a blank area of the window.
- 2 On the **Line** menu, select the line width or pattern.
-  3 On the **Format** toolbar, click the **Line Color** tool.
- 4 Select a color.

To change the color of a line

- 1 Select a line, a group of lines, or an object with an outline.
-  2 On the **Format** toolbar, click the **Line Color** tool.
- 3 Select a color.

To change the style or width of a line or outline

- 1 Select the object.
- 2 On the **Line** menu, click the line style or width.


To remove an outline

- 1 Select the object.
- 2 On the **Line** menu, click **No Line**.


Changing Fill

Filled shapes include shape surrounded by a line. Examples of filled shapes are rectangles, rounded rectangles, circles, ellipses, and polygons.

To change the fill color of an object

- 1 Select the object.
- 2  On the **Format** toolbar, click the **Fill Color** tool.
- 3 Select a color.

To set the default color for filled shapes

- 1 Click a blank space on the window.
- 2  On the **Format** toolbar, click the **Fill Color** tool.
- 3 Select a color.

Deleting Objects

You can delete one or more objects.

To delete an object

- ◆ Do either of the following:
 - Right-click the object and click **Erase**.
 - Select the object and then press Delete.

Undoing Changes

WindowMaker records your editing and formatting changes for each window. By default, WindowMaker supports 10 levels of undo/redo, where one level represents one action. You can set WindowMaker to retain up to 25 levels of actions. You can also turn off undo/redo by setting the undo/redo level to zero.

If you close the window, all recorded actions are cleared.

To undo a command

- ◆ On the **Edit** menu, click **Undo**.

To redo a command

- ◆ On the **Edit** menu, click **Redo**.

To set the number of undo/redo levels

- 1 On the **Special** menu, point to **Configure**, and then click **WindowMaker**. The **WindowMaker Properties** dialog box appears.
- 2 In the **Levels of Undo** box, type the number of levels.

Special Manipulations for All Objects

You can modify and manipulate simple objects. You can:

- Cut, copy, and paste objects.
- Cut, copy, and paste object links.
- Duplicate objects.

Cutting, Copying, and Pasting Objects

Cut, copy, and paste operations in WindowMaker are much the same as in other Windows based applications but there are some significant differences you need to be aware of.

When you cut, copy, or paste an object, attributes and animation links are also cut, copied, or pasted with it.

All pasted objects remain selected after being pasted and you can move them to adjust their location.

To cut an object

- ◆ Right-click the object and then click **Cut**.

To copy an object

- ◆ Right-click the object and then click **Copy**.

To paste an object

- 1 Right-click a blank space in the window and click **Paste**. The cursor changes to a corner symbol.
- 2 Hold down the left mouse button. The cursor changes to a dotted rectangle the size of the copied object.
- 3 Drag the rectangle to locate the object.
- 4 Release the mouse button.

Cutting, Copying, and Pasting Object Links

The Clipboard is a temporary storage area for links you cut or copy.

- The Clipboard only stores the links for your most recent cut or copy action.
- You can paste the links to any object or symbol that supports the links in the Clipboard.
- If a pasted link is not supported by the object, for example, a line color link on a text object, the link is not pasted.
- If you select multiple objects for pasting, the links are pasted to all objects.

To cut, copy, paste and clear links

- ◆ Right-click the object, point to **Links** and click the appropriate command.

Duplicating Objects

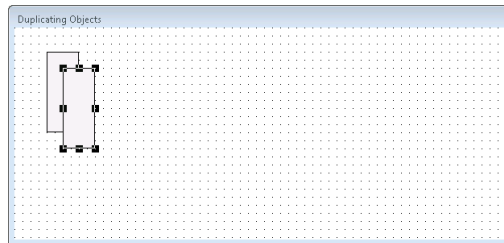
Duplicating objects is similar to copying objects and their animation links, but has the advantage of also duplicating the offset distance and direction when the objects are duplicated more than once.

When you move a duplicated object without un-selecting it, then duplicate it again, the third iteration is offset the same distance and direction between the first two iterations.

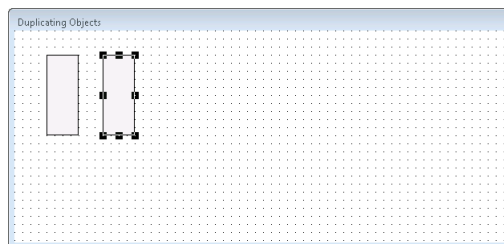
You can repeat this procedure as many times as necessary.

To duplicate an object

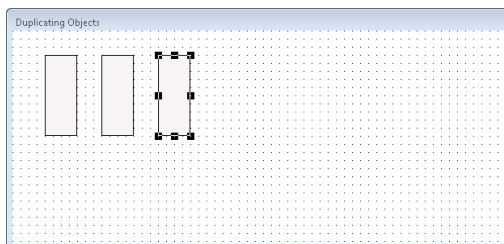
- 1 Right-click the object and click **Duplicate**. The object is copied and pasted to an offset position from the original object.



- 2 Keeping the duplicated object selected, drag it to a different position.



- 3 Again, without un-selecting the duplicated object, right click the object and click **Duplicate** again. The third iteration of the object appears in the same relative position as the first two.



Special Manipulations for Special Objects

The following object types have unique attributes that you can edit:

- Polylines and polygons
- Bitmap containers
- Bitmap transparencies
- Rounded rectangles
- Object text

Reshaping Polyline and Polygon Objects

You can adjust the shapes of polylines and polygons.

To reshape a polyline or polygon

- 1 Select the object.
- 2 Do one of the following.
 - On the **Edit** menu, click **Reshape Object**.
 - Right-click the object and click **Reshape Object**.
 - On the **Arrange** toolbar, click the **Reshape Object** button.



Each shape definition point becomes a handle.

- 3 Drag a handle to reshape.

To add a point to a polygon

- 1 Select the object.
- 2 Do one of the following.
 - On the **Edit** menu, click **Add Point**.
 - Right-click the object and click **Add Point**.
- 3 Click on an edge of the polygon, then drag the point to change the shape of the polygon.

To delete a point from a polygon

- 1 Select the object.
- 2 Do one of the following.
 - On the **Edit** menu, click **Del Point**.
 - Right-click the object and click **Del Point**.
- 3 Click on a point of the polygon, the point is deleted and the shape of the polygon changes.

Working with Bitmap Containers

Bitmap containers are objects that allow you to import graphic objects such as pictures, screen captures, and drawings into your application.

Allowable file types are .bmp, .jpeg, .jpg, .pcx and .tga.

When you import a bitmap, it automatically fills the bitmap container, but you can resize it to its original size and proportions.

You can rotate bitmaps in 90 degree increments.

You can include bitmaps in a cell but not in a symbol.

Using WindowMaker, you are able to place more bitmaps into a window than can load in WindowViewer. If you need to place a large number of bitmaps in a window, be sure to test the window in WindowViewer before releasing the application.

To import a bitmap image



- 1 On the **Drawing** toolbar, click the **Bitmap** tool. The cursor turns into a cross-hair.
- 2 Drag the cursor to draw a bitmap container.
- 3 On the **Edit** menu, click **Import Image**. The **Select Image File** dialog box appears.
- 4 Select the image filename and click **OK**.

To make the bitmap its original size

- 1 Select the image.
- 2 On the **Edit** menu, click **Bitmap - Original Size**.

To paste a bitmap image



- 1 Copy the graphic to the Windows Clipboard.
- 2 Click the **Bitmap** tool and draw a bitmap container in your window.
- 3 Right-click the bitmap container and click **Paste Bitmap**.

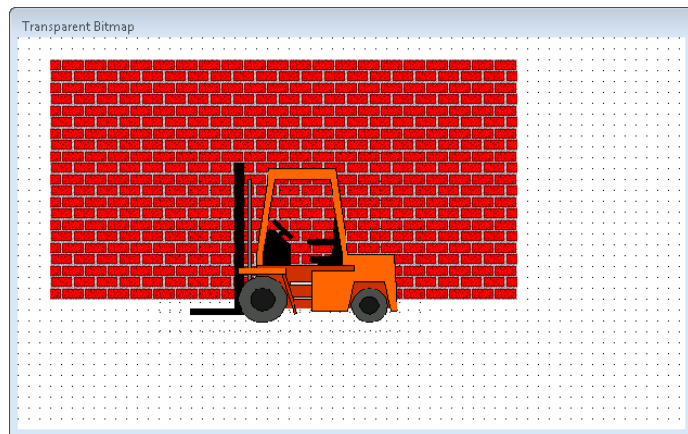
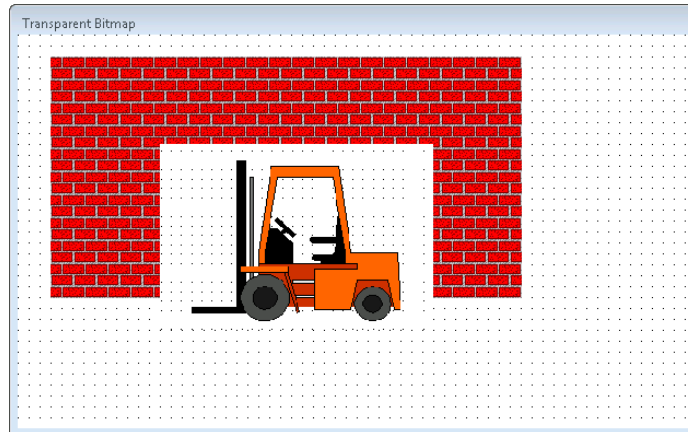
To edit a bitmap

- 1 Select the bitmap.
- 2 On the **Edit** menu click **Edit Bitmap**. Microsoft Paint opens showing the bitmap.
- 3 Edit the bitmap in MS Paint.
- 4 Save and close MS Paint.

Defining Bitmap Transparency

When you define a transparent color in a bitmap, the window background or any objects behind the bitmap shows through it everywhere the transparent color is used.

You can define only one transparent color per image.



To create a transparent bitmap



- 1 With the bitmap selected, click the **Transparent Color** button on the **Format** toolbar to open the transparent color palette.

- 2 Right-click a color square in the **Custom Palette**. The **Edit Custom Color** dialog box appears.



- 3 Click the eye dropper tool.

- 4 Click the color in the bitmap that you want to make transparent. The color is copied to the color square that you selected in the color palette.

- 5 Click the color square to apply the transparent color to the bitmap. All the pixels in the image that are that color become transparent.

Changing the Radius of a Rounded Rectangle

You can increase and/or decrease the corner radius of a Rounded Rectangle.

To increase or decrease a rounded object's radius

- 1 Select the object.
- 2 On the **Edit** menu, click **Enlarge Radius** or **Reduce Radius**.

Substituting Object Text

You can edit the text of objects that have text, such as symbols, cells, or buttons.

When you change a text string, it retains all of its original attributes, font, style, color, and so on. Text formatting also applies to numeric values.

To change the text in an object

- 1 Select the object or button with the text. Do either of the following:
 - On the **Special** menu, click **Substitute Strings**.
 - Right-click the text object, point to **Substitute** and then click **Substitute Strings**.
- 2 In the **New String** box, type the new string and click **OK**.

To change a portion of text in a series of text objects

- 1 Select all the text objects.
- 2 On the **Special** menu, click **Substitute Strings**.
- 3 Click **Replace**. The **Replace Text** box appears.
- 4 In the **Old Text** box, type the portion of text to replace.
- 5 In the **New** box, type the new text string.
- 6 Click **OK**. The new text string replaces the old text string in all the selected objects.

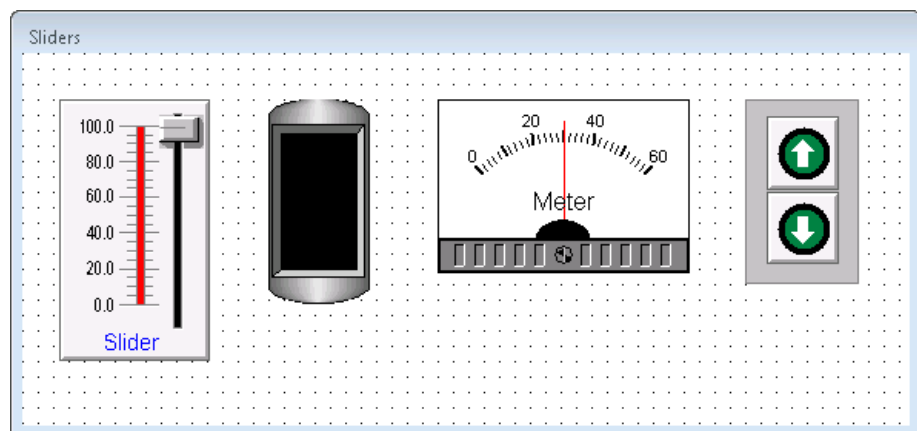
Chapter 4

Animating Objects

You can animate objects or symbols by means of animation links. Animation links connect the values of tags or expressions to your objects or symbols.

For example, you can:

- Create a slider or tank symbol that shows the liquid level in the tank.
- Create a meter that shows a range of values.
- Create touch screen symbols for operator control.



Two Types of Animation Links

There are two basic types of animation links: display links and touch links.

- Display links show information to operators. Examples of display links are changing colors, changing fill levels, horizontal or vertical movements, and blinking objects.
- Touch links allow operator input into the system. Examples of touch links are sliders or push buttons that respond to operator input.

You can define multiple links for objects or symbols. By combining various links, you can create almost any screen animation effect.

Data Display Animations

Data display animations only show information to operators. These animations do not allow for operator input.

Creating Value Displays

Use a value display text object to show the value of a tag. This lets you show things like fill levels, on/off status, or alarm messages.

You can use any one of three types of value display links to show messages at runtime.

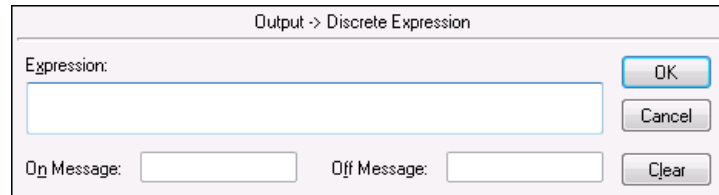
Value Display Type	Shows
Discrete	Discrete values such as on or off
Analog	The value of an analog expression such as fill level or speed.
String	The value of a string expression such as "Fill Level = 100".

You can use up to 1023 characters in an expression. If you need a larger expression, create a QuickFunction and call it in your expression.

Messages appear in the location of the original text object using the font, size, color, alignment, and linked attributes set for that object. The original contents of the field have no effect on the message at runtime.

To create a discrete value display link

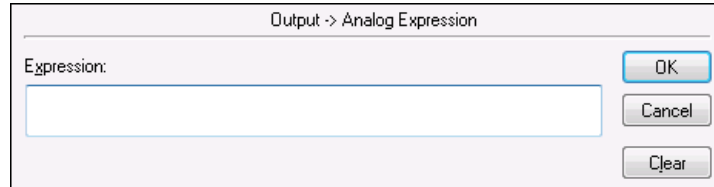
- 1 Right-click the text object and click **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Value Display** area, click **Discrete**. The **Output -> Discrete Expression** dialog box appears.



- 3 In the **Expression** box, type the name of a discrete tag or an expression that equates to a discrete value. For example:
`Cooling_Pump`
- 4 In the **On Message** box, type the message to appear when the value of the expression equals 1, true, on, or yes. For example:
`Pump is ON`
- 5 In the **Off Message** box, type the message to appear when the value of the expression equals 0, false, off, or no. For example:
`Pump is OFF`
- 6 Click **OK**.

To create an analog value display link

- 1 Right-click the text object and click **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Value Display** area, click **Analog**. The **Output -> Analog Expression** dialog box appears.



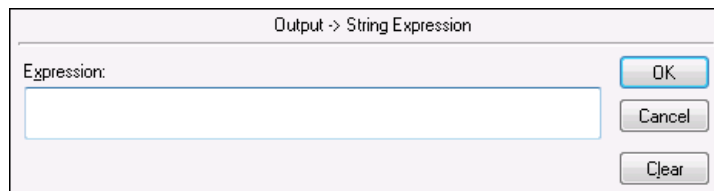
- 3 In the **Expression** box, type an analog (integer or real) tagname or an expression that equates to an analog value. For example:

```
Tank_CV*0.06
```

- 4 Click **OK**.

To create a string value display link

- 1 Right-click the text object and click **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Value Display** area, click **String**. The **Output -> String Expression** dialog box appears.



- 3 In the **Expression** box, type the name of a message tag or an expression that uses a message tag. For example:

```
"The Tank Level is:" + Text(TankLevel, "#")
```

- 4 Click **OK**.

Creating Movement

You make objects move at run time by using location links. You can make an object move horizontally, vertically, or both, as the value of an analog tag or expression changes. For example, as a tank level increases and decreases, an indicator moves up and down.

To create horizontal movement

- 1 Place the object on the screen in the starting location.
- 2 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 3 In the **Location** area, click **Horizontal**. The **Horizontal Location** dialog box appears.

