Wonderware® FactorySuite™ InTouch™ User's Guide

For Version A

Last Revision: September 2002

Invensys Systems, Inc.

Invensys Systems, Inc.
33 Commercial Street
Foxboro, MA 02035
(949) 727-3200
http://www.wonderware.com

Contents

CHAPTER 6: Tagname Dictionary................243

Welcome to InTouch

Welcome to Wonderware® InTouch®, the quickest and easiest way to create human-machine interface (HMI) applications for the Microsoft® Windows™ 2000 and Windows XP operating systems. InTouch is a component of the Wonderware FactorySuite™. InTouch applications span the globe in a multitude of vertical markets including food processing, semiconductors, oil and gas, automotive, chemical, pharmaceutical, pulp and paper, transportation, utilities, and more.

For more information on the operating system versions supported, see "System Requirements."

By using InTouch, you can create powerful, full-featured applications that exploit the key features of Microsoft Windows, including ActiveX controls, OLE, graphics, networking and more. InTouch can also be extended by adding custom ActiveX controls, wizards, generic objects, and creating InTouch QuickScript extensions.

InTouch consists of three major programs, the InTouch Application Manager, WindowMaker™ and WindowViewer™.

The InTouch Application Manager organizes the applications you create. It also is used to configure WindowViewer as an NT service, to configure Network Application Development (NAD) for client-based and server-based architectures, to configure Dynamic Resolution Conversion (DRC) and/or distributed alarming. The DBDump™ and DBLoad™ database utilities are also launched from the Application Manager.

WindowMaker is the development environment, where object-oriented graphics are used to create animated, touch-sensitive display windows. These display windows can be connected to industrial I/O systems and other Microsoft Windows applications.

WindowViewer is the runtime environment used to display the graphic windows created in WindowMaker. WindowViewer executes InTouch QuickScripts, performs historical data logging and reporting, processes alarm logging and reporting, and can function as a client and a server for both DDE and SuiteLink™ communication protocols.

Contents

- Running InTouch for the First Time
- Using the InTouch Application Manager

System Requirements

To run InTouch, we recommend the following hardware and software:

- Any IBM® compatible PC with a Pentium II processor or higher (minimum: 400MHz on a single node system, recommended: 1.2GHz or higher).
- At least 2GB of free hard disk space.
- At least 256MB of random-access memory (RAM), 512MB of RAM is recommended.

Note We recommend 5MB of RAM per 5K tagnames. For example, 32MB of RAM for 32K tagname support and 128MB of RAM for 60K tagname support.

- SVGA display adapter (2MB RAM recommended).

Note We recommend you view WindowMaker in 800x600 resolution or higher to ensure full visibility of all dialog boxes.

- Pointing device. For example, mouse, trackball, touch screen.
- Microsoft® Windows® 2000 Professional with Service Pack 1 or Windows XP™ Professional with Service Pack 3 operating systems.
- For the Windows 2000 operating system to implement the distributed functionality of InTouch, Wonderware NetDDE must be installed and operational.

Note Wonderware FactorySuite InTouch Version 8.0 (or later) does not support the Microsoft Windows 3.x or Microsoft Windows for Workgroups or Windows 9x operating systems.

Installing InTouch

The Wonderware FactorySuite installation program is used to install InTouch. InTouch runs on Microsoft Windows 2000 or Windows XP operating systems. The installation program creates directories as needed, copies files from the compact disk to your hard drive.

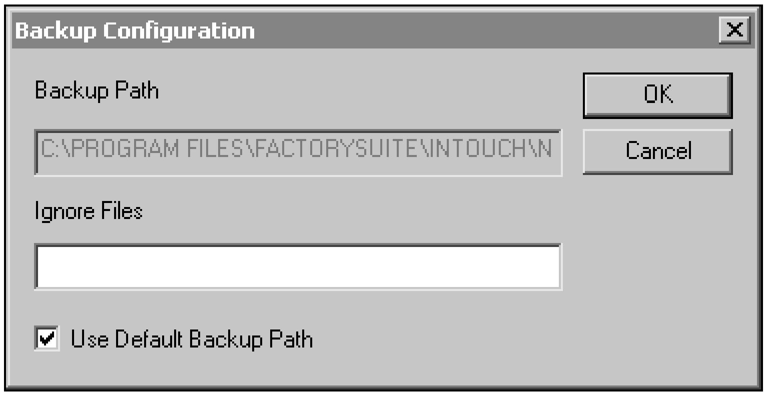
Note For complete InTouch installation instructions, read the InTouch Release Notes (ITRELNOTES.TXT). For complete FactorySuite installation instructions, read the FactorySuite Release Notes (FSRELNOTES.TXT). Additionally, your online *FactorySuite System Administrator's Guide* provides you with detailed installation instructions for most products included your FactorySuite software package.

Upgrading From Previous InTouch Versions

All versions of InTouch will be able to upgrade applications from any previous version through two dialogs that will ask you to confirm the upgrade and confirm that your application has been backed up. These dialogs appear when you attempt to open an application (in either WindowMaker or WindowViewer) that you created in a previous version of InTouch.

Backing up Older Applications

When you attempt to open an older application, Window Maker will detect that it is older and prompt you to back it up prior to converting it by displaying the **Backup Configuration** dialog box:



Backup Path: C:\PROGRAM FILES\FACTORYSUITE\INTOUCH\N

[✓] Use Default Backup Path

To change the default backup path (<Application Directory>\Bak), turn off the **Use Default Backup Path** option and then, in the **Backup Path** box, type the path to the existing directory where you want the backup copy of your application saved. If the directory does not exist, you must first create it, then continue the backup.

In the **Ignore Files** box, you can specify any files that you want ignored during backup. (By default, all the files in the application directory are backed up.) Type each file name separated by a semicolon (;).

Tip You can use the standard wild card characters ('*' and '?') with the filenames.

About this Manual

This manual is divided into a series of logical building block chapters that describe the various aspects of building an InTouch application. It is written in a "procedural" format that tells you in numbered steps how to perform most functions or tasks.

If you are viewing this manual online, when you see text that is green, click the text to "jump" to the referenced section or chapter. When you jump to another section or chapter and you want to come back to the original section, a "back" option is provided.

Tip These are "tips" that tell you an easier or quicker way to accomplish a function or task.

To familiarize yourself with the WindowMaker development environment and its tools, read Chapter 1, "WindowMaker Program Elements." To learn about working with windows, graphic objects, wizards, ActiveX controls and so on, read Chapter 2, "Using WindowMaker." For details on the runtime environment (WindowViewer), read Chapter 2, "Using WindowMaker."

In addition, the *InTouch Reference Guide* provides you with an in-depth reference to the InTouch QuickScript language and functions, system tagnames, and tagname **.fields**.

For details on the add-on program, SPC Pro, see your *SPC Pro User's Guide*.

For details on the add-on program, Recipe Manager, see your *Recipe Manager User's Guide*.

For details on the add-on program, SQLAccess Manager, see your *SQL Access Manager User's Guide*.

The *FactorySuite Systems Administrator's Guide* also provides you with complete information on the common components in the FactorySuite, system requirements, networking considerations, product integration, technical support, and so on.

Online manuals are also included in your FactorySuite software package for all FactorySuite components.

Note You must install the Adobe Acrobat Reader (version 4.0 or later) to view or print the online manuals.

Assumptions

This manual assumes you are:

- Familiar with the Windows 2000 and/or Windows XP operating system working environment.
- Knowledgeable of how to use of a mouse, Windows menus, select options, and accessing online Help.
- Experienced with a programming or macro language. For best results, you should have an understanding of programming concepts such as variables, statements, functions and methods.

Recommended Reading

For additional information on building effective human-computer interfaces, the following sources are recommended:

The Windows Interface: An Application Design Guide, Microsoft Press, 1992.

Dreyfuss, Henry. *Symbol Sourcebook: An Authoritative Guide to International Graphic Symbols.* Van Nostrand Reinhold, 1984.

Laurel, Brenda. *The Art of Human-Computer Interface Design*. Addison-Wesley, 1990.

Norman, Donald A. *The Design of Everyday Things*. Doubleday, 1990.

Tufte, Edward. *The Visual Display of Quantitative Information*. Graphics Press, 1983.

Chappell, David. *Understanding Active X and OLE - A Guide for Developer's and Managers*. Microsoft Press, Strategic Technology Series 1996.

Technical Support

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

Prior to contacting technical support, please refer to the relevant chapter(s) in your *InTouch User's Guide* for a possible solution to any problem you may have with your InTouch system. If you find it necessary to contact technical support for assistance, please have the following information available:

1. Your software serial number.
2. The version of InTouch you are running.
3. The type and version of the operating system you are using. For example, Microsoft Windows NT Version 4.0 SP5 (or later) workstation.
4. The exact wording of system error messages encountered.
5. Any relevant output listing from the Logger, the Microsoft Diagnostic utility (MSD), or any other diagnostic applications.
6. Details of the attempts you made to solve the problem(s) and your results.
7. Details of how to recreate the problem.
8. If known, the Wonderware Technical Support case number assigned to your problem (if this is an on-going problem).

For more information on Technical Support, see your online *FactorySuite System Administrator's Guide*.

Your FactorySuite License

Your FactorySuite system license information can be viewed through the license viewing utility that is launched from the WindowMaker Help **About** dialog box.

To open license utility program

1. On the WindowMaker **Help** menu, click **About**.
2. Click **View License**. The **License Utility - LicView** dialog box appears.

For more information on the licensing viewing utility, see your online *FactorySuite System Administrator's Guide*.

Running InTouch for the First Time

The first time you run INTOUCH.EXE, the INTOUCH.INI file is automatically created. This file contains the default configuration settings for your application. As you configure your application, your settings are written to the INTOUCH.INI file.

Once you have customized your application, when you create a new application, you can copy your customized INTOUCH.INI file to your new application's directory. This eliminates the need for you to reset your customized parameters each time you create a new application.

For more information on customizing your application, see Chapter 2, "Using WindowMaker."

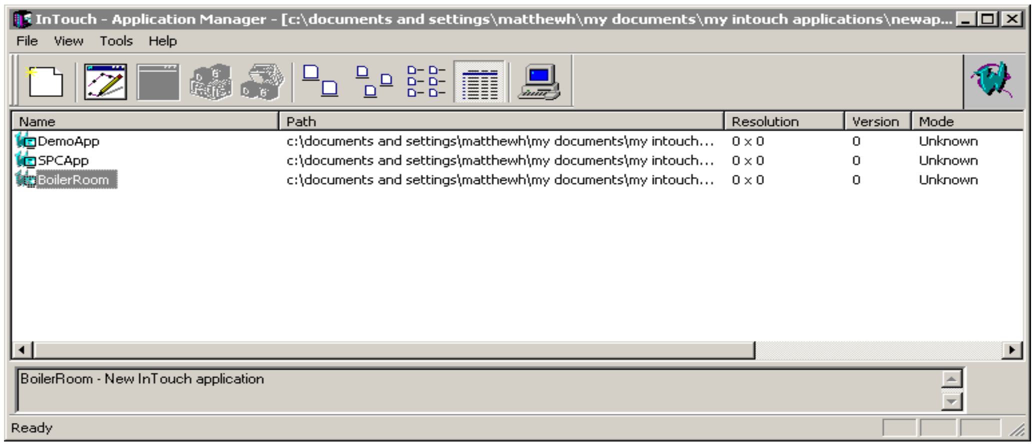
To run InTouch for the first time

1. Start the InTouch program (intouch.exe). The **Welcome to InTouch Application Manager** dialog box appears.
2. Click **Next**. A second **Welcome to InTouch Application Manager** dialog box appears displaying the default path for the starting directory. For example, **C:\Documents and Settings\CPUName\My Documents\My InTouch Applications**.
3. To specify a different directory, type the path to the directory in the input box, or click **Browse** to locate the directory. When a user specifies a different directory, InTouch defaults to the directory location accessed by the most recent user.
4. Click **Finish**.
5. The **InTouch - Application Manager** appears and automatically search your computer for any current InTouch applications. If an application(s) is found, an icon with the application's name appears in the dialog box.

To create a new application

1. On the **File** menu, click **New**, or click the **New** tool in the toolbar. The **Create New Application** wizard appears.
2. Click **Next**. A second **Create New Application** wizard appears. By default, the system will display the path to your InTouch directory followed by "**NewApp**."
3. In the input box, type the path to the directory in which you want your application to be created or click **Browse** to locate the directory.
4. Click **Next**. If the directory you specify does not exist, a message dialog box appears asking if you want to create it. Click **OK**. A third **Create New Application** wizard dialog box appears.
5. In the **Name** box, type a unique name for the new application's icon that appears when the application is listed in the **InTouch Application Manager** window.
6. In the **Description** box, type a description of the application. (The description is optional. However, if you do type a description, it can be a total of 255 characters.)

7. Click **Finish**. The **InTouch - Application Manager** reappears displaying an icon with the name you specified for the new application.


InTouch - Application Manager - [c:\documents and settings\matthewh\my documents\my intouch applications\newap...
File View Tools Help

Name	Path	Resolution	Version	Mode
DemoApp	c:\documents and settings\matthewh\my documents\my intouch...	0 x 0	0	Unknown
SPCApp	c:\documents and settings\matthewh\my documents\my intouch...	0 x 0	0	Unknown
BoilerRoom	c:\documents and settings\matthewh\my documents\my intouch...	0 x 0	0	Unknown

BoilerRoom - New InTouch application

Ready

8. To open an application, click the right mouse button as you select it and then, click the name of the program you want to use for the application in the **File** menu, or select the application in the list and then, click the WindowMaker tool in the toolbar. (WindowViewer cannot be executed for a new application.)

Tip To quickly open the application, double-click it's icon or select it and then, press **Enter**.

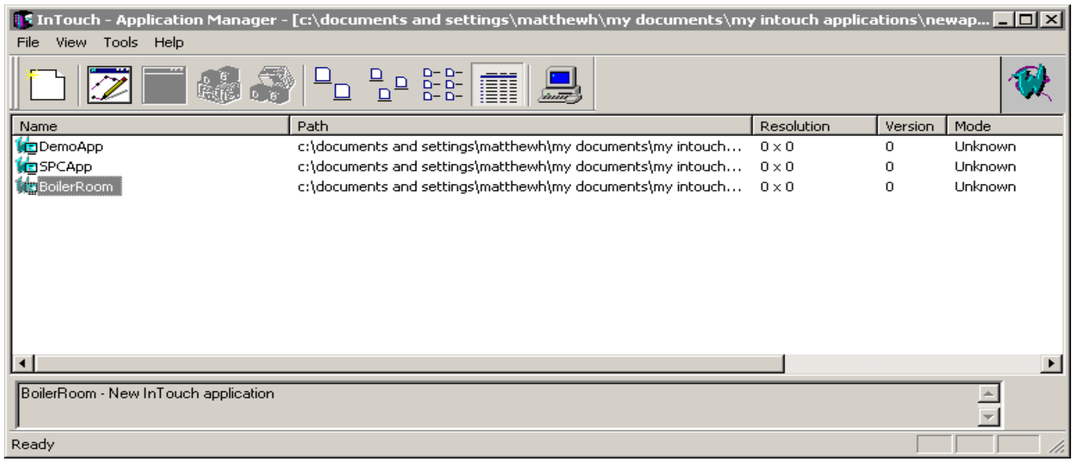
Using the InTouch Application Manager

You will use the InTouch Application Manager to create new applications, open existing applications in either WindowMaker or WindowViewer, delete applications, and run the InTouch DBDump and DBLoad Tagname Dictionary utility programs.

For more information on the DBDump and DBLoad programs, see Chapter 6, "Tagname Dictionary."

To run the InTouch Application Manager

Start the InTouch program (intouch.exe). The InTouch Application Manager dialog box appears.

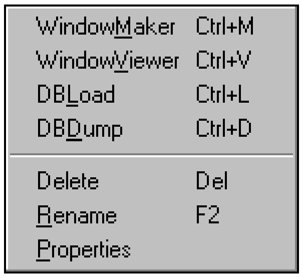

InTouch - Application Manager - [c:\documents and settings\matthewh\my documents\my intouch applications\newap...
File View Tools Help
Ready

Name	Path	Resolution	Version	Mode
DemoApp	c:\documents and settings\matthewh\my documents\my intouch...	0 x 0	0	Unknown
SPCApp	c:\documents and settings\matthewh\my documents\my intouch...	0 x 0	0	Unknown
BoilerRoom	c:\documents and settings\matthewh\my documents\my intouch...	0 x 0	0	Unknown

BoilerRoom - New InTouch application

When you select an application in the list, it's name and it's description appears in the box at the bottom of the screen. If you right-click the description box, a menu appears displaying the commands that you can apply to the selected text.

You can also execute several of the InTouch Application Manager's menu commands from the menu that appears when you click the right mouse button as you select an application. For example:



To rename an application's icon, right-click the application in the list and then, click **Rename**. Type the new name, and then press **Enter**.

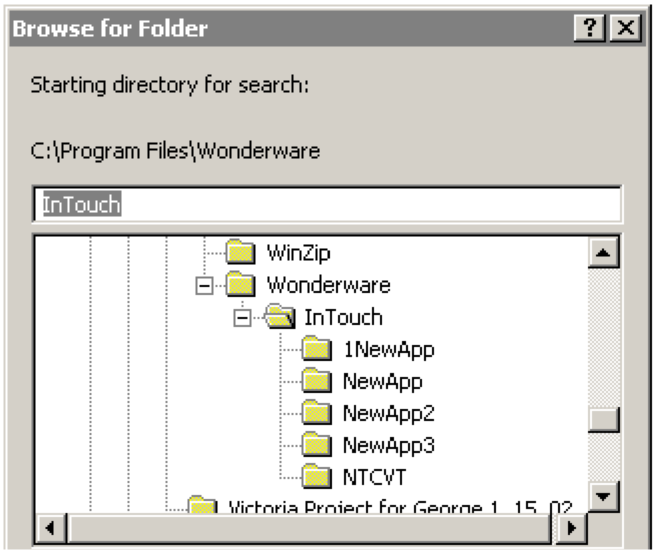
To delete an application's icon, right-click the application in the list and then, click **Delete**. A message box appears asking you to confirm the deletion. Click **Yes** to delete the application from the window or click **No** to cancel the deletion.

Note If you delete an application from the list, it does not delete your files or the application directory.

To find applications

1. On the Tools menu, click Find Applications. The Starting directory for search dialog box appears.

Tip To quickly find an application, right-click the mouse on a blank area of the window and then, click **Find Applications** on the popup menu.


Browse for Folder
?
Starting directory for search:
C:\Program Files\Wonderware
InTouch
WinZip
Wonderware
InTouch
1NewApp
NewApp
NewApp2
NewApp3
NTCVT
Victoria Project for George 1_15_02

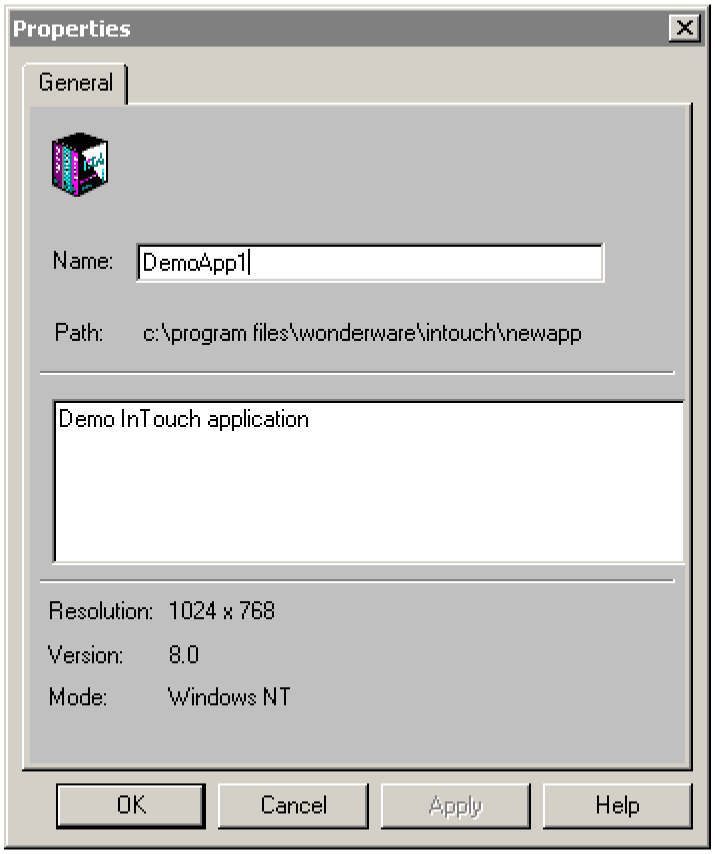
2. Locate the directory in which you want to search for applications and then, click **OK**.

   The InTouch Application Manager reappears displaying icons for all applications that were found in the selected directory.

To view an application's properties

1. Select the application in the list.

2. On the **File** menu, click **Properties**. The **Properties** dialog box appears.



Name: DemoApp1

Path: c:\program files\wonderware\intouch\newapp

Demo InTouch application

Resolution: 1024 x 768

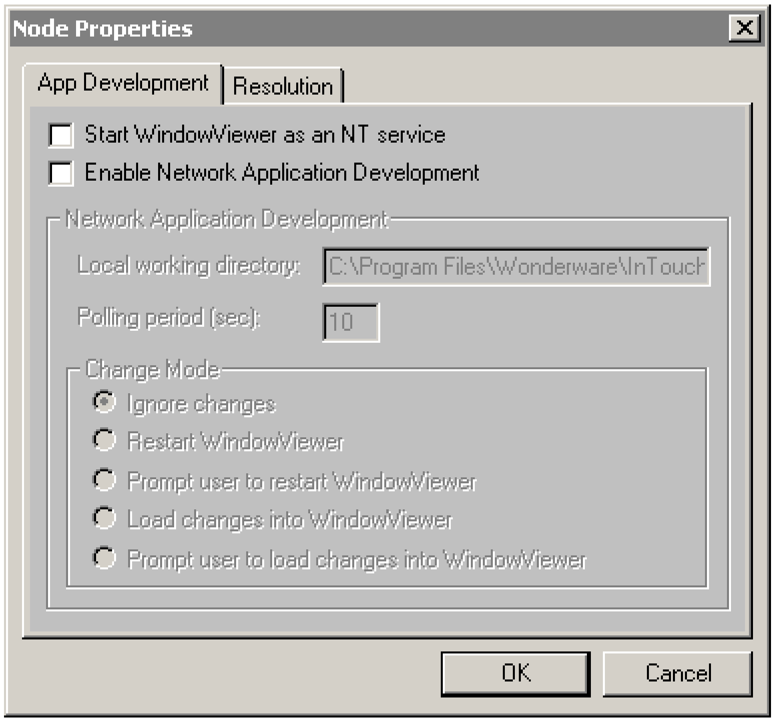
Version: 8.0

Mode: Windows NT

To view a node's properties

On the **Tools** menu, click **Node Properties**. The **Node Properties** dialog box appears.

Tip To quickly access the **Node Properties** dialog box, right-click any blank area of the window and then, click **Node Properties**.


Node Properties
App Development
Resolution
Start WindowViewer as an NT service
Enable Network Application Development
Network Application Development
Local working directory: C:\Program Files\Wonderware\InTouch
Polling period (sec): 10
Change Mode
Ignore changes
Restart WindowViewer
Prompt user to restart WindowViewer
Load changes into WindowViewer
Prompt user to load changes into WindowViewer
OK
Cancel

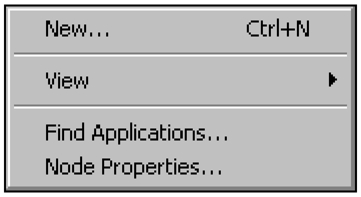
You will use this dialog box to configure the following:

- WindowViewer as an NT Service
- Network Application Development (NAD)
- Dynamic Resolution Conversion (DRC)

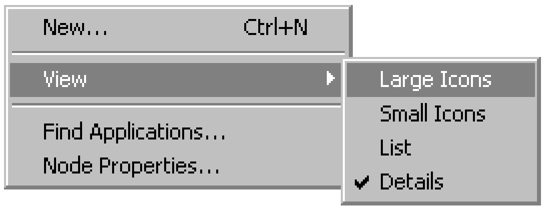
For more information, see Chapter 5, "Building a Distributed Application."

To configure the Application Manager's display window

1. On the **View** menu, click the command that you want to apply or, right-click any column header, or click a blank area of the Application Manager's window or, click a detail (other than the application name) the following popup menu appears.



2. Point to **View**, the following sub-menu appears



3. Select the command that you want to apply.

For more information on the display commands, see "The Application Manager's Tools."

The Application Manager's Tools

By default, when InTouch is initially run, the Application Manager's toolbar and status bar are displayed.

To hide the toolbar

On the **View** menu, select **Toolbar**. To show it again, repeat this step.

To hide the status bar

On the **View** menu select **Status Bar**. To show it again, repeat this step.

The following briefly describes each tool on the Application Manager's toolbar:

Tool	Description
🗋	Executes the **New** command on the **File** menu to create a new application.
📝	Executes the **WindowMaker** command on the **File** menu to open the selected application in WindowMaker. **Tip** To quickly open an application in WindowMaker, double-click it's icon in the display list or, select it and then, press ENTER.
🖥	Executes the **WindowViewer** command on the **File** menu to open the selected application in WindowViewer.
📦	Executes the **DBLoad** command on the **File** menu to run the DBLoad utility used to load a Tagname Dictionary input file.
📦	Executes the **DBDump** command on the **File** menu to run the DBDump utility program used to extract an application's Tagname Dictionary. For more information on the DBDump and DBLoad programs, see Chapter 6, "Tagname Dictionary."

Tool	Description
	Executes the **Large Icons** command on the **View** menu to display large icons for the listed applications.
	Executes the **Small Icons** command on the **View** menu to display small icons for the listed applications.
	Executes the **List** command on the **View** menu to change the dialog box to the list view mode.
	Executes the **Details** command on the **View** menu to change the dialog box to the details view mode.
	Executes the **Node Properties** command on the **Tools** menu to open the **Node Properties** dialog box that is used to set the computer's properties for: WindowViewer as an NT Service Network Application Development (NAD) Dynamic Resolution Conversion (DRC) For more information, see Chapter 5, "Building a Distributed Application."

C H A P T E R 1

WindowMaker Program Elements

WindowMaker is the development environment for InTouch. The WindowMaker graphical user interface adheres to Windows 2000 and Windows NT GUI standards. WindowMaker supports floating and docking toolbars, right-mouse click menus throughout for quick access to frequently used commands and a customizable color palette that provides 16.7 million color support. (The color support is limited only by your video card capability.)

WindowMaker's Application Explorer provides you with a powerful, graphical method for navigating and configuring your InTouch applications. It provides you with easy access to WindowMaker's most commonly used commands and functions such as, all windows commands, all configuration commands and all InTouch QuickScript editors. Additionally, the Application Explorer will display all installed add-on programs such as SQL Access Manager, SPC Pro and Recipe Manager and it provides you with a customizable application launcher.

You can configure the Application Explorer to launch any other FactorySuite program or Windows program to quickly switch between HMI configuration, I/O Server configuration and control configuration.

Contents

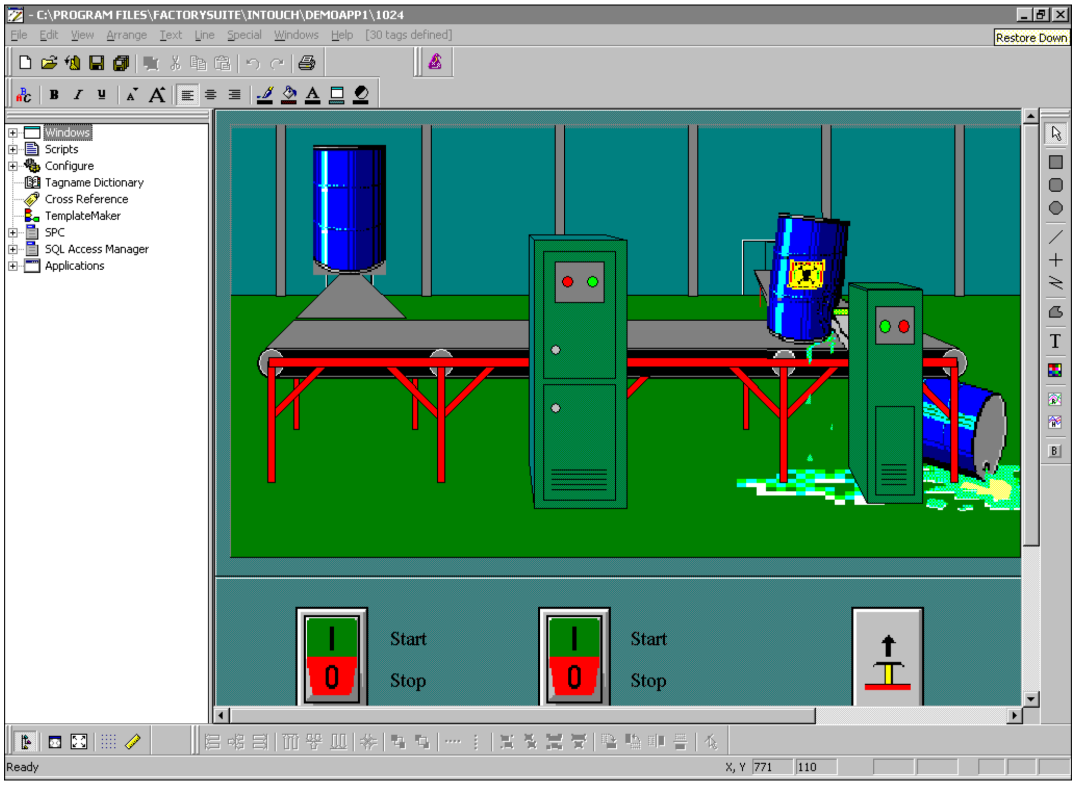
The WindowMaker GUI

WindowMaker supports Windows 2000and Windows XP operating systems graphic user interface (GUI) standards including, right-click mouse support, floating and docking toolbars, pull down menus, context-sensitive help and so on.

The WindowMaker development environment is configurable. By default when you initially open WindowMaker, most of the available elements are automatically displayed including, all toolbars, the Application Explorer and the status bar. However, you can show or hide any or all of these elements and, you can move the toolbars and the Application Explorer to any location that you desire within the WindowMaker window. You can also display the optional ruler and you can turn on and off the visible grid in your windows.

For more information on moving the toolbars see, "Working with the Floating/Docking Toolbars."

The following illustrates the elements of the WindowMaker development environment:



When you create a new application, and run WindowMaker for the first time, its program elements will automatically appear in the default configuration shown in the illustration above.

Many of the tools will not become active until a window is opened and objects are placed in the window and then selected. When a tool is not active, its functionality is not applicable for the current state of the window or the selected object.

When you close WindowMaker, the toolbar floating or docked positions and sizes, Application Explorer and, WindowMaker window size preferences are all saved. When you subsequently run WindowMaker the last size and position is retained.

The Application Explorer

WindowMaker's Application Explorer is a hierarchical graphical view of your application. It shows you what items you have configured in your application and provides you easy access to those items. It also provides you with quick access to many of WindowMaker's most commonly used commands and functions.

Note You can configure the Application Explorer to launch any other FactorySuite program or Windows program. This powerful feature allows you to quickly switch between your HMI configuration, I/O Server configuration and control configuration.

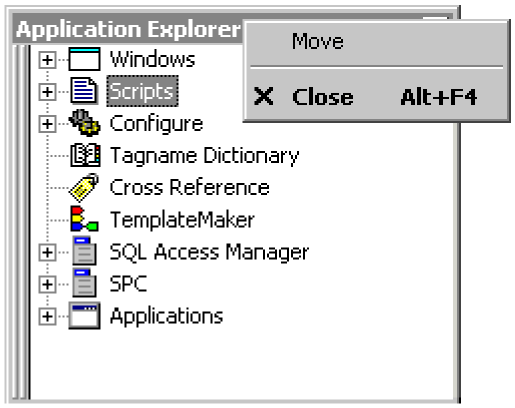
Do not add WindowViewer (VIEW.EXE) to the Application Explorer. If you add WindowViewer, new windows you create in WindowMaker may not be synchronized with the windows in WindowViewer. The proper way to launch WindowViewer is by executing the **WindowViewer** command on the **File** menu, or by clicking the **Runtime** fast switch in the WindowMaker toolbar.

Like all WindowMaker's toolbars, the Application Explorer can be "docked" to any edge of the WindowMaker window or, "floated" anywhere within the WindowMaker window.

When you dock the Application Explorer to an edge of the WindowMaker window, it will automatically size itself accordingly and, if required, scroll bars will be displayed. When you float the Application Explorer within the WindowMaker window its title bar will be displayed. Like all WindowMaker toolbars, when the Application Explorer is floating, you can change its size.

For more information on docking/floating the Application Explorer see, "Working with the Floating/Docking Toolbars."

If you right-click the Application Explorer's title bar, the following menu appears.



For more information on this menu see, "Popup Menus."

For more information on the right-click functionality within the Application Explorer, see "Navigating in the Application Explorer."

To show/hide the Application Explorer

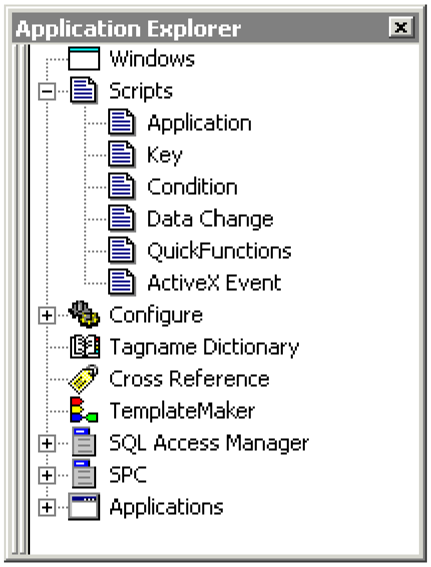
1. On the **View** menu, click **Application Explorer**. (When you initially start WindowMaker, by default, the Application Explorer is displayed.)
2. Repeat step 1 to close the Application Explorer.

Tip To quickly hide the Application Explorer, click the Application Explorer tool on the **View** toolbar. To quickly hide the Application Explorer when it is floating in the WindowMaker window, click the ☒ button on its title bar or, right-click the title bar then, click **Hide** on the menu. When you show the Application Explorer again, it reappears in its previous size and location in the window.

Navigating in the Application Explorer

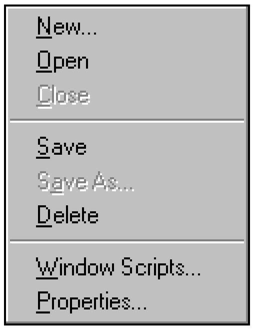
You can expand or collapse the groups listed in the Application Explorer hierarchical graphical view. For example, if you double-click on a group, the view will expand and display the group's members. If you double-click on a member, it will open that member. For example, in the **Windows** group, if you double-click on a member window name, the window will open. If you double-click on **Tagname Dictionary**, the **Tagname Dictionary** dialog box appears, and so on.

Tip All groups that contain members will be preceded with a ⊞. You can click the ⊞ to quickly expand the group and view its members. Likewise, you can click the ⊟ to collapse the group and hide its members. For example:



The following section briefly describes the behavior of each group listed in the Application Explorer when you perform the described action:

Window	
Double-click or **click** ⊞	Expands the view to display the names of all existing windows in your application.
Double-click or **click** ⊟	Collapses the view.
Right-click	Opens a popup menu:
New	Opens the **Windows Properties** dialog box.

Window Name	
Double-click	Opens the window.
Right-click	A popup menu of commands that you can apply to the selected window appears.  New... Open Close Save Save As... Delete Window Scripts... Properties...

Scripts	
Double-click or **click** ⊞	Expands the view to display all InTouch QuickScript types.
Double-click or **click** ⊟	Collapses the view.

Application	
Double-click	Opens the Application Script in the Application Script editor
Right-click	An **Open** button appears. Click to open the Application Script editor.

Key	
Click ⊞	Expands the view to display all Key Scripts in the application. (They are listed by the Key assigned to them.)
Click ⊟	Collapses the view.
Double-click	Opens the Key Script editor. (If scripts exist, the most recently edited script opens.)
Right-click	A popup menu appears:
New	Opens the Key Script editor. (If scripts exist, the most recently edited script opens.)
Open	opens the most recently edited script.

Key Script Name	
Double-click	Opens the script in the Key Scripts editor.
Right-click	A popup menu appears:
New	Opens the Key Script editor. (The script you right-clicked opens.)
Edit	Opens the selected script.
Delete	Deletes the selected script.

Condition	
Click ⊞	Expands the view to display all Condition Scripts in the application.
Click ⊟	Collapses the view.
Double-click	Opens the Condition Script editor. (If scripts exist, the most recently edited script opens.)
Right-click	A popup menu appears:
New	Opens the Condition Script editor. (If scripts exist, the most recently edited script opens.)
Open	Opens the most recently edited script.
List By	Opens a sub-menu:
Name	Lists scripts by their condition
Description	Lists scripts by their comments

Condition Script Name	
Double-click	Opens the script in the Condition Script editor.
Right-click	A popup menu appears:
New	Opens the Condition Script editor. (The script you right-clicked opens.)
Edit	Opens the selected script.
Delete	Deletes the selected script.

Data Change	
Click ⊞	Expands the view to display all Data Change Scripts in the application. (They are listed by the tagname assigned to them.)
Click ⊟	Collapses the view.
Double-click	Opens the Data Change Script editor. (If scripts exist, the most recently edited script opens.)
Right-click	A popup menu appears:
New	Opens the Data Change Script editor. (If scripts exist, the most recently edited script opens.)
Open	Opens the most recently edited script.

Data Change Script	
Double-click	Opens the script in the Data Change Scripts editor.
Right-click	A popup menu appears:
New	Opens the Data Change Script editor. (The script you right-clicked opens.)
Edit	Opens the selected script.
Delete	Deletes the selected script.

QuickFunction	
Click ⊞	Expands the view to display all QuickFunctions in the application. (They are listed by their function.)
Click ⊟	Collapses the view.
Double-click	Opens the QuickFunctions editor. (If QuickFunctions exist, the most recently edited QuickFunction opens.)
Right-click	A popup menu appears:
New	Opens the QuickFunctions editor. (If QuickFunctions exist, the most recently edited QuickFunction opens.)
Open	opens the most recently edited QuickFunction.

QuickFunction Name	
Double-click	Opens the QuickFunction in the QuickFunctions editor.
Right-click	A popup menu appears:
New	Opens the QuickFunctions editor. (The script you right-clicked opens.)
Edit	Opens the selected QuickFunction.
Delete	Deletes the selected QuickFunction.

ActiveX Event	
Click ⊞	Expands the view to display all ActiveX Event Scripts in the application. (They are listed by their event.)
Click ⊟	Collapses the view.

ActiveX Event Script Name	
Double-click	Opens the script in the ActiveX Event Script editor.
Right-click	A popup menu appears:
Edit	Opens the selected script.
Delete	Deletes the selected script.

Configure	
Double-click or **click** ⊞	Expands the view to display many of WindowMaker's configuration commands and the **Wizard/ActiveX Installation** command.
Double-click or **click** ⊟	Collapses the view.

Configuration Item Name	
Double-click	Opens its respective dialog box.
Right-click	Open button appears. Click to open the item's respective dialog box.

Tagname Dictionary	
Double-click	Opens the **Tagname Dictionary** dialog box displaying the last modified tagname's definition. Otherwise, the default **$AccessLevel** system tagname is displayed.
Right-click	**Open** button appears. Click to open the **Tagname Dictionary** dialog box displaying the last modified tagname's definition. Otherwise, the default **$AccessLevel** system tagname is displayed.

Cross Referencing	
Double-click	Opens the **Cross Reference** utility.
Right-click	**Open** button appears. Click to open the **Cross Reference** utility.

TemplateMaker	
Double-click	Opens the SuperTag **TemplateMaker** utility.
Right-click	**Open** button appears. Click to open the SuperTag **TemplateMaker** utility.

Add-on Programs	
Double-click, or **click** ⊞	Expands the view to display the add-on program's configuration commands.
Double-click, or **click** ⊟	Collapses the view.

Program Name	
Double-click	Opens respective dialog box.
Right-click	**Open** button appears. Click to open the command's respective dialog box.
Note The add-on programs must be installed to appear in the Application Explorer.	

Applications	
Double-click or **click** ⊞	Expands the view to display all other applications that you can launch from WindowMaker.
Double-click or **click** ⊟	Collapses the view.
Right-click	**New** button appears. Click to add an application to the Application Explorer.

Application Name	
Double-click	Launches the application without exiting WindowMaker.
Right-click	A pop-up menu appears:
New	Opens a blank Application Properties dialog box. Use this to add a new application.
Run	Will start the application.
Delete	Will remove the application from the Application Explorer.
Properties	Opens the Application Properties dialog box for the selected application.

Adding Applications to the Application Explorer

One of the most powerful features of the WindowMaker Application Explorer, is its ability to launch other FactorySuite and third-party Windows applications from within WindowMaker.

For example, you can run your I/O Server program and configure it at the same time that you are developing your application. You can launch third-party Windows programs you frequently use such as Windows Notepad, Wordpad, Microsoft Excel, Microsoft Word, Microsoft Paint, and so on.

Tip The InTouch add-on programs, SQL Access, SPC Pro and Recipe Manager are automatically added to the Application Explorer when you install them.

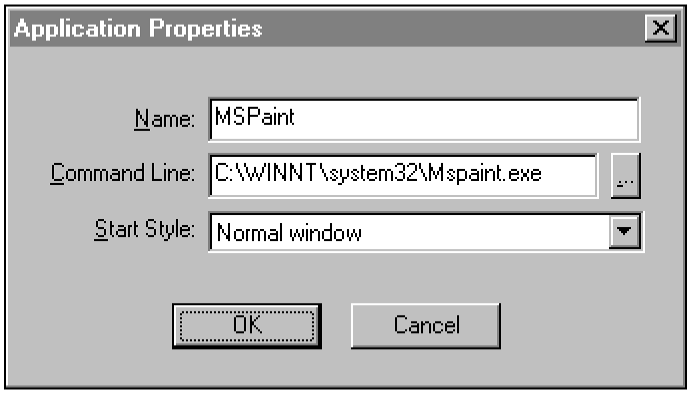
Caution! Do not add WindowViewer (VIEW.EXE) to the Application Explorer. If you add WindowViewer, new windows you create in WindowMaker may not be synchronized with the windows in WindowViewer. The proper way to launch WindowViewer is by executing the **WindowViewer** command on the **File** menu, or by clicking the **Runtime** fast switch in the WindowMaker toolbar.

You can also configure the Application Explorer to open a specific document or spreadsheet in a program. For example, if you select a specific Microsoft Word document or Microsoft Excel spreadsheet, when you double-click that application's icon in the Application Explorer, the application starts up and automatically displays the document or spreadsheet you selected. These documents display the icon of the application in which they were originally created, or the .exe configured as the associated application.

To add an application to the Application Explorer

1. Display the Application Explorer.
2. Right-click **Applications**. A **New** button appears.

3. Click **New**. The **Application Properties** dialog box appears.

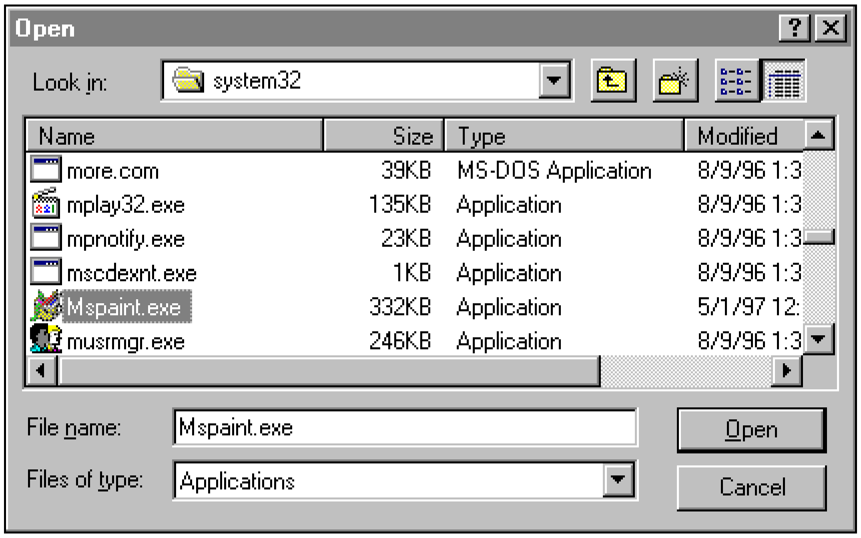

Application Properties

Name: MSPaint

Command Line: C:\WINNT\system32\Mspaint.exe

Start Style: Normal window

4. In the **Name** box, type the name that you want to display in the Application Explorer for the application.
5. In the **Command Line** box, type the full path for the application or, click the ellipsis button. The **Open** dialog box appears.


Open

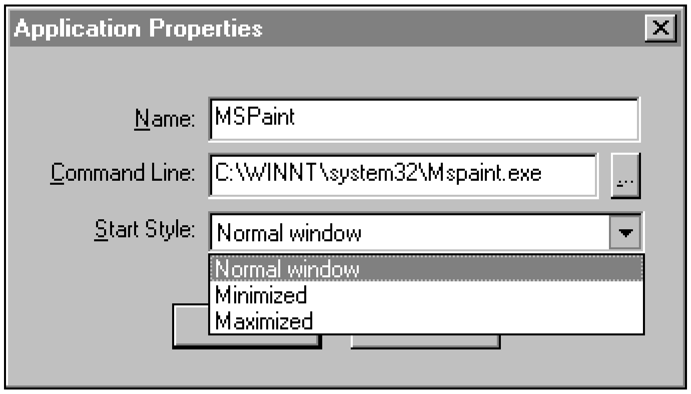
Look in: system32

Name	Size	Type	Modified
more.com	39KB	MS-DOS Application	8/9/96 1:3
mplay32.exe	135KB	Application	8/9/96 1:3
mpnotify.exe	23KB	Application	8/9/96 1:3
mscdexnt.exe	1KB	Application	8/9/96 1:3
Mspaint.exe	332KB	Application	5/1/97 12:
musrmgr.exe	246KB	Application	8/9/96 1:3

File name: Mspaint.exe

Files of type: Applications

6. Locate the application and then, click **Open**. The **Application Properties** dialog box reappears.


Application Properties
Name: MSPaint
Command Line: C:\WINNT\system32\Mspaint.exe
Start Style: Normal window
Normal window
Minimized
Maximized

Tip You can add optional command line parameters for the application in the **Command Line** field.

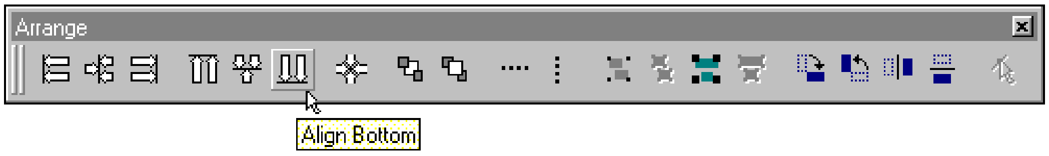
7. Click the **Start Style** arrow and select the style that you want for the application when you run it from WindowMaker.
8. Click **OK**.

The application is added to the Application Explorer under **Applications**. You can now run the application at any time from WindowMaker.

The WindowMaker Toolbars

The tools on the WindowMaker toolbars are grouped by common functionality. For example, the **Arrange** toolbar contains tools that you can use to quickly apply most of the commands found on the **Arrange** menu.

If you rest the cursor on a tool, a tool tips box appears displaying the name of the tool. For example:



Working with the Floating/Docking Toolbars

The WindowMaker toolbars have "floating and docking" capability. Meaning you can move any toolbar from its default "docked" position and dock it again on any edge of the WindowMaker window or, in the toolbar area at the top of WindowMaker's window. Docked toolbars can also be moved from their docked position at the edge of the window and floated within the window. Floating toolbars have title bars that allow you to change its size.

Tip The Application Explorer can also be docked or floated anywhere in the window and its size can also be changed when it is floating just like any other toolbar.

For more information on the Application Explorer, see "The Application Explorer."

To change a docked toolbar's location in the window

1. Click the toolbar's "cool bars" or, on a blank area of the docked toolbar.
2. Hold down the left mouse button as you move the toolbar away from the edge of the window or, out of the toolbar area, or any edge of the WindowMaker window.
3. Move the toolbar to another edge of the window or, to a new position in the toolbar area.

Tip If you move a horizontally docked toolbar to the left or right edge of the WindowMaker window, it will automatically change to its default vertical shape when in position for docking to that edge. Likewise, if you move a vertical toolbar to the toolbar area at the top of the window or, to the bottom edge of the window, it will change to its default horizontal shape when in position for docking.

4. Release the mouse. The toolbar will be docked in the new location.

Note You cannot change the size nor can you access the right-click menu when a toolbar is docked.

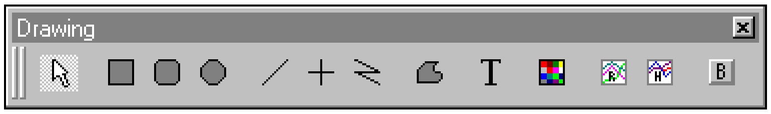
To show/hide a docked toolbar

1. On the **View** menu select the toolbar's name. (When you initially start WindowMaker, by default, all toolbars are showing.)
2. Repeat step 1 to reverse your selection.

Tip When you show a docked toolbar that has been hidden again, it reappears in its last docked location in the window.

To float a docked toolbar

1. Click the toolbar's "cool bars" or, a blank area of the docked toolbar.
2. Hold down the mouse button as you move the toolbar from its docked position to a new location within the WindowMaker window.
3. Release the mouse button. The toolbar appears as follows:



Tip You can change the size of a floating toolbar.

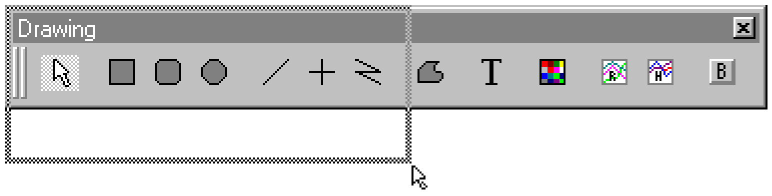
When you dock a floating toolbar on the left or right edge of the WindowMaker window, it will automatically change to its default vertical shape when it is in position for docking to that edge. Likewise, if you move it to the toolbar area at the top of the WindowMaker window or, to the bottom edge of the WindowMaker window, it will change to its default horizontal shape when in position for docking.

This will also occur when you dock a floating toolbar whose size you have changed. However, when you float the toolbar in the window again, it will return to its previous floated size.

To change the size of a floating toolbar

1. Move the mouse over any edge of the toolbar. The cursor will change to a double-ended arrow.
2. Click on the edge and hold down the mouse button as you move the mouse to size the toolbar.

Tip As you move the mouse, a box appears to indicate the size the toolbar will be if you release the mouse button. For example:



3. Release the mouse button when the toolbar is the desired size.

Tip When you right-click the title bar of a floating toolbar, a menu appears displaying the commands you can apply to the toolbar.

For more information on this menu, see "Popup Menus."

To hide/show a floating toolbar

1. To hide a floating toolbar, on the **View** menu, select the toolbar's name or, right-click the toolbar's title bar then, click **Hide** on the menu.

Tip To quickly hide the toolbar, click the ☒ button on the toolbar's title bar.

2. To show a hidden floating toolbar, on the **View** menu select the toolbar's name.

Tip The toolbar reappears at its previous location and in its previous size.

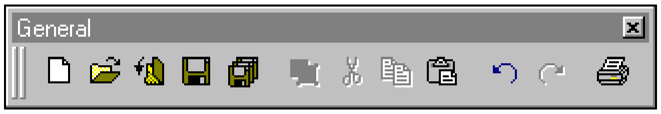
To hide all toolbars at once

1. On the **View** menu, click **Hide All** or, click the Hide/Restore All tool on the **View** toolbar. All toolbars and the Application Explorer will be hidden.
2. Repeat step 1 to show them again.

Tip All displayed toolbars will have a check mark preceding their names on the **View** menu.

General Toolbar

The **General** toolbar is grouped with tools that execute most of the window commands found on the **File** menu and the Microsoft Windows Clipboard tools found on the **Edit** menu:



Tip When you right-click a blank area of an open window, or right-click a window name under **Windows** in the Application Explorer, a menu appears that also contains most of these same windows commands.

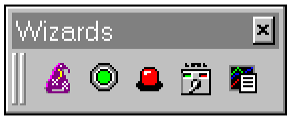
The following briefly describes each tool:

Icon Name	Icon	Description
New Window Tool	🗋	Executes the **New Window** command on the **File** menu to open the **Windows Properties** dialog box to create a new window.
Open Window Tool	📂	Executes the **Open Window** command on the **File** menu to open the **Windows to Open** dialog box listing the names of existing windows that you can select to open.
Close Window Tool		Executes the **Close Window** command on the **File** menu to open the **Windows to Close** dialog box listing the names of all currently open windows that you can select to close.
Save Window Tool	💾	Executes the **Save Window** command on the **File** menu to open the **Windows to Save** dialog box listing the names of all currently open windows that have been modified since they were last saved.
Save All Tool		Automatically saves all currently open windows that have been modified since they were last saved. This tool does not ask for confirmation on a per window basis. It saves all modified windows automatically.

Icon Name	Icon	Description
Duplicate Tool	⧉	Executes the **Duplicate** command on the **Edit** menu to duplicate the currently selected object(s) in the window.
Cut Tool	✂	Executes the **Cut** command on the **Edit** menu to cut the currently selected objects(s) from the window and copies them to the Windows Clipboard.
Copy Tool	⎘	Executes the **Copy** command on the **Edit** menu to copy the currently selected objects(s) and copies them to the Windows Clipboard. (Copied objects are not erased from the window.)
Paste Tool	📋	Executes the **Paste** command on the **Edit** menu to paste any object that has been cut or copied to the Windows Clipboard. (The cursor changes to the paste mode. Click in the window to paste the copied or cut object.)
Undo Tool	↶	Executes the **Undo** command on the **Edit** menu that reverses the last action or command applied to an object.
Redo Tool	↷	Executes the **Redo** command on the **Edit** menu that reverses the last undo action or command applied to an object. By default the number of undo/redo levels is set to 10. You can increase the **Levels of Undo** to 25 in the **WindowMaker Properties** dialog box. To access this dialog box, in the Application explorer, under **Configure**, double-click **WindowMaker** or, on the **Special** menu, point to **Configure** then, click **WindowMaker** on the submenu.
Print Tool	🖨	Executes the **Print** command on the **File** menu to open the **WindowMaker Printout** dialog box used to print database, window information, and QuickScripts.

Wizards/ActiveX Toolbar

The **Wizards/ActiveX** toolbar, by default, only contains the wizard tool that you use to access the wizard **Selection Dialog** box. However, you can add any installed wizard or ActiveX control to the toolbar. For example:

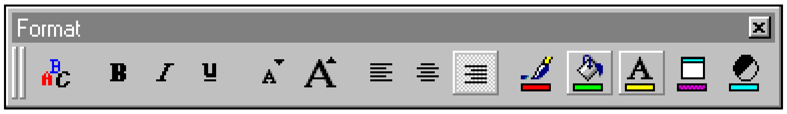


The following briefly describes the wizard tool:

Tool	Description
	Displays the **Wizard Selection** dialog box used for selecting wizard to paste into your windows.

Format Toolbar

The **Format** toolbar is grouped with tools that execute most of the text object formatting commands found on the **Text** menu. It also contains the tools you use to access the color palette to select line, text, fill, window background and transparent object color.



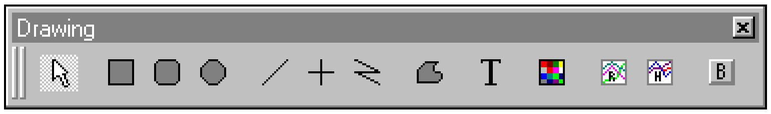
The following briefly describes each tool:

Icon Name	Icon	Description
Font Button		Executes the **Font** command on the **Text** menu to open the **Font** dialog box to select the font, its style and size.
Bold Button		Executes the **Bold** command on the **Text** menu to apply **bold** styling to single or multiple text string selections and numeric value fields.
Italic Button		Executes the **Italic** command on the **Text** menu to apply *italic* styling to single or multiple text string selections and numeric value fields.
Underline Button		Executes the **Underline** command on the **Text** menu to apply underline styling to single or multiple text string selections and numeric value fields.
Reduce Font Button		Executes the **Reduce Font** command on the **Text** menu to reduce the point size of any font. This command can be applied by selecting the text string(s) and clicking on this tool on the **Text** toolbar.
Enlarge Font Button		Executes the **Enlarge Font** command on the **Text** menu to enlarge point size of any font. This command can be applied by selecting the text string(s) and clicking on this tool on the **Text** toolbar.

Icon Name	Icon	Description
Left Justified Button		Executes the **Left Justified** command on the **Text** menu to align the left edge of single or multiple text string selections and numeric value fields.
Centered Button		Executes the **Centered** command on the **Text** menu to center single or multiple text string selections and numeric value fields.
Right Justified Button		Executes the **Right Justified** command on the **Text** menu to align the right edge of single or multiple text string selections and numeric value fields.
Line Object Color Button		Opens the color palette used to select the color for a line object or an object's outline.
Object Fill Color Button		Opens the color palette used to select an object's fill color.
Text Color Button		Opens the color palette used to select the color for a text object.
Window Color Button		Opens the color palette to select a window's background color.
Bitmap Transparency Color Button		Opens the color palette to select a transparent color for a bitmap object.

Draw Object Toolbar

The **Draw Object** toolbar is grouped with all the tools you use to draw both simple graphic objects such as rectangles, ellipses, lines or text objects and, complex objects such as real-time trends, historical trends, bitmaps and 3-dimensional buttons with labels in your windows:



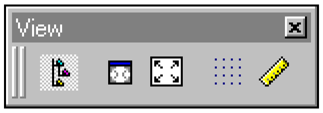
The following briefly describes each tool:

Icon Name	Icon	Description
Selector Mode		Selector mode used to select objects in the window.
Rectangle Tool		Rectangle tool used to draw rectangles or squares.

Icon Name	Icon	Description
Rounded Rectangle Tool		Rounded rectangle tool used to draw rectangles or squares with rounded corners.
Ellipse Tool		Ellipse tool used to draw ellipses or circles.
Line Tool 1		Line tool used to draw lines at any angle.
Line Tool 2		Line tool used to draw horizontal or vertical lines.
Line Tool 3		Line tool used to draw polylines.
Polygon Tool		Polygon tool used to draw polygon objects.
Text Tool		Text tool used to type text objects.
Bitmap Tool		Bitmap tool used to draw a bitmap container for pasting a bitmap directly from the Windows Clipboard or one of the following: .BMP", .JPG, .PCX or .TGA.
Real Time Trend Tool		Real time trend tool used to draw real time trend objects.
Historical Trend Tool		Historical trend tool used to draw historical trend objects.
Button Tool		Button tool used to draw a 3-dimensional button with a label.

View Toolbar

The **View** toolbar is grouped with tools that execute most of the window commands found on the **View** menu. These commands are used to control the state of the WindowMaker window.



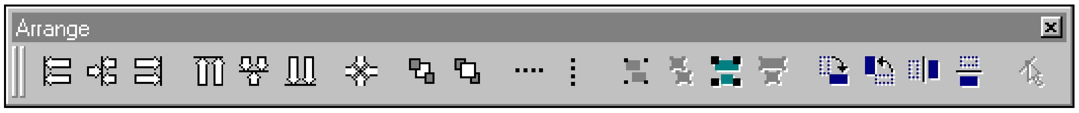
The following briefly describes each tool:

Icon Name	Icon	Description
Application Explorer Button		Toggles the **Application Explorer** command on the **View** menu on and off to show/hide the Application Explorer.
Hide All Button		Toggles the **Hide All** command on the **View** menu on and off to hide/show all docked toolbars. When the hide all mode is active, the overall size of WindowMaker remains the same. To return to normal mode, click the Hide/Restore All tool on the floating **View Toolbar**, or click **Hide All** on the **View** menu. In the hide all mode, all floating toolbars remain visible and the **View Toolbar** automatically floats on top of WindowMaker. If you dock any of the floating toolbars while in the hide all mode, the mode is automatically terminated.
Full Screen Button		Toggles **Full Screen** command on the **View** menu on and off to switch the display mode from normal view to full screen. To return to normal mode, click the Full Screen tool on the floating **View Toolbar**, or click **Full Screen** on the **View** menu. In the full screen mode, all WindowMaker program elements are hidden except, any open windows and floating toolbars. The **View Toolbar** automatically floats on top of WindowMaker. In the full screen mode, the coordinates of the client area will remain the same. For example, the top left is 0,0. The full screen mode automatically sets the coordinates after it maximizes the client area, hides the Title Bar and menu bar and, adjusts the client area to mimic View's full screen mode.

Icon Name	Icon	Description
Snap to Grid Button		Toggles the **Snap to Grid** command on the **Arrange** menu on and off to show/hide the visible grid used to align objects. Works with the **Snap to Grid** command on the **Arrange** menu. If the **Snap to Grid** option in the **WindowMaker Properties** dialog box is not selected, this tool will have no effect.
Ruler Button		Toggles the **Ruler** command on the **View** menu on and off to show/hide the ruler. For more information on the ruler, see "The WindowMaker Ruler."

Arrange Toolbar

The **Arrange** toolbar is grouped with tools that execute most of the object arranging commands found on the **Arrange** menu:



The following briefly describes each tool:

Button Title	Button	Description
Align Left Button		Executes the **Align Left** command on the **Arrange/Align** submenu. Aligns the left edge of all selected objects with the left edge of the left most selected object.
Align Center Button		Executes the **Align Center** command on the **Arrange/Align** submenu. Aligns the vertical centerline of all selected objects with the centerline of the group of objects selected.
Align Right Button		Executes the **Align Right** command on the **Arrange/Align** submenu. Aligns the right edge of all selected objects with the right edge of the right most selected object.
Align Top Button		Executes the **Align Top** command on the **Arrange/Align** submenu. Aligns the top edge of all selected objects with the top edge of the top most selected object.
Align Middle Button		Executes the **Align Middle** command on the **Arrange/Align** submenu. Aligns the middle of all selected objects with the middle of the group of objects.

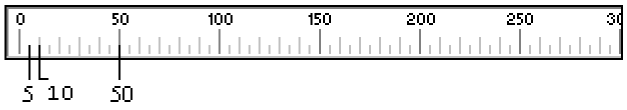
Button Title	Button	Description
Align Bottom Button		Executes the **Align Bottom** command on the **Arrange/Align** submenu. Aligns the bottom edge of all selected objects with the bottom edge of the lowest selected object.
Align Centerpoints Button		Executes the **Align Centerpoints** command on the **Arrange/Align** submenu. Aligns the centerpoint of all the selected objects with the centerpoint of the group of selected objects.
Send to Back Button		Executes the **Send to Back** command on the **Arrange** menu to place all selected objects behind all objects that are not selected.
Bring to Front Button		Executes the **Bring to Front** command on the **Arrange** menu to place all selected objects in front of all objects that are not selected.
Space Horizontal Button		Executes the **Space Horizontal** command on the **Arrange** menu to evenly space all selected objects horizontally between the left most and right most selected objects.
Space Vertical Button		Executes the **Space Vertical** command on the **Arrange** menu to evenly space all selected objects vertically between the top most and bottom most selected objects.
Make Symbol Button		Executes the **Make Symbol** command on the **Arrange** menu to combine multiple objects into a single unit called a symbol.
Break Symbol Button		Executes the **Break Symbol** command on the **Arrange** menu to break a symbol into its individual components.
Make Cell Button		Executes the **Make Cell** command on the **Arrange** menu to combine multiple selected objects into a single unit called a cell. When combining cells each cell will be retained, so when the combined cell is broken the original cells are restored.
Break Cell Button		Executes the **Break Cell** command on the **Arrange** menu to break a selected cell. When combining cells each cell will be retained, so when the combined cell is broken the original cells are restored.
Rotate Clockwise Button		Executes the **Rotate Clockwise** command on the **Arrange** menu to rotate selected objects clockwise 90 degrees.