Properties		Horizontal Movement	
	Value		
At Left End:	<input type="text" value="0"/>	To Left:	<input type="text" value="0"/>
At Right End:	<input type="text" value="100"/>	To Right:	<input type="text" value="100"/>

- 4 In the **Expression** box, type an analog tag name or an expression that equates to an analog value.
- 5 In the **Properties** area, configure how far the object moves. Do the following:
 - a In the **At Left End** box, type the value of the analog tag when the object should be at its farthest left position.
 - b In the **At Right End** box, type the value of the analog tag when the object should be at its farthest right position.
 - c In the **To Left** box, type the number of pixels the object should move to the left of its starting position.
 - d In the **To Right** box, type the number of pixels the object should move to the right of its starting position.
- 6 Click **OK**.

To create vertical movement

- 1 Place the object on the screen in the starting location.
- 2 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 3 In the **Location** area, click **Vertical**. The **Vertical Location** dialog box appears.

Value		Vertical Movement	
At Top:	0	Up:	0
At Bottom:	100	Down:	100

- 4 In the **Expression** box, type an analog tag name or an expression that equates to an analog value.
- 5 In the **Properties** area, do the following:
 - a In the **At Top** box, type the value of the analog tag when the object should be at the top position.
 - b In the **At Bottom** box, type the value of the analog tag when the object should be at the bottom position.
 - c In the **Up** box, type the number of pixels the object should move up from the starting position.
 - d In the **Down** box, type the number of pixels the object should move down from the starting position.
- 6 Click **OK**.

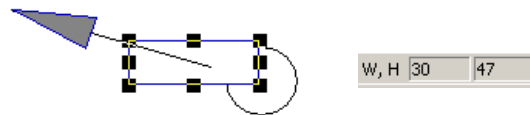
Creating Rotation

You can make an object move around a center point as the value of an analog tag changes by using orientation links. For example, as pressure increases or decreases, a pointer can move around a dial.

The orientation link uses the center of the object or symbol as the default center of rotation. You can offset the center of rotation.

Orientation links are not supported for ArchestrA graphics.

Tip Draw a temporary rectangle from the center of the object to the rotation center point. Now you can read the X and Y offset dimensions in pixels from the W, H boxes in the status bar.



To create an orientation link

- 1 Right-click the object and click **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Miscellaneous** area, click **Orientation**. The **Orientation -> Analog Value** dialog box appears.

The screenshot shows the 'Orientation -> Analog Value' dialog box. It has a title bar with the text 'Orientation -> Analog Value'. Below the title bar is an 'Expression:' field with an empty text box and 'OK', 'Cancel', and 'Clear' buttons. Underneath is a 'Properties' section with four input fields: 'Value at Max CCW:' (0), 'CCW Rotation:' (0), 'Value at Max CW:' (100), and 'CW Rotation:' (360). At the bottom is a 'Center of Rotation Offset from Object Centerpoint' section with 'X Position:' (0) and 'Y Position:' (0) input fields.

- 3 In the **Expression** box, type the name of an analog tag or an expression that equates to an analog value.

- 4 In the **Properties** area, do the following:
 - a In the **Value at Max CCW** box, type the value the expression must be for the object to rotate to the maximum counter-clockwise position.
 - b In the **Value at Max CW** box, type the value the expression must be for the object to rotate to its maximum clockwise position.
 - c In the **CCW Rotation** box, type the degrees the object rotates counter-clockwise when the **Value at Max CCW** is reached.
 - d In the **CW Rotation** box, type the degrees the object rotates clockwise when the **Value at Max CW** is reached.
- 5 In the **Center of Rotation Offset from Object Centerpoint** area, do the following:
 - a In the **X Position** box, type the horizontal offset of the rotation centerpoint. Enter the offset in pixels from the centerpoint of the object.
 - b In the **Y Position** box, type the vertical offset of the rotation centerpoint. Enter the offset in pixels from the centerpoint of the object.
- 6 Click **OK**.

Animating Sizes

You can vary the height and/or width of an object according to the value of an analog tag or expression by using object size links.

For example, a pressure indicator can become larger as pressure increases, or an object on a conveyor can appear to move toward the viewer by becoming larger.

Object size links not only control the size of an object, but the direction in which the object changes size through the use of an anchor for the animation.

To create a object size height link

- 1 Right-click the object and click **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Object Size** area, click **Height**. The **Object Height -> Analog Value** dialog box appears.

The screenshot shows the 'Object Height -> Analog Value' dialog box. It features an 'Expression' text box at the top. Below it is the 'Properties' section, which includes four input fields: 'Value at Max Height' (100), 'Max % Height' (100), 'Value at Min Height' (0), and 'Min % Height' (0). At the bottom is the 'Anchor' section with three radio buttons: 'Top', 'Middle', and 'Bottom', with 'Bottom' selected. On the right side, there are three buttons: 'OK', 'Cancel', and 'Clear'.

- 3 In the **Expression** box, type the name of an analog tag or an expression that equates to an analog value.
- 4 In the **Properties** area, do the following:
 - a In the **Value at Max Height** box, type the value of the tag or expression when the object reaches maximum height.
 - b In the **Value at Min Height** box, type the value of the tag or expression when the object reaches minimum height.
 - c In the **Max % Height** box, type the percentage of its original height that the object will be when the tagname or expression reaches the value set in the **Value at Max Height** box. The percent figures are expressed as a percentage of the drawn size of the object. The drawn size is always the 100% size.
 - d In the **Min % Height** box, type the percentage of its original height that the object will be when the tagname or expression reaches the value set in the **Value at Min Height** box. The percent figures are expressed as a percentage of the drawn size of the object. The drawn size is always the 100% size.
- 5 Select the **Anchor** point from which the object enlarges.
 - Select **Top** for the object to enlarge from its top downward.
 - Select **Middle** for the object to enlarge from its centerpoint outwards in both directions.
 - Select **Bottom** for the object to enlarge from its bottom upwards.
- 6 Click **OK**.

To create a object size width link

- 1 Right-click the object and click **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Object Size** area, click **Width**. The **Object Width -> Analog Value** dialog box appears.

- 3 In the **Expression** box, type the name of an analog tag or an expression that equates to an analog value.
- 4 In the **Properties** area, do the following:
 - a In the **Value at Max Width** box, type the value of the tag or expression when the object reaches maximum width.
 - b In the **Value at Min Width** box, type the value of the tag or expression when the object reaches minimum width.
 - c In the **Max % Width** box, type the percentage of its original width that the object will be when the tagname or expression reaches the value set in the **Value at Max Width** box. The percent figures are expressed as a percentage of the drawn size of the object. The drawn size is always the 100% size.
 - d In the **Min % Width** box, type the percentage of its original width that the object will be when the tagname or expression reaches the value set in the **Value at Min Width** box. The percent figures are expressed as a percentage of the drawn size of the object. The drawn size is always the 100% size.
- 5 Select the **Anchor** point from which the object enlarges in width.
 - Select **Left** for the object to enlarge from its left side.
 - Select **Center** for the object to enlarge from its centerpoint outwards in both directions.
 - Select **Right** for the object to enlarge from its right side.
- 6 Click **OK**.

Animating Colors

You can animate color changes to any object by using color links. Changes can be based on the value of an analog or discrete tag, the value of an analog or discrete expression, or a discrete or analog alarm status.

You can use three kinds of color links to animate objects:

- Line Color
- Fill Color
- Text Color

For each of these three kinds of color links, four types of expressions can control color changes.

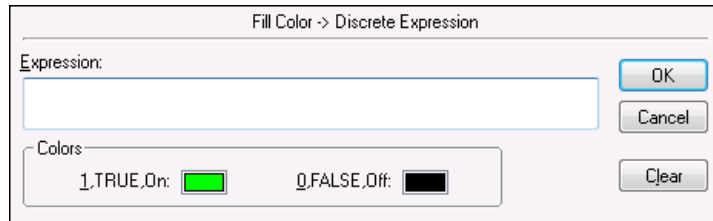
Expression Type	Changes the color based on
Discrete	The value of a discrete tag or expression.
Analog	The value of an analog tag or expression. You can define ten colors to represent differing values.
Discrete Alarm	The alarm state of a tag, Alarm Group, or Group Variable.
Analog Alarm	The alarm state of an analog tag, Alarm Group, or Group Variable. You can define five colors to represent five alarm conditions.

WARNING! Objects do not go into an alarm state when using an analog alarm link if the link is to a remote tag from an unconverted application created before InTouch version 7.11.

All discrete color links are created in the same way. The following procedure describes creating a fill color link.

To create a discrete fill color link

- 1 Right-click the object and click **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Fill Color** area, click **Discrete**. The **Fill Color -> Discrete Expression** dialog box appears.



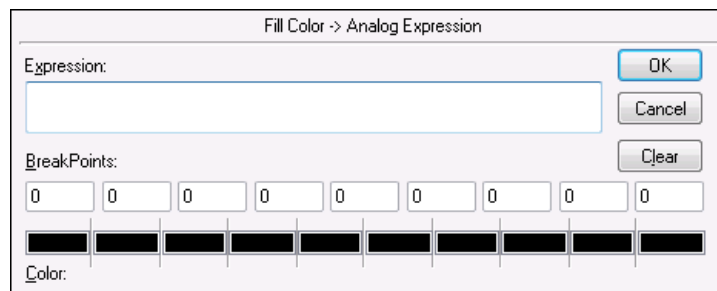
- 3 In the **Expression** box, type the name of a discrete tag or a discrete expression that equates to true or false.

Discrete expressions can contain analog tags. For example, `TankLevel >= 75`. In this example, when the value of the variable "TankLevel" is greater than or equal to 75, the fill color of the object changes.

- 4 In the **Colors** area, click each color box to open the color palette. Select the color to use for each state.
- 5 Click **OK**.

To create an analog expression color link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Fill Color** area, click **Analog**. The **Fill Color -> Analog Expression** dialog box appears.



- 3 In the **Expression** box, type the name of an analog tag or an expression that equates to an analog value.
- 4 In the **Break Points** area, do the following.
 - Specify the breakpoint values where the object changes color.

Tip You do not have to use ten different colors. For example, if you only want the object to change color three times, type three values and use the same color for the remaining values. If you need a more versatile range, review the analog fill capabilities of ArchestrA symbols. For more information see, Chapter 1, About InTouch and ArchestrA Integration, in the *InTouch® HMI and ArchestrA Integration Guide*.

- In the **Colors** area, select a color for each breakpoint.
- 5 Click **OK**.

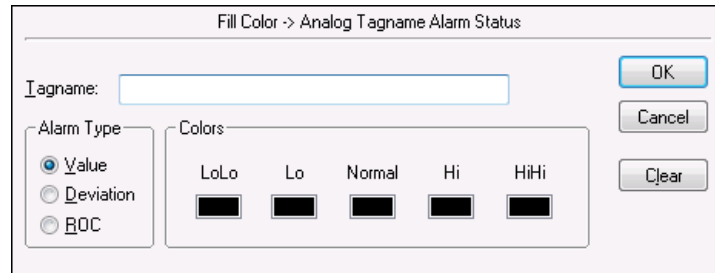
To create a discrete alarm status color link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Fill Color** area, click **Discrete Alarm**. The **Fill Color -> Discrete Tagname Alarm Status** dialog box appears.

- 3 In the **Tagname** box, type the name of the discrete tag to associate with the object.
- 4 In the **Colors** area, select a color for each alarm state.
- 5 Click **OK**.

To create an analog alarm status color link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Fill Color** area, click **Analog Alarm**. The **Fill Color -> Analog Tagname Alarm Status** dialog box appears.



- 3 In the **Tagname** box, type the name of an analog tag to associated with the object.
- 4 In the **Alarm Type** area, select from one of the three types of alarms to associate with the object.

Alarm Type	Use up to
Value	Five colors to show the status of the value alarms.
Deviation	Three colors to show the status of the deviation alarms.
ROC (Rate of Change)	Two colors to show the status of a rate-of change alarm.

- 5 In the **Colors** area, select a color for each alarm state.
- 6 Click **OK**.

Animating Fill Levels

You can vary the fill level of an object using a percent fill link. The percent fill is based on the value of an analog tag or expression. You can create horizontal fills, vertical fills, or both.

For example, you can use a vertical fill link to show the level of liquid in a tank or a horizontal fill link to show the progress of a process.

You create the horizontal and vertical percent fill links the same way.

To create a percent fill link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Percent Fill** area, do one of the following:
 - Click **Vertical**, the **Vertical Fill -> Analog Value** dialog box appears.

- Click **Horizontal**, the **Horizontal Fill -> Analog Value** dialog box appears.

- 3 In the **Expression** box, type the name of an analog tag or an expression that equates to an analog value.

- 4 In the **Properties** area, do the following:
 - a In the **Value at Max Fill** box, type the value of the expression that will result in the object being filled to its maximum level.
 - b In the **Value at Min Fill** box, type the value of the expression that will result in the object being filled to its minimum level.
 - c In the **Max % Fill** box, type the percentage (0-100) that the object will be filled when the expression reaches the level set in the **Value at Max Fill** box.
 - d In the **Min % Fill** box, type the percentage (0-100) that the object will be filled when the expression reaches the level set in the **Value at Min Fill** box.
- 5 In the **Direction** area, click the direction to fill from.
- 6 In the **Background Color** box, select the color of the unfilled portion of the object.
 - The actual fill color is the color that you select for the object when you draw it.
 - If you link both vertical percent fill and horizontal percent fill links to the same object, the last color you select is the background color.
- 7 Click **OK**.

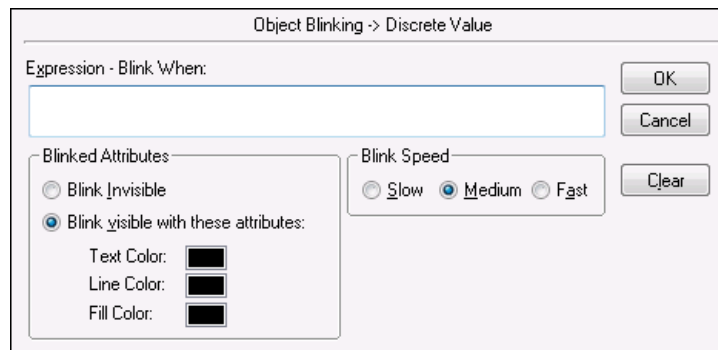
Making Objects Blink

You can create animated objects that blink based on the values of tags by using blink links. For example, you can create an object that blinks red when a certain piece of equipment is on, or when an alarm set point is reached.

Tip Discrete expressions can contain analog tagnames. For example, TankLevel > 75. In this example, when the value of the TankLevel tag is greater than 75, the object begins blinking.

To create a blink link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Miscellaneous** area, click **Blink**. The **Object Blinking -> Discrete Value** dialog box appears.



- 3 In the **Expression - Blink When** box, type the name of a discrete tag or an expression that equates to a discrete value.
- 4 In the **Blinked Attributes** area, do the following:
 - Click **Blink Invisible** to set the object to blink by disappearing and reappearing in the window.
 - Click **Blink visible with these attributes** to set the object to remain visible, but change color when activated.
 - Click the **Text Color**, **Line Color** or **Fill Color** boxes to select colors for those parts of the object. The color palette appears.

Note If you select a fill blink color that is the same as the object's fill color, the object does not appear to blink.

- 5 In the **Blink Speed** area, set the blinking speed of the object. Click either **Slow**, **Medium**, or **Fast**.
- 6 Click **OK**.

To set the blink frequency for WindowMaker

- 1 On the **Special** menu, point to **Configure** and then click **WindowViewer**. The **WindowViewer Properties** dialog box appears.
- 2 In the **Blink Frequency** area, type the number of milliseconds to use for the three speeds.

Note Changes you make to these settings are global and affect all blink speeds in your application.

- 3 Click **OK**.

Enabling Visibility

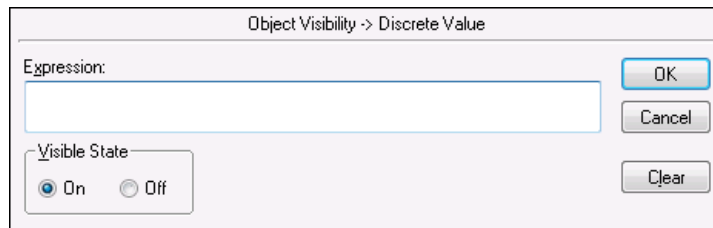
You can create links to hide objects based on the values of various tags by using visibility links. Using visibility links, you can:

- Create the impression that moving objects only move in one direction, by hiding them when they move in the wrong direction.
- Create the impression that a moving object has stopped.
- Cause an object such as an alarm or error message to become visible only when it is activated.

Tip Discrete expressions can contain analog tags, for example, `TankLevel >= 75`. In this example, when the value of the tag, `TankLevel` is greater than or equal to 75, the object becomes visible in the window.

To create a visibility link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Miscellaneous** area, click **Visibility**. The **Object Visibility -> Discrete Value** dialog box appears.



- 3 In the **Expression** box, type the name of a discrete tag or an expression that equates to a discrete value.

- 4 Select the **Visible State** for the object. If you select **Off**, the object is invisible when the value of the expression is true. If you select **On**, the object is visible when the value of the expression is true.
- 5 Click **OK**.