Button Title	Button	Description
Rotate Counterclockwise Button		Executes the **Rotate CounterClockwise** command on the **Arrange** menu to rotate selected objects counter clockwise 90 degree.
Flip Horizontal Button		Executes the **Flip Horizontal** command on the **Arrange** menu to flip selected objects horizontally.
Flip Vertical Button		Executes the **Flip Vertical** command on the **Arrange** menu to flip selected objects vertically.
Reshape Polygon or Polyline Button		Turns on the **Reshape Object** command on the **Edit** menu to reshape a polygon or polyline.

The WindowMaker Ruler

You can use the WindowMaker ruler to do precision alignment of the objects in your windows.

The small tick marks are spaced 5 pixels apart. The medium tick marks are spaced 10 pixels apart. The numbered large tick marks are spaced 50 pixels apart. For example:



The ruler's 10 and 50 pixel spacing increments are equivalent to the distance in pixels that a selected object is moved when you hold down the SHIFT or CTRL key and press an up, down, right or left arrow key.

For example, if you want to move an object 10 pixels at a time, hold down the SHIFT key while you press an arrow key. To move an object 50 pixels at a time, hold down CTRL key while you press an arrow key.

Tip When you select an object and only press an arrow key, the object is moved 1 pixel at a time.

These features can be useful when you need to make fine alignment and location adjustments.

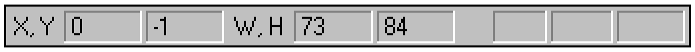
For more information on the using the arrow keys, see "Moving Objects with the Arrow Keys."

To show/hide the ruler

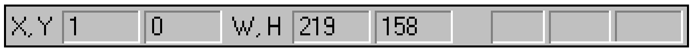
1. On the **View** menu, click **Ruler** or, click the ruler tool in the **View** menu.
2. Repeat step one to hide the ruler.

The WindowMaker Status Bar

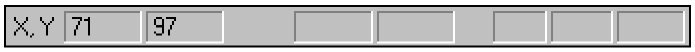
When you select an object in a window, the WindowMaker status bar displays the object's upper left X and Y pixel coordinates and the object's pixel height and width. For example:



When you select multiple objects, the status bar displays the coordinate for the left edge of the left most object (X) and the coordinate for the top edge of the top most object (Y). The width and height are also show for the entire group. For example:



When you click a blank area of a window the status bar displays the X and Y coordinates for the current location of the cursor in the window. For example:



To show/hide the status bar

1. On the **View** menu, click **Status Bar**.
2. Repeat step one to hide the **Status Bar**.

The WindowMaker Color Palette

The WindowMaker palette provides support for up to 16.7 million colors. (The color support is limited only by your video card capability.) By default, the palette offers you a wide range of color selections. However, you can create your own custom palettes. Your custom palettes can be loaded into and exported from the WindowMaker color palette.

Using the Standard Color Palette

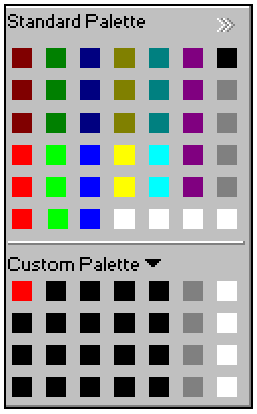
The WindowMaker color palette is used to apply color to static and dynamic properties of lines, rectangles, round rectangles, ellipses, polylines, polygons and text. It is also used to select the background color for your windows and the transparent color for bitmaps that allows you to view objects behind bitmaps.

For more information on transparent bitmaps, see Chapter 2, "Using WindowMaker."

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 use the standard color palette

1. To select a standard color, click the color that you want to use in the **Standard Palette** section. (The color palette will close and the color you selected will be applied.)



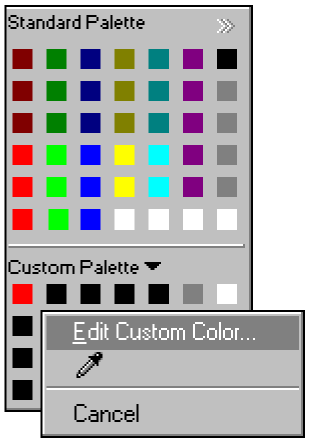
2. To select one of InTouch's 32 classic colors (palette colors prior to InTouch Version 7.0), click the >> in the right corner.

Creating a Custom Color Palette

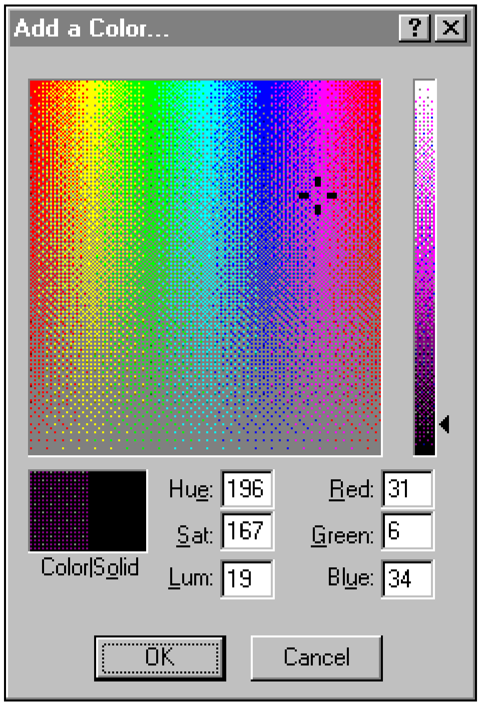
The WindowMaker color palette allows you to define custom colors and add them to your palette. It also allows you to import palettes created in other Windows applications and add them to the standard palette. You can also export your custom palettes to other Windows applications.

To create a custom color

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. The following menu appears.



3. Click **Edit Custom Color**. The **Add a Color** dialog box appears.



Hue: 196
Sat: 167
Lum: 19
Red: 31
Green: 6
Blue: 34

4. Click anywhere in the matrix then, use the slider at the right of the dialog box to adjust the color's attributes.

Hue, Sat, Lum

A combination of hue, saturation and luminosity can be used to define any color. Hue is the value of a color wheel, 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. If you change any of these values, the red, green and blue scales will change to match.

The easiest way to experiment with different colors is to press and hold the mouse then, move the cursor around in the color matrix.

Red, Green, Blue

A combination of red, green and blue levels can be used to define any color. You can see the effect of changing these values in the color matrix. If you change these values, the values for hue, saturation and luminosity will change to match.

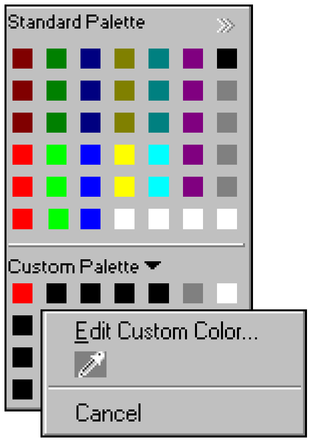
If you define a color using the **Hue**, **Sat**, **Lum** or **Red**, **Green**, **Blue** scales, you can view the color in the **Color|Solid** boxes to make sure you defined the color as you intended.

The **Color** (left) box shows the amount of white and black in the color you specified. The **Solid** (right) box, shows how the color will look if you choose 100% of the color with no white and black. To adjust the color, use the slider at the right of the dialog box. To specify that you want 100% of the color with no white or black, type ALT + O.

5. Click **OK**. The color you selected will be added to the square that you originally clicked in the color palette.

To select a custom color with the blotter 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. The following menu appears.

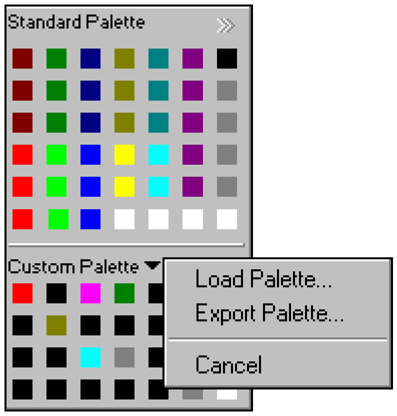


3. Click the blotter tool then, click the color that you want to add to the **Custom Palette** section of the color palette. You can select any color anywhere within the WindowMaker window or outside of WindowMaker completely. You will primarily use this feature when you are creating transparent bitmaps.

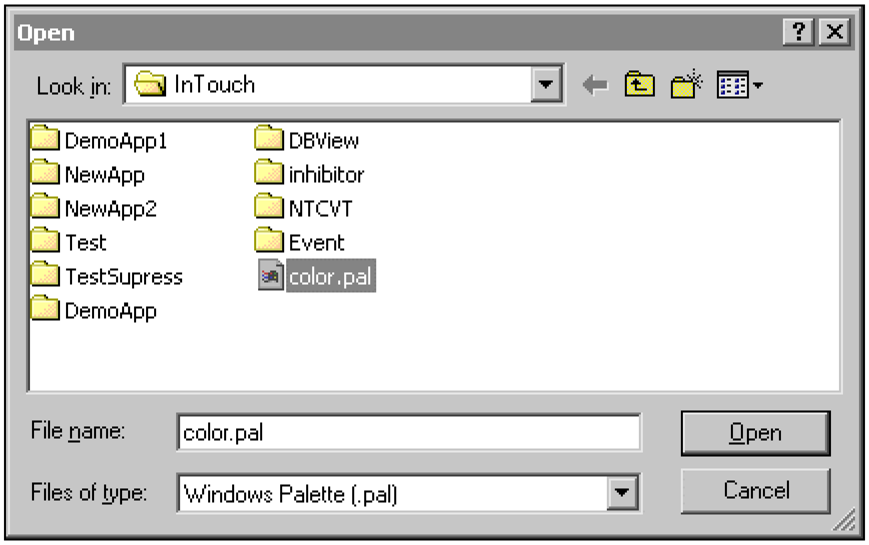
For more information on creating transparent bitmaps, see Chapter 2, "Using WindowMaker."

To import a custom palette

1. Open the color palette.
2. Click the **Custom Palette** arrow. The following menu appears.



3. Click **Load Palette**. The standard Windows **Open** dialog box appears.



4. Locate and select the palette (.PAL) file then, click **Open** or double-click the file name. The colors contained in your palette will be loaded into the **Custom Palette** section of the color palette.

To export a custom palette

1. Open the color palette.
2. Click the **Custom Palette** arrow then, click **Export Palette** on the menu (shown above).
3. The standard Windows **Save As** dialog box appears. Specify the name that you want to save the palette as then, click **Save**.

Tip The palette must be saved with the .PAL extension.

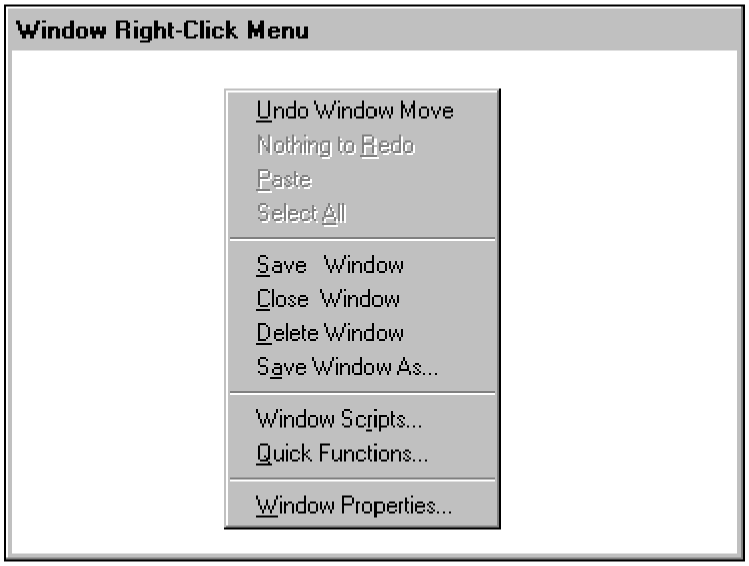
Popup Menus

InTouch supports right-click mouse functionality to display menus for frequently used commands for windows and graphic objects. Popup menus containing the commands that can be applied to selected text in dialog box, text box, an animation link tagname or expression box or, in a QuickScript window is also supported. Instead of using the standard menus to find the command you want to use, you can simply right-click the window, object, dialog box, text box, or the groups and their members in the Application Explorer.

Tip To turn off the right-click functionality, add the line **oldrightmousebehavior=1** to your INTOUCH.INI file.

To access the window right-click menu

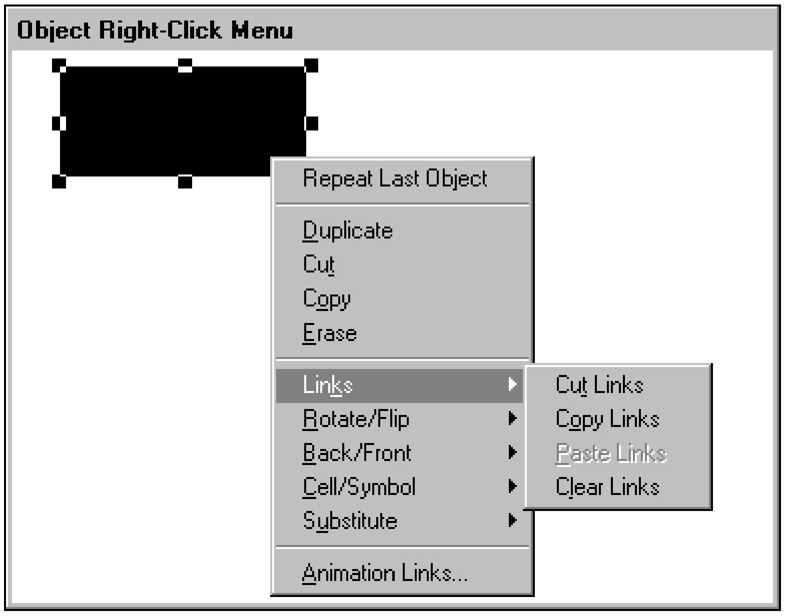
1. Right-click a blank area of a window. The following menu appears.



2. Click the command that you want to use on the menu. (Commands that are not applicable for the current state of the window will not be active.)

To access the graphic object right-click menu

1. Right-click an object in the window. The following menu appears.

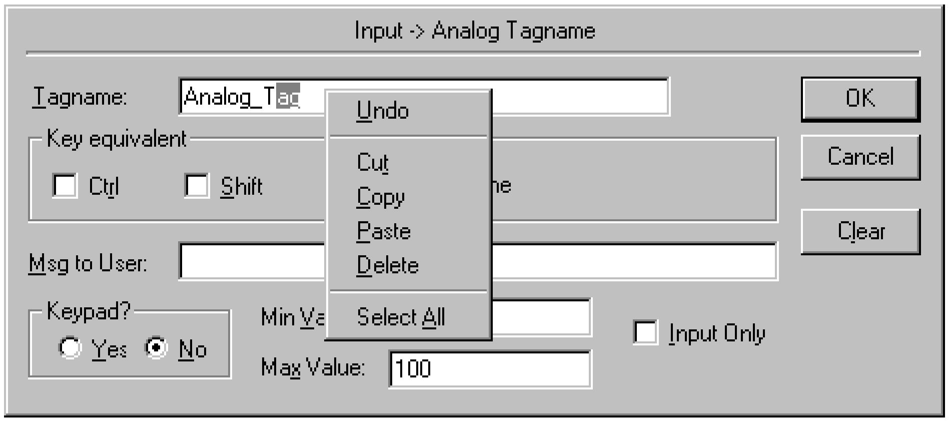


2. Click the command that you want to use on the menu. (Commands that are not applicable for the current state of the object will not be active.)

Tip If a "submenu" exists for the command, an arrow will be displayed. To select a command in a submenu, point at the command in the initial right-click menu then, click the command that you want to apply to the object in the submenu.

To access the dialog box text right-click menu

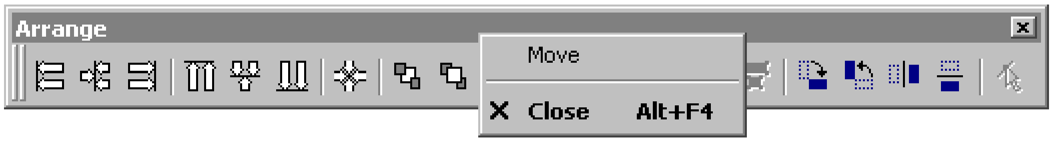
1. In any WindowMaker dialog box, right-click a text box. The following menu appears.



2. Click the command that you want to apply to the selected text. (Commands that are not applicable for the current state of the text will not be active. The **Select All** command will become active when partial or no text is selected.)

To access a floating toolbar right-click menu

1. Right-click the floating toolbar's title bar. The following menu appears.

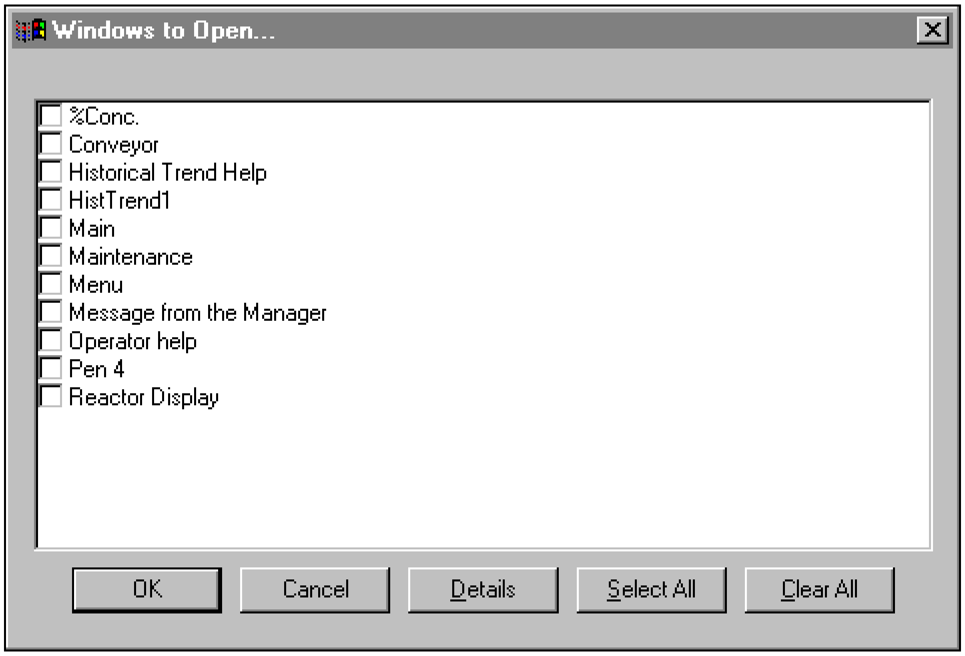


2. Click the command on the menu that you want to apply to the toolbar.

Common Window Dialog Box Features

When you are opening, saving, closing, deleting or duplicating a window(s) using the commands on the **File** menu in WindowMaker, the dialog boxes that you will use are very similar and have many common features. To avoid redundancy in the procedures describing how you perform these actions, the common features of those dialog boxes are described in this section.

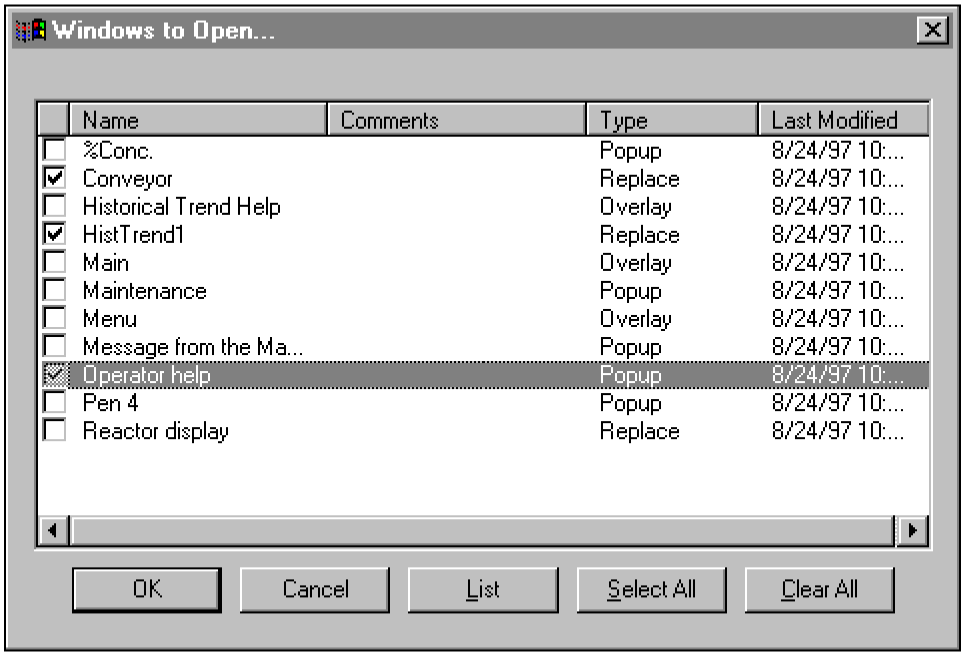
When you right-click on a blank area of an open window, or you click the open, save, close, delete or save as window commands on the **File** menu, by default, the respective dialog box for the command you selected appears in the "list view." Meaning that the names of all the windows that are applicable for the selected command appear in a continuous list. For example:



Tip A horizontal scroll bar appears when the number of window names exceeds the default list space.

Click **Details** to change from the "list view" to the details view.

When you select the details view, the windows and their details are displayed in a multi-column format. The details displayed include any comments you entered for the window in the **Window Properties** dialog box, the window's type, the date and time it was last modified. For example:


Windows to Open...

	Name	Comments	Type	Last Modified
☐	%Conc.		Popup	8/24/97 10:...
☑	Conveyor		Replace	8/24/97 10:...
☐	Historical Trend Help		Overlay	8/24/97 10:...
☑	HistTrend1		Replace	8/24/97 10:...
☐	Main		Overlay	8/24/97 10:...
☐	Maintenance		Popup	8/24/97 10:...
☐	Menu		Overlay	8/24/97 10:...
☐	Message from the Ma...		Popup	8/24/97 10:...
☑	Operator help		Popup	8/24/97 10:...
☐	Pen 4		Popup	8/24/97 10:...
☐	Reactor display		Replace	8/24/97 10:...

In the details view, you can select any unopened window by clicking on any portion of its row, not just the check box. (The entire row will be highlighted.) You can click on a selected window a second time, to deselect it.) A vertical scroll bar will also appear when the number of window names exceeds the default list space.

To sort the list by a detail type, click the column header for that detail. The details view sort sequences:

Column	Sort Sequence
Name	Alphabetically
Comments	Alphabetically
Type	Overlay, Replace then Popup
Last Modified	Earliest date/time (top) to latest date/time (bottom)

Tip Each time you click a column header, the list sort order will toggle from ascending to descending. For example, if the list is currently sorting in ascending order and you click a column header, the list will be resorted in descending order for the column selected.

To return the list to the default display, click the small box on the far left side of the column header.

To size the columns, place the cursor over the vertical lines that separate each detail header. When the cursor changes to an "I" bar, click and drag the header to the width you want for the column.

To quickly auto-size a column, double-click on the column's right vertical line separator.

To open selected window(s) click **OK**.

To cancel your selections and close the dialog box, click **Cancel**.

To return the dialog box to "list view," click List.

To select all listed windows, click **Select All**.

To clear all selected windows, click **Clear All**.

Miscellaneous Mouse Short Cuts

Double-clicking an object or symbol automatically executes the **Animation Links** command (on the **Special** menu) with the object or symbol selected.

For more information on animation links, see Chapter 7, "Creating Animation Links."

Double-clicking a blank expression input field within a link definition dialog box launches the Tag Browser listing all tagnames defined in the application's Tagname Dictionary.

Double-clicking after a period (**.**) in an expression input field on a link definition dialog box will display the **Choose field name** dialog box containing a global listing of all the tagname **.fields**.

Double-clicking a tagname in an animation link tagname or expression opens that tagname's definition in the Tagname Dictionary.

Double-clicking a SuperTag parent template name in an animation tagname or expression opens the Tagname Dictionary details dialog box for that SuperTags member tagnames.

For more information on the Tag Browser and tagname .fields, see Chapter 6, "Tagname Dictionary."

Right-clicking on a blank area of an open window, a text box in any WindowMaker dialog box or, on a graphic object will display a menu with the commands that you can apply to the window, text or object. Right-clicking the title bar of a floating toolbar will also display a menu of commands that you can apply to the toolbar.

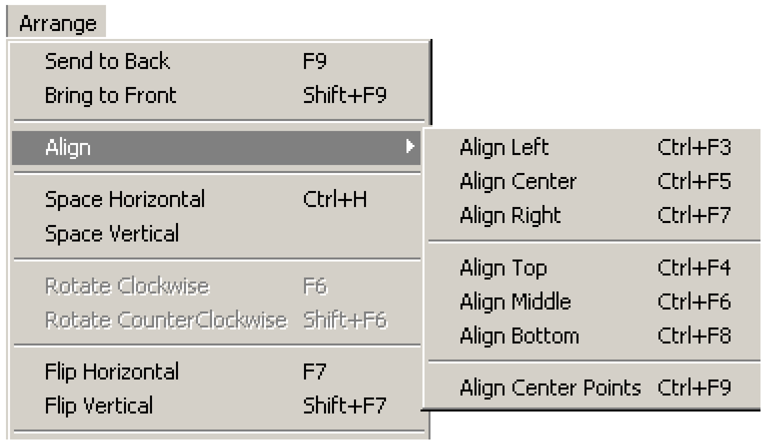
Short Cuts and Accelerators

InTouch provides many mouse and keyboard shortcuts for frequently used functions. Whenever a menu command has a keyboard shortcut, it is displayed on the menu to the right of the command. In addition, all commands may be executed with a three-key sequence beginning with the ALT key. The second key is the underlined character in the menu name, and the third key is the underlined character in the command.

For example, you can execute the New Window command on the File menu by using the sequence alt + fn.

To execute a command on a right-click menu, type the underlined character in the command.

To execute a command on a submenu, you must press a three key sequence. For example, to execute the Align Center command on the Align submenu you would press alt + aac, or ctrl+F5



Note When you select a menu command that is followed by an ellipsis (**...**), a dialog box appears, and you must select or enter more information in order to complete the command.

Moving Objects with the Arrow Keys

In WindowMaker, you can use the up, down, left and right arrow keys to move a selected object or group of objects one pixel at a time in the direction of the arrow key being pressed. This feature can be very convenient when you need to make fine alignment and location adjustments. To move the object 10 pixels at a time, hold down the SHIFT key while pressing the arrow key. To move the object 50 pixels at a time, hold down the CTRL key while pressing the arrow key.

In WindowViewer, when you use the arrow keys, an algorithm is used to move from one touch-sensitive object to another. For example, the left cursor arrow key cuts a path to the left as wide as the height of the selected object. If it intersects any part of another touch-sensitive object within its path, that object takes the focus as the currently selected object. If no other touch sensitive object is encountered, the path wraps around to the left edge of the screen and continues to try to intersect another object. If no other object is encountered, the original object remains selected.

All arrow keys function in the same manner. The up and down arrow keys use the width of the selected object for the width of their paths.

Tip If you know that the user of your application will only use cursor keys to navigate in the windows in your application, you should carefully plan the placement of the touch-sensitive objects in the window to ensure that they have intersecting paths.

In WindowViewer, the TAB key can also be used to transition between the touch-sensitive objects in a window. (However, the tabbing order cannot be guaranteed.)

By careful application design, and understanding how the arrow keys react, you can create applications that can be used without a mouse.

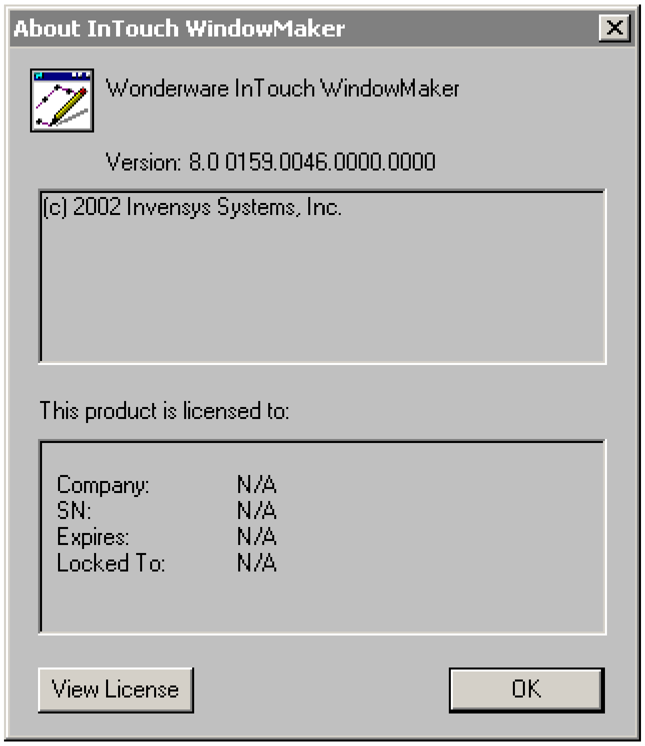
Using WindowMaker Help

WindowMaker supports context-sensitive help as follows:

For help on an open dialog box, press the F1 key. The respective Windows online help topic appears.

For help on the various QuickScript functions, in the QuickScript editor, click **Help** or, on the **Insert** menu, point to **Functions** then, click **Help**. A dialog box appears listing all available QuickScript functions. Click the function that you want help for. The function's respective Windows online help topic appears.

To obtain information about your InTouch software, for example, the Version you are using, your Serial Number and your License expiration date and so on, on the WindowMaker **Help** menu, click **About**. The **About InTouch** dialog box appears.



Wonderware InTouch WindowMaker

Version: 8.0 0159.0046.0000.0000

(c) 2002 Invensys Systems, Inc.

This product is licensed to:

Company:	N/A
SN:	N/A
Expires:	N/A
Locked To:	N/A

Click **View License** to launch the license viewing utility to obtain information regarding your FactorySuite license.

For more information on the License Viewer Utility, see your *FactorySuite System Administrator's Guide*.

C H A P T E R 2

Using WindowMaker

By setting various properties for WindowMaker and WindowViewer, you can customize the functionality and final appearance of your application. For example, you can specify what menus you want available in WindowViewer, you can include company names in the title bar of your application, and so on.

This chapter describes how to configure WindowMaker and WindowViewer, work with WindowMaker's windows, edit and arrange graphic objects, and how to install and use wizards and ActiveX controls.

Contents

Simple Objects

WindowMaker has four basic types of simple objects; lines, filled shapes, text and buttons. Each of these simple object types have attributes that affect their appearance. These attributes include line color, fill color, height, width, orientation, etc. and can be either static or dynamic. A static attribute remains unchanged during the operation of the application. A dynamic attribute is linked to the value of an expression such that a change in the value of the expression results in a change in the attribute. For example, the fill color of an object can be linked to the value of a discrete expression. Based on the state of the expression, the fill would be one color when the expression is true and another color when it is false. Most of the attributes of simple objects can be made dynamic. An object may have more than one dynamic attribute. Dynamic attributes may be combined freely to achieve the desired result. The following briefly describes WindowMaker's simple object types:

Object	Description
Line	A line object is made up of one or more line segments depending on the type of line. Color is the only attribute of a line that you can animate. Width and style cannot be animated. They are simply assigned default attributes.
Filled Shapes	Filled shapes are two dimensional objects made up of a closed interior area surrounded by a line. Examples of filled shapes are rectangles, rounded rectangles, circles, ellipses and polygons. The attributes of a filled shape are: line color, line width, line style, fill color, percent color fill, height, width, location, visibility, orientation and size.
Text	Text is an object made up of a string of characters on a single line. The attributes of a text object are: font, size, color, bold, underline, *italic,* justification, visibility and location.
Buttons	The 3-dimensional buttons can be created for any desired size by using the Button tool on the WindowMaker Draw Object Toolbar. The default "text" string on the button face is edited by using the Substitute Strings command on the Special menu. Many types of links can be attached to buttons such as action scripts, key scripts, analog or discrete value input or output links. If an input or output link is attached to a button the value is displayed on the button as the text string.

Complex Objects

In addition to simple objects, InTouch also supports complex objects which are considerably different. The following briefly describes WindowMaker's complex object types:

Object	Description
Bitmap	The Bitmap tool is used to copy and paste bitmaps into your application. Once pasted in an application window, a bitmap can be rotated and, it can be defined with a transparent background so that it can float over other objects.
Trends	There are two trend tools: one is for creating trends that display real-time data and the second is for creating trends that display historical data (retrieved from historical log files). Both the real-time and the historical trends can be configured to display graphical representations of multiple tagnames over time.
Symbols	A symbol is a combination of simple objects (lines, filled shapes, and text) which is treated as a single object. Any attribute change applied to a symbol, whether it is a change in a static attribute in WindowMaker, or a change in a dynamic attribute in WindowViewer, will affect all the component objects of a symbol. For example, if you create a pump symbol from two circles and two rectangles, and then you define a fill Color Link on the symbol, the fill color will apply to all four objects. Similarly, in WindowMaker, a Fill Color selection would also change the fill color of all the component objects. The component objects of a symbol can have different values for the same attribute if these attributes were different before the objects were combined into the symbol and were not changed after they became a symbol. Symbols cannot contain bitmaps, buttons, cells, alarms or trends.

Object	Description
Cells	A cell is a collection of two or more objects, symbols, or other cells that are joined together to form a single unit. Cells maintain a fixed spatial relationship between their individual graphic elements. Each component of a cell can have its own links. Cells are used to create virtual devices such as a slide controller. A cell is created by selecting two or more objects, symbols, and/or cells and then executing the **Make Cell** command on the **Arrange** menu. Once a set of objects is made into a cell, its interior details such as colors, sizes, animation links, and so forth, cannot be changed. The only way to change a cell's appearance or operation is by "breaking" it apart using the **Break Cell** command on the **Arrange** menu. The attributes of the components of a cell are changed in WindowViewer by the operation of links. Cells can be duplicated, copied, pasted, aligned, spaced, and so on. Cells cannot be sized. They must be broken apart, sized and made back into a cell. Cells are very useful in creating multiple similar devices that are associated with different tagnames.

Object	Description
Wizards	Wizards save you a considerable amount of time during application development. They are easy to use and easy to configure. When you select a wizard and paste it into a window and then double-click it, its configuration dialog box appears and you can configure it. For example, in the case of a "slider" wizard, the configuration would include items such as the tagname to effect, the minimum and maximum range labels for the slider, the fill color, and so on. Once the required configuration information is entered, the Wizard is ready to use. By 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. "Complex" Wizards can be developed to provide "behind the scenes" types of operations. These operations may include creating complete display windows, creating or converting a database, importing an AutoCad drawing, and configuring other applications such as, the InTouch Recipe Manager and SPC add-on programs. **Note** A proficient "C" programmer can develop custom Wizards by using Wonderware's Extensibility Toolkit. This toolkit is available through your distributor. For more information on Wizards, see "Working with Wizards."

Object	Description
ActiveX Controls	WindowMaker supports ActiveX controls which, in their simplest form, are mini-applications that talk to or run within your application. WindowMaker supports all ActiveX controls that are included in Wonderware FactorySuite components, for example, all InTrack ActiveX controls. WindowMaker also supports third-party ActiveX controls such as those installed with Office97. You install the InTouch ActiveX controls just like any other wizard. If desired, you can add frequently used ActiveX controls and thento your WindowMaker **Wizards/ActiveX Toolbar**. When 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 specify a unique control name for it in order to reference it in a script (a default name is created when you initially add the control). All ActiveX controls have properties, methods and events associated with them. You can associate an ActiveX property with a tagname of a corresponding data type. You can execute ActiveX methods through InTouch QuickScript functions. You can associate an ActiveX event with an ActiveX Event Script that executes when the event occurs. In other words, you can use InTouch QuickScript functions to handle control events, call control methods, and control properties. In runtime, the tagnames and QuickScripts you defined in WindowMaker control the behavior of your ActiveX controls.

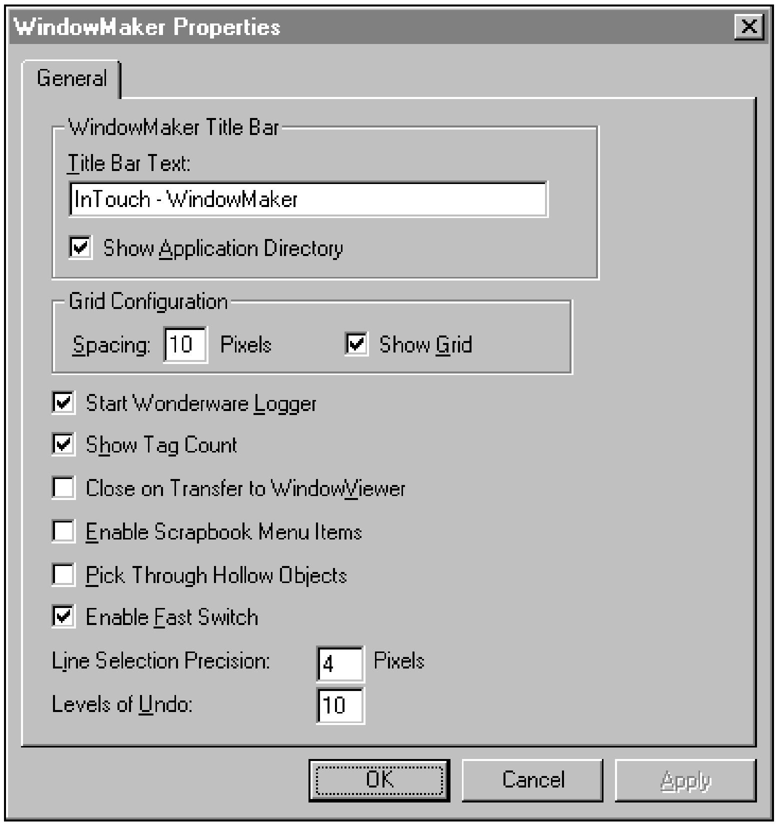
Customizing Your Development Environment

There are many properties that you can set to customize WindowMaker. For example, you can customize your application's title bar text to include the company name. You can set the pixel spacing for the grid and so on.

To set the properties for WindowMaker

1. On the **Special** menu, point to **Configure** and then click **WindowMaker** or in the Application Explorer under **Configure**, double-click **WindowMaker**. The **WindowMaker Properties** dialog box appears with the **General** properties sheet active.

Tip In the Application Explorer under **Configure**, you can also right-click **WindowMaker** and then click **Open**.


WindowMaker Properties

General

WindowMaker Title Bar

Title Bar Text: InTouch - WindowMaker

[✓] Show Application Directory

Grid Configuration

Spacing: 10 Pixels [✓] Show Grid

[✓] Start Wonderware Logger

[✓] Show Tag Count

[ ] Close on Transfer to WindowViewer

[ ] Enable Scrapbook Menu Items

[ ] Pick Through Hollow Objects

[✓] Enable Fast Switch

Line Selection Precision: 4 Pixels

Levels of Undo: 10

Tip If you right-click any of the text entry boxes in any dialog box, a menu appears displaying the commands that you can apply to the selected text.

2. In the **Title Bar Text** box, type the title that you want to appear in your application's title bar in runtime. For example:

   ABC Company, Paint APP1

   Note You cannot change the title bar if you are using a "Promotional License."

   For more information on FactorySuite licensing, see your *FactorySuite System Administrator's Guide*.

3. **Show Application Directory** if you want to include the path to the application's directory in the title bar. For example:

   ABC Company, Paint APP1 - C:\DEMOAPP1

4. In the **Spacing** box, type the number of pixels that you want spaced between the snap to grid's coordinates.

5. Select **Show Grid** if you want a visible grid in your windows when you turn on WindowMaker's "snap to grid" functionality.

Tip If you do not select **Show Grid**, no grid will be visible in your windows when you turn snap to grid on.

Click the Snap to Grid tool on the **View** toolbar or on the **Arrange** menu, click **Snap to Grid** to turn the snap to grid functionality on and off.

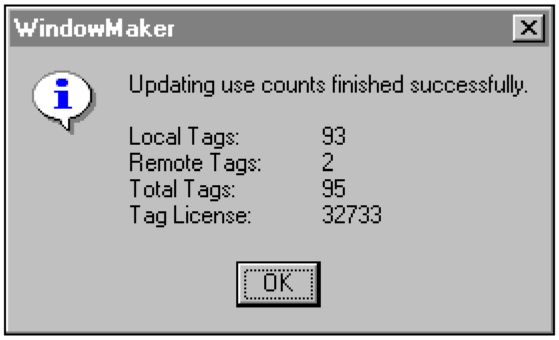
Objects are snapped to the grid by their upper left corner. If you select multiple objects, the snapping will be applied to the upper left corner of the first object selected in the group.

6. Select **Show Tag Count**, if you want the number of tagnames defined in your Tagname Dictionary to be displayed in the WindowMaker menu bar.

Tip This can be very useful information if you are creating an application with a limited Tagname Dictionary size.

Note Showing the tagname count will seriously impact performance of the Tagname Dictionary in WindowMaker.

The displayed tagname count does not include remote tagname references. To determine your remote tagname reference usage, execute the **Update Use Counts** command on the **Special** menu. Once the system has completed updating the use counts, the following dialog box appears.



```
WindowMaker

Updating use counts finished successfully.

Local Tags:      93
Remote Tags:     2
Total Tags:      95
Tag License:     32733
```

For more information on updating use counts, see Chapter 6, "Tagname Dictionary."

7. Select **Close on Transfer to WindowViewer**, if you want WindowMaker to automatically close when you start WindowViewer.

Note If memory is not an issue and you are moving often between WindowViewer and WindowMaker, this option should not be selected.

Tip When you select this option, the **Close WindowViewer** option on the **General** properties sheet in the **WindowViewer Properties** dialog box is automatically selected too.

8. Select **Enable Scrapbook Menu Items** if you are copying and pasting objects between WindowMaker and Scrapbook+.

   Note The Windows Scrapbook+ program must be installed on your computer in order to use these menu commands. This option is used by legacy InTouch systems. It is not supported for FactorySuite nor is Scrapbook+ sold through Wonderware.

9. Select **Pick Through Hollow Objects**, if you want to be able to select objects that are behind "hollow" objects.

   Tip If you select this option and then draw four lines and join them to make a frame around another object, you can select the object within the frame without having to send the frame to the back.

10. Select **Enable Fast Switch**, if you want the fast switch between WindowMaker and WindowViewer to be displayed in your menu bar for both programs.

    Tip If you select this option, in WindowMaker, the fast switch is the word **Runtime** displayed in the upper right hand of your screen. In WindowViewer, it is the word **Development**. To quickly switch between the two programs, click their fast switch.

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

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

    Tip In most cases, the default setting of 4 should be sufficient.

12. In the **Levels of Undo** box, type the number of undo/redo levels that you want saved. You can have up to 25 levels. If you type zero, the undo/redo functionality 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 save the window.

    For more information on undo and redo, see "Undoing Object Edits."

13. Click **OK**.

Note After you modify any of these settings, you must restart WindowMaker to apply your changes.

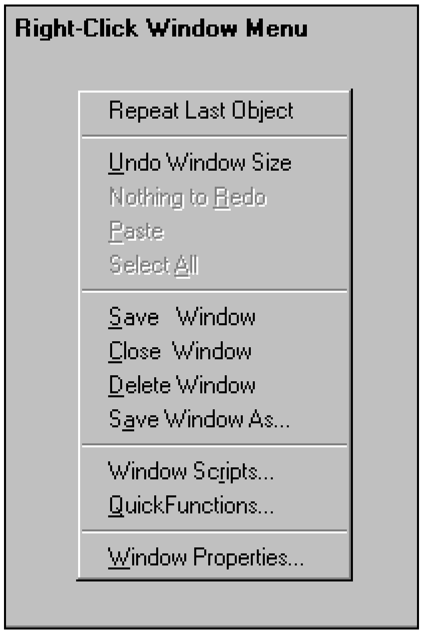
Working with WindowMaker Windows

Your InTouch application will more than likely be comprised of numerous windows that you create to hold your graphics and text objects. When you create a new window in WindowMaker, you will be required to define certain properties for that window such as, its background color, title, screen position and so on. You can also create QuickScripts that execute based upon whether the window is opening, showing or closing.

This section contains the procedures that you will follow to create, open, save, close, delete, and duplicate windows.

Tip The **General Toolbar** contains tools that you can use to quickly apply most of the windows commands on the **File** menu.

To quickly access the various commands that can be applied to a window, right-click a blank area of the open window and then click the appropriate command on the right-click menu. For example:

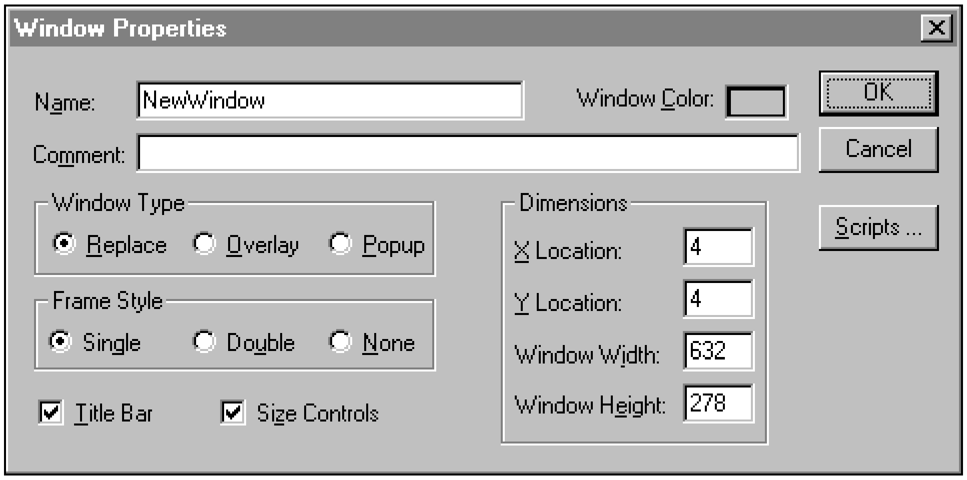


Creating a New Window

To create a new window

1. On the **File** menu, click **New Window** or clickor select the **New Window** tool on the **General Toolbar**. The **Window Properties** dialog box appears.

Tip To quickly create a new winor clickdow, in the Application Explorer, right-click **Windows** and then click **New**.



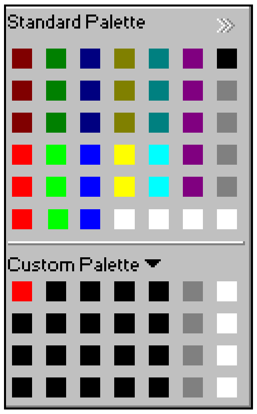
Name: NewWindow
Window Color:
Comment:
Window Type: [✓ Replace] Overlay Popup
Frame Style: [✓ Single] Double None
[✓ Title Bar] [✓ Size Controls]
Dimensions
X Location: 4
Y Location: 4
Window Width: 632
Window Height: 278

Note By default, the settings in this dialog box will reflect those of the previously created window or if you select this command while a window is open in WindowMaker, the settings will reflect those of the active window. If a Window script(s) is attached to the active window, a message box appears asking you if you want the window script(s) copied to the new window.

Tip If you right-click any of the text entry boxes in any dialog box, a menu appears displaying the commands that you can apply to the selected text.

2. In the **Name** box, type the name that you want to appear in the new window's title bar. The name can be up to 32 characters long. It can include embedded spaces, punctuation marks, and any other character on the keyboard except, quotation marks (").
3. In the **Comment** box type any miscellaneous comments that you want associated with the window (optional). This information is for documentation purposes only and is not used by the application.

4. Click the **Window Color** box to select the background color for the window. The color palette appears:



Tip If no change is desired, click on the current color selection or press the ESC key to close the palette.

For more information on using the color palette, see Chapter 1, "WindowMaker Program Elements."

5. Click the color that you want to use for the window's background.
6. Select the **Window Type** that you want to use. There are three window types:

Window	Description
Replace	Automatically closes any window(s) it intersects when it appears on the screen including popup and other replace type windows.
Overlay	Appears on top of currently displayed window(s) and can be larger than the window(s) it is overlaying. When an overlay window is closed, any window(s) that were hidden behind it reappear. Clicking on any visible portion of a window behind an overlay window will bring that window to the foreground as the active window.
Popup	Similar to an overlay window, except it always stays on top of all other open windows (even if another window is clicked). Popup windows usually require a response from the user in order to be removed.

Tip You can change a window's type by opening its **Window Properties** dialog box again. There are three ways to do this:

Open the window and then on the **Windows** menu, click **Window Properties**.

In the Application Explorer under **Windows**, right-click the window name and then click **Properties**. (If the window is not open when you execute the command, it is automatically opened behind the dialog box.)

Open the window, right-click a blank area of the window and then click **Window Properties**.

7. Select the **Frame Style** for the window. There are three styles:

Frame	Description
Single	3D bordered window that can have a title bar and **Size Controls**.
Double	3D bordered window that has no title bar and cannot be sized without Size Controls.
None	A window without a border that cannot be sized without Size Controls. (With Size Controls it becomes a 3D bordered window that can be sized.)

8. Select **Title Bar** if you want the window to have a title bar. The title bar is also used to move the window by clicking the mouse on it and then dragging the mouse.

Note If your window has a title bar, you cannot select **Double** or **None** for the **Frame Style**.

9. Select **Size Controls** if you want the user to be able to resize the window in WindowMaker.
10. In the **Dimensions** group, type the pixel location that you want for each of the window's coordinates:

Location	Description
X Location	The number of pixels between the left edge of the WindowMaker design area and the left edge of the window being defined.
Y Location	The number of pixels between the top edge of the WindowMaker design area and the top edge of the window being defined.
Window Width	The window's width in pixels.
Window Height	The window's height in pixels.

Note Windows limits the minimum height of a window according to your display monitor. For example, for standard VGA, the minimum is 26 pixels. Windows limits the minimum width of a window according to your display monitor. For example, for the standard VGA, the minimum is 102 pixels.

Note By default, the values in these boxes will be set to the dimensions of the previously created window. They are also automatically modified if you manually change the windows size in WindowMaker by using the window's border.

11. Click **Scripts** to access the **Window Script** editor. There are three types of scripts that you can apply to a window:

Script	Description
On Show	Executes one time when the window is initially shown.
While Showing	Executes continuously at the specified frequency while the window is showing.
On Hide	Executes one time when the window is hidden.

Note If you attach a Window Script to the active window and then you create a new window, the script(s) from the active window can be copied to the new window. A message dialog box appears asking if you want to copy the window script(s).

Tip If you later decide that you need to attach a script to an open window, right-click a blank area of the open window and then click **Window Scripts**. If the window is not open, in the Application Explorer, double-click **Windows** to show all window names. Right-click the window name and then click **Window Scripts**.

For more information on creating window scripts, see Chapter 8, "Creating QuickScripts in InTouch."

Creating a Window to Hide the Title and Menu Bars

The WindowMaker design area is the entire area below the title and menu bars and within the window frame. The design area becomes the display area in WindowViewer. The specific location X=0 and Y=0 is always the upper left corner just below the title and menu Bars. The title and menu bars are each 19 pixels high and are above the design area. For example, if WindowMaker is maximized, and you are using a 1024x768 video display, the visible design area would 1024x730 (768 less 19 pixels for the title and 19 pixels for the menu bar equals 730 pixels in the visible design area). If WindowViewer is configured to show its title bar and menu bar, the display area in WindowViewer will fill the screen below the title bar and menu bar exactly as seen in WindowMaker.

To take advantage of the additional space used by the title and menu bars, you can design an application with the title bar and menu bars hidden. When the title bar and menu bar are hidden, the upper left corner of the window references a new position on the screen. This increases the visible display area and provides you with more display area. If you configure WindowViewer in this manner, all of your windows will automatically appear to move up, and a gap appears at the bottom of the window. To fill this gap, you need to increase the window height by setting the Y location of the window to a negative value. This places the window under the title bar and menu bar in WindowMaker, and on top of the them in WindowViewer.

You can use this technique with a popup window to cover the title bar and menu bar in WindowViewer. You can also create a touch link push button or QuickScript to hide this popup window whenever you need to reveal the title bar and menu bar to the user. In addition, by applying security and a password, you can restrict certain users from hiding the window and gaining access to the menus which includes the ability to exit WindowViewer.

Note A Promotional InTouch license does not allow the title bar to be hidden. Therefore, if an application is developed under a Promotional InTouch license and WindowViewer is configured with the **Hide Title Bar** option selected when that application is viewed on a standard runtime license, the title bar will be hidden as configured and, all the windows in the application will move up.

Opening Windows

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

To open a window(s)

1. On the **File** menu, click **Open Window** or click the Open Window tool on the **General Toolbar**. The **Windows to Open** dialog box appears listing the names of all windows in your application.

   Tip To quickly open a single window, in the Application Explorer, double-click **Windows** to open the list of all the window names in your application and then double-click the window name. You can also right-click the window name and then click **Open**.

2. Click the check box next to the name of the window(s) that you want to open.

   Tip By default, all currently opened windows will already be checked.

3. Click **OK** to close the dialog box and open the selected window(s).

Saving Windows

Once you have created a window, you will need to save it before you can close it or exit your application. All graphics, QuickScripts, properties, and so on associated with the window are also saved.

To save a window(s)

1. On the **File** menu, click **Save Window** or click the Save Window tool on the **General Toolbar**. The **Windows to Save** dialog box appears listing the names of all windows that need to be saved.

Tip To quickly save a single window, in the Application Explorer, right-click the window name and then click **Save**. You can also right-click any blank area of the window and then click **Save Window**.

To quickly save all currently open windows, click the **Save All** Windows tool on the **General Toolbar**, or the **Save All** command on the **File** menu.

2. Click the check box next to the name of window(s) that you want to save.
3. Click **OK** to close the dialog box and save the selected window(s).

Closing Windows

If you attempt to close a window(s) that has been modified since it was last saved, you will be prompted to save your changes before WindowMaker will close the window.

To close a window(s)

1. On the **File** menu, click **Close Window** or click the Close Window tool on the **General Toolbar**. The **Windows to Close** dialog box appears listing the names of all currently open windows.

Tip To quickly close a single window, in the Application Explorer, right-click the window name and then click **Close**. You can also right-click any blank area of the window and then click **Close Window**.

2. Click the check box next to the name of the window(s) that you want to close.
3. Click **OK** to close the dialog box and close the selected window(s).

Deleting Windows

To delete a window(s)

1. On the **File** menu, click **Delete Window**. The **Windows to Delete** dialog box appears listing the names of all currently open windows.

Tip To quickly delete a single window, in the Application Explorer, right-click the window name and then click **Delete**. You can also right-click any blank area of the window and then click **Delete Window**.

2. Click the check box next to the name of the window(s) that you want to delete.
3. Click **OK** to close the dialog box and delete the selected window(s).

Note Deleted windows cannot be restored unless you have backed them up. You will be prompted to confirm the deletion of each window name you select.

Duplicating a Window

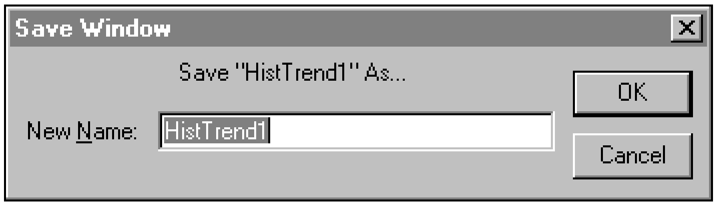
When you want to create a duplicate copy of an existing window, the window that you want to duplicate must be open.

To duplicate a window

1. On the **File** menu, click **Save Window As**. The **Window to save under new name** dialog box listing the names of all currently open windows appears.

Tip To quickly duplicate a window, in the Application Explorer, right-click the window name and then click **Save As**. You can also right-click in any blank area of the window and then click **Save Window As**.

2. Click the check box next to the name of window that you want to duplicate. (Only one window name can be selected.) The **Save "*Window Name*" As** dialog box appears.



3. In the **New Name** box, type a valid name for the new window.
4. Click **OK** to close the dialog box and create the duplicate window.

Exporting Windows

Exporting windows is very useful when you need to create or maintain a library application or you need to quickly create remote tagname references in another application. To move windows from one InTouch application to another, you <u>must</u> use the **Export Window** command on the **File** menu.

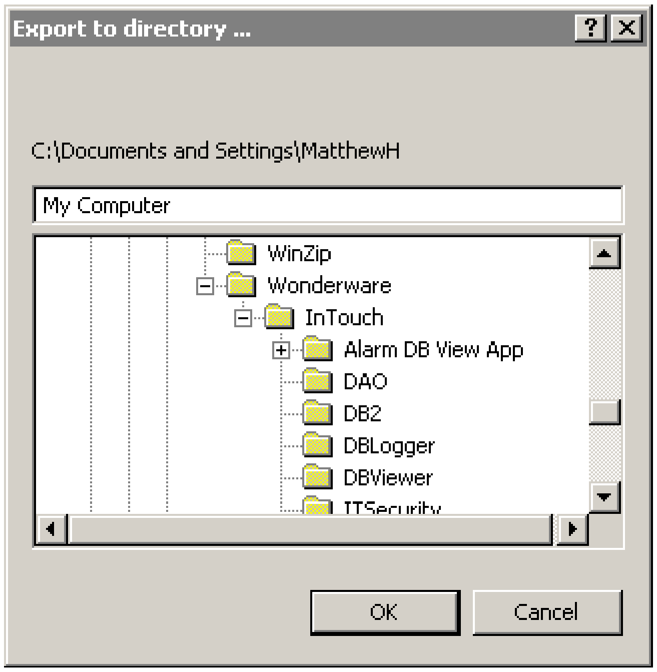
For more information on remote tagname references, see Chapter 6, "Tagname Dictionary."

Caution! If you attempt to copy InTouch window files by using any other copy methods, such as the File Manager or Windows Explorer copy commands, you may corrupt your application's Tagname Dictionary!

To export a window

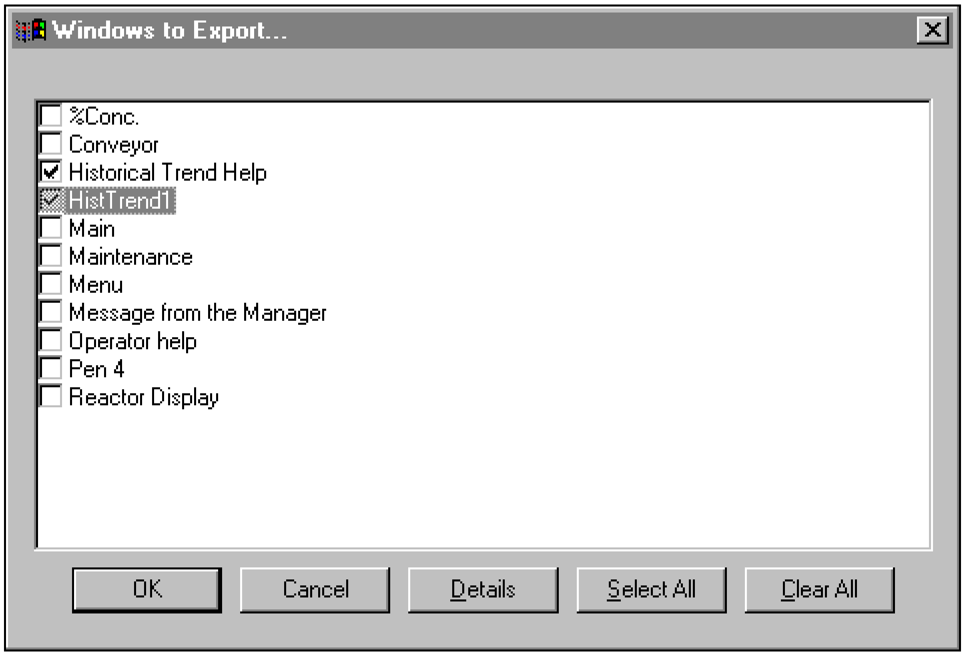
1. Close all windows in your current application.

2. On the **File** menu, click **Export Window**. The **Export to directory** dialog box appears.



3. Locate and select the application directory (folder) that you want to export the window(s) and then click **OK**.

4. The **Windows to Export** dialog box appears.



5. Select the window(s) that you want to export.

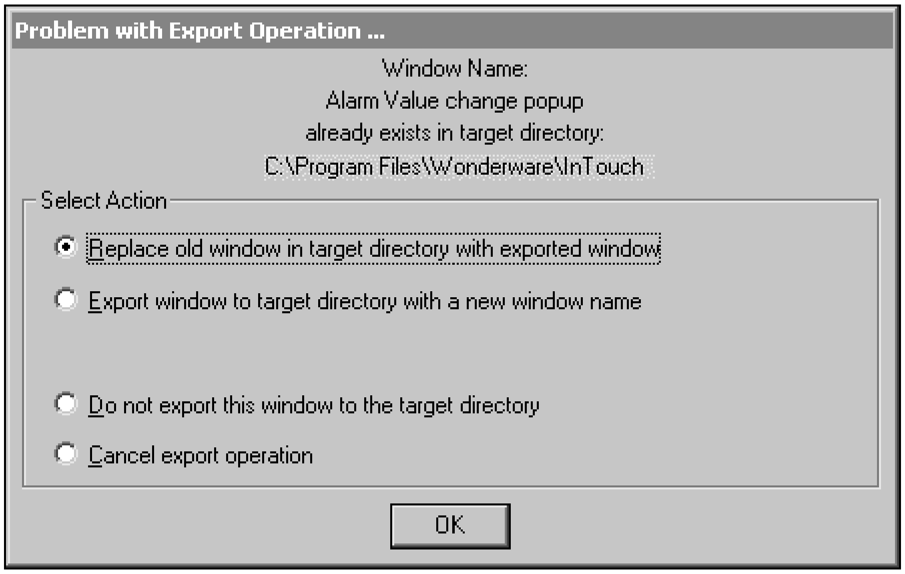
6. Click **OK**. The export operation will take place.

Note When you export a window, all of the objects and animation links associated with that window are exported with the window. However, the tagnames associated with the objects in the window are converted into "placeholder" tagnames. Placeholder tagnames are used to avoid any problems that might occur when the destination application's Tagname Dictionary does not contain the same tagnames.

For more information on converting placeholder tagnames, see Chapter 6, "Tagname Dictionary."

Problem with Export Operation

If a problem is encountered by the system when exporting the window, the **Problem with Export Operation** dialog box appears.



Problem with Export Operation ...

Window Name:
Alarm Value change popup
already exists in target directory:
C:\Program Files\Wonderware\InTouch

Select Action

● Replace old window in target directory with exported window
○ Export window to target directory with a new window name
○ Do not export this window to the target directory
○ Cancel export operation

In the **Select Action** group, select the action that you want to take and then click **OK**.

Importing Windows

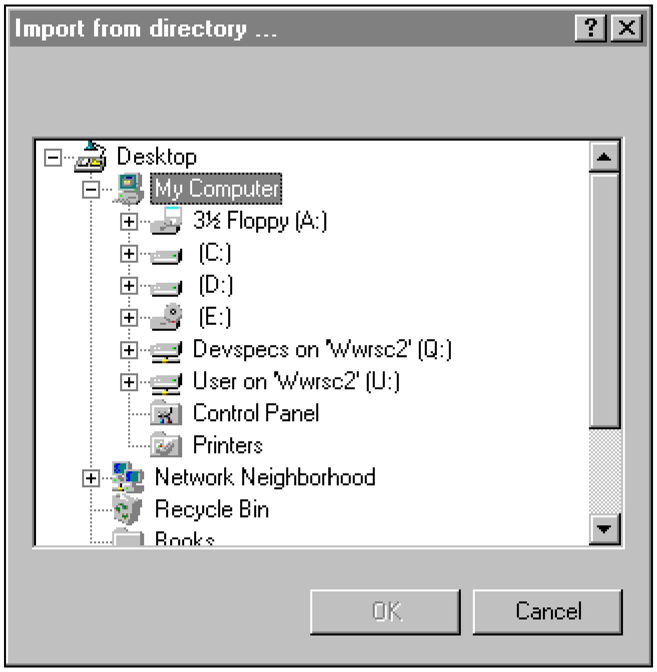
Importing windows from one InTouch application to your current application, can save you a considerable amount of development time. It also provides you with a quick and easy method for creating remote tagname references. It allows you to reuse your previously created windows, objects and window scripts. When you move windows from one InTouch application to another, you must use the **Import** command on the **File** menu.