Disabling Objects

You can impose a level of security on your application with disable links. For example, you can disable touch sensitive objects based on operator access level or name. Or you can secure a button from tampering if no one is logged on.

A disabled state of ON means the touch functionality of the object or button is turned off and is not active as long as the expression is true.

Tip Discrete expressions can contain analog tagnames. For example, TankLevel >= 75. In this example, when the value of the variable "TankLevel" is greater than or equal to 75, the object is disabled.

To create a disable link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Miscellaneous** area, click **Disable**. The **Object Disabled -> Discrete Value** dialog box appears.

- 3 In the **Expression** box, type the name of a discrete tag or expression that equates to a discrete value.
- 4 In the **Disabled State** area, do one of the following:
 - Select **On** to set the disabled state so that the object does not activate while the discrete tag or expression is true.
 - Select **Off** to remove the disabled state and allow the object to function while the discrete tag or expression is true.
- 5 Click **OK**.

Configuring ToolTips

You can create ToolTips to give users information about an object on the screen by using tooltip links. ToolTips appear when the pointer moves over the object, and disappears when the pointer is moved away. The duration of time the tooltip appears and the positioning of the tooltip are determined by the operating system.

You can set either an expression or static text for your tooltips.

- Create a static tooltip to show the same message every time the tooltip appears.
- Create an expression tooltip so that every time the tooltip appears, the expression is evaluated and shown as the tooltip text.

For the following example expression, the text appears as the current value of the msgTooltipTag01 message tag.

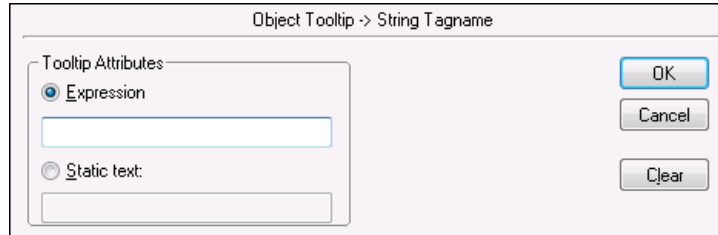
```
msgTooltipTag01
```

For this example, the literal string appears followed by the current value of the iTemp tag and the result appears as the tooltip text:

```
"Current temp. is " + StringFromIntg (iTemp,10)
```

To create a tooltip link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box opens.
- 2 In the **Miscellaneous** area, click **Tooltip**. The **Object Tooltip -> String Tagname** dialog box appears.



- 3 In the **Tooltip Attributes** area, select either **Expression** or **Static text**.
 - If you select **Expression**, enter an expression that evaluates to a message value. This can be a simple message tagname or a more complex expression.
 - If you select **Static text**, enter a static message, up to 131 characters, as the tooltip text.
- 4 Click **OK**.

Positioning a Touch-Sensitive Window

You can make a window appear during run time in a precise location relative to a touch sensitive object. For example, an operator can select an object to see the status, name, or other data linked to that object. When the operator selects the object, either by a click or mouse-over, the window appears in the location you specify.

Use the script functions `ShowAt()` or `ShowTopLeftAt()` with the system read only tags `$ObjHor` and `$ObjVer` to locate the window relative to the object. You can also use fixed positions in these functions.

If the Windows **Display Properties** is set to the **Windows XP** theme, this functionality is erratic under certain circumstances.

The syntax looks like this:

```
ShowTopLeftAt (windowname, $ObjHor, $ObjVer);
```

where

`windowname`: the name of the window to be opened.

`$ObjHor`: the horizontal position of the center of the object selected.

`$ObjVer`: the vertical position of the center of the object selected.

The new window appears with its top left corner at the center of the selected object.

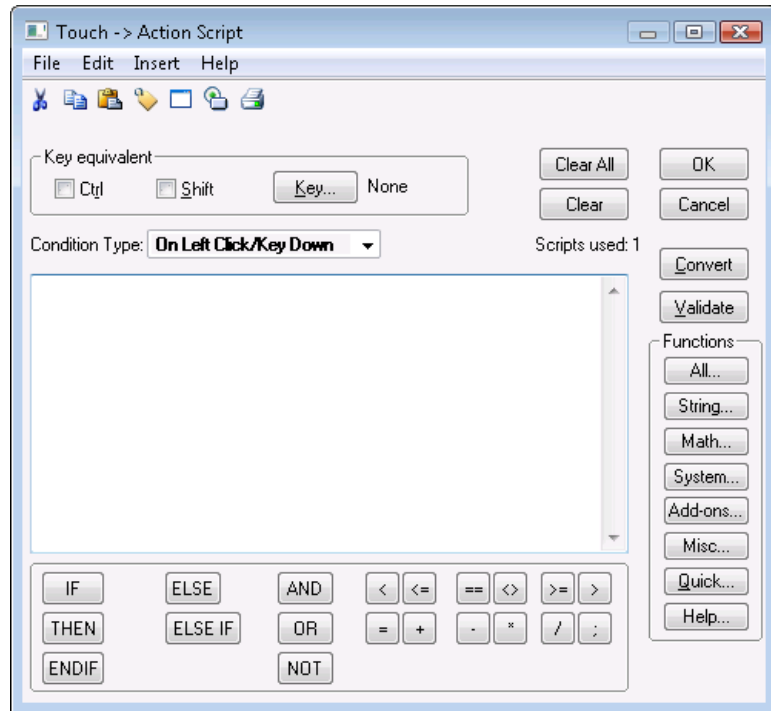
A similar script function opens the window with its center at the center of the selected object. The syntax looks like this:

```
ShowAt (windowname, $ObjHor, $ObjVer);
```

To open a window at the selected object

- 1 Design, name, and create the window to appear.
- 2 Right-click the object to trigger the window, and click **Animation Links**. The **Animation Links** dialog box appears.

- In the **Touch Pushbuttons** area, select **Action**. The **Touch -> Action Script** dialog box appears.



- Type the one of following scripts according to the syntax defined previously:

```
ShowTopLeftAt (windowname, $ObjHor, $ObjVer);
```

or

```
ShowAt (windowname, $ObjHor, $ObjVer);
```
- In the **Condition Type** box, click the mouse action to open the window.
- Click **OK**.

\$ObjHor System Tag

Contains the horizontal pixel location of the center of the object in focus.

Category

system

Usage

\$ObjHor

Data Type

Integer (read only)

See Also

\$ObjVer

\$ObjVer System Tag

Contains the vertical pixel location of the center the object in focus.

Category

system

Usage

\$ObjVer

Data Type

Integer (read only)

See Also

\$ObjHor

Data Entry Animations

To create objects that enable operator interaction, use touch links. Touch links allow operators to input data into the system. For example, an operator can log on with a keyboard, turn a valve on or off, enter a new alarm setpoint, or start or stop a process, and so on.

A frame appears around a touch-sensitive object when it has the focus. An object gets the focus when the user moves the cursor over it or presses the TAB or arrow keys to move the focus to the object.

If you expect your users to use the TAB key to select touch sensitive objects, try to arrange them in horizontal patterns. Pressing the Tab key moves the focus from one object to another from left to right beginning at the top of the window then moving down.

If a touch link object contains text objects that are placed on top of each other, the top text object is the only one that appears.

An operator can activate a touch-sensitive object by clicking it, pressing an assigned key equivalent, pressing **Enter** when the object frame appears, or by actually touching it if using a touch screen display device.

You can create nine types of user input touch links:

Touch Link	Action
User Inputs	<ul style="list-style-type: none"> • Discrete • Analog • String
Sliders	<ul style="list-style-type: none"> • Vertical • Horizontal
Pushbuttons	<ul style="list-style-type: none"> • Discrete Value • Action • Show Window • Hide Window

When a text field is used for input, text appears on the screen as the keys are pressed.

If you don't want text to appear as you type it, select the **Input Only** option in the configuration panel for the link.

Enabling Discrete Input

You can design operator input touch links to change the value of discrete tags from one state to another. For example to turn a pump on or off, use discrete links.

To create a discrete input link

- 1 Right-click an object and select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Touch Links** area, under **User Inputs**, select **Discrete**. The **Input -> Discrete Tagname** dialog box appears.

- 3 In the **Tagname** box, type the name of a discrete tag.
- 4 Optionally, in the **Key Equivalent** area, assign a key equivalent to the link. For more information, see [Creating Keyboard Shortcuts](#) on page 102.
- 5 Configure the discrete input details. Do the following:
 - In the **Msg to User** box, type the message to appear in the **Input** dialog box.
 - In the **Set Prompt** and **Reset Prompt** boxes, type the messages to appear on the buttons the operator will click to turn the discrete value on and off.
 - In the **On Message** and **Off Message** boxes, type the messages to appear in the text field associated with the object when the value is on or off.
- 6 Select the **Input Only** check box to prevent the input from appearing in a text field associated with the object.
- 7 Click **OK**.

Enabling Analog Input

You can create an object for operators to input real values such as alarm set point or conveyor speed by using an analog input link.

To create an analog input link

- 1 Right-click an object and select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Touch Links** area, under **User Inputs**, select **Analog**. The **Input -> Analog Tagname** dialog box appears.

- 3 In the **Tagname** box, type the name of an analog tag.
- 4 In the **Key Equivalent** area, assign a key equivalent to the link. For more information, see [Creating Keyboard Shortcuts](#) on page 102.
- 5 Configure the analog input details. Do the following:
 - In the **Keypad?** area, click **Yes** if you want to show an on-screen numeric keypad for inputting the new value. In the **Msg to User** box, type the prompt message that you want to appear in keypad.
 - In the **Min Value** and **Max Value** boxes, type the minimum and maximum input values for the tag.
- 6 Select the **Input Only** check box to prevent the input from appearing in a text field associated with the object.
- 7 Click **OK**.

Enabling String Input

You can create an object to enable users to input text strings such as batch names, operator ID, or passwords by using string input links.

To create a string input link

- 1 Right-click an object and select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Touch Links** area, under **User Inputs**, select **String**. The **Input -> String Tagname** dialog box appears.

- 3 In the **Tagname** box, type the name of a message tag.
- 4 Optionally configure a key equivalent and/or keyboard.
 - In the **Key Equivalent** area, assign a key equivalent to the link. For more information, see *Creating Keyboard Shortcuts* on page 102.
 - In the **Keypad?** area, click **Yes** if you want to show a keyboard for inputting the new value. In the **Msg to User** box, type the message to appear in keyboard.
- 5 In the **Echo Characters?** area, select if you want characters to appear in the input box as the user types the string.
 - Click **Yes** to show the typed text in the input box.
 - Click **No** to not show the typed text.
 - Click **Password** to use a “masking” character instead of the typed text. In the **Password Char** box, type the masking character. Select the **Encrypt** check box to encrypt the password.

Important Password encryption only works within the context of the InTouch HMI. Do not encrypt the string if you want to pass it to an external security system, such as the operating system or a SQL Server database. The external security system cannot read the encrypted password string and access will fail.

- 6 Select the **Input Only** check box to prevent the input from appearing in a text field associated with the object.
- 7 Click **OK**.

Enabling Sliders

You can create objects that users can drag back and forth through the use of slider touch links. As the user moves the object, it alters the value of the tag linked to it. You can link an object to a horizontal or a vertical slider.

When you make an object into a slider, you set the reference location, which is the point on the object the cursor uses to lock onto it.

You can use both horizontal and vertical links on a single object, so that the value of two analog tags are altered simultaneously.

To create a horizontal slider link

- 1 Right-click the object and click **Animation Links**. The **Animation Link** selection dialog box appears.
- 2 In the **Slider** area, click **Horizontal**. The **Horizontal Slider** dialog box appears.

- 3 In the **Tagname** box, type the name of an analog tag.
- 4 In the **Properties** area, do the following:
 - a In the **At Left End** box, type the value for the tag when the slider is in its farthest left position.
 - b In the **At Right End** box, type the value for the tag when the slider is in its farthest right position.
 - c In the **To Left** box, type the number of pixels the slider can move to the left.
 - d In the **To Right** box, type the number of pixels the slider can move to the right.
- 5 In the **Reference Location** area, click the location on the object that the cursor will use to lock onto the object.
- 6 Click **OK**.

To create a vertical slider link

- 1 Right-click the object and click **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Slider** area, click **Vertical**. The **Vertical Slider** dialog box appears.

Vertical Slider

Tagname:

OK

Cancel

Clear

Properties

	Value		Vertical Movement
At Top:	<input type="text" value="0"/>	Up:	<input type="text" value="0"/>
At Bottom:	<input type="text" value="100"/>	Down:	<input type="text" value="100"/>

Reference Location

Top Middle Bottom

- 3 In the **Tagname** box, type the name of an analog tag.
- 4 In the **Properties** area, do the following:
 - a In the **At Top** box, type the value for the tag when the slider is in its farthest up position.
 - b In the **At Bottom** box, type the value for the tag when the slider is in its farthest down position.
 - c In the **Up** box, type the number of pixels the slider can move up.
 - d In the **Down** box, type the number of pixels the slider can move down.
- 5 In the **Reference Location** area, click the location on the object that the cursor will use to lock onto the object.
- 6 Click **OK**.

Enabling Push Buttons

You can create touch sensitive objects that start action scripts by using touch push button links.

Action scripts can set tags to specific values, start and control other applications, execute functions and so on.

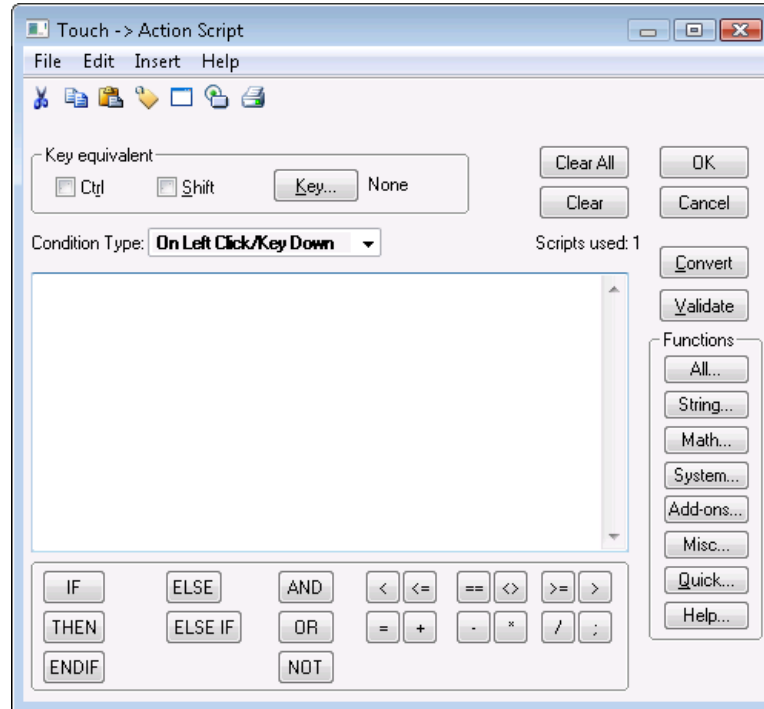
To create a discrete value touch link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Touch Pushbutton** area, click **Discrete Value**. The **Pushbutton -> Discrete Value** dialog box appears.

- 3 In the **Tagname** box, type a discrete type tagname.
- 4 Click **Key** to assign a key equivalent to the link.
- 5 In the **Action** area, click one of the following types:
 - Click **Direct** to set the value equal to 1 as long as the push button is pressed and held down. The value automatically resets to 0 when the button is released.
 - Click **Reverse** to set the value equal to 0 when the push button is pressed and held down. The value automatically resets to 1 when the button is released.
 - Click **Toggle** to reverse the state of the discrete tag.
 - Click **Reset** to set the value equal to 0.
 - Click **Set** to set the value equal to 1.
- 6 Click **OK**.

To create an action script link

- 1 Right-click the object and then select **Animation Links**. The Animation Links dialog box appears.
- 2 In the **Touch Pushbutton** area, click **Action**. The **Touch -> Action Script** editor appears.



- 3 In the **Condition Type** list, select a script type. For more information on the types of action scripts, see Chapter 3, *Script Triggers*, in the *InTouch® HMI Scripting and Logic Guide*.

Note If you assign a key equivalent link to an action push button and to the same key is used for a key script, the key equivalent link you assign takes precedence over the key script.

- 4 In the **Script Editor** window, type the script to execute when the object is activated.
- 5 Click **OK**.

Opening and Closing Windows

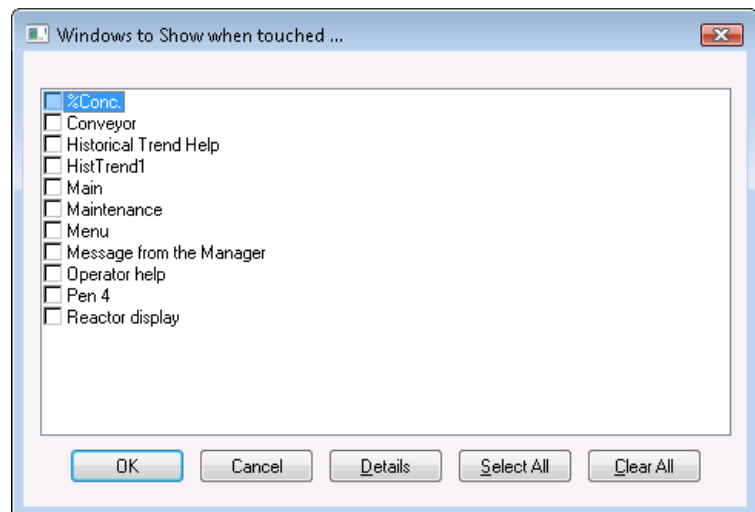
You can create touch links to open and close other InTouch windows by using show window and hide window links.