For more information on remote tagname references, see Chapter 6, "Tagname Dictionary."

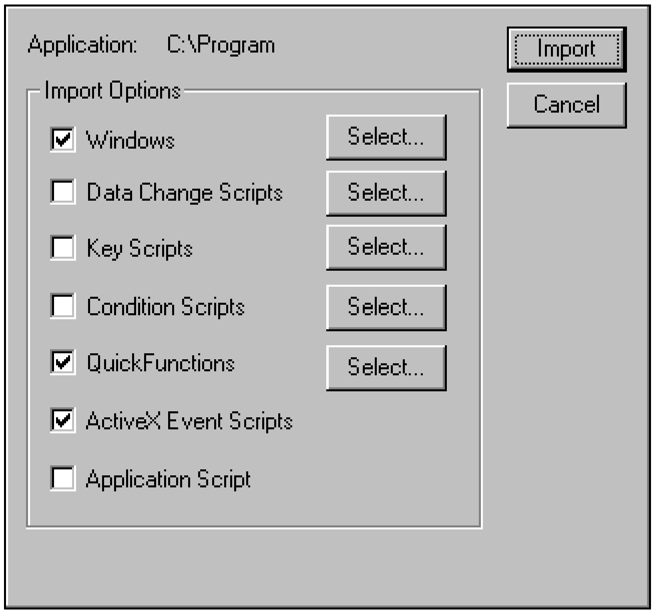
Note If you attempt to move InTouch window files by using any other move methods, such as the File Manager or Windows Explorer move commands, you may corrupt your application's Tagname Dictionary!

To import a window or QuickScript

1. Close all windows in your current application
2. On the **File** menu, click **Import**. The **Browse for Folder** dialog box appears.



3. Locate and select the application directory (folder) containing the windows that you want to import and then click **OK**.
4. The Import Options dialog box appears.


Application: C:\Program

Import Options
- [✓] Windows
- [ ] Data Change Scripts
- [ ] Key Scripts
- [ ] Condition Scripts
- [✓] QuickFunctions
- [✓] ActiveX Event Scripts
- [ ] Application Script

5. Select the item(s) that you want to import and then click **Select**. A dialog box appears for you to select the window or QuickScript(s) that you want to import.
6. Once you have selected the window(s) or QuickScript(s) that you want to import, click **Import**. The system will automatically begin to import the selected items into your current application.

Note To import a window script, you must import the entire window. When you import a window, all of the objects and links associated with that window are imported with the window. However, the tagnames associated with the objects in the window (and the tagnames used in an imported script) are converted into "placeholder" tagnames.

When the tagnames in an imported script or window are converted to placeholder tagnames, three index characters are added to the beginning of each tagname. For example, when a discrete tagname is imported, the tagname is prefixed with the three characters **?d:**. When a tagname of 30, 31 or 32 characters in length is imported, the three indexing characters will still be added to the beginning of each tagname. However, the addition of these three characters will not truncate the length of your existing tagname. For example, for placeholder tagnames only, a 32 character tagname is increased to 35 characters. These three additional spaces are allotted only for placeholder tagnames. This increase in tagname length is not supported for standard tagnames.

For more information on converting placeholder tagnames, see Chapter 6, "Tagname Dictionary."

Note When you import a window from an application that contains SuperTags, only the SuperTag instances actually used in the imported window are imported into the new application. The entire SuperTag template structure is not imported. For example, if the application has hundreds of SuperTag member tagnames defined in it, and only 50 of those are used in the imported window, only those 50 are imported.

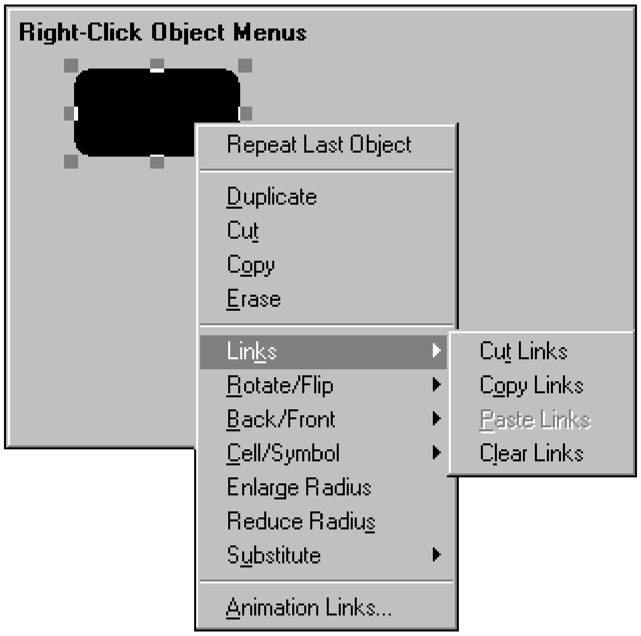
Working with Graphic Objects

Once you have created a new window in your application, you are ready to populate it with graphic objects. WindowMaker provides you with numerous tools for editing and arranging the various graphic objects that you will draw and paste into your windows. This section describes the procedures you will use to perform various editing functions on your graphic objects.

Object	Description
Draw Object Toolbar	Contains the graphic drawing tools that you will use to create graphics in your windows.
View Toolbar	Contains the Ruler tool that you can use to display a ruler for assisting you in alignment of your graphic objects in your windows. This toolbar also contains the tools that you will use to hide or show the Application Explorer, the toolbars, or a visible grid in your windows. It also contains the tool for switching to and from full screen mode.
General Toolbar **Format Toolbar**	Contain the tools that you can click to quickly apply many of the commands on the **Edit** menu and the **Text** menu to your graphic objects.
Arrange Toolbar	Contains the tools that you can click to quickly apply the alignment commands on the **Arrange** menu.
Wizards/ActiveX Toolbar	You can also customize this toolbar by adding any wizards or ActiveX controls that you repeatedly use.

For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."

Tip If you right-click an object, a menu appears displaying the valid commands or actions that you can apply to the selected object. For example:



Note The commands on the right-click menus will vary. They are based upon the type of object that is selected.

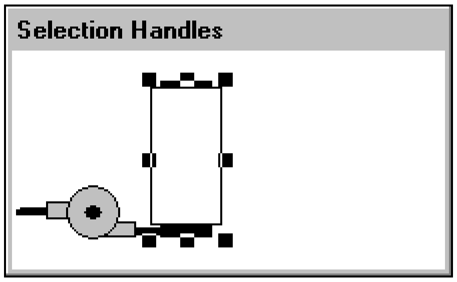
Selecting and Sizing Objects

After you draw an object, and you click it, several little boxes will surround it. These boxes are called "handles." You use these handles to resize and/or reshape the object.

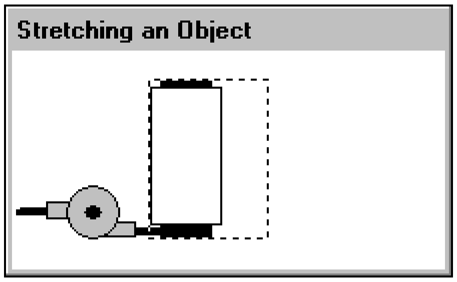
The notion of "selected" is a key concept of WindowMaker graphics editing. The presence of "handles" around an object indicates that it is "selected." Clicking directly on an object selects it. Clicking on a blank area of the window deselects any currently selected object(s) in that window. In general, any command that you execute is applied to all selected objects, if the command is valid for the object.

To change the size of an object

1. Select the object and then position the point of the arrow cursor in the center of a "handle."

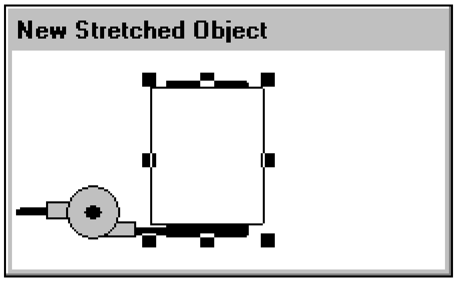


2. Press and hold down the left mouse button while you drag the handle to either stretch or shrink the object:



You can stretch/shrink the object in either direction, depending upon which handle is selected. If you use one of the four corner handles to size the object, it will be sized vertically and horizontally simultaneously.

3. Release the mouse button. The object will be redrawn at its new size:



If the size is not correct, on the **Edit** menu, click **Undo**, or click the Undo tool on the **General Toolbar** and try again.

For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."

For more information on undoing and redoing edits, see "Undoing Object Edits."

Selecting all Objects in a Window

To select all objects in the active window, on the **Edit** menu, click **Select All**, or press F2, or right-click a blank area of the open window and then click **Select All**.

Selecting Multiple Objects

To select multiple objects, select your first object, then hold down the SHIFT key while you click the other objects that you want to select. To deselect a specific object from a group of selected objects, while all objects are selected, hold down the SHIFT key and click the object that you want to deselect.

Selecting a Group of Objects

Move the cursor to a blank area of your window. Depress the mouse button and drag the mouse. A dotted selection rectangle with a small hand cursor appears. Drag the mouse until you have completely surrounded all of the objects that you want to select. Release the mouse button. All the objects that were completely within the rectangle will be selected.

Deselecting a Group of Selected Objects

If you hold the **Shift** key down while you draw a selection rectangle, all enclosed selected objects will become deselected. This technique may also be used to start a selection rectangle on top of another object.

If you hold the **Shift** key down while you click the left mouse button, the object under the cursor will not be dragged when the cursor is moved. Instead, a selection rectangle will be drawn.

Undoing Object Edits

WindowMaker keeps track of the editing and formatting changes that you make. You can configure WindowMaker to support up to 25 undo/redo levels. You can also disable the undo/redo functionality by setting the level to zero (0). By default, WindowMaker is set to support 10 levels where, one level represents one action.

Note The undo and redo stacks are empty when you create a new window or open an existing window. Both stacks are also cleared when you save the window.

For more information on configuring the undo/redo levels, see "Customizing Your Development Environment."

The **Undo** and **Redo** commands are located on the **Edit** menu. These commands can also be accessed by right-clicking the object. The commands are dynamic and will change to reflect the last action to which they can be applied. For example, if you move an object then decide that you want to return it to its original location in the window, right-click a blank area of the window and click **Undo Move** or on the **Edit** menu, click **Undo Move**.

The Undo and Redo tools are located on the **General Toolbar**.

You will use undo and redo to reverse actions or commands that you have applied to an object. You can undo or redo the following actions or commands:

Command	Action
Basic	Create, Select, De-select, Move and Re-size Object Line, Fill, Text and Window Color Window Move and Window Size
Editing	Duplicate, Cut, Copy, Paste, Delete Paste Bitmap and Adjust Bitmap - Original size Select All, Link Cut, Link Copy, Link Paste, Link Clear Enlarge Radius, Reduce Radius Reshape Object, Add Point, Delete Point
Arranging	Send to Back, Bring to Front, Align (all commands) Space Horizontal, Space Vertical Rotate CW, Rotate CCW Flip Horizontal, Flip Vertical Make Symbol, Break Symbol Make Cell, Break Cell
Text	All operations (size, style, pitch, justification)
Line	All operations (width and styles)
Special	Animation Links (double-click on object), Substitute Strings

Duplicating Objects

To redraw the previously drawn object

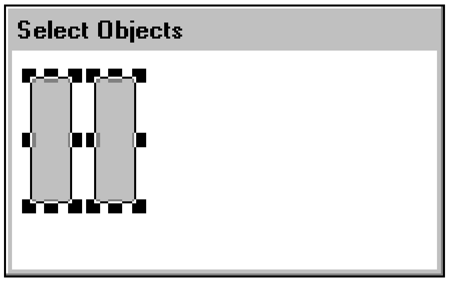
1. Draw the object.
2. Right-click the object and then click **Repeat Last Object**.
3. Click the left mouse button and then draw the same object again.

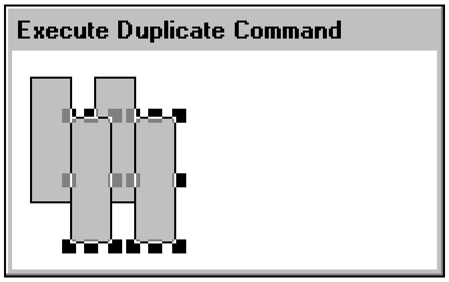
To duplicate an object

1. Select the object(s) you want to duplicate.

2. On the **Edit** menu, click **Duplicate** or click the Duplicate Selection(s) tool on the **General Toolbar**.

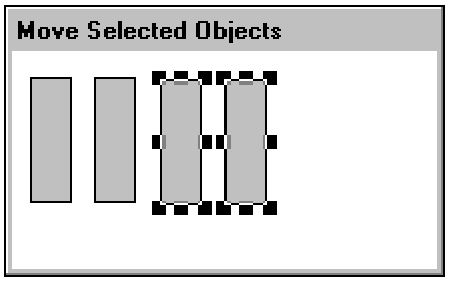
   For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."

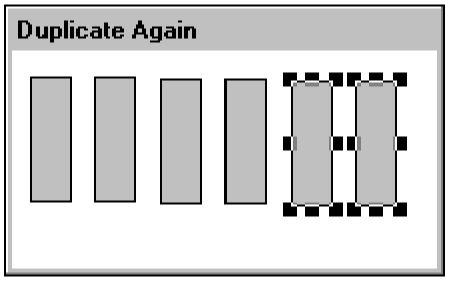




Tip To quickly duplicate an object(s), right-click the object and then click **Duplicate**.

If you move the duplicated object(s) without deselecting it, and you duplicate it again, the second (and all subsequent duplications) will automatically be offset the same distance that the first duplicate was moved. For example:





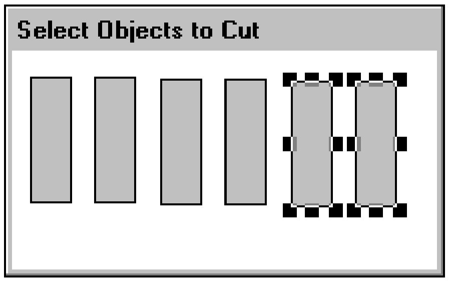
You can repeat this procedure as many times as necessary.

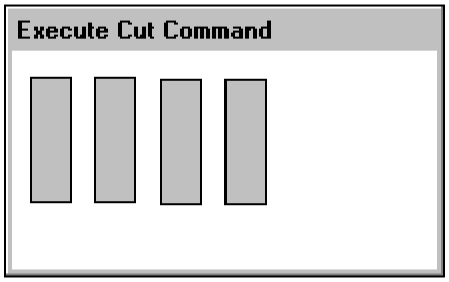
Cutting Objects to the Windows Clipboard

To cut an object

1. Select the object(s) you want to cut.

2. On the **Edit** menu, click **Cut** or click the Cut to Clipboard tool on the **General Toolbar**.





Tip To quickly cut an object(s), right-click the object and then click **Cut**.

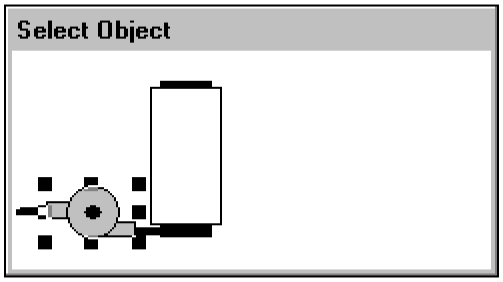
Note When you cut an object, it is erased from your window and copied to the Windows Clipboard. The object's attributes and animation links are also copied with it.

Copying Objects to the Windows Clipboard

To copy an object

1. Select the object(s) you want to copy.
2. On the **Edit** menu click **Copy** or click the Copy to Clipboard tool on the **General Toolbar**.

   For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."



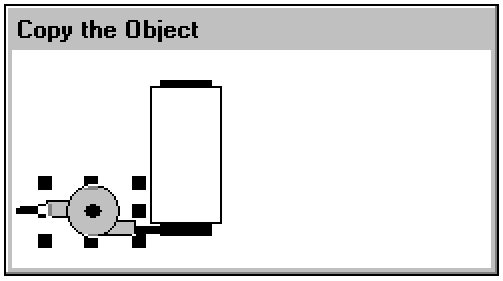
Tip To quickly copy an object(s), right-click the object and then click **Copy**.

Note When you copy an object, it is not erased from your window. It is copied to the Windows Clipboard. The object's attributes and animation links are also copied with it.

Pasting Objects from the Windows Clipboard

To paste an object from the Windows Clipboard

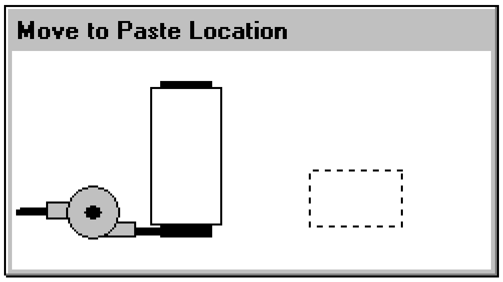
1. Copy or cut the object:



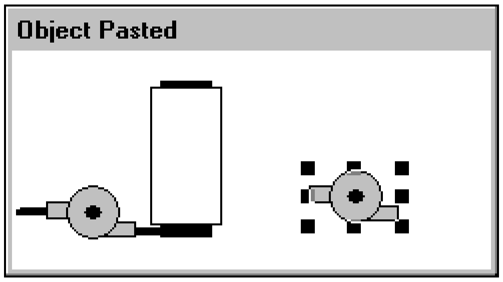
2. On the **Edit** menu, click **Paste** or click the Paste from Clipboard tool on the **General Toolbar**.

Tip To quickly paste a copied object, right-click the object and then click **Paste**.

3. The cursor will change to a corner symbol.
4. Hold down the left mouse button, a dotted line rectangle the size of the copied object appears. Drag the rectangle to the location in the window that you want to paste the object:



5. Release the mouse button to paste the object:



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

Note When you copy or cut an object to the Windows Clipboard, all the object's attributes and animation links are copied with it. If you subsequently paste that object into a window, all of its attributes will still be intact.

Cutting and Pasting Object Links

WindowMaker's link paste buffer is a temporary storage area that stores links that you cut or copy from an object. (The buffer only stores the links for your most recent cut or copy action.) You can paste the links stored in the link paste buffer to any object or symbol. If you select multiple objects, the links are pasted to each individual object.

If a pasted link has no apparent value to the object, for example, a line color link on a text object, the link is not pasted.

For more information on animation links, see Chapter 7, "Creating Animation Links."

To cut, copy, paste and clear links

1. Select the object that you want to apply the links command to.
2. On the **Edit** menu, point to **Links** and then click the appropriate command.

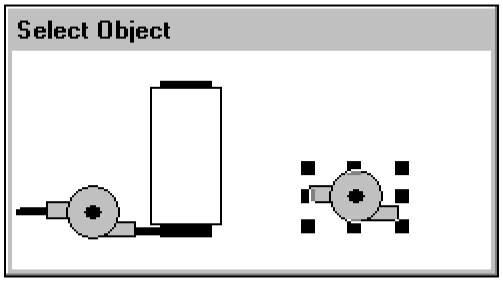
Tip To quickly access the link commands, right-click the object, then point to **Links** and then click the appropriate links command.

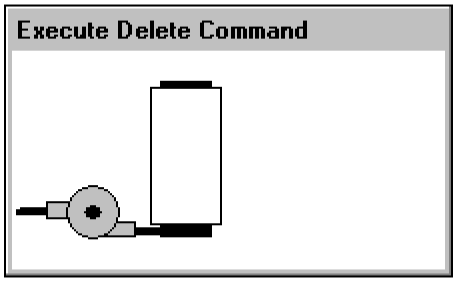
Deleting Objects

To delete an object

1. Select the object(s) you want to delete.

2. On the **Edit** menu, click **Erase**.





Tip To quickly delete an object(s), right-click the object and then click **Erase**, or select the object and press the DEL key.

Note Deleted objects are <u>not</u> copied to the Windows Clipboard.

Increasing or Decreasing a Rounded Object's Radius

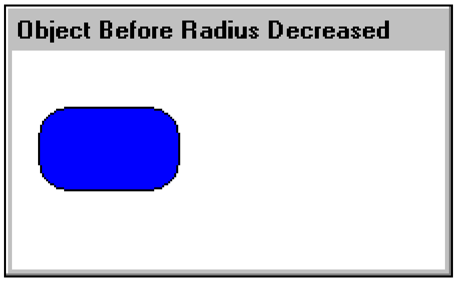
You can increase and/or decrease the corner radius of any object that you have drawn with the Rounded Rectangle tool.

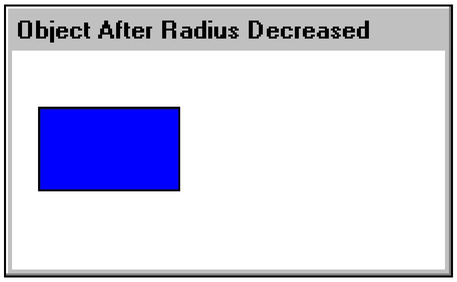
To increase (or decrease) a rounded object's radius

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

Tip To quickly increase or decrease the radius, right-click the rounded object and then click the appropriate command.

3. Repeat the command until the radius has increased to the desired degree. For example:



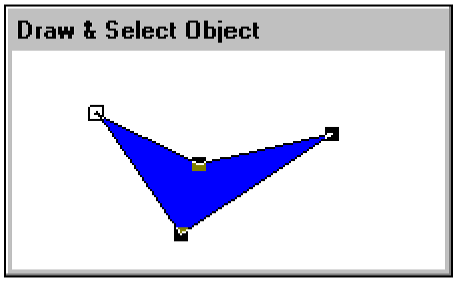


Tip You can also use the keyboard short cut keys Shift+Plus, (+ symbol on the numeric keypad), to increase the radius or Shift+Minus, (**-** symbol on the numeric keypad) to decrease the radius. Each time you press these key combinations, the command will be performed. If you hold the keys down, the command will execute continuously until the maximum increased or decreased radius for the object is reached.

Reshaping a Polyline or Polygon Object

To adjust the shape of polylines or polygons

1. Select the polyline or polygon object.

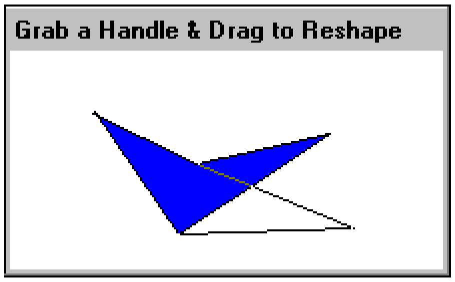


Tip Each "point" where you clicked the mouse as you were drawing the object reappears as a "handle."

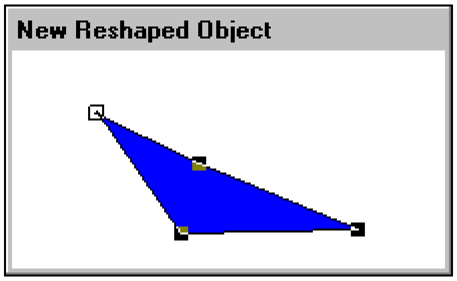
2. On the **Edit** menu, click **Reshape Object** or click the Reshape Object tool on the **Arrange Toolbar**.

Tip To quickly reshape a polyline or polygon, right-click the object and then click the appropriate command.

3. To reshape the object, grab a handle and drag it to the desired location:

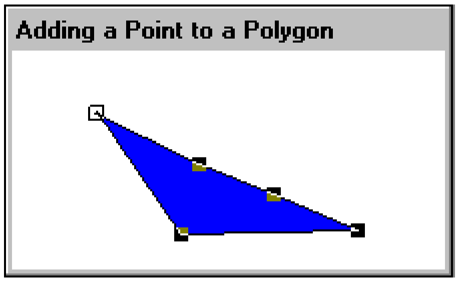


4. When you release the mouse, the object will be redrawn to its new shape:



To add or delete "points" on a polygon or polyline

1. Select the polyline or polygon object.
2. On the **Edit** menu, click **Add Point** or **Del Point** then click the spot of the object where you want the new point added or click the point that you want to delete.

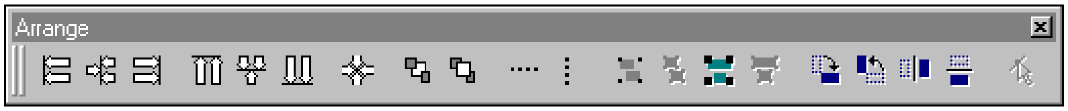


Tip To quickly add (or delete) points on a polyline or polygon, right-click the object and then click the appropriate command.

Arranging Objects in your Window

WindowMaker provides you with numerous tools that you will use to arrange your objects in your windows. This section describes the various arranging tools that are available in WindowMaker.

Tip The **Arrange Toolbar** contains tools that you can use to quickly apply most of the commands found on the **Arrange** menu to selected objects. For example:



For more information on the **Arrange Toolbar**, see Chapter 1, "WindowMaker Program Elements."

Aligning Objects

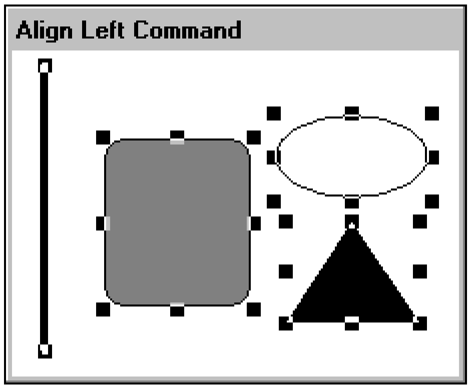
You can align objects by their left or right edges, centers, centerpoints, tops, middles or bottoms.

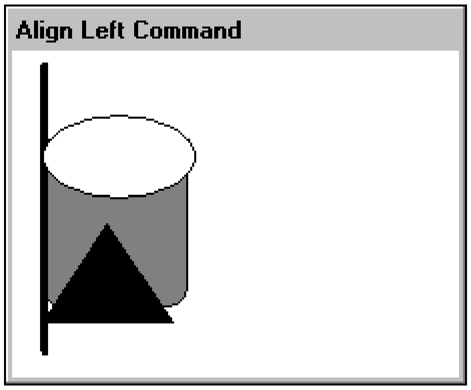
To align all selected objects

1. Select the objects(s).
2. On the **Arrange** menu, point to **Align** and then click the appropriate align command. The selected object(s) will be aligned according to your selection.

The following examples illustrate the behavior for each alignment command:

Align Left aligns the left edge of all selected objects with the left edge of the object that is farthest left in the group:

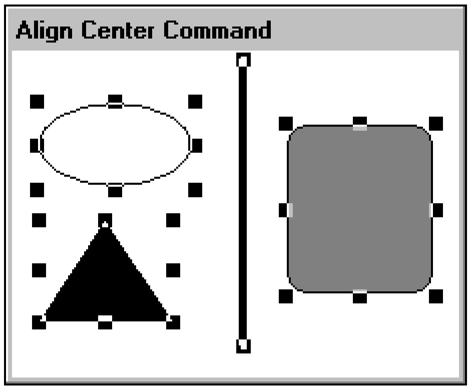


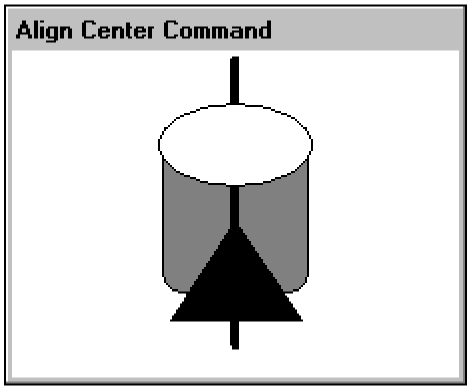


Tip To quickly align objects, select the objects and then click the appropriate tool on the **Arrange Toolbar**.

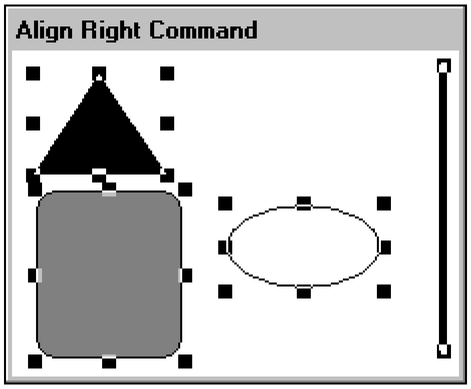
For more information on the toolbars, see Chapter 1, "WindowMaker Program Elements."

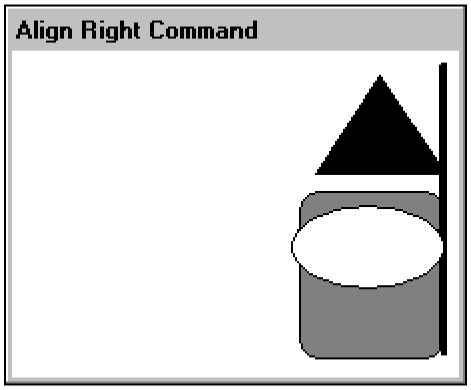
Align Center aligns all the selected objects with the vertical centerline of the group:



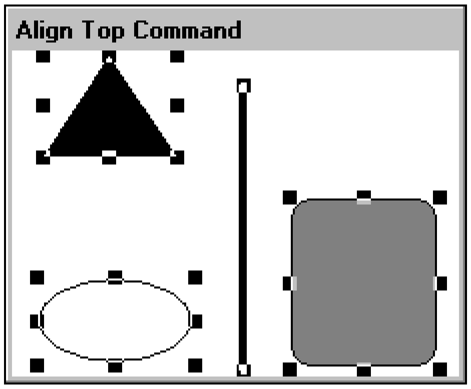


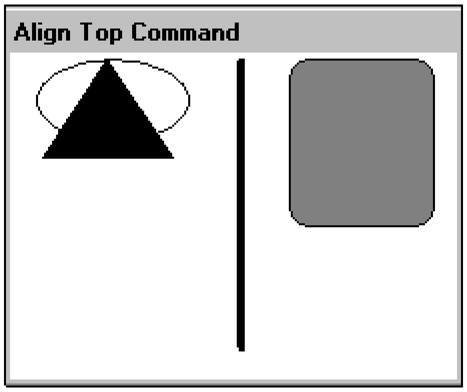
Align Right aligns the right edge of all selected objects with the right edge of the object that is farthest right in the group:



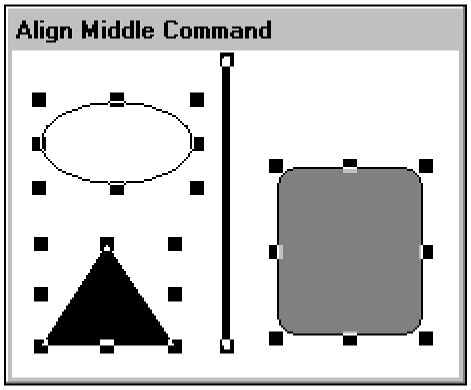


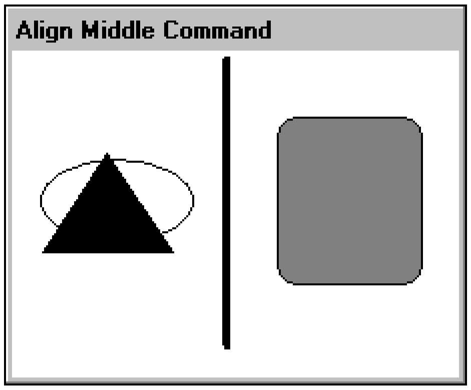
Align Top the top edge of all selected objects with the top edge of the object that is highest in the group:



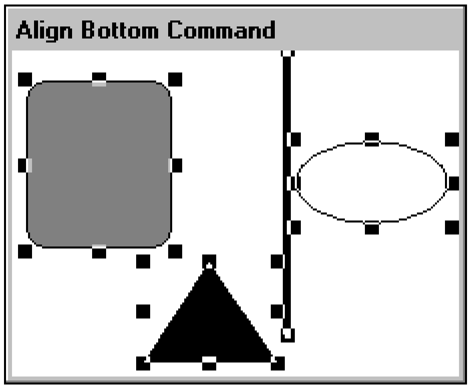


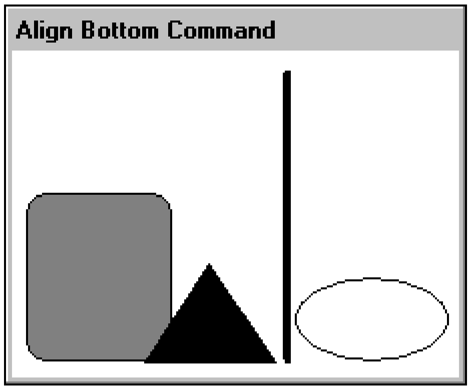
Align Middle aligns the middle of all the selected objects with the middle of the group:



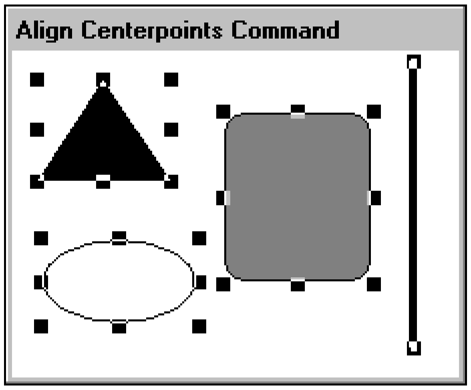


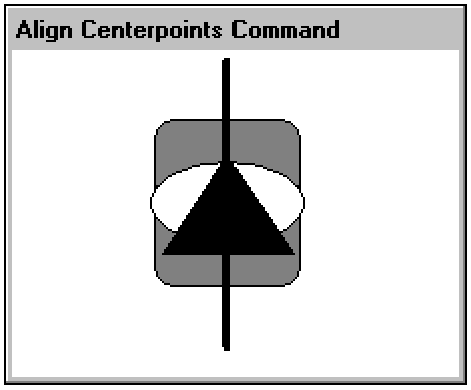
Align Bottom the bottom edge of all selected objects with the bottom edge of the object that is lowest in the group:





Align Centerpoints aligns the centerpoints of all the selected objects with the centerpoint of the group:





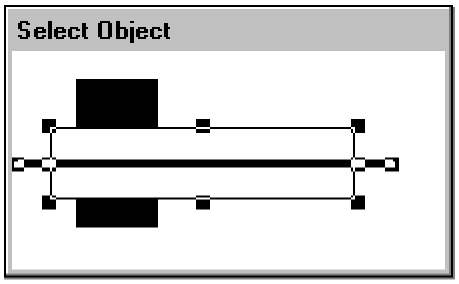
Layering Objects

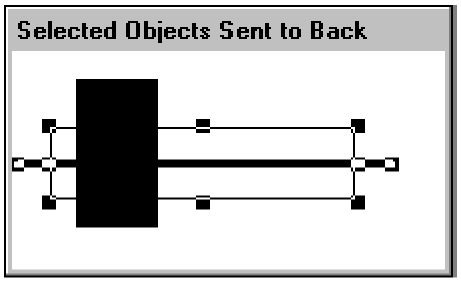
You can layer the objects in your window by positioning objects in front or behind each other.

To position an object behind another object

1. Select the objects(s).

2. On the **Arrange** menu, click **Send to Back** or click the Send to Back tool on the **Arrange Toolbar**. The selected object(s) will be redrawn behind the object(s) not selected in your window:



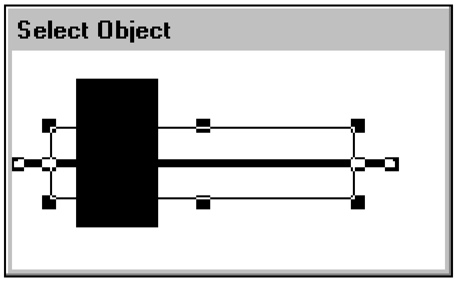


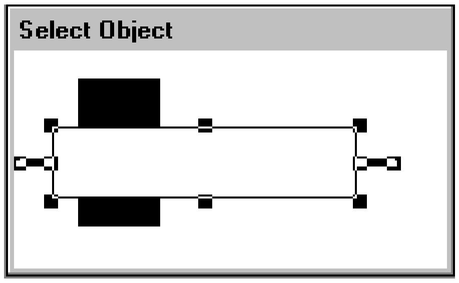
Tip To quickly send an object(s) to the back, right-click the object, then point to **Front/Back** and then click **Send to Back**.

To position an object in front of another object

1. Select the objects(s).

2. On the **Arrange** menu, click **Bring to Front**, or click the Bring to Front tool on the **Arrange Toolbar**. The selected object(s) will be redrawn in front of the object(s) not selected in your window:





Notice that the object farthest back moved to the front.

Tip To quickly bring an object(s) to the front, right-click the object, then point to **Front/Back** and then click **Bring to Front**.

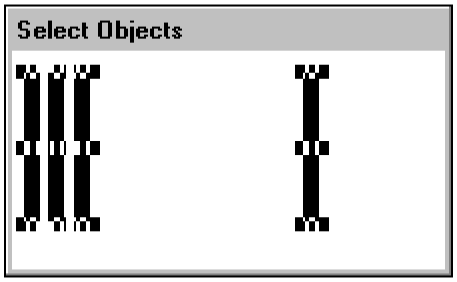
Controlling Horizontal and Vertical Spacing

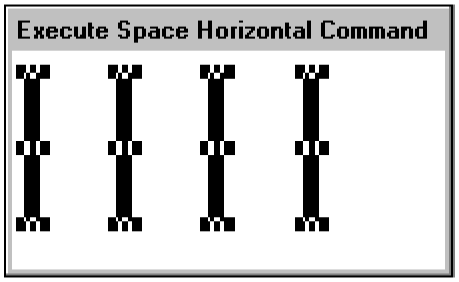
You can evenly space objects horizontally between the left most selected object and the right most selected object. You can also control the vertical spacing between the top most selected object and the bottom most selected object.

To evenly space objects horizontally or vertically

1. Select the objects.

2. On the **Arrange** menu, click **Space Horizontally** (or **Space Vertically**) or click the appropriate tool on the **Arrange Toolbar**. The selected object(s) will be redrawn spaced evenly between the two farthest placed objects. For example:





For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."

Rotating Objects

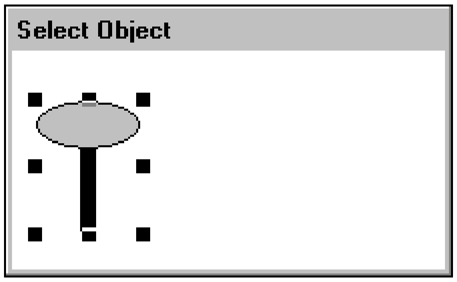
In WindowMaker, you can rotate most objects including bitmaps, JPEG, PCX, and TGA images and text objects. Objects can be rotated clockwise or counter clockwise 360 degrees in 90 degree increments (90 degrees, 180 degrees, 270 degrees and 360 degrees). Any links attached to the object are rotated with the object. You cannot rotate cells. However, you can rotate symbols.

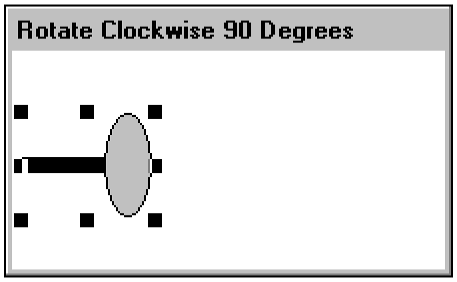
Note Rotating objects in WindowMaker has nothing to do with dynamically rotating objects in runtime. Objects are rotated in WindowViewer by linking them to an **Orientation** animation link. Text objects cannot be rotated in WindowViewer. However, bitmaps or images can be rotated by assigning an **Orientation** animation link to them.

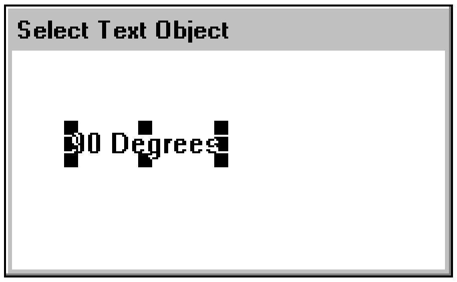
To rotate a selected object(s) 90 degrees

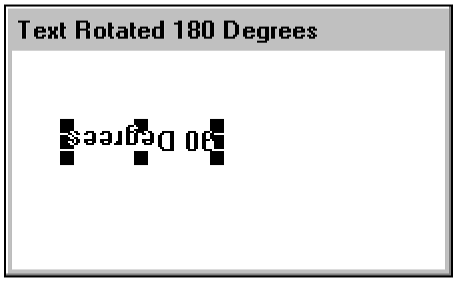
1. Select the object(s).

2. On the **Arrange** menu, click **Rotate Clockwise** (or **Rotate CounterClockwise**). The selected object(s) will be rotated 90 degrees in the direction you chose:









To rotate an object 180 degrees, repeat this procedure. To rotate an object 270 degrees, perform this procedure twice, and so on.

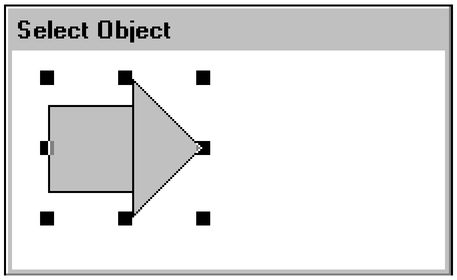
Tip To quickly rotate an object, right-click the object, then point to **Rotate/Flip** and then click the appropriate command.

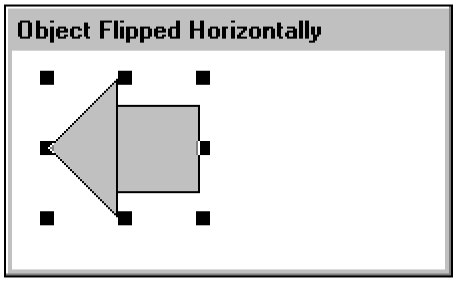
Mirroring Objects

You can flip most WindowMaker objects horizontally or vertically including, bitmaps , JPEG PCX, and TGA images (text cannot be flipped, it can only be rotated). When you flip an object, you transform it into its horizontal or vertical mirror image. Any links attached to the object are flipped with the object.

To flip a selected object

1. Select the object(s).
2. On the **Arrange** menu, click **Flip Horizontal** (or **Flip Vertical**) or selector the appropriate tool on the **Arrange Toolbar**. The selected object(s) will flip. For example:





Tip To quickly flip an object(s), right-click the object, then point to **Rotate/Flip** and then click the appropriate command.

For more information on toolbars, see Chapter 1, "WindowMaker Program Elements."

Creating Cells and Symbols

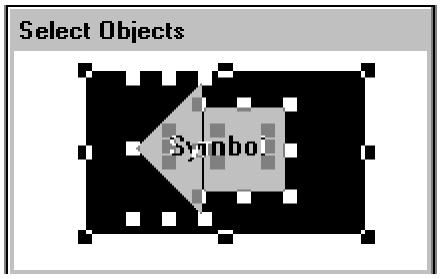
You can combine multiple objects into two different types of single units; cells and symbols. Multiple cells can be combined into a single cell. Cells are objects that maintain a fixed spatial relationship between individual graphic elements. The individual components within a cell, except another cell, can be animated. Cells cannot be resized, nor can animation links be attached to cells. However, animation links can be attached to symbols, and symbols can be included in a cell. All animation links associated with a symbol or an object(s) within a cell are unchanged. The attributes of objects such as text, font, line width, radius and relative positions within a cell cannot be sized until the cell is broken into its individual components.

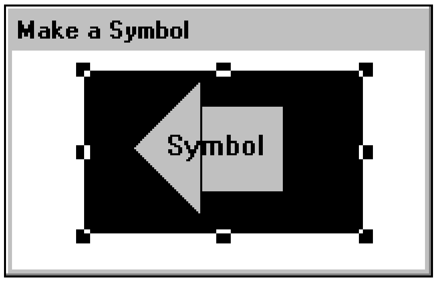
Tip Double-clicking a cell will cause the **Substitute Tagnames** dialog box to appear (not the animation links selection dialog box as with objects and symbols).

For more information on substituting tagnames, see Chapter 6, "Tagname Dictionary."

Note When combining cells, each cell will be retained, so when the combined cell is broken the original cells are restored.

A symbol can be made up of multiple symbols and/or multiple simple objects as illustrated below:





If one of the objects selected has animation links attached to it, the links will be attached to the new symbol. (If the link paste buffer has links in it, you will be asked if you want to paste the links on the new symbol.)

Note 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. Therefore, if you break the new symbol, it will be broken into the individual components of each original symbol. The two original symbols are lost.

For more information on animation links, see Chapter 7, "Creating Animation Links."

To create a symbol or cell

1. Select the objects that you want to include in the cell or symbol...
2. On the **Arrange** menu, click **Make Cell** (or **Make Symbol**), or click the appropriate tool on the **Arrange Toolbar**.

Tip To quickly create a cell or symbol, select all the objects. Right-click one of the selected objects, point to **Cell/Symbol** and then click the appropriate command.

To break a symbol or cell

1. Select the symbol or cell...
2. On the **Arrange** menu, click **Break Cell** (or **Break Symbol**), or click the appropriate tool on the **Arrange** toolbar.

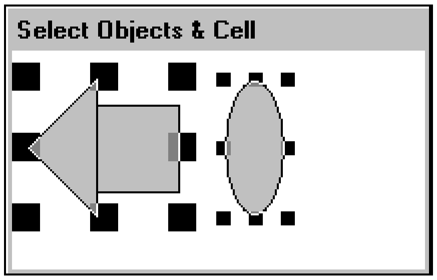
Tip If the symbol has links defined for it, the links are automatically saved to the link paste buffer.

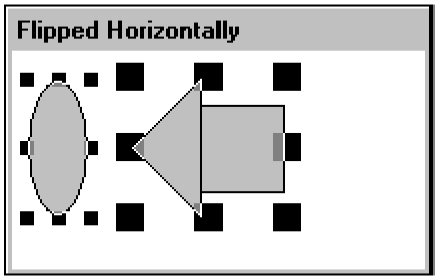
To quickly break a cell or symbol, right-click the cell or symbol, then point to **Cell/Symbol** and then click the appropriate command.

For more information on the WindowMaker toolbars, seeChapter 1, "WindowMaker Program Elements."

Flipping Cells

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





Compare the location of the cell (on the left) and direction it is facing before it is flipped to the direction it is facing after it is flipped. Its position was flipped, but not its contents. The ellipse object was mirrored. The same applies to cells flipped vertically.

Snapping Objects to the Grid

When you are arranging objects in your windows, turning on the grid will cause your graphic to snap at the upper left pixel interval on the grid. If you select multiple objects, the snapping will be applied to the upper left corner of the first object selected in the group.

Click the Snap to Grid tool on the **View** toolbar to turn snap to grid on and off, or on the **Arrange** menu, click **Snap to Grid**.

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

For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."

To configure the grid

1. On the **Special** menu, point to **Configure** then click **WindowMaker** or in the Application Explorer under **Configure**, double-click **WindowMaker**. The **WindowMaker Properties** dialog box appears.

2. In the **Spacing** box, type the number of pixels that you want spaced between the snap to grid's coordinates.
3. Select **Show Grid** if you want a visible grid in your windows when you turn on WindowMaker's "snap to grid" functionality.

Tip If you do not select **Show Grid**, no grid will be visible in your windows when you turn snap to grid on.

For more information on configuring the grid, see "Customizing Your Development Environment."

Working with Images and Bitmaps

All graphic objects such as pictures, screen captures, AutoCad drawings, JPEG, PCX and TGA file types and so on, that are created in other Windows programs must be pasted into a bitmap container in WindowMaker.

WindowMaker sees a bitmap as a single object. Therefore, you cannot animate the individual elements of a bitmap, nor can you include bitmaps in symbols. However, you can include them in a cell.

In WindowMaker, you can rotate bitmaps, JPEG, PCX, and TGA images. They can be rotated clockwise or counter clockwise 360 degrees in 90 degree increments (90 degrees, 180 degrees, 270 degrees and 360 degrees). Any links attached to the bitmap are rotated with it.

Note Rotating bitmaps in WindowMaker has nothing to do with dynamically rotating them in runtime. Bitmaps or images are rotated in WindowViewer by linking them to an **Orientation** animation link.

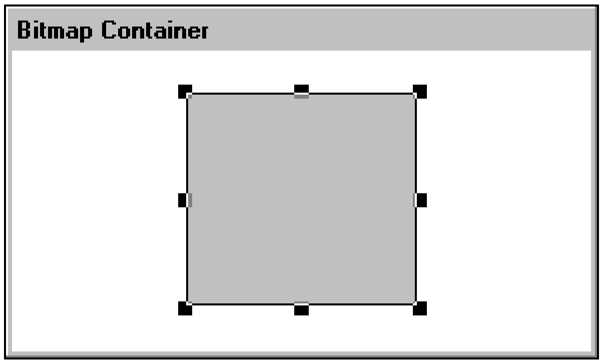
You can also define a bitmap with a transparent color, so that you can float it over other objects. When you define a bitmap with a transparent color, the window background color or any objects behind the bitmap will show through it everywhere the transparent color is used. (Only one transparent color may be used per bitmap.)

For more information on transparent bitmaps and images, see "Creating a Transparent Bitmap."

To import a bitmap or JPEG, PCX or TGA file type

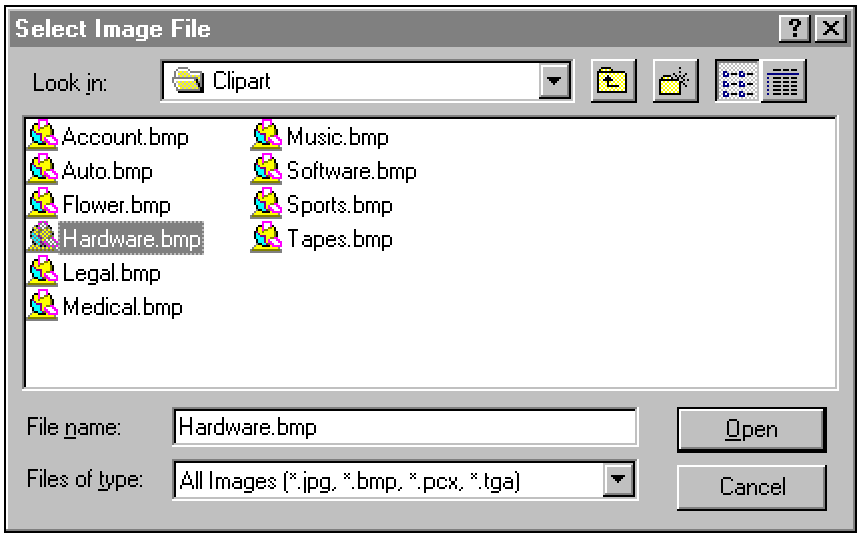
1. Click the Bitmap tool (your cursor turns into a cross-hair) then draw a bitmap container in your window (the size is irrelevant).

2. Select the bitmap container:



3. On the **Edit** menu, click **Import Image**. The Windows **Select Image File** dialog box appears.

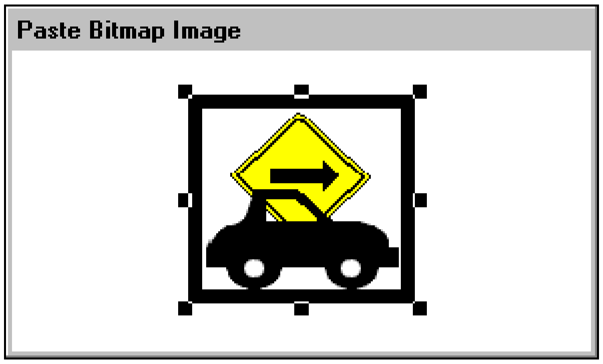
Tip To quickly paste the image, right-click the bitmap container and then click **Import Image**.



File name: Hardware.bmp

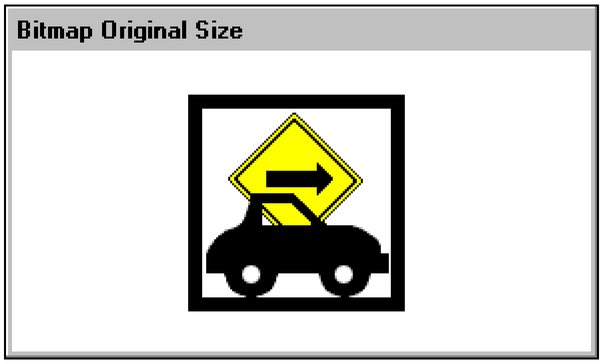
Files of type: All Images (*.jpg, *.bmp, *.pcx, *.tga)

4. Locate and select the .BMP, .PCX, .TGA or .JPG file that you want to import as a bitmap, then click **Open** or double-click the image filename. The image will be pasted into your bitmap container.



5. To make the bitmap its original size, select it, then on the **Edit** menu, click **Bitmap - Original Size**. The bitmap will be redrawn at its original size.

Tip To quickly size the bitmap, right-click the bitmap and then click **Bitmap - Original Size**.

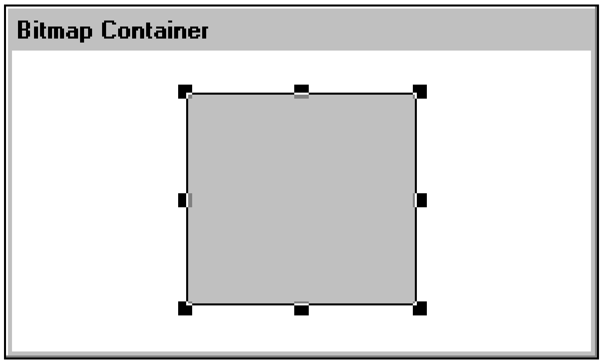


Pasting a Bitmap from the Windows Clipboard

To paste a bitmap from the Windows Clipboard into a window

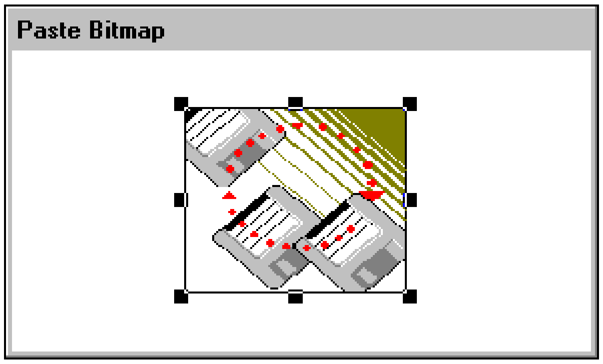
1. Copy the graphic to the Windows Clipboard. For example, display the graphic, then hold down the ALT key while you press the PRINT SCRN key to copy it to the Windows Clipboard.
2. Click the Bitmap tool (your cursor turns into a cross-hair), then draw a bitmap container in your window (the size is irrelevant).

3. Select the bitmap container.

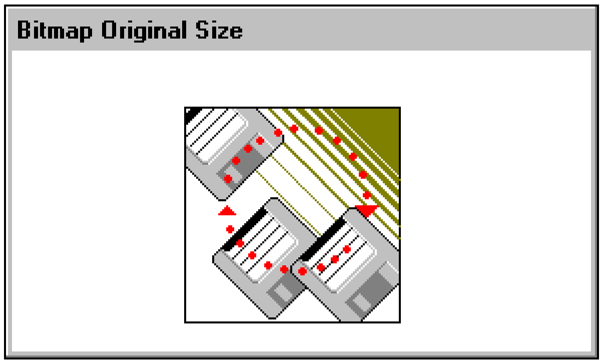


4. On the **Edit** menu, click **Paste Bitmap**. The bitmap from the Windows Clipboard will be pasted into the bitmap container:

Tip To quickly paste the bitmap, right-click the bitmap container and then click **Paste Bitmap**.



5. To make the bitmap its original size, select it and then on the **Edit** menu, click **Bitmap - Original Size**. The bitmap will be redrawn at its original size.



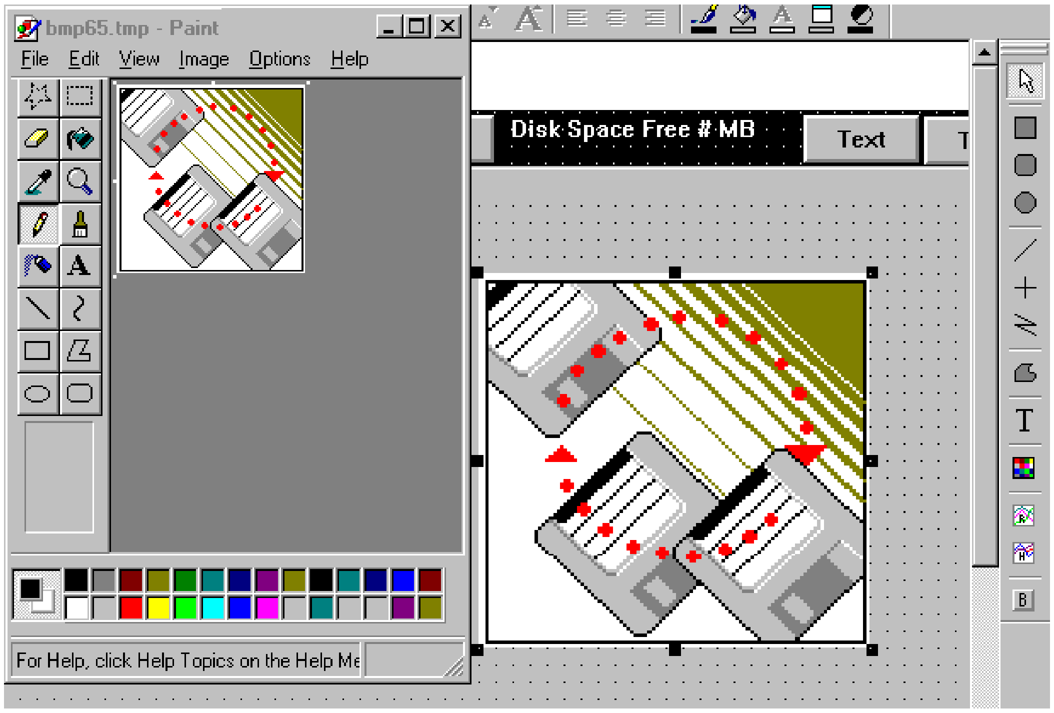
Tip To quickly size the bitmap, right-click the bitmap and then click **Bitmap - Original Size**.

Editing a Bitmap

You can edit bitmaps directly from WindowMaker. When you edit a bitmap in WindowMaker, the application that is associated with the .bmp extension is launched. For example, if the bitmap was created using MSPaint, when you edit the bitmap in WindowMaker, MSPaint will automatically be launched as described below.

To edit a bitmap from WindowMaker using Windows Paint

1. Select the bitmap. On the **Edit** menu click **Edit Bitmap**. Microsoft Paint is opened displaying the bitmap:


bmp65.tmp - Paint
File Edit View Image Options Help
For Help, click Help Topics on the Help Me
Disk Space Free # MB
Text

Tip To quickly edit a bitmap, right click the bitmap and then click **Edit Bitmap**.

2. Edit the bitmap in MS Paint.
3. Exit MS Paint to return to WindowMaker.

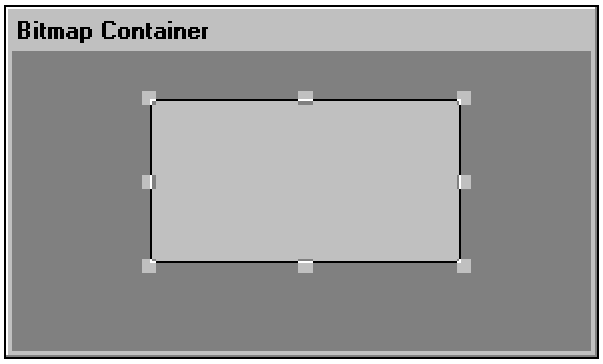
Note You cannot use WindowMaker while editing the bitmap in its native application. To return to WindowMaker, exit the native application.

Creating a Transparent Bitmap

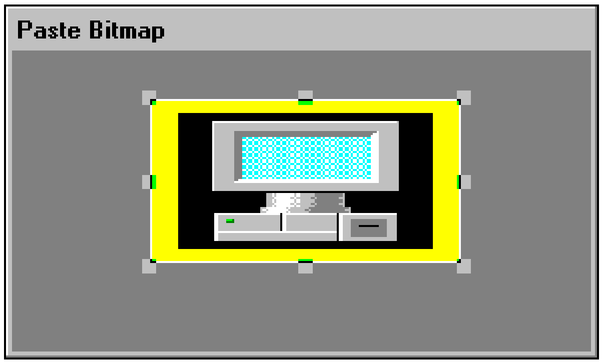
You can define a bitmap or image with a transparent color, so that you can float it over other objects. When you define a bitmap or image with a transparent color, the window's background or any objects behind the bitmap will show through it everywhere the transparent color is used.

To create a transparent bitmap

1. Click the Bitmap tool (your cursor turns into a cross-hair) then draw a bitmap container in your window (the size is irrelevant).
2. Select the bitmap container:

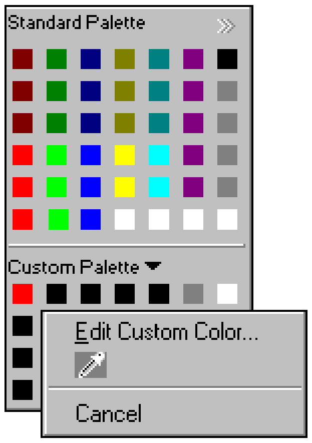


3. Right-click the bitmap container and then click **Paste Bitmap** (if you have copied the graphic to the Windows Clipboard), otherwise click **Import Image** (to locate and select the .BMP, .PCX, .TGA or .JPG file to open). The bitmap image will be pasted into the bitmap container:

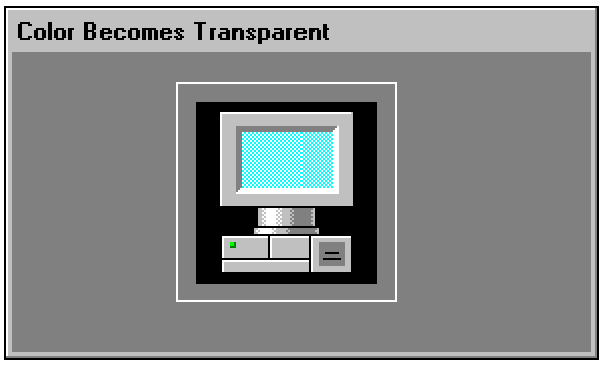


4. Right-click the bitmap and then click **Bitmap - Original Size** to return the bitmap to its original size.
5. With the bitmap selected, click the Transparent Color tool on the **Format** toolbar to open the transparent color palette.

6. Right-click a blank color square in the **Custom Palette** section at the bottom of the color palette. The **Edit Custom Color** dialog box appears.



7. Click the Blotter tool (the **Edit Custom Color** dialog box will close).
8. Click the color in the bitmap that you want to make transparent. The color will be copied to the color square that you selected in the transparent color palette.
9. Click the color square to apply the transparent color to the bitmap.



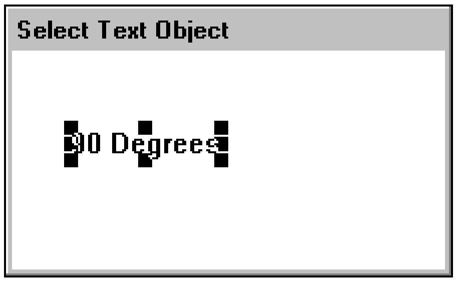
In this example, we made the wide border area of the bitmap transparent. Therefore, the window background color now shows through the bitmap. If objects were behind the now transparent area, they would also show through the bitmap.

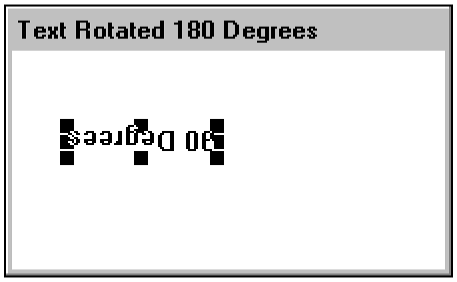
Note Only one transparent color can be applied per bitmap.