You can program objects to open more than one window at a time. However, if you program your link to open more than one window, be aware of any intersections and window types.

If one of the opening windows is a replace type window and intersects another opening window, the other window closes before opening.

To create a show (or hide) window link

- 1 Right-click the object and then select **Animation Links**. The **Animation Links** dialog box appears.
- 2 In the **Touch Pushbutton** area, click **Show Window** or **Hide Window**. The **Windows to Show when touched** or **Windows to Hide when touched** dialog box appears.



- 3 Select the window(s) to open or hide.
- 4 Click **OK**.

Configuring On-Screen Keyboards

On-screen keyboards allow operator input in situations where a keyboard is not connected to the computer.

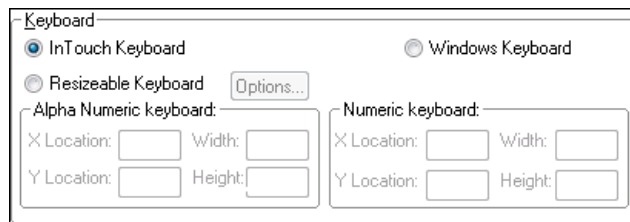
You can designate one of three on-screen keyboard types.

- The standard InTouch keyboard or keypad. This is the default keyboard.
- The Windows system keyboard. The Windows keyboard is a fully functional QWERTY-type keyboard with function keys, print screen key, number lock key, directional arrows, and so on.
- The re-sizable keyboard or keypad. This keyboard can be resized during run time.

You can also open the keyboard by using the `DialogStringEntry()` and `DialogValueEntry()` function in a script. For more information, see `DialogStringEntry()` Function on page 93 and `DialogValueEntry()` Function on page 94.

To configure the on screen keyboard type

- 1 On the **Special** menu, point to **Configure** and then click **WindowViewer**. The **WindowViewer Properties** dialog box appears.
- 2 In the **Keyboard** area, select type of keyboard you want.



- 3 If you select the re-sizeable keyboard, click **Options** to select font, location, and dimension properties of the keyboard.
- 4 Click **OK**.

To make an on screen keyboard appear

- 1 Configure the on-screen keyboard type.
- 2 Right-click on a text object and select **Animation Links**. The **Animation Links** dialog box appears.

- 3 In the **Touch Links** area, under **User Inputs**, select **String**. The **Input->String Tagname** dialog box appears.

- 4 In the **Keypad?** area, select **Yes**.
- 5 Click **OK**.

DialogStringEntry() Function

Shows an alphanumeric keyboard on the screen, allowing the operator to change the current string value of a message tag in the Tagname Dictionary.

Category

misc

Syntax

```
[Result=]DialogStringEntry(MessageTag_Text,
    UserPrompt_Text);
```

Parameters

MessageTag_Text

The name of the message tag to be modified. This value is a string value. Specify the tagname within quotes or use the .Name dotfield without quotes. You can also use a message tag as a pointer.

UserPrompt_Text

The user message to show at the top of the keyboard.

Return Value

Returns one of the following integer values:

- 0 = Cancel was pressed.
- 1 = OK was pressed.
- 1 = Internal error.
- 2 = Could not initiate.
- 3 = Tagname not defined.
- 4 = Tagname is not a Message type.
- 5 = Unable to write.

Remarks

This function is used primarily in applications containing touch screens.

Example

```
Errmsg=DialogStringEntry(MyMessageTag.Name, "Enter a  
new string...");  
Errmsg=DialogStringEntry("MyMessageTag","Enter a new  
string...");
```

For example, the following script opens an alphanumeric keyboard, allowing modification of MyMessageTag while showing the message "Enter a new string..." at the top of the keyboard:

```
MessageTagX="MyMessageTag"; {assign the string  
MyMessageTag (which is actually the tagname to be  
modified) to the Memory Message tagname MessageTagX}  
MessageDisplay="Enter a new string..."; {assign the new  
message string to the Memory Message tagname  
MessageDisplay}  
Errmsg=DialogStringEntry(MessageTagX, MessageDisplay);  
{quotes are not required because MessageTagX was  
defined as a Message tagname}
```

See Also

DialogValueEntry()

DialogValueEntry() Function

Shows the numeric keypad on the screen, allowing the user to change the current value of a discrete, integer or real tag.

Category

misc

Syntax

```
[Result=] DialogValueEntry(ValueTag_Text, LowLimit,  
HighLimit, UserPrompt_Text);
```

Parameters*ValueTag_Text*

The name of the discrete, integer, or real tag to modify. This value is a string value. Specify the tagname within quotes or use the .Name field without quotes. You can also use a message tag as a pointer.

LowLimit

The minimum allowable value for the tag. (This should be \geq the tagname's definition for minimum value, minimum raw or minimum engineering unit, as applicable).

HighLimit

The maximum allowable value for the tag. (This should be \leq the tagname's definition for maximum value, maximum raw or maximum engineering unit, as applicable).

UserPrompt_Text

The user message to show at the top of the keypad.

Return Value

Returns one of the following integer values:

- 0 = Cancel was pressed.
- 1 = OK was pressed.
- 1 = Highlimit \leq Lowlimit.
- 2 = Could not initiate.
- 3 = Tagname not defined.
- 4 = Tagname is not a Discrete, Integer or Real type.
- 5 = Write failed.

Remarks

This function is used primarily in applications containing touch screens.

Example(s)

```
Errmsg=DialogValueEntry(MyIntegerTag.Name,  
    MyIntegerTag.MinEU, MyIntegerTag.MaxEU, "Enter a new  
    value...");  
  
Errmsg=DialogValueEntry("MyIntegerTag", -100, 100,  
    "Enter a new value...");
```

For example, the following script brings up the numeric keypad, allowing modification of `MyIntegerTag` using a minimum and maximum limit of -100 and 100 (respectively), while showing the message "Enter a new value..." at the top of the keypad:

```
TagnameX="MyIntegerTag"; {assign the string  
    MyIntegerTag (which is actually the tagname to be  
    modified) to the Memory Message tagname TagnameX}  
  
Min=-100; {assign the minimum value allowed for the  
    tagname to the Memory Real/Integer tagname Min}  
  
Max=100; {assign the minimum value allowed for the  
    tagname to the Memory Real/Integer tagname Max}  
  
MessageDisplay="Enter a new value..."; {assign the new  
    message string to the Memory Message tagname  
    MessageDisplay}  
  
Errmsg=DialogValueEntry(TagnameX, Min, Max,  
    MessageDisplay); {quotes are not required because  
    TagnameX was defined as a Message tagname. By  
    assigning a Discrete, Integer or Real tagname to  
    TagnameX, the function will modify that assigned  
    tagname}
```

See Also

`DialogStringEntry()`

Common Animation Tasks

Common animation tasks include selecting tags, creating keyboard shortcuts, changing tagname references and replacing placeholder tags.

Selecting Tags or Attributes

Use the **Select Tag** dialog box, also called the tag browser, to select:

- Tags defined in a local or remote InTouch application.
- ArcestrA object attributes with the Attribute Browser.

You open the **Select Tag** dialog box by double-clicking in any text box that requires a tag name for input.

Selecting an InTouch Tag

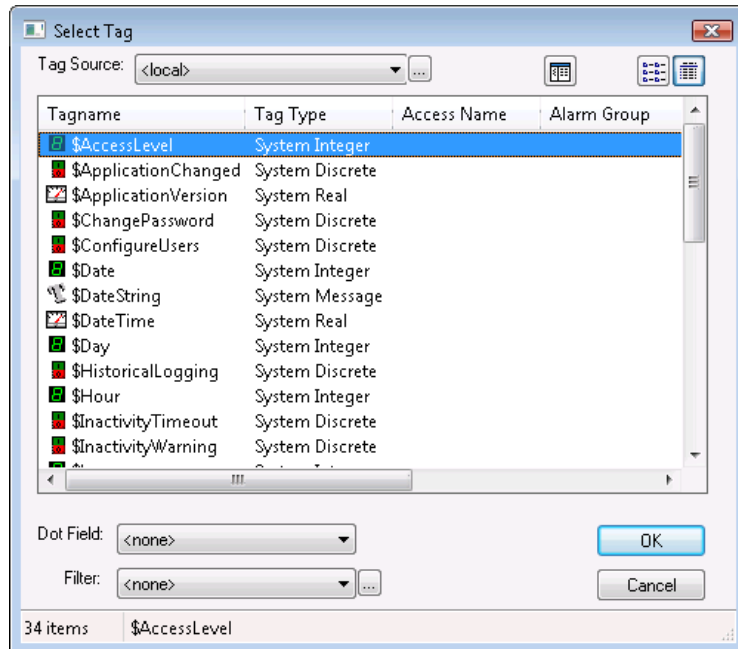
You can select tags defined in a local or remote InTouch application. You can create references to tags in any tag source that supports the Tagname Dictionary interface.

For example, remote tag references allow your application to access data from I/O Servers without creating tags in the local Tagname Dictionary.

You can set a dotfield for each InTouch tag you select. Dotfields can access, monitor, and modify tag properties. If you do not select a dotfield, the .Value dotfield is used. For more information about dotfields, see Chapter 4, Using Tag Dotfields to View or Change Tag Properties, in the *InTouch® HMI Data Management Guide*.

To select an InTouch tag

- 1 Open the **Select Tag** dialog box by double-clicking in any text box that requires a tag name for input.



- 2 In the **Tag Source** list, click the name for the tag source or click the browse button to define a new tag source to use.

For more information on defining tag sources, see Chapter 5, Data Access with I/O, in the *InTouch® HMI Data Management Guide*.

- 3 In the **Filter** list, click a filter to reduce the number of tagnames shown in the window. To define a filter, click the ellipsis button. For more information, see *Creating a Tag Filter* on page 100.
- 4 Select a tagname in the window.

You can change the way the tagnames are shown in **Select Tag** dialog box. For more information, see *Changing the View in the Select Tag Dialog Box* on page 101.

- 5 In the **Dot Field** list, click a dotfield to append to the selected tagname.

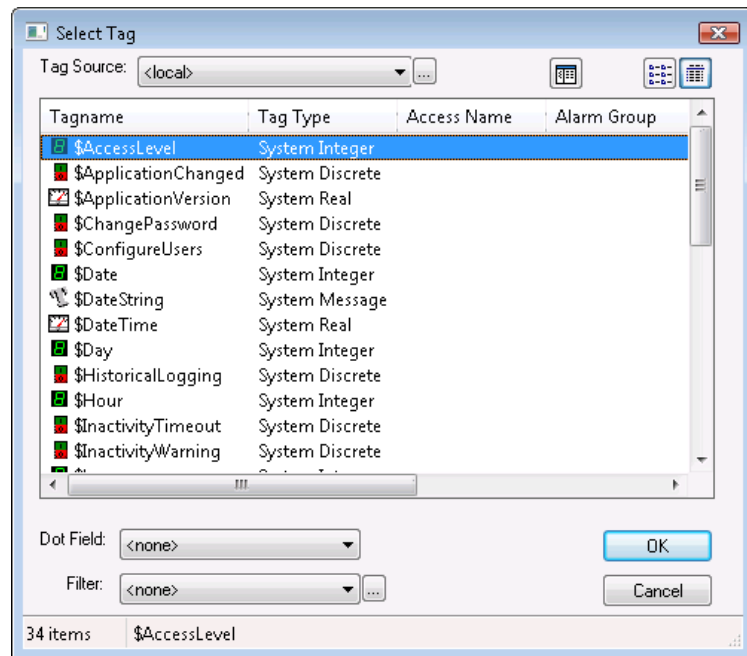
Dotfields can access, monitor and modify tag properties. If you do not select a dotfield, the `.Value` dotfield is used. Click **OK**.

Selecting an ArchestrA Object Attribute

You can select attributes that are associated with an ArchestrA object. To do this, you must first add the Galaxy that contains the object as a data source for your InTouch application. For more information about data sources, see Chapter 5, Data Access with I/O, in the *InTouch® HMI Data Management Guide*.

To select an object attribute

- 1 Open the **Select Tag** dialog box by double-clicking in any text box that requires a tag name for input.



- 2 In the **Tag Source** list, click the name of the tag source that uses the Galaxy or click the browse button to define a new tag source to use. The **Attribute Browser** dialog box appears.
- 3 Use the **Attribute Browser** to find and select an ArchestrA object attribute. For more information, see the Wonderware Application Server documentation.
- 4 Click **OK** to close the **Attribute Browser**. The attribute reference appears in the text box that requires a tag name.

Note To return to the **Select Tag** dialog box from the **Attribute Browser**, click the **Back** button in the bottom right corner of the **Attribute Browser**.

Creating a Tag Filter

If you have a large number of tags in your Tagname Dictionary, finding the tag for an animation can be cumbersome. For example, you may only want to see the tags assigned to a particular Access Name or Alarm Group. You can configure a filter so that only a subset of tags are shown in the **Select Tag** dialog box.

You can use the following wildcard expressions in your filter.

- The multiple character wildcard is the asterisk (*). For example, “Asyn*” searches for all tagnames beginning with the characters “Asyn”.
- The single character wildcard is the question mark (?). For example, the “Tag?” filter searches for all four-character tagnames that begin with “Tag”. The “Tag*” filter searches for all tagnames that begin with “Tag”.
- Any sequence of valid tagname characters, together with the two wildcard characters, is acceptable in a filter. Valid tagname characters are: A-Z, a-z, 0-9, !, @, -, #, \$, %, _ and &.

To define a search filter

- 1 In the **Select Tag** dialog box, click the ellipsis button next to the **Filter** list. The **Define Tag Filter** dialog box appears.

- 2 In the **Filter Name** box, type a filter name.




- 3 In the **Filter Options** area, configure the filter criteria. Do any of the following:
 - In the **Tagname** box, type the tag name.
 - In the **Tag Type** box, type the tag type.
 - In the **Access Name** box, type the local Access Name.
 - In the **Alarm Group** box, type the Alarm Group name.
 - In the **Comment** box, type the comment expression.
- 4 Click **OK**. The **Filter Name** appears in the **Filter** list in the **Select Tag** dialog box. You can select the filter to show the tags meeting the filter criteria.

To delete a search filter

- 1 In the **Filter Name** box, click the filter.
- 2 Click **Delete**.

Changing the View in the Select Tag Dialog Box

The **Select Tag** dialog box has three different views: List, Details, and Tree View.

To see the	Click the	Description
List view	List view button 	Small icons appear next to the tagnames according to the type of each tagname.
Details view	Details view button 	You see the same small icons and tagnames, plus Tag Types, Access Name, Alarm Group, and Comments. Sort the list by clicking the column headers.
Tree View	Tree view icon 	The Tree View displays the tagnames in two views. You can access the member tagnames in any SuperTag template.

Creating Keyboard Shortcuts

You can assign a specific key on the keyboard to activate certain animation links by using key equivalent links. The key equivalent is only operational when the object with the link is visible or selected. If the object has a visibility or disable link, the key equivalent is not active when the object is disabled.

You can define the same key in multiple windows. However, the definition in the most recently opened window is the active one. In the case of overlay windows, the key is active in the window on top.

Note If any object or action push button in the active window is assigned to the same key used for a key action script, the key equivalent link on the key in the active window takes precedence over the execution of the key action script.

The animation links that support key equivalents appears the **Key Equivalent** area in their link dialog boxes.

Key links are only available for function keys 1-16. If you are using a custom keyboard that has more than 16 function keys, get a device driver from your manufacturer that allows you to access the extended function keys on your system.

To assign a key to a link

- 1 Open the **Animation Links** dialog box for the type of link you are configuring.
- 2 Select **Ctrl** and/or **Shift** if you want the operator to hold down either or both of these keys when pressing the key equivalent.
- 3 Click **Key**. The **Choose key** dialog box appears.
- 4 Click the key to assign to the link. The **Animation Link** dialog box reappears with the name of the selected key next to the **Key** button.
- 5 Click **OK**.

Changing Tagname References

When you duplicate an object, you get an exact copy of the original including links, animation, scripts and so on. However, if you need to use a different tag on a duplicated object, you must substitute a tag.

You can select and substitute a tag for any object and you can select multiple objects and substitute all their tags at the same time.

If your system license supports a limited number of tagnames, convert your local tags to remote tag references to reduce the number of tags defined in your local Tagname Dictionary.

Substituting a tag works for all tags and references.