Working with Text Objects

In WindowMaker you can change the font, font style, font size, justification and rotation of any selected text object. You can also rotate it 360 degrees by 90 degree increments (90 degrees, 180 degrees, 270 degrees and 360 degrees).

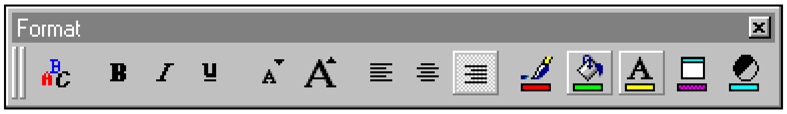
For example:





Note Rotating text objects in WindowMaker has nothing to do with dynamically rotating objects in runtime. **Orientation** animation links cannot be applied to text objects. Therefore, text objects cannot be rotated in WindowViewer.

The **Format Toolbar** contains tools that you can use to quickly apply most of the commands found on the **Text** menu to selected objects. For example:



For more information on the Format Toolbar, see Chapter 1, "WindowMaker Program Elements."

Formatting Text Objects

All WindowMaker text commands operate on single or multiple text string selections and numeric value fields. If no text object is selected when a command on the **Text** menu is executed, the command is automatically applied to the respective text tool's default setting on the **Format Toolbar** and the default setting of the Text tool on the **Draw Object Toolbar**.

The text justification attribute settings are particularly important for text objects used for outputting dynamic values. The justification determines how fields of varying length will be displayed in runtime.

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

For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."

Displaying Numeric Values

Text objects are also used to display static or dynamic numeric values. By attaching a **Touch Links User Inputs** - **Analog** or **Value Display - Analog** animation link to a text object you can display the value of an analog (integer or real) tagname.

To determine the display format of the analog value, the following four characters are used:

Character	Description
0	zero
#	number or pound sign
,	comma
.	period or decimal point

The following illustrates field formatting for analog values:

Character	Description
#	Displays any whole number. For example: **1234** would display as **1234** (Only one # sign is necessary)
0.0	Forces one leading zero and one decimal place. For example: **.1** would display as **0.1** **77.1** would display as **77.1**
00000	Forces leading zeros as required. For example: **123** would display as **00123** **1234** would display as **01234** **12345** would display as **12345**
#,##0.0	Inserts comma and leading zero if required, and one decimal place. For example: **1234.56** would be displayed as **1,234.6** **123.4** would display as **123.4**
0,000.0	Forces comma, leading zeros and one decimal place. For example: **12.3** would display as **0,012.3**

Note If you use a zero in the format, it must be followed by zeros. All places to the right of the decimal point must always be zeros. For example, 000.00 is correct, while #0#0.0# is incorrect.

Tip All normal text formatting applies to numeric values. These include font, size, color justification and bolding.

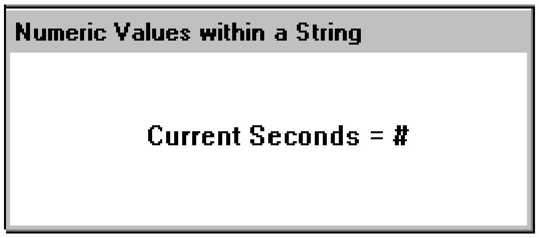
To create a text object

1. Click the Text tool in the **Draw Object Toolbar**.
2. Click in the window and type the text string.

Tip To quickly access the various commands that can be applied to a text object, right-click the text object and then click the appropriate command.

To display a numeric value within a text string

1. Click the Text tool and then type a text object in the window using one of the previously described valid numeric formats. For example:



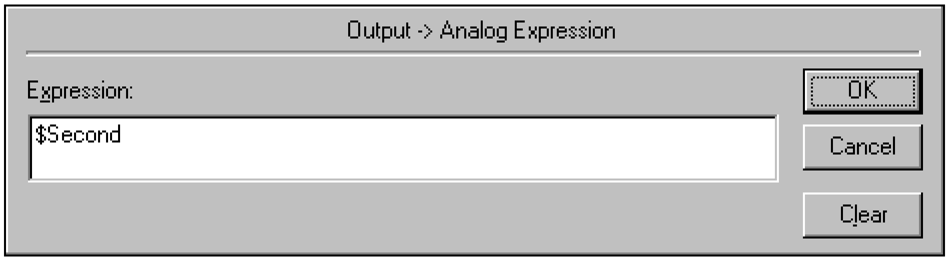
Numeric Values within a String

Current Seconds = #

2. Select the object and then on the **Special** menu, click **Animation Links** or double-click the text object. The **Animation Links** selection dialog box appears.

Tip To quickly access the dialog box, right-click the text object and then click **Animation Links**.

3. In the **Value Display** section, click **Analog**. The **Output -> Analog Expression** dialog box appears.



Output -> Analog Expression

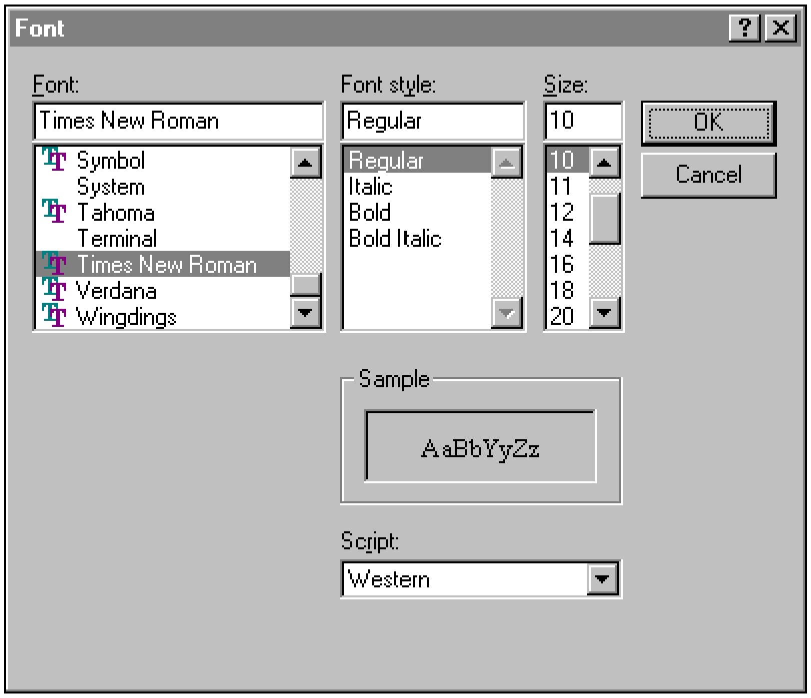
Expression: $Second

4. In the **Expression** box, type an analog tagname or expression. (In this example, the system tagname **$Second** is being used.)
5. Click **OK**.

6. Click the **Runtime** fast switch in the upper right hand corner of the menu bar (or use the short cut keys ALT + !) to switch to WindowViewer or on the **File** menu, click **WindowViewer**.
7. If you used this example, you will see the current system seconds (a value between 0-59) displayed in place of the pound (#) sign in the text string.
8. Click the **Development** fast switch in the upper right hand corner of the menu bar (or use the short cut keys ALT + !) to return to WindowMaker or on the **File** menu, click **WindowMaker**.

To change the font, font style and font size of a string

1. Select the text string and then on the **Text** menu, click **Fonts** or click the Fonts tool on the **Format Toolbar**. The standard Windows **Font** dialog box appears.


Font

Font: Times New Roman
Symbol
System
Tahoma
Terminal
Times New Roman
Verdana
Wingdings

Font style: Regular
Regular
Italic
Bold
Bold Italic

Size: 10
10
11
12
14
16
18
20

Sample
AaBbYyZz

Script: Western

2. Select the desired font from the **Font** list (the font name appears in the **Font** field). Once a font is selected, the styles and sizes available for it appear in the **Font Style** and **Size** fields. When a font size is selected a sample of the font in the selected style and size appears in the **Sample** field (see above example).
3. Click **OK**.

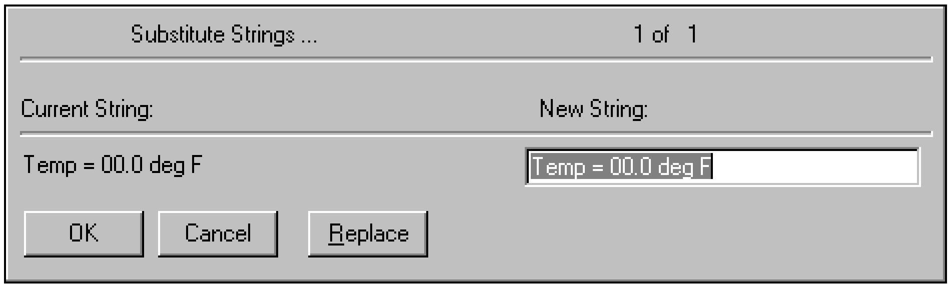
Note A font's point size will be enlarged or reduced in accordance with the range of point sizes available for the selected font. The default font for WindowMaker is System and cannot be sized. Choose a Windows True-Type font before changing the size.

Editing Text Objects

To change the text in an object

1. Select the object or button with the text.
2. On the **Special** menu, click **Substitute Strings**. The **Substitute Strings** dialog box appears.

Tip To quickly access the dialog box, right-click the text object, point to **Substitute** and then click **Substitute Strings**.



Current String: Temp = 00.0 deg F

New String: Temp = 00.0 deg F

For more information on Analog Input/Display links, see Chapter 7, "Creating Animation Links."

3. In the **New String** box, type the new string and then click **OK**.

Tip You can also use this command on strings that are included in a symbol or cell and to change the label on buttons drawn with the Button tool.

When you change a text string, it retains all of it's original attributes, that is font, style, color, and so on. All normal text formatting also applies to numeric values.

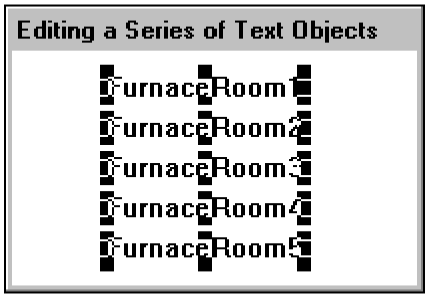
You can also select and edit multiple string objects at the same time.

Replacing a Portion of a Text Object

You can change a portion of a text object's text and InTouch will automatically make the change to all selected text objects using the same text.

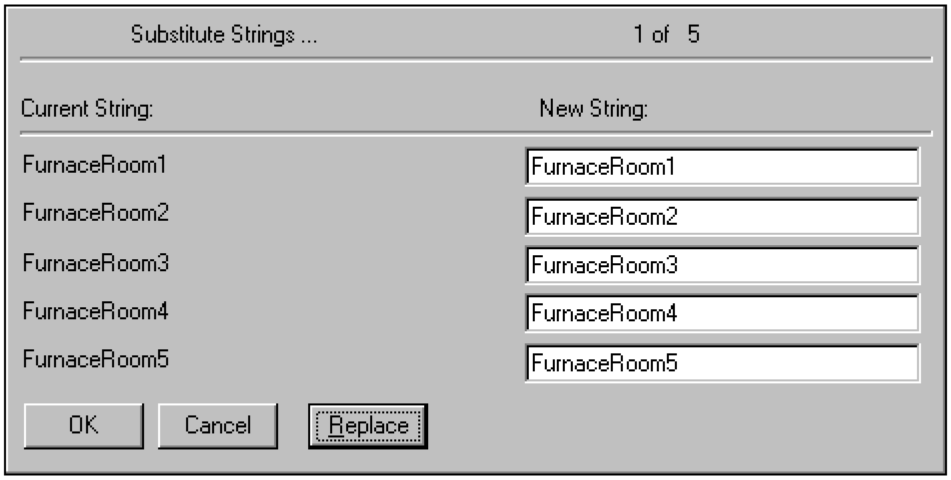
To change a portion of text in a series of text objects

1. Select all of the text objects.



2. On the **Special** menu, click **Substitute Strings**. The **Substitute Strings** dialog box appears.

Tip To quickly access the dialog box, right-click a text object, point to **Substitute** and then click **Substitute Strings**.

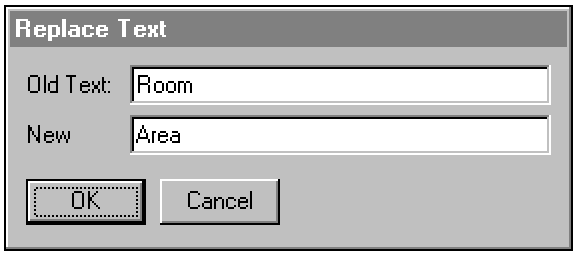


Substitute Strings ... 1 of 5

Current String:	New String:
FurnaceRoom1	FurnaceRoom1
FurnaceRoom2	FurnaceRoom2
FurnaceRoom3	FurnaceRoom3
FurnaceRoom4	FurnaceRoom4
FurnaceRoom5	FurnaceRoom5

Tip If you right-click any of the text entry boxes in any dialog box, a menu appears displaying the commands that you can apply to the selected text.

3. Click **Replace**. The **Replace Text** dialog box appears.

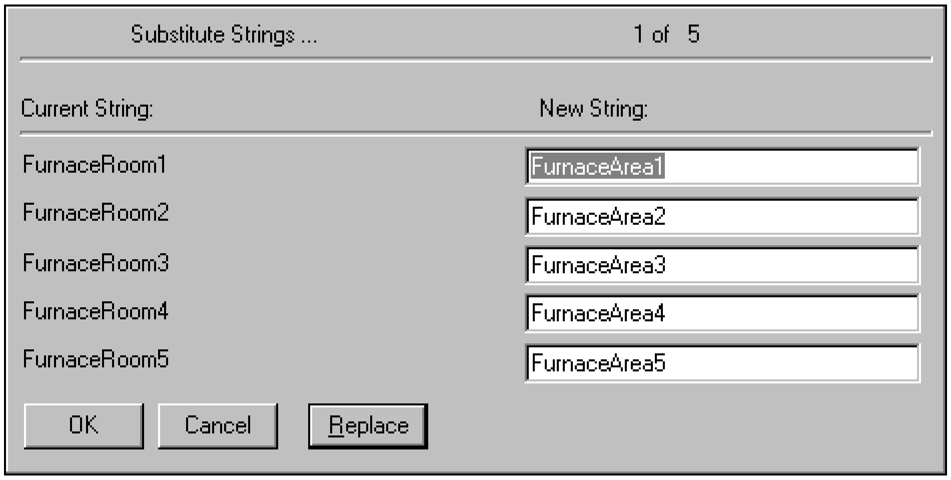


Old Text: Room

New: Area

4. In the **Old Text** box, type the portion of the string that you want to replace.

5. In the **New** box, type the replacement text.
6. Click **OK**. The **Substitute Strings** dialog box reappears showing the change made to the selected text strings:



Substitute Strings ... 1 of 5

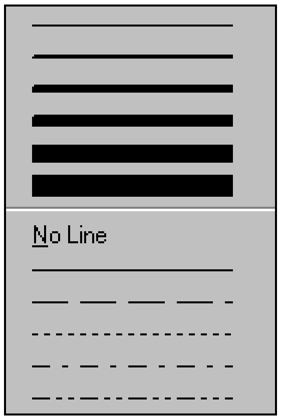
Current String:	New String:
FurnaceRoom1	FurnaceArea1
FurnaceRoom2	FurnaceArea2
FurnaceRoom3	FurnaceArea3
FurnaceRoom4	FurnaceArea4
FurnaceRoom5	FurnaceArea5

7. Click **OK**. All of the selected text objects will automatically be modified.

Working with Lines and Outlines

You can change the style and width of a line object including the outlines around ellipses, rectangles, polygons and bitmaps or images. You can apply a line style or width change to a single selected object or multiple selected objects.

The **Line** menu is divided into two sections. The top section contains the line widths and the bottom section contains the line styles. For example:



To apply a line command

Select the object and then on the **Line** menu, click the desired line style or width.

Tip If you do not select an object when you select a line style or width, the change will be applied to the default settings for all line tools in the **Wizard Toolbar**.

Note You can only change the width of solid lines. Broken lines can only be single pixel wide. Wider lines take longer to draw in runtime.

To remove an object's outline

Select the object and then on the **Line** menu, click **No Line**. The object's outline will be removed.

Working with Wizards

Wizards save you a considerable amount of time during application development. They are easy to use and easy to configure. To configure a wizard, you install it, select it in the **Wizard Selection** dialog box, paste it into your window and then double-click it. Its respective configuration dialog box appears (if it is a wizard that can be configured).

For example, if you wanted to use a slider wizard, you would need to configure items such as the tagname effect, the minimum and maximum range labels for the slider, the fill color, and so on. You can save a considerable amount of development time by using Wizards because you don't have to draw the individual components for the object, or set the value ranges for the object, or animate the object.

The FactorySuite InControl program includes the following five wizards that you can place in an InTouch window. These wizards allow easy and effective interaction between InControl and InTouch.

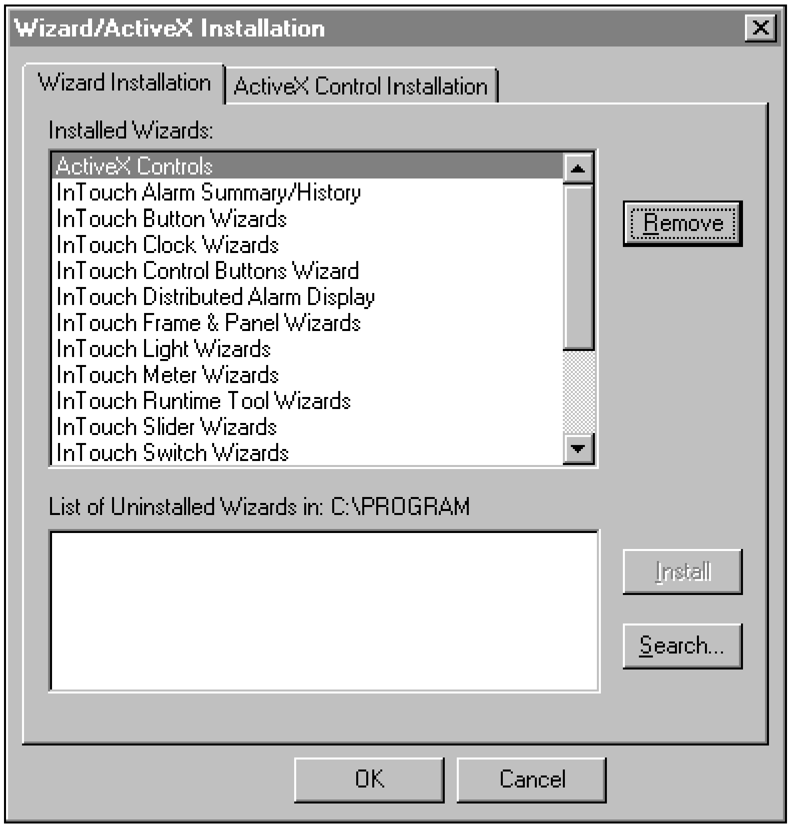
Wizard	Description
InControl Project	Launches an InControl project. This starts InControl for the specified project and allows the operator to use all the InControl functions to edit, compile, download, and run the programs in that project.
Configure Runtime Engine	Starts the runtime engine and select the node on which it runs.
InControl Mode	Used to set programs currently downloaded to the runtime engine to the specified mode (Run, Pause, Single Step).

Wizard	Description
InControl Edit	Launches an individual program in a project. This starts InControl within the development environment, at a specified line within the specified program. You can use all the tools available in the development environment to edit, compile, download, and run the program.
InControl Runtime Add Tag	Associates InTouch tagnames with InControl symbols (variables).

To install or remove wizards

1. On the **Special** menu, point to **Configure** and then click **Wizard/ActiveX Installation** or in the Application Explorer, double-click **Wizard/ActiveX Installation**. The **Wizard/ActiveX Installation** dialog box appears with the **Wizard Installation** property sheet active.

Tip In the Application Explorer, you can also right-click **Wizard/ActiveX Installation** and then click **Open**.



Installed Wizards:

ActiveX Controls
InTouch Alarm Summary/History
InTouch Button Wizards
InTouch Clock Wizards
InTouch Control Buttons Wizard
InTouch Distributed Alarm Display
InTouch Frame & Panel Wizards
InTouch Light Wizards
InTouch Meter Wizards
InTouch Runtime Tool Wizards
InTouch Slider Wizards
InTouch Switch Wizards

List of Uninstalled Wizards in: C:\PROGRAM

Note The maximum number of Wizards that can be installed concurrently is 43.

2. In the **Installed Wizards** list, select the wizard(s) that you want to remove from your application and then click **Remove**. A message box appears asking you to confirm the deletion.

Note The **Remove** button is only active when wizards are displayed in the Installed Wizards list.

Tip To select a group of wizards, click the first wizard in the list, hold down the shift key and click the last wizard that you want to select. All wizards in the list between your first and last selection will be selected. To select multiple wizards that are not consecutively listed, click the first wizard, hold down the ctrl key and then click the next wizard. Repeat this for all wizards that you want to select.

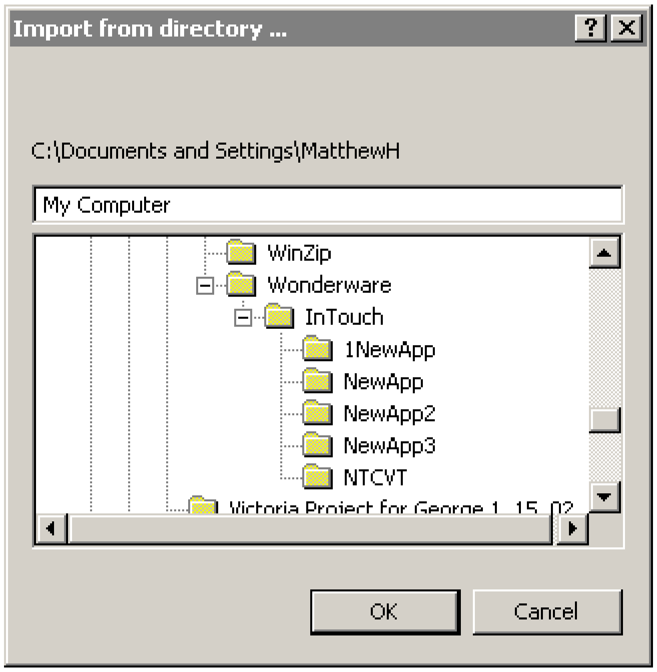
3. Click **Yes** to remove the wizard. The removed wizard(s) is moved to the **List of Uninstalled Wizards** list.

Note When you remove a wizard, it is not deleted. However it is no longer loaded into memory.

4. To install wizards, select them in the **List of Uninstalled Wizards** and then click **Install**.

Note The **Install** button is only active when wizards are displayed in the **List of Uninstalled Wizards** list.

5. Click **Search** if you want to install wizards from another directory. The **Search for Wizard files** dialog box appears.


Import from directory ...

C:\Documents and Settings\MatthewH

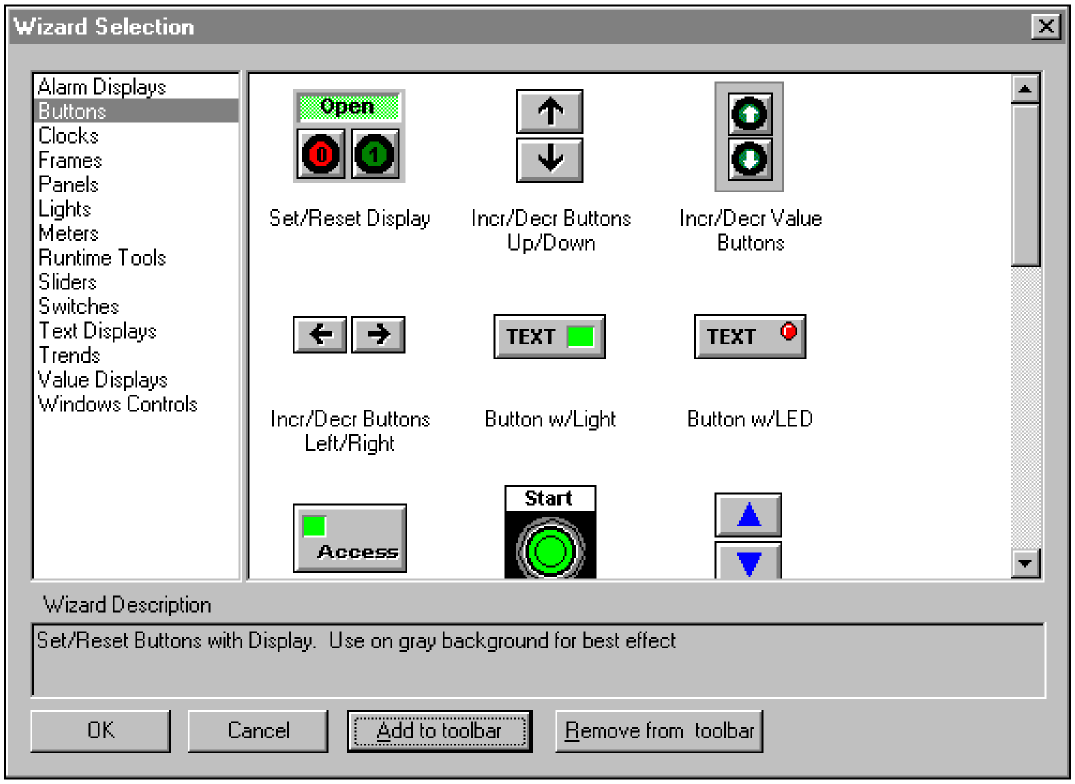
My Computer

6. Locate the directory containing the wizards that you want to install and then click **OK**. The wizard installation dialog box reappears.

   Any Wizards that were found appear in the **List of Uninstalled Wizards** list and you can now install them as previously described.

To place a wizard in a window

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



Set/Reset Buttons with Display. Use on gray background for best effect

2. In the list of wizards, click the category of wizards that you want to use.

   All available wizards in that category will be shown the display area. For example, if you select **Buttons**, all available button wizards will immediately be shown in the display area.

3. Select the wizard that you want to use and then click **OK** or double-click the wizard. The dialog box will close and your window reappears.

Tip To add the wizard to the **Wizards/ActiveX Toolbar**, click **Add to toolbar**. Once you add a wizard to the **Wizards/ActiveX Toolbar**, you can select it and paste it into your open window at any time.

Note The number of wizards that you can add to the toolbar is limited to your system resources.

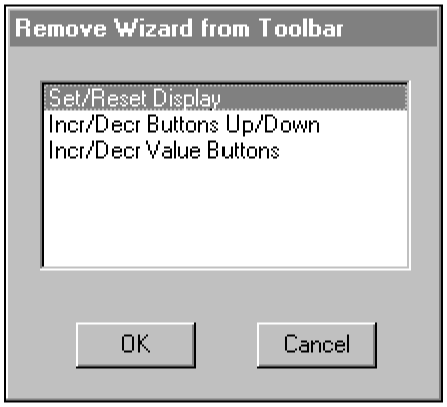
4. The cursor will change to a corner symbol ⌈W when you return to the window. Click the location in the window where you want to paste the wizard.
5. Double-click the wizard to configure it (if applicable).

Note Some toolbar functions may be used to modify applicable wizards directly. For example, the **Reduce Font** tool, **Line Color** tool, **Fill Color** tool, and so on.

For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."

To remove wizards 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.


Remove Wizard from Toolbar

Set/Reset Display
Incr/Decr Buttons Up/Down
Incr/Decr Value Buttons

3. Select the wizard(s) that you want to remove from the toolbar.
4. Click **OK**.

InTouch Windows Control Wizards

The windows control wizards are complex objects. Unlike other wizards, they provide enhanced functionality through InTouch QuickScripts. They can be used for editing data objects and operator inputs. Windows control wizards also have InTouch tagname **.fields** and some of their properties are accessible both during development and runtime when their QuickScript functions are used.

You can use windows control wizards in your InTouch application to display text/data, gather user input or offer choices for the user at runtime. Choices may be in the form of list boxes, check boxes, combo boxes and radio (option) buttons. You can use the text boxes to display or input text/data.

When you configure a windows control wizard, you must specify a **Control Name** to identify the control. InTouch uses the **Control Name** to identify the control when you execute a windows control QuickScript function. Therefore, you must also specify the **Control Name** parameter in the QuickScript function. For example:

```
SetPropertyD ( "ControlName.Property", Discrete );
```

For more information on using the windows control QuickScript functions, see your *InTouch Reference Guide*.

Control Names do not add to the application's tagname count and must be unique for each control. Tagnames, although not required, are essential for productive use of the control. For example, selecting an item in a list box is not useful if the selected item is not automatically assigned to a tagname, thus making it accessible to InTouch.

Windows controls have properties (similar to tagname **.fields**) that can be modified at development (WindowMaker) and runtime (WindowViewer). They also support specific QuickScript functions that can be processed at runtime to modify lists, load files, disable controls, and so on.

Note To function properly, windows control objects cannot overlap each other. For verification, select the object in WindowMaker and insure that the object's handles do not touch another object.

Windows Controls that accept keyboard input do not work in SuiteVoyager.

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.

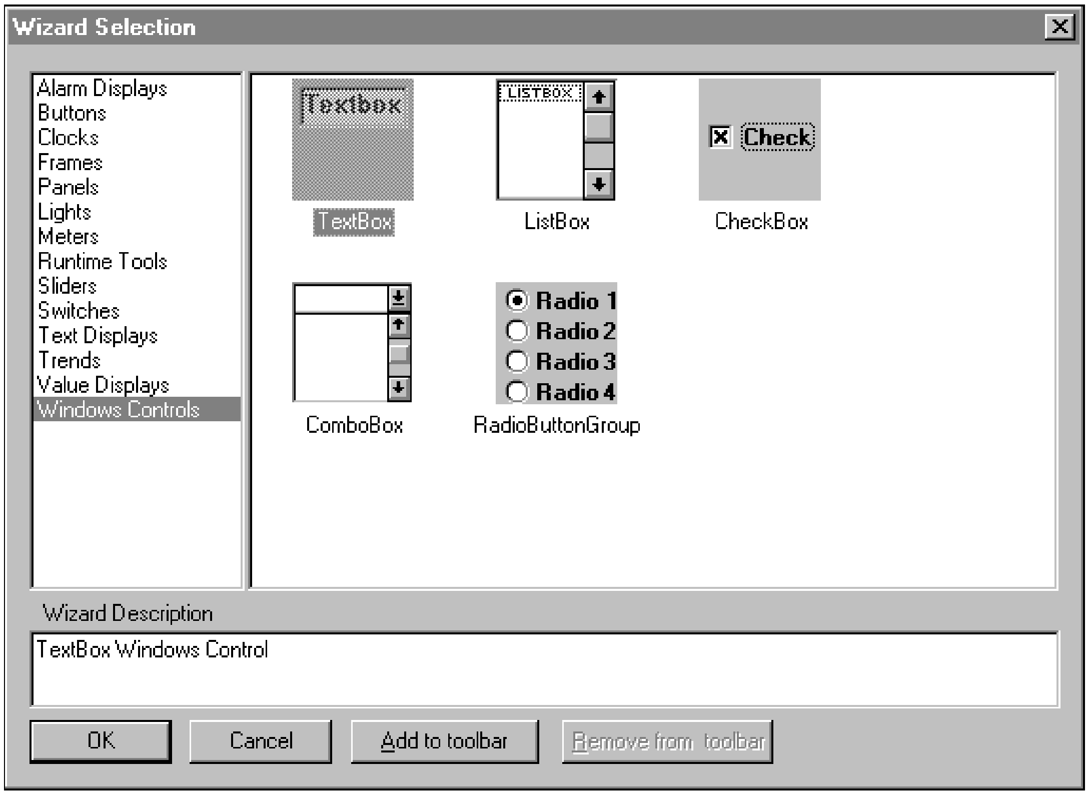
For more information on using the windows control **.fields**, see your online *InTouch Reference Guide*.

Using InTouch Windows Control Wizards

The Windows control wizards available include: Text Boxes, Check boxes, Combo Boxes, List boxes and Radio (Option) Buttons. The windows control wizards also have tagname **.fields** and QuickScript functions that you can use to dynamically control them in runtime.

Tip Windows control wizards are pasted into your windows just like any other wizard.