To substitute a tag with another tag

- 1 Select the object(s) associated with the tag to change to another tag.
- 2 On the **Special** menu, click **Substitute Tags**. The **Substitute Tagnames** dialog box appears.
- 3 In the **New Name** box, type the new tagname.
 - If you double-click a tag in the **New Name** box, its definition in the Tagname Dictionary appears.
 - If you erase the tagname then double-click the blank box, the **Select Tag** dialog box appears.
- 4 Click **OK**. The tag associated with the object is automatically changed.

Converting Placeholder Tags

If you import or export a window or QuickScript to or from your current application, all the tags associated with that window or QuickScript are transferred with the window.

The local tags are not automatically added to your application database. Instead, they are automatically marked as placeholder tags.

Remote tag references are not affected, and are not marked as placeholder tags.

You must convert the placeholder tags to existing local tags, define them in the new application's Tagname Dictionary, or make them remote tag references.

Notice the placeholders **?d:**, **?i:**, **?m:** and **?r:** preceding the tags. They indicate the type that the tag was originally defined as:

Placeholder Symbol	Tagname Type
d	Discrete
i	Integer
m	Message
r	Real

Note Remote tag references are not shown as placeholders but as remote tag references such as: **PLC2:Temperature**.

You can use several methods in the **Substitute Tagnames** dialog box to convert placeholder tags to local tags. For more information, see Chapter 4, Exporting and Importing Tag Definitions, Windows, and Scripts, in the *InTouch® HMI Application Management and Extension Guide*.

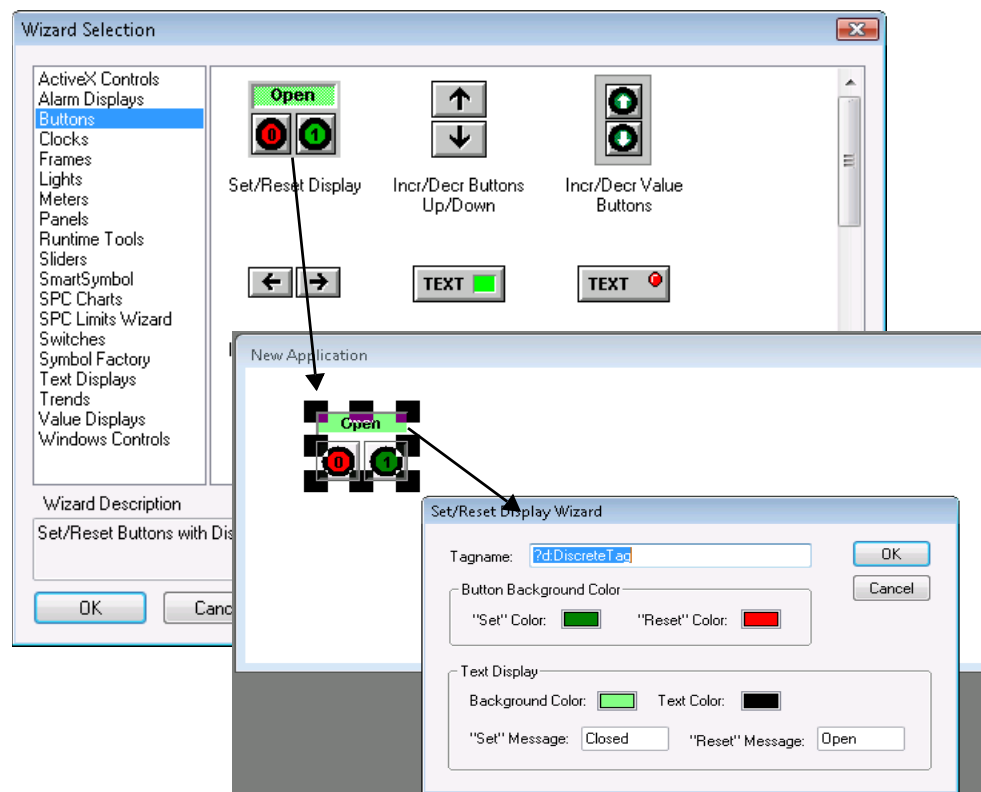
Tip If you manually convert tagnames and you no longer need the original tag defined in the original Tagname Dictionary, you can update the tagname use counts and then delete the unused tag.

By importing a window or QuickScript from another application, and converting all of the tagnames associated with the animation links or QuickScript(s) to remote tagname references, you can create an application that instantly receives data from hundreds of remote tagnames without defining a single tag in your local Tagname Dictionary.

Chapter 5

Wizards

Wizards are pre-designed, pre-built, and pre-programmed objects you only need to select, place and configure for your application.



Working with Wizards

Using wizards, you do not spend time drawing the individual components for the object, or entering the value ranges for the object, or animating the object.

- You select wizards from the Wizards/ActiveX toolbar.
- You configure wizards by typing tags and QuickScripts in configuration dialog boxes.
- When you paste a selected wizard into a window and then double-click it, the **Configuration** dialog box appears.

For example, in the case of a slider wizard, the configuration includes items such as the tagname to effect, the minimum and maximum range labels for the slider, the fill color, and so on. After the required configuration information is provided, the wizard is ready to use.

You can also develop your own complex wizards to provide behind the scenes types of operations. These operations can include creating complete display windows, creating or converting a database, importing an AutoCAD drawing, and configuring other applications.

Before creating your own wizards, you should investigate ArchestrA symbols, which offer wizard functionality, but do not require programming.

Types of Wizards

Categories of wizards are shown in the **Wizard Selection** dialog box.



Trend Objects and Windows Controls Wizards are special wizards with unique parameters. For more information, see Trend Objects on page 110 and Windows Controls Wizards on page 111.

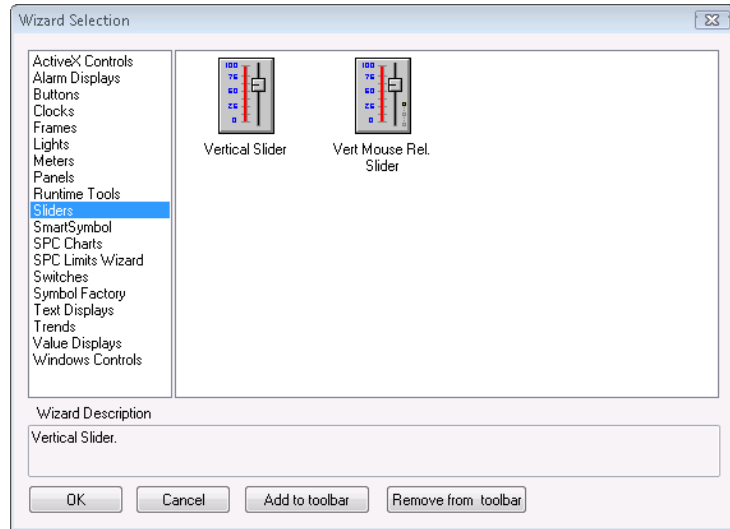
Adding Wizards to the Toolbar

You can add frequently used wizards to the Wizard/ActiveX Toolbar for quick availability.

To add a wizard to the Wizards/ActiveX Toolbar



- 1 Click the **Wizard** button in the **Wizards/ActiveX Toolbar**. The **Wizard Selection** dialog box appears.



- 2 In the left pane, select a wizard category, such as **Sliders**.
- 3 In the right pane, select a wizard, and then click **Add to toolbar**. The wizard button appears in the toolbar.

To remove wizards from the toolbar



- 1 Click the **Wizard** button in the **Wizards/ActiveX Toolbar**. The **Wizard Selection** dialog box appears.
- 2 Click **Remove from toolbar**. The **Remove Wizard from Toolbar** dialog box appears.
- 3 Select the wizard to remove and click **OK**.

Pasting Wizard Instances

You can place instances of a wizard in a window.

To place a wizard in a window



- 1 Click the Wizard button in the **Wizards/ActiveX Toolbar**. The **Wizard Selection** dialog box appears. In the left pane, select a wizard category.
- 2 In the right pane, select a wizard.
- 3 Click **OK**. The dialog box closes, and the cursor becomes a corner symbol.
- 4 Click the location to put the wizard.

Configuring Wizards

After you place a wizard in your application, double-click it to configure its properties. A properties dialog box appears that is custom to the selected wizard.

For more information on each particular type of wizard, see the wizard Help, if available.

Performing Standard Operations on Wizards

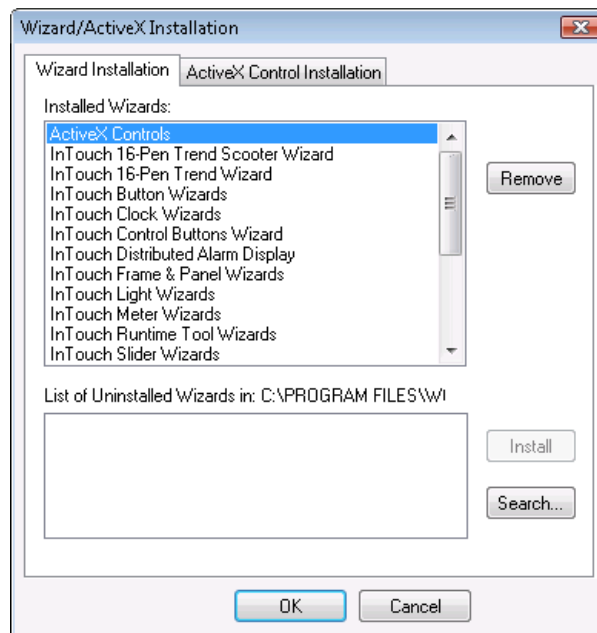
You can cut, copy, paste, delete, and duplicate wizards, in the same way and with the same results as with other objects.

Installing and Removing Wizards

You must install a wizard to WindowMaker so that you can use it in your application. When you remove a wizard from WindowMaker, it is not deleted from your computer.

To install a wizard

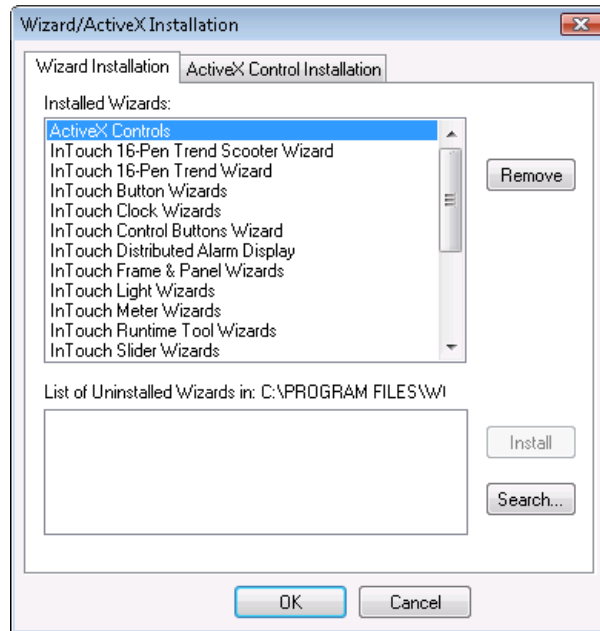
- 1 On the **Special** menu, point to **Configure** and then click **Wizard/ActiveX Installation**. The **Wizard/ActiveX Installation** dialog box appears.



- 2 Select from the **List of Uninstalled Wizards** and then click **Install**.

To remove a wizard

- 1 On the **Special** menu, point to **Configure** and then click **Wizard/ActiveX Installation**. The **Wizard/ActiveX Installation** dialog box appears.



- 2 In the **Installed Wizards** list, select the wizard to remove and then click **Remove**. A message box appears asking you to confirm the deletion.

Tip SHIFT+click or CTRL+click to make multiple selections.

- 3 Click **Yes** to remove the wizard. The wizard appears in the **List of Uninstalled Wizards** list.

To import wizards from another directory

- 1 On the **Special** menu, point to **Configure** and then click **Wizard/ActiveX Installation**. The **Wizard/ActiveX Installation** dialog box appears.
- 2 Click **Search**. The **Browse for Folders** dialog box appears.
- 3 Browse to the directory containing the wizards to install and click **OK**. The **Wizard Installation** dialog box reappears with the imported wizards in the **List of Uninstalled Wizards**.

Trend Objects

Trend objects are wizards that chart the values of tags over time.

There are three main types of trend objects:

- Real-time trends chart up to four tags in real time.
- Historical trends chart up to eight tags over a past time period.
- 16 Pen trends chart real-time and historical data for up to sixteen tags.

There is no limit to the number of trend objects, real-time or historical, you can place in a window.

Configure trend objects with the following items:

- Time span
- Value range
- Grid resolution
- Location of time and value stamps
- Pens and colors

Before you can use a trend wizard, you must enable logging for each tag to track, and also enable logging within the InTouch application.

To enable logging for tags

- 1 From the **Tagname Dictionary**, select a tag and then select **Log Data**.
- 2 If you have not done so previously, enable logging in the InTouch application.
 - a On the **Special** menu, point to **Configure**, and then click **Historical Logging**. The **Historical Logging Properties** dialog box appears.
 - b Select the **Enable Historical Logging** check box and click **OK**.

For more information on configuring and using trend objects, see Chapter 10, Trending Tag Data, in the *InTouch® HMI Data Management Guide*.

Windows Controls Wizards

The Windows Controls Wizards are user interface objects such as drop-down lists, combo boxes, option selections or check boxes.

Use Windows Control Wizards to present predefined sets of choices to your users. For example, you might create a drop down list of processes, recipes, or operator IDs. You can also enable and disable specified controls. You can even load and modify the contents of drop-down lists.

Run-time properties for Windows Controls Wizards are accessed through QuickScript functions rather than animation link expressions.

Windows Controls Wizards have properties like InTouch tag dotfields. They can be read-write or read-only. Some properties are accessible at development and some at run time. They are identified as ControlName.x, where x is the property.

For example, if the .Visible property of a windows control is equal to 0, the control is not visible in the window. As with InTouch tags, .Value is the default property for Windows Control Wizards.

Note A more robust and flexible set of .NET based Windows controls is available if you use ArchestrA Symbols.

Creating and Configuring Windows Controls

When you create and configure Windows Control Wizards, keep in mind that:

- Windows Control Wizards cannot overlap one another.
- Each Windows Control Wizard has a unique control name—which does not add to the tag count.
- The initial value of tags assigned to either a list box or combo box does not initialize the value of the list box or combo box. You must use the `SetPropertyX QuickScript` functions in a script to assign initial values that need to be different than the default values.

Tip Paste Windows Controls Wizards into your windows just like other wizards. To achieve the best readability, select a gray background for your windows controls. If your background color cannot be gray, place a gray Panel Wizard behind the Windows Control Wizards.

For each Windows Controls Wizard, you must specify an alphanumeric control name, where the first character is a letter. Underscores are allowed, but other special characters are not. For example, “Checkbox_1” is allowed, but “Checkbox#1” is not.

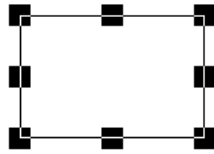
You can configure Windows Control Wizards using InTouch QuickScripts.

Creating a Text Box

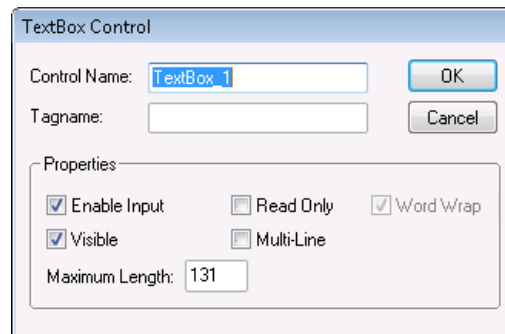
You use text boxes in your application to allow operators to enter text strings.

To create and configure a text box

- 1 Create and place a text box. Do the following:
 - a From the **Wizard Selection** dialog box, select **Windows Control Wizards**. The control wizard icons appear.
 - b Double-click the text box icon. Your application window reappears and the cursor changes to a left corner symbol.
 - c Click a location in the application window to place the wizard. The text box wizard appears with heavy black handles at the corners.



- d Drag and resize the wizard to suit your application.
- 2 Double-click the wizard. The **Textbox Control** dialog box appears.



- 3 Configure the dialog box. Do the following:
 - a Type a control name, such as *TextBox_1*, in the **Control Name** box.
 - b Type a memory message tag name, such as *New_Value*, in the **Tagname** box.
 - c In the **Properties** area, select **Enable Input** and **Visible**.
- 4 Click **OK**. The **Textbox Control** dialog box closes.

Creating a List Box

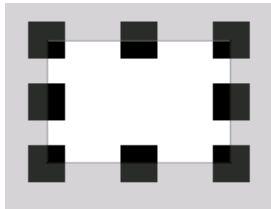
You can create applications that enable your users to select items from a list of options.

When adding a list box to your application, you place the control on the screen, set the properties to configure the list, and then write any scripts you may need.

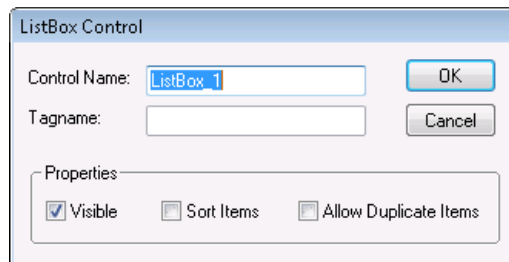
List boxes can be loaded with items from a file or from keyboard input during run time. For more information, see *Scripting Windows Controls* on page 119.

To create a list box

- 1 Create and place the list box control. Do the following:
 - a In the **Wizard Selection** dialog box, select **Windows Controls**.
 - b Double-click the list box icon. Your application window reappears and the cursor changes to a left corner symbol.
 - c Click a location in the application window to place the control. The list box control appears.



- 2 Double-click the control. The **ListBox Control** dialog box appears.



- 3 Configure the control. Do the following:
 - a Type a control name, such as *ListBox_1*, in the **Control Name** box.
 - b Type a memory message tag name, such as *LB1_Value*, in the **Tagname** box.
 - c In the **Properties** area, configure how the control appears and functions.
- 4 Click **OK**.

Creating a Combo Box

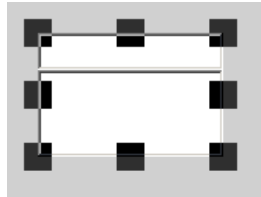
You can create applications that enable your users to select items from lists of options with a combo box. A combo box is a Windows control combining a text box and a list box.

When adding a combo box to your application, place the control on the screen, set the properties to configure the combo box, and then write any scripts you may need.

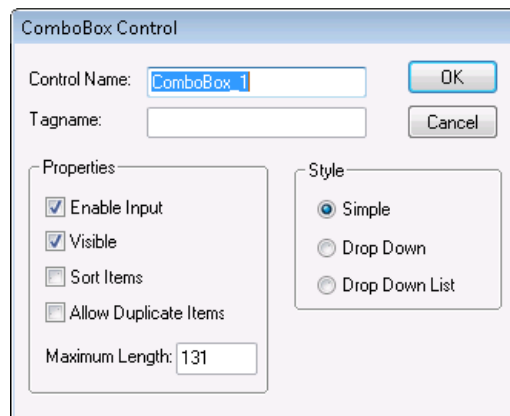
Combo boxes can be loaded with items from a file or from keyboard input during run time. For more information, see *Scripting Windows Controls* on page 119.

To create a combo box

- 1 Create and place the combo box control. Do the following:
 - a In the **Wizard Selection** dialog box, select **Windows Controls**.
 - b Double-click the combo box icon. Your application window reappears and the cursor changes to a left corner symbol.
 - c Click a location in the application window to place the control. The combo box control appears.



- 2 Double-click the control. The **ComboBox Control** dialog box appears.



- 3 Configure the control. Do the following:
 - a Type a control name, such as *ComboBox_1*, in the **Control Name** box.
 - b Type a memory message tag name, such as *CBI_Value*, in the **Tagname** box.
 - c In the **Properties** area, configure how the control appears and functions.
 - d In the **Style** area, select the type of combo box.
- 4 Click **OK**.

Creating a Check Box

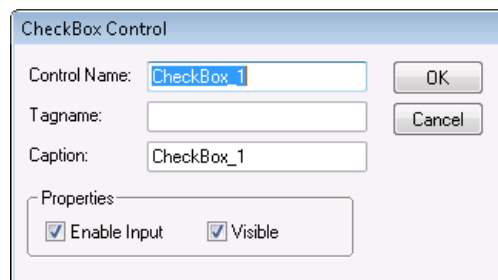
A check box control allows an operator to select an option.

To configure a check box control

- 1 Create the check box. Do the following:
 - a From the **Wizard Selection** dialog box, select **Windows Control Wizards**. The control wizard icons appear.
 - b Double-click the check box icon. Your application window reappears and the cursor changes to a Left corner symbol.
 - c Click a location in the application window to place the wizard. The check box wizard appears.



- d Drag and resize the wizard.
- 2 Double-click the wizard. The **Checkbox Control** dialog box appears.



- 3 Configure the wizard. Do the following:
 - a Type the control name.
 - b Type the discrete tag name, or double-click the empty tagname box to show the **Select Tagname** dialog, and select a tag.
 - c Type the caption to appear on the face of the button.
- 4 Click **OK**.

Creating a Radio Button Group

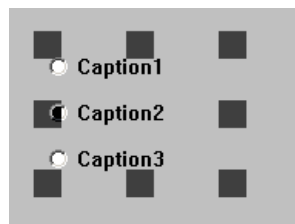
You use radio buttons when the user must select from one of several choices. When a user selects an option, the previously selected option is un-selected.

You create the options for a user as a group of radio buttons. Each radio button has a caption, and provides a unique value to your script.

You can only assign integer tags to radio button controls.

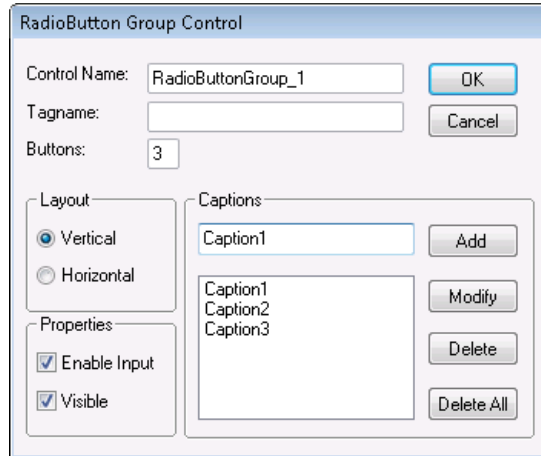
To create a Radio Button group

- 1 Create the Radio Button control wizard. Do the following:
 - a From the **Wizard Selection** dialog box, select **Windows Control Wizards**. The control wizard icons appear.
 - b Double-click the Radio button icon. Your application window reappears and the cursor changes to a left corner symbol.
 - c Click a location in the application window to place the wizard. The Radio Group control wizard appears with three radio buttons.



- d Drag and resize the wizard to suit your application.

- 2 Double-click the wizard. The **Radio Button Group Control** dialog box appears.



- 3 Configure the wizard. Do the following:
 - a Type the control name.
 - b Type an integer type tag name to link to this control.
 - c Type the number of buttons to show.
 - d Type the captions for each button.
 - e Set the layout and properties.
- 4 Click **OK**.

Scripting Windows Controls

You can use QuickScript functions in scripts to:

- Get or set the value of a control.
- Enable, disable, or hide a control.
- Work with items in combo boxes, list boxes, text boxes, and check boxes.

The run-time properties can be either read-write or read-only, depending on the property.

Use the `GetPropertyX()` and `SetPropertyX()` functions to control or retrieve these properties.

Getting or Setting the Value of a Control

The `.Value` property is the default property for all InTouch Windows Controls Wizards.

Changes made to this property are synchronized in the InTouch tagname and the Windows Controls Wizards.

.Value Dotfield

Default property for all InTouch windows control wizards. Changes made to this property are synchronized in the InTouch tagname and the windows control wizards.

Category

windows control

Usage

The M, I and D are for the Memory, Integer and Discrete versions of the `GetProperty` and `SetProperty` functions.

```
[ErrorNumber=]GetPropertyM("ControlName[.Value]",
    Tagname);

[ErrorNumber=]SetPropertyM("ControlName[.Value]",
    Value);

[ErrorNumber=]GetPropertyI("ControlName[.Value]",
    Tagname);

[ErrorNumber=]SetPropertyI("ControlName[.Value]",
    Value);

[ErrorNumber=]GetPropertyD("ControlName[.Value]",
    Tagname);

[ErrorNumber=]SetPropertyD("ControlName[.Value]",
    Value);
```

Parameters*ControlName*

Name of the windows control, e.g., ChkBox_4.

Tagname

The tagname to which the value of the property is written.

[.Value]

This property is optional. If not specified, the function always assumes the .Value property is being used.

Value

The actual value to be written or a valid InTouch tagname (of the same type as the property to be written to) that holds the property value to be written when the function is processed.

Remarks

The initial value of tagnames assigned to either a list box or combo box cannot be used to initialize the value of the list box or combo box.

This dotfield is read/write during development and run time. If the .Value dotfield is accessed by associating a tagname to either a list box or a combo box, it is read-only. If the .Value dotfield is assigned to a check box, radio button, or text box, it is read/write. The value you specify at development serves as the default for run time.

Data Type

Message (read/write) for text boxes, list boxes and combo boxes.

Integer (read/write) for radio buttons.

Discrete (read/write) for check boxes.

Applies To

Text boxes, list boxes, combo boxes, check boxes and radio buttons.

Example(s)

The following statement sets the .Value dotfield of the radio button object "RadioButton_1" to a value of 4:

```
SetPropertyI( "RadioButton_1.Value", 4 );
```

See Also

GetPropertyM(), SetPropertyM(), GetPropertyI(), SetPropertyI(), GetPropertyD(), SetPropertyD()

Enabling or Disabling a Control for User Input

Use the `.Enabled` property to determine whether the control object can respond to operator-generated events.

.Enabled Dotfield

Determines whether the control object can respond to user-generated events.

Category

windows control

Usage

```
[ErrorNumber=] GetPropertyD("ControlName.Enabled",  
    Tagname);  
  
[ErrorNumber=] SetPropertyD("ControlName.Enabled",  
    Discrete);
```

Parameters

ControlName

Name of the windows control. For example, `ChkBox_4`.

Tagname

A discrete tagname that holds the property requested.

Discrete

A discrete value or a discrete tagname that holds the value to be written when the function is processed. For a discrete value:

0 = Control is disabled.

1 = Control is enabled.

Remarks

This property is read/write during both development and run time.

Data Type

Discrete (read/write)

Applies To

Text boxes, list boxes, combo boxes, check boxes and radio buttons.

Example(s)

The following statement disables the list box object named "ListBox_1".

```
SetPropertyD("ListBox_1.Enabled", 0);
```

See Also

GetPropertyD(), SetPropertyD()

Hiding Windows Controls Dynamically

Use the `.Visible` property to determine whether the windows control is visible in the window.

.Visible Dotfield

Determines whether the windows control is visible in the window.

Category

windows control

Usage

```
[ErrorNumber=]GetPropertyD("ControlName.Visible",  
    Tagname);  
[ErrorNumber=]SetPropertyD("ControlName.Visible",  
    Number);
```

Parameters

ControlName

Name of the windows control. For example, `ListBox_1`.

Tagname

A tagname (of the same type to be returned) that holds the property value when the function is processed.

Number

A discrete value or a discrete tagname that holds the value to be written when the function is processed. For a discrete value:

0 = Control is invisible.

1 = Control is visible.

Remarks

This property is read/write in both development and run time.

Data Type

Discrete (read/write)

Valid Values

Applies To

Text boxes, list boxes, combo boxes, check boxes and radio buttons.

Example(s)

The following statement hides the text box named "TextBox_1".

```
SetPropertyD("TextBox_1.Visible", 0);
```

See Also

GetPropertyD(), SetPropertyD()

Adding and Deleting Items in Combo Boxes

Use the following script functions to add and delete items from combo boxes and lists

Script function	Effect
<code>wcAddItem()</code>	Adds an item to the end of the list of a list box or combo box. If sorting is enabled, the list is sorted after the item is added.
<code>wcInsertItem()</code>	Adds an item at a specified position in the list of a list box or combo box.
<code>wcDeleteItem()</code>	Deletes an item from a specified position in the list of a list box or combo box.
<code>wcDeleteSelection()</code>	Deletes the currently selected item from the list or combo box.
<code>wcClear()</code>	Removes all items from the list or combo box.

`wcAddItem()` Function

Adds an item to the end of the list of a list box or combo box. If sorting is enabled, the list is sorted after the item is added.

Category

windows control

Syntax

```
[ErrorNumber=]wcAddItem("ControlName", "MessageTag");
```

Parameters

ControlName

The name of the windows control object. For example, `ListBox_1`. Actual string or message tagname.

MessageTag

The message string to be shown. Actual string or message tagname.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Applies To

List boxes and combo boxes.

Example(s)

The following statement adds the contents of the message string to the list box when the window (using On Show Window QuickScript) containing the list box is opened:

```
wcAddItem("ListBox_1", "Chocolate");  
wcAddItem("ListBox_1", "Vanilla");  
wcAddItem("ListBox_1", "Strawberry");
```

See Also

wcInsertItem()

wcInsertItem() Function

Inserts the specified string into the list of a list box or combo box at the specified position. Unlike the wcAddItem() function, the wcInsertItem() function does not sort a list, even if it is created as a sorted list box or combo box.

Category

windows control

Syntax

```
[ErrorNumber=]wcInsertItem("ControlName", ItemPosition,  
    "Message");
```

Parameters*ControlName*

The name of the windows control object. For example, ListBox_1. Actual string or message tagname.

ItemPosition

A number corresponding to the position of the item to be added. If this parameter is -1, the string is added to the end of the list. Any number or Integer tagname.

Message

Contains the string to insert at the position indicated by ItemPosition. Actual string or message tagname.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Applies To

List boxes and combo boxes.

Example(s)

The following statement inserts a new item called "Blueberry" into a list box at fourth position from the top when an action script runs.

```
wcInsertItem("ListBox_1", 4, "Blueberry");
```

See Also

wcAddItem()

wcDeleteItem() Function

Deletes the item at a specified position from the list of either a list box or combo box.

Category

windows control

Syntax

```
[ErrorNumber=]wcDeleteItem("ControlName",  
    ItemPosition);
```

Parameters

ControlName

The name of the windows control object. For example, ListBox_1. Actual string or message tagname.

ItemPosition

A number corresponding to the position of the item. Any number or Integer tagname.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Applies To

List boxes and combo boxes.

Example(s)

The following statement deletes the third item in a list when an action script runs:

```
wcDeleteItem("ListBox_1", 3);
```

wcDeleteSelection() Function

Deletes the currently selected item from the list.

Category

windows control

Syntax

```
[ErrorNumber =]wcDeleteSelection("ControlName");
```

Parameter

ControlName

The name of the windows control object. For example, ListBox_1. Actual string or message tagname.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Applies To

List boxes and combo boxes.

Example(s)

The following statement deletes the currently selected item in a list box list when an action script runs:

```
wcDeleteSelection("ListBox_1");
```

wcClear() Function

Removes all items from the list box or combo box.

Category

windows control

Syntax

```
[ErrorNumber=]wcClear("ControlName");
```

Parameters*ControlName*

The name of the windows control object. For example, ListBox_1. Actual string or message tagname.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Applies To

List boxes and combo boxes.

Example(s)

The following statement clears all items in a list box when an action script runs:

```
wcClear("ListBox_1");
```

Loading and Saving List Items From or To a File

Use the following script functions to load and or save items in a combo box or list from or to a file.

Script function	Effect
wcLoadList()	Loads the contents of a list box or combo box with new items from a file.
wcSaveList()	Saves the contents of a list box or combo box to a file.

wcLoadList() Function

Loads a list box or combo box with new items from a file.

Category

windows control

Syntax

```
[ErrorNumber=]wcLoadList("ControlName", "Filename");
```

Parameters

ControlName

The name of the windows control object. For example, ListBox_1. Actual string or message tagname.

Filename

Contains the name of a file. If you do not supply a complete path name as part of the message parameter, the function checks the application directory for the message file. Actual string or message tagname.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

If you use external files to fill the list and combo boxes, they must follow specific formatting and contain specific information. Format:

ControlType, ListCount

ListItem, ItemIndex

ListItem, ItemIndex

: :

: :

ListItem, ItemIndex

The ControlType is either COMBOBOX or LISTBOX.

For example, you want to load a list file to a combo box and it contains three items to select from and those items have no item data assigned. The format of the file appears as:

COMBOBOX, 3

Chocolate, 0

Vanilla, 0

Strawberry, 0

The COMBOBOX is the control type. The list count is 3 for Chocolate, Vanilla, and Strawberry. Chocolate is then listed as the first item or position 1. Vanilla as position 2, and Strawberry as position 3. Each of these items has a data value of 0.

For more information on item data, see `wcSetItemData()` Function on page 135.

Applies To

List boxes and combo boxes.

Example(s)

The following statement loads a properly formatted list (located in `c:\wclist.txt`.) into a combo box:

```
wcLoadList("Combobox_1", "c:\wclist.txt");
```

See Also

`wcAddItem()`, `wcSaveList()`

wcSaveList() Function

Saves the items of a list box or combo box to a file.

Category

windows control

Syntax

```
[ErrorNumber=]wcSaveList("ControlName", "Filename");
```

Parameters*ControlName*

The name of the windows control object. For example, `ListBox_1`. Actual string or message tagname.

Filename

Contains the name of a file. If the file does not exist, it is created. Actual string or message tagname.

Remarks

For a list of returned error numbers, see [Understanding Windows Controls Error Messages](#) on page 141.

Applies To

List boxes and combo boxes.

Example(s)

The following statement saves the current items in a list box in a file (`c:\newlist.txt`) when an action script runs:

```
wcSaveList("ListBox_1", "c:\newlist.txt");
```

See Also

`wcLoadList()`, `wcSetItemData()`

Finding Items In a Combo Box or List

Use the `wcFindItem()` function to search for a specified item in a list box or combo box. If the item is found, this function returns the corresponding position to an integer tagname as the fourth parameter.

`wcFindItem()` Function

Determines the corresponding position of the first item in the list box or combo box that matches the supplied message string.

Category

windows control

Syntax

```
[ErrorNumber=]wcFindItem ("ControlName", "MessageTag",  
    CaseSens, Tagname);
```

Parameters

ControlName

The name of the windows control object. For example, `ListBox_1`. Actual string or message tagname.

MessageTag

The message string to be compared. Actual string or message tagname.

CaseSens

Determines the type of the string comparison. It can either be a discrete value or tagname. The following values are valid:

0 = case-insensitive.

1 = case-sensitive.

Tagname

Integer tag into which the position of the matching item is returned. If no matching item is found, -1 is returned.

Remarks

For a list of returned error numbers, see [Understanding Windows Controls Error Messages](#) on page 141.

Applies To

List boxes, combo boxes

Example(s)

Assuming that `ListBox_1` is a list box that contains “ItemA”, “ItemB”, and “ItemC”, the function returns the following values into `Result`:

```
wcFindItem("ListBox_1", "ItemB", 0, Result);
```

returns 2

```
wcFindItem("ListBox_1", "Itemb", 1, Result);
```

returns -1

```
wcFindItem("ListBox_1", "itemc", 0, Result);
```

returns 3

```
wcFindItem("ListBox_1", "XYZ", 0, Result);
```

returns -1

Working with Item Indexes in a Combo Box or List

Use the following dotfields to work with the item index of a list box or combo box.

Dot Field	Effect
<code>.TopIndex</code>	The integer index of the topmost item in the list box.
<code>.NewIndex</code>	The integer index (tagname) of the last item added to the list box or combo box through the <code>wcAddItem()</code> or <code>wcInsertItem()</code> functions.
<code>.ListIndex</code>	The index (tagname or number) of the currently selected item in the list.

.TopIndex Dotfield

Sets or reads the integer index of the top-most item in a list box.

Category

windows control

Usage

```
[ErrorNumber=]GetPropertyI ("ControlName.TopIndex",  
    Tagname);
```

```
[ErrorNumber=]SetPropertyI ("ControlName.TopIndex",  
    Number);
```

Parameters*ControlName*

Name of the windows control. For example, *ListBox_1*.

Tagname

An Integer tagname that holds the property value when the function is processed.

Number

The index number that defines the top-most item in the list box. Can be a literal integer value or a integer tagname or expression that provides an integer value.

Remarks

This property is available only in run time.

Data Type

Integer (read/write)

Applies To

List boxes.

Example(s)

The following statement sets the `TopIndex` of the list box object "ListBox_1" to a value of 14:

```
setPropertyI ("ListBox_1.TopIndex", 14);
```

See Also

`GetPropertyI()`, `setPropertyI()`, `.ListIndex`, `.NewIndex`

.NewIndex Dotfield

Returns the integer index (`Tagname`) of the last item added to the list box or combo box via the `wcAddItem()` or `wcInsertItem()`.

Category

windows control

Usage

```
[ErrorNumber=]GetPropertyI ("ControlName.NewIndex",  
    Tagname);
```

Parameters*ControlName*

Name of the windows control. For example, *ListBox_4*.

Tagname

A tagname containing the integer index of the last item added to the list or combo box. For empty lists, a value of -1 is returned.

Remarks

This property is only available in run time.

Data Type

Integer (read-only)

Applies To

List boxes and combo boxes.

Example

The following statement retrieves the index of the most recently added item in the list box named "ListBox_1" and writes that value to the memory integer tagname `NewItemIndex`.

```
GetPropertyI("ListBox_1.NewIndex", NewItemIndex);
```

See Also

`GetPropertyI()`, `wcAddItem()`, `wcInsertItem()`, `.ListIndex`, `.TopIndex`

.ListIndex Dotfield

Sets or reads the index (*Tagname* or *Number*) of the currently selected item in the list.

When using a list box, an index of -1 indicates that no item is currently selected.

When using a combo box, an index of -1 indicates that the user has typed new text into the text entry field of the control.

Syntax

```
[ErrorNumber=]GetPropertyI("ControlName.ListIndex",  
    Tagname);
```

```
[ErrorNumber=]SetPropertyI("ControlName.ListIndex",  
    Number);
```

Parameter

ControlName

Name of the windows control. For example, `ListBox_4`.

Tagname

The tagname to which the index of the currently selected item is written.

Number

The index number that defines a specific item in the list.

Remarks

The index number defines a specific item in a list. Use the `.ListIndex` dotfield to set or determine the index of the currently selected item in a list or combo box.

This property is available only in run time.

Data type

Integer (read or write)

Applies to

List boxes and combo boxes.

Example

This statement retrieves the index of the currently selected item in the list box named "ListBox_1" and writes that value to the memory integer tagname MyListBoxIndex.

```
GetPropertyI ( "ListBox_1.ListIndex",
    MyListBoxIndex );
```

See Also

GetPropertyI(), SetPropertyI(), .NewIndex, .TopIndex

Counting List Box or Combo Box Items

The .ListCount dotfield contains the number of items in a list box or combo box.

.ListCount Dotfield

Reads the number of items in the list box or combo box.

Category

Windows control

Syntax

```
[ErrorNumber=]GetPropertyI ("ControlName.ListCount",
    Tagname);
```

Parameter

ControlName

Name of the windows control.

Tagname

A valid tagname that contains the integer count of the items in the list.

Remarks

This property is available only in runtime.

Data Type

Integer (read-only)

Applies To

List boxes and combo boxes.

Example(s)

The following statement retrieves the number of items in the list box named *ListBox_1* and writes that value to the memory integer tag *MyListBoxCount*.

```
GetPropertyI ("ListBox_1.ListCount", MyListBoxCount);
```

See Also

GetPropertyI(), .ListIndex

Getting or Setting the Value of a List Item

Use the `wcGetItemData()` function to find the integer value associated with the list item identified by the item index.

Use the `wcSetItemData()` function to assign an integer value to the item in the list specified by item index. This assigns a number to a string.

wcGetItemData() Function

Reads the integer value associated with the list item identified by the `ItemIndex` parameter.

Category

windows control

Syntax

```
[ErrorNumber=]wcGetItemData("ControlName", ItemIndex,  
    Tagname);
```

Parameters

ControlName

The name of the windows control object. For example, `ListBox_1`. Actual string or message tagname.

ItemIndex

A number corresponding to the position of the item. Any number or integer tagname.

Tagname

Actual name of a real or integer tagname. The `wcGetItemData()` function places the numeric value corresponding to the item into this tagname upon return from the function.

Remarks

For a list of returned error numbers, see [Understanding Windows Controls Error Messages](#) on page 141.

Applies To

List boxes and combo boxes.

Example(s)

The following statement retrieves the numeric value associated with the fifth item in a list box and returns it to the `ItemValue Integer` tagname when an action script runs:

```
wcGetItemData("ListBox_1", 5, ItemValue);
```

If the fifth item in the list is assigned the integer value 4500, the `ItemValue` tagname contains 4500.

See Also

`wcSetItemData()`

wcSetItemData() Function

Assigns an integer value of the item (the Number parameter) to the item in the list specified by the ItemIndex parameter. This function allows the assignment of a number to a string.

Category

windows control

Syntax

```
[ErrorNumber=]wcSetItemData("ControlName", ItemIndex,  
    Number);
```

Parameters

ControlName

The name of the windows control object. For example, ListBox_1. Actual string or message tagname.

ItemIndex

An integer value specifying the list item you want to edit. Any number or Integer tagname.

Number

An integer value representing the item data. Any number or Integer tagname.

Remarks

You can create complete lists containing the items using a program like Notepad and then load them using one function call. Format the list as required by the wcSaveList() function.

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Use the wcGetItemData() function to return the value (item data) associated with the list item. The tagname parameter contains the returned numeric value. This parameter could be an I/O Integer tagname that writes directly to the real world device.

Example(s)

A recipe has three ingredients; flour, sugar and salt. The quantity of flour is 4500 grams, sugar is 1500 and salt is 325 grams. The values are assigned to each of the list box items by using a data change script triggered by what recipe (tagname, RecipeName) is selected:

```
wcSetItemData("ListBox_1", 1, 4500); {set 1st item in  
    the list (flour)=4500}  
wcSetItemData("ListBox_1", 2, 1500); {set 2nd item in  
    the list (sugar)=1500}  
wcSetItemData("ListBox_1", 3, 325); {set 3rd item in  
    the list (salt)=325}
```

See Also

wcLoadList(), wcSaveList(), wcGetItemData()

Getting the Name of a List Item

Use the `wcGetItem()` function to return the item string associated with the corresponding item index in the list box or combo box.

`wcGetItem()` Function

Returns a string containing the contents of the item corresponding to the `ItemIndex` in the list box or combo box.

Category

windows control

Syntax

```
[ErrorNumber=]wcGetItem("ControlName", ItemIndex,  
    Tagname);
```

Parameters

ControlName

The name of the windows control object. For example, `ListBox_1`. Actual string or message tagname.

ItemIndex

A number corresponding to the position of the item. Any number or Integer tagname.

Tagname

Message tagname. The `wcGetItem` function will place the data corresponding to the item index into this tagname upon return from the function.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Applies To

List boxes and combo boxes.

Example(s)

The following statement returns the string value of the tenth item in a combo box to the `ListSelection` message tag when an action script runs:

```
wcGetItem("Combobox_1", 10, ListSelection);
```

If item ten in the list is "Vanilla," then the `ListSelection` tag contains the string "Vanilla".

Loading the Contents of a Text Box

Use the `wcLoadText()` function to load the contents of a text box from a file. Use the `wcSaveText()` function to save the contents of a text box to a text file.

Note If the tag name has been defined with a maximum length, only that number of characters can be assigned from the text box contents to the tag. If no tag is assigned to the text box, its contents can be up to 65,535 characters.

`wcLoadText()` Function

Replaces the contents of the text box with the contents of the file.

Category

windows control

Syntax

```
[ErrorNumber=]wcLoadText ("ControlName", "Filename");
```

Parameters

ControlName

The name of the windows control object. For example, `ListBox_1`. Actual string or message tagname.

Filename

Contains the name of a file. If a complete path name is not supplied as part of the message parameter, the function will check the application directory for the file. Actual string or message tagname.

Remarks

For a list of returned error numbers, see [Understanding Windows Controls Error Messages](#) on page 141.

Applies To

Text boxes.

Example(s)

The following statement loads a text file (`c:\InTouch.32\readme.txt`) into a text box when the window (On Show Window script) containing the text box opens:

```
wcLoadText ("Textbox_1", "c:\InTouch.32\readme.txt");
```

wcSaveText() Function

Saves the text contained in the text box to the specified file. If the file doesn't exist, it is created. If it does exist, it must be read/write.

Category

windows control

Syntax

```
[ErrorNumber=]wcSaveText("ControlName", "Filename");
```

Parameters

ControlName

The name of the windows control object. For example, ListBox_1. Actual string or message tagname.

Filename

Contains the name of the destination file. If you do not supply a complete path name, the file is saved in the application directory. If the file exists, it is overwritten. If the file does not exist, it is created. The resulting file can subsequently be loaded into a text box object using the wcLoadText() function. Actual string or message tagname.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Applies To

Text boxes.

Example(s)

The following statement saves the current information entered in a text box to a file in c:\InTouch.32\newtext.txt when an action script runs:

```
wcSaveText("Textbox_1", "c:\InTouch.32\newtext.txt");
```

See Also

wcLoadText()

Checking If a Text Box is Read-Only

Use the `.ReadOnly` dotfield to determine whether the content of the text box is read-only or read/write.

.ReadOnly Dotfield

Determines whether the content of the text box is read-only or read/write.

Category

windows control

Usage

```
[ErrorNumber=]GetPropertyD("ControlName.ReadOnly",  
    Tagname);
```

Parameters

ControlName

Name of the windows control. For example, *Textbox_1*.

Tagname

A discrete tagname that holds the property value when the function is processed.

0 = Contents of the text box is read/write

1 = Contents of the text box is read-only

Remarks

This property is available in both development and run time.

Data Type

Discrete (read-only)

Applies To

Text boxes.

Example(s)

The following statement retrieves the read-only status of the Text box named "TextBox_1":

```
GetPropertyD("TextBox_1.ReadOnly", A_Tagname);
```

See Also

GetPropertyD(), SetPropertyD()

Getting or Setting the Label of a Check Box

The `.Caption` dotfield defines the message text of a check box.

.Caption Dotfield

Determines the message to be displayed with the check box.

Category

windows control

Usage

```
[ErrorNumber=]GetPropertyM ("ControlName.Caption",  
    Tagname);  
  
[ErrorNumber=]SetPropertyM ("ControlName.Caption",  
    "Message");
```

Parameters

ControlName

Name of the windows control. For example, *ChkBox_4*.

Tagname

A message tagname that holds the property requested.

Message

A message string surrounded in quotes.

Remarks

This property is read/write during both development and run time.

Data Type

Message (read/write)

Applies To

Check boxes.

Example

This statement sets the caption of the check box object "CheckBox_1" to "Blue Paint Option."

```
SetPropertyM("CheckBox_1.Caption","Blue Paint Option");
```

See Also

GetPropertyM(), SetPropertyM()

Understanding Windows Controls Error Messages

Given an error number, `wcErrorMessage()`, returns a string message describing the error. It applies to list boxes, text boxes, combo boxes, radio buttons and check boxes.

The Window Controls functions return values based on the result of processing QuickScript functions. The return value is used for error diagnostics. You can assign these values to integer tag names. For example:

```
ErrorNumber = wcGetItem("ControlName", Number,
    Tagname);
```

In this script, `ErrorNumber` is an integer tag that contains the returned error value. The returned value of the function can be passed to the `wcErrorMessage()`. The `wcErrorMessage()` will return a string description of the error. For example:

```
ErrorMsg = wcErrorMessage(ErrorNumber);
```

In this script, `ErrorMsg` is a message type tag that contains the text of the returned error. The following table identifies numeric error values and their definitions.

Error Message	Definition
0	Success
-1	General failure
-2	Insufficient memory available
-3	Property is read-only
-4	Specified item already present
-5	Object name unknown
-6	Property name unknown
-x	Unknown error.

wcErrorMessage() Function

Returns a message string describing the error.

Category

windows control

Syntax

```
ErrorMessage=wcErrorMessage (ErrorNumber) ;
```

Parameters

ErrorMessage

Message tagname.

ErrorNumber

Number returned by all windows control functions. Any number or Integer tagname.

Remarks

For a list of returned error numbers, see Understanding Windows Controls Error Messages on page 141.

Applies To

List boxes, text boxes, combo boxes, check boxes and radio buttons.

Example(s)

If an error occurs while a list is being loaded, show the text description of the error into the ErrorDescription message tagname. In this example, a Value Display animation link is assigned to the ErrorDescription tagname to show the error message.

In the “On Show” window QuickScript:

```
ErrorNumber=wcLoadList ("ListBox_1", "c:\recipe.txt");  
ErrorDescription=wcErrorMessage (errornumber) ;
```

You can use this function with all windows control functions to show error messages:

```
ErrorNumber=wcAddItem ("ListBox_1", "AM_4A4356");  
ErrorMsg=wcErrorMessage (ErrorNumber) ;
```

Chapter 6

ActiveX Controls

ActiveX controls are stand-alone software components that bring additional functionality to InTouch applications.

You can use several types of ActiveX controls in your InTouch application.

- The InTouch HMI includes ActiveX controls for alarming.
- Other Wonderware products, such as ActiveFactory, provide controls for manipulating and analyzing data.
- You can use third-party ActiveX controls.
- You can develop your own ActiveX controls in Visual Basic or C.

You can use any number of ActiveX controls in your InTouch application. You can:

- Select and paste an ActiveX control into any application window.
- Adjust the size of the control, if sizing is supported by the control.
- Duplicate, cut, copy, paste and delete ActiveX controls.
- Align ActiveX controls: left, right, top, bottom, and center point.
- Add ActiveX controls to the Wizards/ActiveX Toolbar.
- Combine ActiveX controls with other objects when creating a cell.
- Use the properties, methods, and events supported by the particular ActiveX control properties.

InTouch applications do not support the following types of ActiveX controls:

- Windowless controls
- Simple frame site or area box
- Containers
- Data controls
- Dispatch objects

InTouch applications only support these data types: discrete (Boolean), integer (32-bit numbers), real (floating point IEEE notation with 32 bits), and message (strings up to 131 characters). Unsupported data types include variant, pointers, arrays, structures, and parameterized properties.

ActiveX controls cannot overlap other InTouch objects, such as window controls or graphic objects. Too many ActiveX controls on one window can reduce system performance.

Using ActiveX Controls

You select an ActiveX control and paste it into a window and then double-click it. Its configuration dialog box appears.

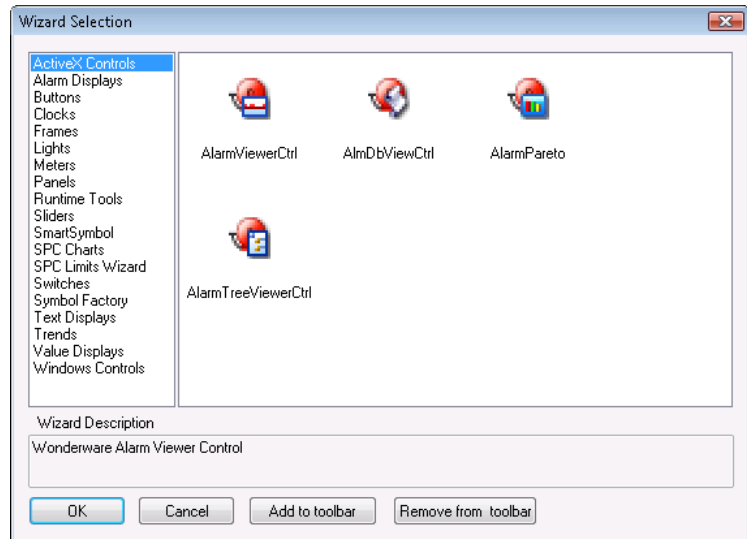
When you configure an ActiveX control, you can give it a unique control name. The control name can then be referenced in scripts.

To use an ActiveX control in an InTouch application

- 1 Install the ActiveX control. See *Installing and Removing ActiveX Controls* on page 148.
- 2 Select and paste the ActiveX control into an InTouch window.
- 3 Configure the ActiveX control properties and bind them to tags.
- 4 Associate ActiveX events to ActiveX Event scripts.
- 5 Call ActiveX methods and set ActiveX control properties in ActiveX Event scripts, or other InTouch QuickScripts.

To place an ActiveX control in a window

- 1 Click the Wizard Dialog tool in the **Wizards/ActiveX Toolbar**. The **Wizard Selection** dialog box appears.



- 2 In the list of wizards, click the **ActiveX Controls** category. All available ActiveX controls appear in the display area.
- 3 Double-click the ActiveX control to use. The dialog box closes and the cursor changes to the corner symbol.
- 4 Click the location to paste the ActiveX control.

To add ActiveX controls to the toolbar

- 1 Click the **Wizard Dialog** tool in the **Wizards/ActiveX Toolbar**. The **Wizard Selection** dialog box appears.
- 2 Select the ActiveX control to add.
- 3 Click **Add to toolbar**.

To remove ActiveX controls from the toolbar

- 1 Click the **Wizard Dialog** tool in the **Wizards/ActiveX Toolbar**. The **Wizard Selection** dialog box appears.
- 2 Click **Remove from toolbar**. The **Remove Wizard from Toolbar** dialog box appears.
- 3 Select the ActiveX control to remove.
- 4 Click **OK**.

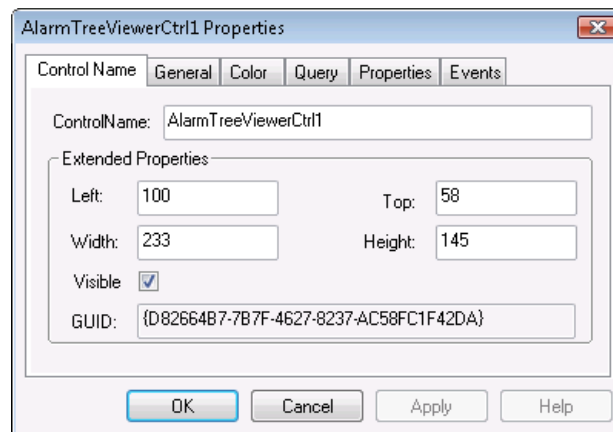
Configuring ActiveX Controls

The properties that you can configure for a particular ActiveX control are determined by the ActiveX control itself. All properties have a default value.

You must paste an ActiveX control into an InTouch window before you can configure its properties.

- A default control name, such as Calendar1, is generated when you paste the ActiveX control in a window. You reference the control name in scripts. An ActiveX control must be running in an open application window for any script to run against it.
- You can assign the ActiveX control properties to InTouch tags. You must assign each property type to a corresponding InTouch tag type.

ActiveX controls have three standard tabs: **Control Name**, **Properties**, and **Events**.



Use the **Events** tab to assign scripts to available control events, such as when the user double-clicks the mouse.

Any other tabs and their configurations are unique to the control and depend upon its properties. For example, some controls may require you to configure colors and fonts, while others may not have these properties.

Naming ActiveX Controls

A new instance of a control is created and given a unique name when you:

- Select **Duplicate** on the **Edit** menu.
- Select **Cut** or **Copy** and then **Paste** on the **Edit** menu.
- Select **Save Window As** on the **File** menu.
- Click **Undo** and then **Redo**.
- Import a window that contains a control.

ActiveX controls must have unique names. If you try to use a name that already exists for a control, an error message appears.

To name an ActiveX control

- 1 Paste the ActiveX control into your development window.
- 2 Double-click the control. The control **Properties** dialog box appears.
- 3 Click the **Control Name** tab and then type a name for the ActiveX control in the **ControlName** box.

Standard Operations on ActiveX Controls

You can perform standard operations on ActiveX controls just like any other InTouch object. For more information, see [Special Manipulations for All Objects](#) on page 51.

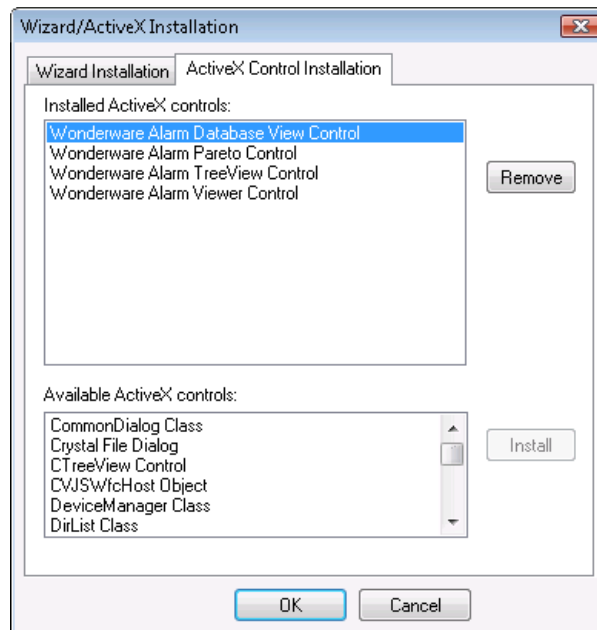
Installing and Removing ActiveX Controls

Even though you have already installed an ActiveX control on your computer, you must also make it known to the InTouch HMI by installing it to WindowMaker.

When you remove a control from WindowMaker, it is not deleted from your computer. It is simply removed from memory and does not function.

To install an ActiveX control

- 1 On the **Special** menu, point to **Configure** and then click **Wizard/ActiveX Installation**. The **Wizard/ActiveX Installation** dialog box appears.
- 2 Click the **ActiveX Control Installation** tab. The **ActiveX Control Installation** property sheet appears.



- 3 In the **Installed ActiveX controls** list, select the control to install in the **Available ActiveX controls** list and then click **Install**.

Tip To select multiple controls, use the **SHIFT** key or **CTRL** key.

- 4 Click **OK**.

To remove an ActiveX control

- 1 On the **Special** menu, point to **Configure** and then click **Wizard/ActiveX Installation**. The **Wizard/ActiveX Installation** dialog box appears.
- 2 Click the **ActiveX Control Installation** tab. The **ActiveX Control Installation** property sheet appears.
- 3 In the **Installed ActiveX controls** list, select the control to remove from your application and then click **Remove**. A message appears asking you to confirm the deletion.

Tip To select multiple controls, use the **SHIFT** key or **CTRL** key.

- 4 Click **Yes** to remove the control. The control is moved to the **Available ActiveX controls** list.
- 5 Click **OK**.

Index

Symbols

\$ObjHor system tag 80, 82

\$ObjVer system tag 80, 82

A

action script link, creating 90

ActiveX controls

about 143

adding to toolbar 145

configuring 146

definition 39

installing and uninstalling 148

naming 147

placing on a window 145

removing from toolbar 145

standard operations on 147

using 144

alarm types 72

aligning objects 44

analog data display, creating 62

analog input, creating links 85

anchor points 67, 68

animating

colors 69

fill levels 73

sizes 66

animation links, types of 60

animation tasks 97

animations

data display 60

data entry 83

application windows, creating 30

ArchestrA Symbol, definition 38

B

bitmap containers

definition 38

working with 55

bitmaps

creating a transparent bitmap 56

defining transparency 56

importing 55

pasting, editing 55

blink attributes 75

blink frequency, setting 76

blinking objects 75

bring to front, object 45

buttons

creating 37

default fonts 13

substitute strings 37

C

- .Caption dotfield 140
- cells
 - creating 40
 - definition 38
 - identifying 39
 - use of 39
- check boxes
 - creating 116
 - getting or setting the label 140
- circle button 36
- closing windows 91
- color links
 - creating analog alarm status link 72
 - creating analog expression link 70
 - creating discrete alarm status link 71
 - creating discrete fill link 70
 - defining 69
 - expression types 69
- color palettes
 - custom 22
 - importing and exporting 23
 - opening 21
 - using 21
- colors, animating 69
- combo boxes
 - adding and deleting items 123
 - creating 115
 - finding items 129
 - working with item indexes 130
- controls
 - enabling or disabling user input 121
 - getting and setting the value 119

D

- data display
 - analog value, creating 62
 - animations 60
 - discrete value, creating 61
 - string value, creating 62
 - values 60
- data entry
 - analog input 85
 - discrete 84
 - string input 86
- data entry animations 83
- dialog boxes, features 25

- DialogStringEntry() function 93
- DialogValueEntry() function 94
- disable link 77
- discrete input, creating links 84
- discrete value display, creating 61
- display links 60
- documentation conventions 7
- dotfields
 - .Caption dotfield 140
 - .Enabled dotfield 121
 - .NewIndex dotfield 131
 - .ReadOnly dotfield 139
 - .TopIndex dotfield 130
 - .Value dotfield 119
 - .Visible dotfield 111, 122
- double, frame style 31
- drawing buttons 36
- drop-down lists 111

E

- ellipse button 36
- .Enabled dotfield 121
- error messages
 - about 141
 - definitions 141

F

- fast switch 12
- fill levels
 - animating 73
 - percent 73
- fill, changing 50
- flipping objects or cells 46
- fonts
 - changing 48
 - configure 13
 - setting defaults 24
- frame style 31
- functions
 - DialogStringEntry() function 93
 - DialogValueEntry() function 94
 - ShowAt() function 80
 - ShowTopLeftAt() function 80
 - wcAddItem() function 123
 - wcClear() function 126
 - wcDeleteItem() function 125
 - wcDeleteSelection() function 125
 - wcErrorMessage() function 142

- wcFindItem() function 129
 - wcGetItem() function 136
 - wcGetItemData() function 134
 - wcInsertItem() function 124
 - wcLoadList() function 127
 - wcLoadText() function 137
 - wcSaveList() function 128
 - wcSaveText() function 138
 - wcSetItemData() function 135
- G**
- grid
 - show 11, 14
 - snap to 14
 - spacing 11, 14
- H**
- home, windows 32
 - horizontal movement 63
 - horizontal line button 36
 - horizontal spacing of objects 45
- I**
- importing bitmaps 55
- K**
- keyboard shortcuts 102
 - keyboards, configuring on-screen keyboards 92
- L**
- layering objects 45
 - line button 36
 - lines and outlines, changing 49
 - lines and shapes, creating 36
 - lines, select precision 13
 - links
 - action script 90
 - analog alarm status color, creating 72
 - analog expression color, creating 70
 - analog input 85
 - animate objects 59
 - animation 26, 39
 - animation types 60
 - blink 75
 - cannot make symbol 40
 - clipboard limitations 52
 - color 69
 - color link types 69
 - creating movement 63
 - creating rotation 65
 - cut, copy, or paste 52
 - disabling 77
 - discrete alarm status, creating 71
 - discrete color fill, creating 70
 - discrete input 84
 - discrete value touch links 89
 - display 60
 - keyboard shortcuts 102
 - location links 63
 - on screen keyboards 92
 - opening or closing windows 91
 - orientation 65
 - percent fill 73
 - push buttons 89
 - size links 66
 - slider, creating 87
 - string input 86
 - symbol in a cell 40
 - ToolTips 78
 - touch 60, 83
 - user input types 83
 - value display types 60
 - visibility 76
 - list boxes
 - creating 114
 - lists
 - finding items 129
 - getting and setting a list item value 134
 - getting the name of a list item 136
 - loading from a file 126
 - saving to a file 126
 - working with item indexes 130
- M**
- mouse short cuts 26
 - movement
 - creating 63
 - horizontal 63
 - vertical 64
- N**
- .NewIndex dotfield 131
 - none, frame style 31

O

- object links, cutting, copying, and pasting 52
- objects
 - alarm types 72
 - aligning 44
 - blinking 75
 - complex 38
 - controlling horizontal and vertical spacing 45
 - cutting, copying, and pasting 52
 - deleting 50
 - disabling 77
 - duplicating 53
 - flipping 46
 - grouping to cells 40
 - grouping to symbols 40
 - horizontal movement 63
 - layering 45
 - moving 42, 63
 - moving with arrow keys 27
 - pick through hollow objects 12
 - reshaping polyline and polygon 54
 - resizing 47
 - rotating 47, 65
 - selecting 41
 - simple 36
 - size height link 67
 - size width link 68
 - snap to grid 14
 - special manipulations 51
 - substituting text 57
 - trend 110
 - vertical movement 64
- opening windows 91
- orientation link, creating 65
- overlay, windows type 31

P

- palettes, color 21
- pan and zoom
 - limitations 17
 - show or hide toolbar 15
- pasting wizard instances 107
- percent fill 73

- polygons, adding or deleting points 54
- polylines, creating 37
- popup, windows type 31
- pushbuttons, creating 89

R

- radio button group control 117
- radio buttons, creating 117
- .ReadOnly dotfield 139
- rectangle button 36
- rectangle, changing the radius of 57
- replace, window type 31
- resizing objects 47
- rotating objects 47
- rotation 65
- rounded rectangle button 36
- ruler, show or hide 15
- runtime 12

S

- screen mode, full 24
- script functions for windows controls 119
- select tag dialog box 101
- send to back, objects 45
- shapes, creating 36
- show hide windows 91
- ShowAt() function 80
- ShowTopLeftAt() function 80
- simple objects 36
- single, frame style 31
- size animating 66
- size controls 31
- size height link 67
- size width link 68
- sliders, creating 87
- SmartSymbol, definition 38
- snap to grid 14
- spacing of objects, horizontal and vertical 45
- string input, creating links 86
- string value display, creating 62
- substitute strings
 - creating buttons 37
 - text objects 57
- substituting tags 103

symbols

- combining and breaking 40
- creating or breaking 40
- definition 38
- identifying 39
- use of 40

system tags

- \$ObjHor 80, 82
- \$ObjVer 80, 82

T

tag filters 100

tagname references, changing 103

tags

- converting placeholder 104
- enable logging 110
- finding and selecting 97
- show tag count 12
- substituting 103

technical support, contacting 8

text

- changing fonts 48
- creating 37
- default fonts 13

text boxes

- checking if read-only 139
- creating 113
- loading or saving the contents of 137

thumbnails 16

tool tips

- configuring 78
- creating 79
- static or expression 78

toolbars

- changing size 18
- show or hide 18

.TopIndex dotfield 130

touch links 60, 83

trend objects 39, 110

U

undo, levels of 13

update use counts 12

user input, enabling or disabling 121

V

value displays, creating 60

.Value dotfield 119

vertical line button 36

vertical movement 64

vertical spacing of objects 45

visibility link, creating 76

.Visible dotfield 111, 122

W

wcAddItem() function 123

wcClear() function 126

wcDeleteItem() function 125

wcDeleteSelection() function 125

wcErrorMessage() function 142

wcFindItem() function 129

wcGetItem() function 136

wcGetItemData() function 134

wcInsertItem() function 124

wcLoadList() function 127

wcLoadText() function 137

wcSaveList() function 128

wcSaveText() function 138

wcSetItemData() function 135

WindowMaker 9

- about 28
- close on transfer 12
- fast switch 12
- levels of undo 13
- preferences 10
- title bar changing 11

windows

- closing 33, 91
- creating 30
- deleting 34
- dimensions 31
- duplicating 34
- exporting 32
- frame style 31
- height 31
- home 32
- importing 32
- modifying application 32
- name 30

- opening 33, 91
- opening at selected object 80
- positioning a touch-sensitive window 80
- saving 33
- selecting background color 30
- setting to appear at run time 32
- setting type and appearance 32
- show hide 91
- size controls 31
- type 31
- undoing changes 51
- width 31
- windows control wizards
 - about 111
 - configuring 112
 - creating 112
 - hiding 122
- windows controls, script functions 119
- windows type, replace, overlay, or popup 31
- Windows XP 30, 80

- wizards
 - adding to the toolbar 107
 - configuring 108
 - configuring windows controls 112
 - creating windows controls 112
 - definition 39
 - importing 109
 - installing and uninstalling 108
 - pasting instances 107
 - removing from the toolbar 107
 - types of 106
 - windows controls 111
 - working with 106
- workspace
 - customizing 10

X

- X location 31

Y

- Y location 31