For more information on pasting wizards, see "To place a wizard in a window."



TextBox Windows Control

Tip To achieve the best windows control 3-D effect, select a gray background for your window. If your window color cannot be gray, place a gray "Panel Wizard" behind the windows control wizard.

Windows Control Usability Guidelines

It is very important that you follow the guidelines below when you are using windows control wizards:

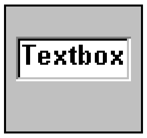
1. Windows control wizards work properly when they do not overlap other windows control wizards or other normal graphic objects.

   Tip To verify that the windows control wizard is not overlapping any other object, select it in WindowMaker. Verify that none of its selection handles are touching another graphic object on the screen.

2. Windows control wizards should be used sparingly and intelligently.

   Note Placing 10 to 20 windows control wizards in one window results in non-intuitive, hard to navigate displays. When it is necessary for you to use numerous windows control wizards, we recommend that you call other dialog boxes with additional windows control wizards.

Text Box Control Wizard

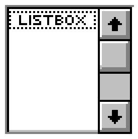
	Text boxes control wizards are versatile controls that can be used to get input from the user or to display text such as a notepad file (ASCII flat files only). You can configure text boxes to allow user input or as read-only for display purposes only You can only assign **Message** type tagnames to a text box control wizard.

Text box control wizard QuickScript examples:

```
wcLoadText("TextBox_1",FileName);
wcSaveText("TextBox_1",FileName);
```

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

List Box Control Wizard

	List box control wizards display a list of choices to the user. The choices are displayed vertically in a single column. If the number of items exceeds what can be displayed in the list box, scroll bars will automatically appear on the control. List boxes require the user to choose from a list and do not allow user input. You can only assign **Message** type tagnames to a list box control wizard.

List box control wizard QuickScript example:

```
IF (ItemToAdd == "") THEN
    Show "Cannot Add Blank";
ELSE
    wcAddItem("ListBox_1",ItemToAdd);
```

{Get the index of the item we just added.}

{Since the list is sorted, we cannot assume anything about the new items location.}

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

{Now, set the Item Data specified on the screen by the user.}

{From this point on, this item will have this data associated with it.}

{It allows you to associate a number with a string; the string being displayed in the list.}

```
    wcSetItemData("ListBox_1",ListBox_NewIndex,
        ListBox_ItemData);
```

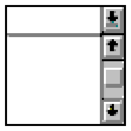
```
{Since we just added an item, update the "NumItems" variable.}
    GetPropertyI("ListBox_1.ListCount",ListBox_NumItems);
ENDIF;
```

List box and combo box control wizards use an internal, one-based numbering system (item index) that automatically assigns a number to each item in the list. For example, the first item in the list is assigned the number 1, the second is number 2, and so on. The item index is a 32-bit integer that is used as a parameter for windows control "item"

Note When using list boxes and combo boxes with the **wcLoadList()** and **wcSaveList()**, specific formatting and information must be provided.

For more information, on the windows control QuickScript functions, see your *InTouch Reference Guide*.

Combo Box Control Wizard



Combo box control wizards combine the features of a text box and a list box. The choices are displayed vertically in a single column. If the number of items exceeds what can be displayed in the list box, scroll bars will automatically appear on the control. Combo box control wizards allow the user to make a selection, either by typing text or selecting an item from the list. You can only assign **Message** type tagnames to a combo box control wizard.

There are three styles of combo boxes:

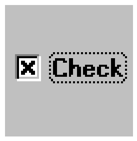
Type	Description
Simple	Combo boxes display their list at all times. To display the entries in the list box, the list box must be drawn large enough to display all entries. A vertical scroll bar is automatically inserted if there are more entries than can be displayed. Simple combo boxes allow the user to type in choices that are not in the list or will display the first item in the list matching the typed letters. If no match is found, the top of the list is displayed.
Drop Down	Combo boxes allow the user to either type text directly or click an arrow to open a list of choices. As with the simple combo box, this control allows the user to type choices not on the list or will display the first item in the list matching the typed letters. If no match is found, the top of the list is displayed.
Drop Down List	Combo boxes are similar to simple list boxes. They display a list of choices to select. Unlike list boxes, however, the list is not displayed until the arrow is clicked. This type of control is used to conserve screen space.

Combo box control wizard QuickScript example:

```
wcAddItem("ComboBox_1", UserMessage );
```

Where: *UserMessage* is a tagname assigned to a string input link. When the operator types a new message and then clicks this action push button, linked to the **"On Down"** QuickScript, the message is displayed in the combo box wizard with the control name "ComboBox_1."

Check Box Control Wizard

	A check box indicates whether a particular condition is on/off, true/false or yes/no. Check boxes work independently of each other, allowing the user to select or deselect any number of check boxes at the same time. Check boxes return a discrete value. They return 0 if not selected and 1 if selected. You can only assign **Discrete** type tagnames to a check box control wizard.

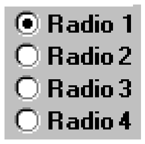
Check box control wizard QuickScript example:

```
{ Clear any previous machine }

Machine = "";

IF (Cutter_Selected) THEN
   Machine = Machine + "Cutter";

ENDIF;

IF (Mixer_Selected) THEN
   Machine = Machine + "Mixer";

ENDIF;
```

Where: *Cutter_Selected* is the tagname assigned to the control name "Checkbox_1" in the first check box wizard. *Mixer_Selected* is the tagname assigned to the control name "Checkbox_2" in the second check box wizard.

Machine is a tagname assigned to a string output link that displays the name for the check box selected.

Radio (Option) Buttons Control Wizard

	Radio, or option buttons present a set of choices for the user. Unlike check boxes, radio buttons operate as part of a group. Selecting one radio button immediately clears all of the other buttons in the group. Radio buttons return an integer value. The value of a radio button control corresponds to the selected radio button.

For example, if Radio 1 option is selected, the current value is 1. If the Radio 4 option is selected, the value is 4. You can only assign **Integer** type tagnames to a radio button control wizard.

Radio (option) button control wizard QuickScript example:

```
SelectedMachine=1;
```

Where: *SelectedMachine* is an integer tagname assigned to the radio button wizard with the control name "RadioButtonGroup1. This is a Window **"On Show"** QuickScript that sets the value of the *SelectedMachine* tagname to 1. (This sets the default to select the first radio button in the group when the window is initially shown.) When the operator selects another radio button, the value of *SelectedMachine* will change according to the button selected. For example, if the radio button group had 4 choices and the operator selected the third button, *SelectedMachine*'s value would be set to 3.

Configuring a Windows Control Wizard

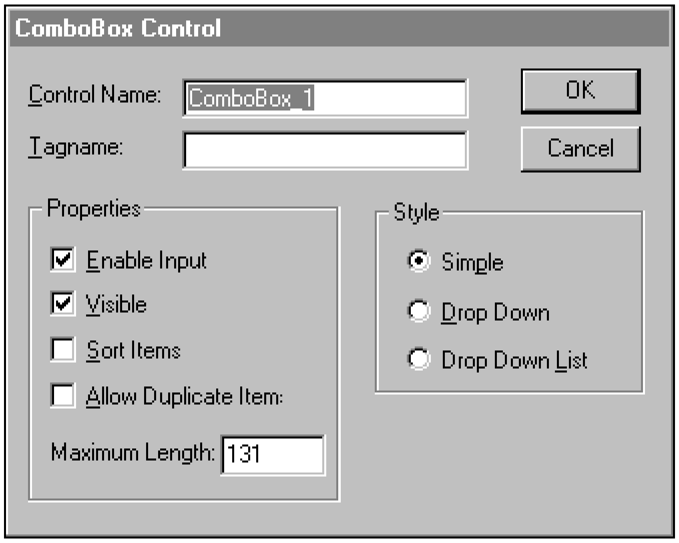
Each windows control wizard has its own unique configuration dialog box based on its intended functionality. The options shown in the dialog box are configurable properties not available through other WindowMaker tools. However, properties such as color, font style and size are modified by using the respective WindowMaker tools.

Most of the windows control wizards support QuickScript functions. For example, you could create a Data Change QuickScript to load and clear the lists, add and delete items in the lists, and so on.

For more information on the windows control QuickScript functions and **.fields** see, your *InTouch Reference Guide*.

To configure a windows control wizard

1. Paste the wizard in your window.
2. Double-click it. Its respective configuration dialog box appears. For example:



ComboBox Control

Control Name: ComboBox_1

Tagname:

Properties

[✓] Enable Input

[✓] Visible

[ ] Sort Items

[ ] Allow Duplicate Item:

Maximum Length: 131

Style

● Simple

○ Drop Down

○ Drop Down List

Tip If you right-click any of the text entry boxes in any dialog box, a menu appears displaying the commands that you can apply to the selected text.

3. In the **Control Name** box, type a unique name to identify the windows control.

   You must specify a unique **Control Name** for each windows control wizard you use for correct operation at runtime. WindowViewer uses the **Control Name** to identify the control when you execute a windows control QuickScript function. **Control Names** must start with an alpha character (underscores and numbers can be used after the first alpha character) and cannot include any special characters.

4. Specifying a **Tagname** when you configure the windows control is optional. However, if you do specify a tagname, its value is automatically set to the **.Value** property of the control (that is, the item index for the item selected in a list box).

5. Type in all other required entries and set all parameters for the particular windows control that you are configuring, as applicable.

6. Click **OK**.

Windows Control Wizard Properties

Windows control wizards have properties like InTouch tagname **.fields**. They can be read-write or read-only. Some properties are accessible at development and some at runtime. 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 will not be visible in the window. Similar to InTouch tagnames, **.Value** is the default property for the windows control wizard.

In WindowMaker, windows control wizard properties such as text font, size and color are modified using the respective WindowMaker toolbars or menu commands. The properties that are not supported by the toolbar or menu commands, are configured from within the wizard's configuration dialog box. Other properties of windows control wizards are dynamic and are read-write or read-only in runtime. This is similar to runtime properties of InTouch tagnames (**.fields**) such as **.Value** and **.Name**. Unlike InTouch tagnames, runtime properties for windows control wizards are accessed through QuickScript functions not animation link expressions.

The runtime properties may be either read-write or read-only depending on the property. The **GetProperty()** and **SetProperty()** QuickScript functions must be used to control or retrieve these properties. The following briefly describes each windows control property:

Property	Description
.Caption	Determines the "message" to be displayed with the check box.
.Enabled	Determines whether the control object can respond to operator-generated events.
.ListCount	Determines the number of items in a list box or combo box.

Property	Description
.ListIndex	Determines the corresponding index (*tagname* or *number*) of the currently selected item in the list. **Note** Index is a number that defines a specific item in a 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 entered new text into the text entry field of the control.
.NewIndex	Returns the corresponding integer index (*tagname*) of the last item added to the list box or combo box through the **wcAddItem()** or **wcInsertItem()** functions.
.ReadOnly	Determines whether the contents of the text box are read-only or read-write.
.TopIndex	Determines the corresponding integer index of the top most item in the list box.
.Value	The default property for all InTouch windows control wizards. Changes made to this property are synchronized in the InTouch tagname and the windows control wizards.
.Visible	Determines whether the windows control is visible in the window. The windows control wizard properties do not appear in the **Choose field name** dialog box.

For example:

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

Where:

Parameter	Description
ControlName	The **Control Name** configured for the windows control wizard, for example, CheckBox_1 or the name of an alarm object, for example, AlmObj_1.
.Property	Windows control or alarm object property.
Tagname	A valid InTouch tagname (of the same type to be returned) that will hold the property value when the function is processed.

For more information on windows control wizards, see "InTouch Windows Control Wizards. "

Windows Control Wizard Functions

The following briefly describes the InTouch QuickScript functions that are available for use with the InTouch Windows Control wizards:

Function	Description
wcAddItem	Adds the supplied string to the list box or combo box.
wcClear	Removes all items from the list box or combo box.
wcDeleteItem	Deletes the item associated with the item index argument in both list or combo boxes.
wcDeleteSelection	Deletes the currently selected item from the list. Applies to list boxes and combo boxes
wcErrorMessage	Given an error number, wcErrorMessage(), returns a string message describing the error. Applies to list boxes, text boxes, combo boxes, radio buttons and check boxes.
wcFindItem	Determines the corresponding index of the first item in the list box or combo box that matches the supplied string.
wcGetItem	Returns the value property of an item string associated with a corresponding index in a list box or combo box.
wcGetItemData	Retrieves the integer value associated with a list item in a list box or combo boxes.
wcInsertItem	Inserts a string into a list box or combo box.
WcLoadlist	Replaces the contents of the list box or combo box with new items.
wcLoadText	Replaces the contents of the text box with a new string.
wcSavelist	Replaces the contents of a filename with the items in a list object.
wcSaveText	Saves the text contained in a text box to a filename.
wcSetItemData	Assigns an integer value to an item in a list box.

Working with ActiveX Controls

ActiveX controls, originally known as OLE controls or OCXs, are standalone software components that perform specific functions in a standard way. ActiveX controls define standard interfaces for reusable components. ActiveX controls are not separate applications. Instead, they are servers that are placed into a control container. To use ActiveX controls, they must be placed in an ActiveX container. InTouch is an ActiveX container. Microsoft VisualBasic and internet browsers are also ActiveX containers.

ActiveX controls behave exactly like InTouch Wizards, except they bring compelling new functionality to InTouch applications. You can create ActiveX controls by using Visual Basic, Microsoft VC++ or other 3rd party development tools. You can also buy ActiveX controls from third-parties for specific functionality. These controls are packaged in the OCX form. Wonderware's FactorySuite InTrack component also provides you with several ActiveX controls. Additionally, IndustrialSQL's ActiveTrend allows you to run the IndustrialSQL Trend program (or a functional subset) from within InTouch and ActiveEvent allows you to notify the IndustrialSQL Event sub-system when an event has occurred in another application.

There are three main components of ActiveX controls: *properties*, *methods* and *events*.

- Properties are very similar to variables that you can modify, for example, Calendar.day, Control.height, and so on.
- Methods are similar to script function calls that you can call from the container. For example, **Browser.Navigate("http://www.wonderware.com")**, **Engine.start()**.
- Events occur through the ActiveX container. For example, **Control.click (shift)**. **FileViewer.DoubleClick (name)**, and so on.

InTouch allows you to access ActiveX control properties, methods and events. You can associate these properties with InTouch tagnames or you can access them through InTouch QuickScripting.

Note In order for an ActiveX Event script to function properly, the ActiveX control for which the script was created, must be loaded into memory. If the window containing an ActiveX control is closed, its ActiveX Event scripts, or any other InTouch QuickScripts containing script functions associated with that ActiveX control, will not execute properly.

You can use one or more ActiveX controls in your InTouch application. InTouch allows you to easily select and paste an ActiveX control into any application window and to add them to your **Wizards/ActiveX Toolbar**. You can also import ActiveX Event scripts from one application to another.

To use an ActiveX control in InTouch

1. Install the ActiveX control(s) that you want to use.
2. Select and paste the ActiveX control into a WindowMaker window.
3. Configure the ActiveX control's properties and assign them to tagnames.
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.

The following WindowMaker edits can be made to an ActiveX control:

- An ActiveX control's size can be changed, if sizing is supported by the control.
- ActiveX controls can be duplicated, cut, copied, pasted and deleted.
- All aligning commands (left, right, top, bottom, centerpoint) can be applied to an ActiveX control.

- ActiveX controls can be added to the **Wizards/ActiveX Toolbar**.
- ActiveX controls can be included with other objects when creating a cell.
- The WindowMaker menu commands and their equivalent toolbar tools can be used to directly modify many ActiveX properties. For example, Reduce Font, Line Color, Fill Color, and so on.

InTouch does not support the following types of ActiveX controls:

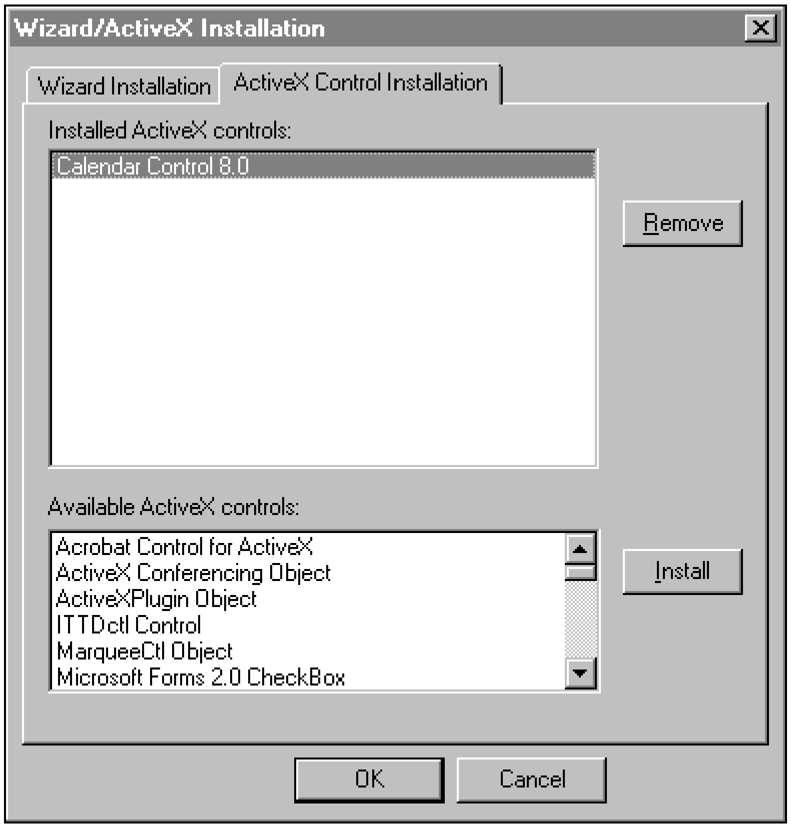
- Windowless Controls
- Simple Frame Site (Group Box)
- Containers
- Data Controls
- Dispatch Objects
- Arrays, Blobs, Objects, Variant Types

To install or remove an ActiveX control

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

Tip In the Application Explorer, you can also right-click **Wizard/ActiveX Installation** and then click **Open**.

2. Click the **ActiveX Control Installation** tab to activate the **ActiveX Control Installation** property sheet:



Installed ActiveX controls:

Calendar Control 8.0

Available ActiveX controls:

Acrobat Control for ActiveX
ActiveX Conferencing Object
ActiveXPlugin Object
ITTDctl Control
MarqueeCtl Object
Microsoft Forms 2.0 CheckBox

3. In the **Installed ActiveX controls** list, select the control(s) that you want to remove from your application and then click **Remove**. An interactive message box appears asking you to confirm the deletion.

Tip To select a group of controls, click your first selection, hold down the **Shift** key and select your last selection. All controls in between will be selected as well. To select multiple controls that are not consecutively listed, click the first control and then hold down the **Ctrl** key as you click another.

4. Click **Yes** to remove the control(s). The removed control(s) is moved to the **Available ActiveX controls** list.

Note When you remove a control, it is not deleted. However it is no longer loaded into memory. Therefore, it will not function properly.

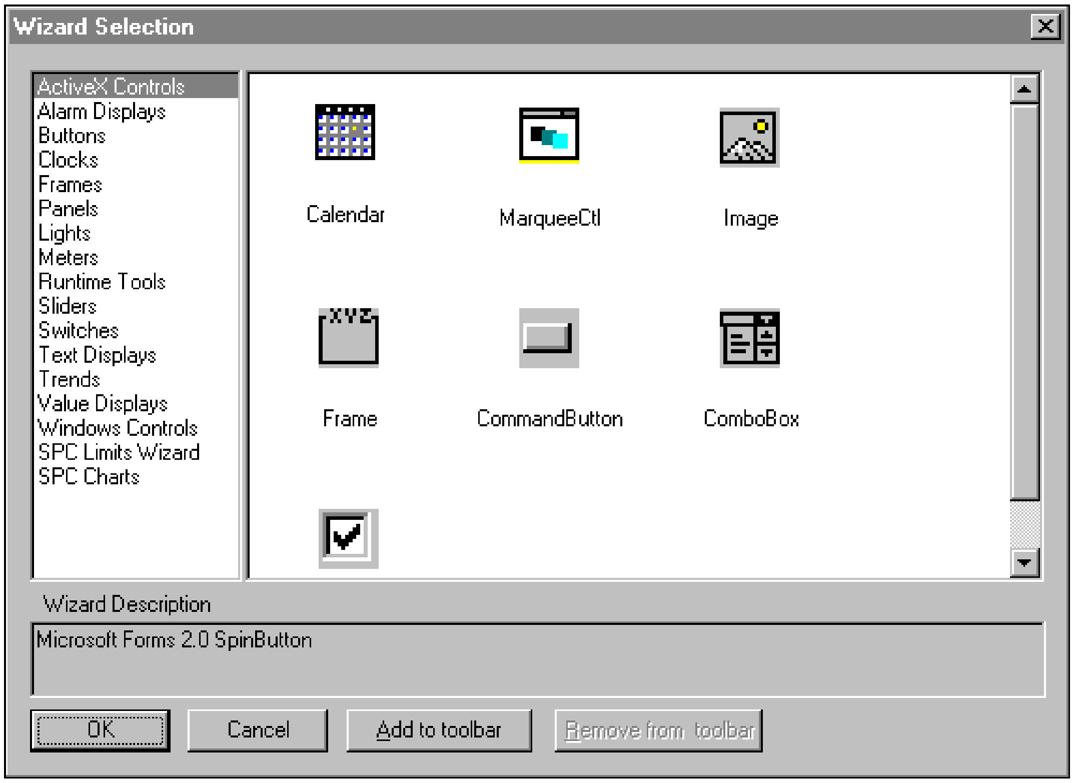
5. To install ActiveX controls, select them in the **Available ActiveX controls** list and then click **Install**.

Note The **Install** button is only active when controls are displayed in **Available ActiveX controls** list.

6. Click **Close**.

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.



Microsoft Forms 2.0 SpinButton

2. In the list of wizards, click the **ActiveX Controls** category. All available ActiveX controls will be shown the display area.
3. Select the ActiveX control that you want to use and then click **OK**, or double-click the control. The dialog box will close and your window reappears.

Tip To add the ActiveX control to the **Wizards/ActiveX Toolbar**, click **Add to toolbar**. Once you add a control to the **Wizards/ActiveX Toolbar**, you can select it and paste it into your open window at any time.

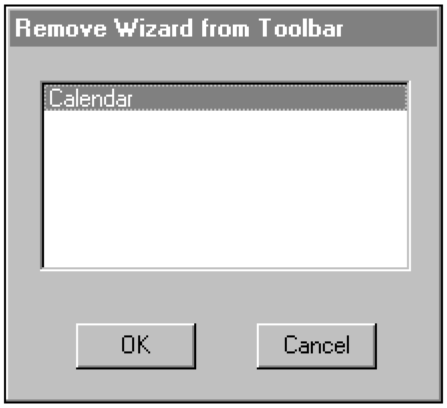
Note The number of ActiveX controls that you can add to the toolbar is limited to your system resources.

4. The cursor will change to the corner symbol, ⌜W, when you return to the window. Click the location in the window where you want to paste the ActiveX control.
5. Double-click the control to configure it properties.

For more information on the WindowMaker toolbars, see Chapter 1, "WindowMaker Program Elements."

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(s) that you want to remove from the toolbar.
4. Click **OK**.

Configuring an ActiveX Control

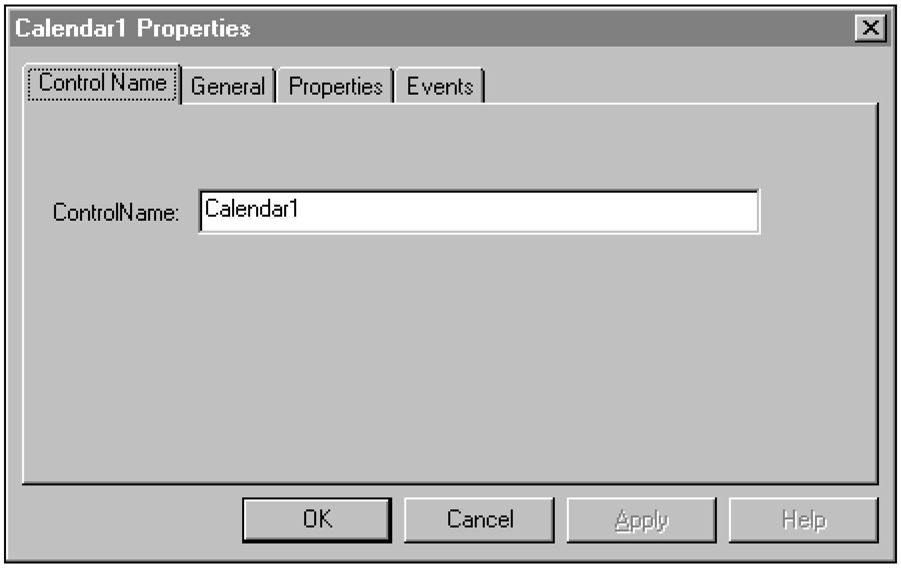
When you paste an ActiveX control into an InTouch window, you must configure its properties to interact with InTouch. Each control must be named for reference from InTouch QuickScripts. A default control name, such as Calendar1, will be generated when you paste the ActiveX control. (This control name will be global within your InTouch application.)

The ActiveX control's properties must be assigned to InTouch tagnames. Each property type must be assigned to an equivalent InTouch tagname type.

To name an ActiveX control

1. Paste the ActiveX control into your WindowMaker window.
2. Double-click the control, or right-click the control and then click **Properties**. The control's respective **Properties** dialog box appears.

Note Each ActiveX control's Properties dialog box is unique to the control. The number of tabs displayed is based upon the properties of the particular control. Some ActiveX controls may require you to configure more properties than others do. For example some controls may require you to configure their Colors and Fonts, while others may not have these properties. However, for all ActiveX controls, InTouch adds three tabs; **Control Name**, **Properties** and **Events**. For example:



ControlName: Calendar1

3. Click the **Control Name** tab and then type a unique name for the ActiveX control in the **ControlName** box.

   You must define a unique name for each ActiveX control used in your InTouch application. The Control Name is used in script functions to identify the control. For example:

```
#Calendar1.day = Tag1;
#Calendar1.year = 1997;
```

Note By default, the Control Name is determined by the ProgID for that control. ProgIDs are names that are entered into the system registry when ActiveX controls are installed on a computer. When an instance of that control is placed in an InTouch application, the control's ProgID is obtained from the system registry and an index number is appended to it, resulting in a Control Name, such as **Calendar1**.

If you use the default Control Name, a new instance of a control is created and given a unique name under the following circumstances:

- Selecting **Duplicate** on the **Edit** menu
- Selecting **Cut** or **Copy** and then **Paste** on the **Edit** menu
- Selecting **Save Window As** on the **File** menu
- Clicking **Undo** and then **Redo**
- Importing a window that contains a control

Changing an ActiveX Control Name

It is best not to change the names of ActiveX controls; however, there are special cases where this may be necessary. Let's assume that you use the default Control Name, for example, **Calendar1**. Later you delete the ActiveX control and then recreate it. InTouch will automatically increment the Control Name. In this case, the duplicate ActiveX control's name will be **Calendar2**. In order for your existing scripts to work with this new control, you must rename the new control from **Calendar2** to **Calendar1**.

ActiveX controls must have unique names. When you edit the Control Name and click **OK** or **Apply**, the name is checked against a table of existing Control Names. If the name does not already exist in the table, the name of the control will be changed and saved. If the name already exists, an error message appears. You must specify a unique name for the control.

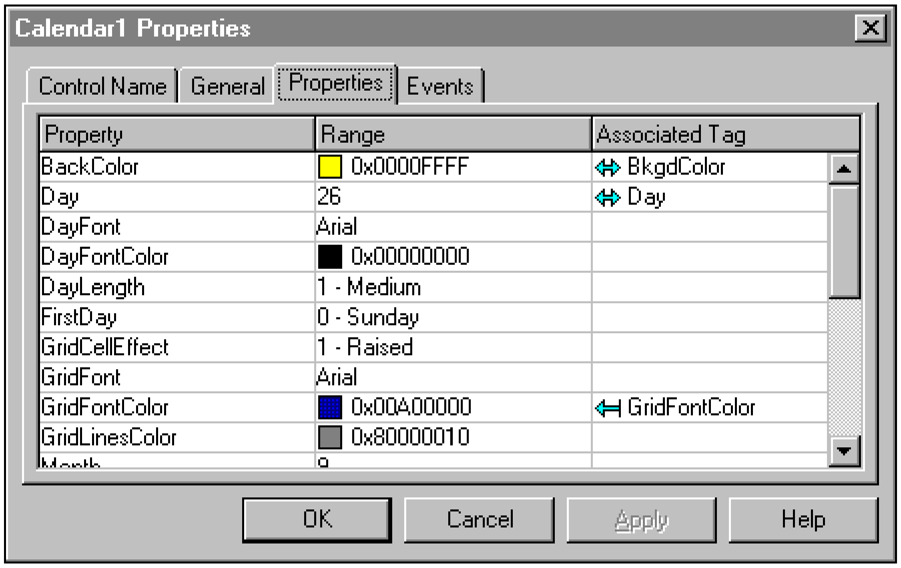
Configuring ActiveX Control Properties

The properties that you can configure for a particular ActiveX control are determined by the ActiveX control designer. Each ActiveX control's **Properties** property sheet displays three columns: **Property**, **Range** and **Associated Tag**. The **Property** and **Range** columns are read-only. The **Associated Tag** column is used to associate InTouch tagnames with the respective property in the **Property** column.

Note When you click certain items in the **Range** column an arrow appears that you can click to view the list of possible values for the item. The items in the list area for viewing purposes only and cannot be changed.

To configure an ActiveX control's properties

1. Click the **Properties** tab in the ActiveX control's **Properties** dialog box to activate the **Properties** property sheet:


Calendar1 Properties

Property	Range	Associated Tag
BackColor	🟨 0x0000FFFF	⇔ BkgdColor
Day	26	⇔ Day
DayFont	Arial	
DayFontColor	⬛ 0x00000000	
DayLength	1 - Medium	
FirstDay	0 - Sunday	
GridCellEffect	1 - Raised	
GridFont	Arial	
GridFontColor	🟦 0x00A00000	⇐ GridFontColor
GridLinesColor	▨ 0x80000010	
Month	9	

2. Click in the middle of each cell in the **Associated Tag** column and then type a tagname for the respective property.

Tip If you type in a tagname that is not defined in the Tagname Dictionary, you will be prompted to define it now.

If you double-click a blank cell, or click the ... button, the Tag Browser appears displaying the tagnames for the selected tag source. Double-click the tagname that you want to use, or select it and then click **OK**. The tagname is automatically inserted into the cell.

Note There is a timing problem between the initialization of a tagname and the creation of an ActiveX control. Due to this timing problem, it is impossible to guarantee that the initial value of the tagname will be the value of the associated property in the ActiveX control.

To resolve this issue, create an association between the Startup event and an InTouch QuickScript. In the QuickScript, implement the logic that sets the tagname's value into the property of the control. This can be done by using an assignment statement. For example:

```
#ThisControl.Property = MyTag;
```

Once the event fires, the QuickScript will execute and the property of the control and the tagname value will be in sync. Thereafter, normal notifications will occur between the property and the tagname.

For more information on the Tag Browser, see Chapter 6, "Tagname Dictionary."

3. Once you specify the tagname, double-click in the cell to the left of the tagname to select the association direction for the tagname to its respective property. (Continuously double-clicking will cycle you through the various association direction choices. The association direction choices are described below.)

Tip There are actually two fields in each cell in the **Associated Tag** column. The association direction selection and the tagname entry. The ActiveX control determines the association direction and the property type determines the tagname type that must be used.

You can select one directional or bi-directional association. However, if the association direction you select is not valid for the property or tagname, the control will automatically change it accordingly. For example, if you select ←, when the tagname's value changes, its associated property is changed accordingly. A certain subset of the symbols below appears based upon the potential relationship between the property and the tagname.

For example, if you associate a property to a writeable tagname, you will see a different set of associations than if you associate the same property to a read-only tagname. Select the appropriate association symbol as follows:

Symbol	Description
←	The tagname sets the value of the associated property.
|—	This symbol indicates that the property is read-only and the tagname cannot change the property's value.
→	The property sets the value of the associated tagname.
—|	This symbol indicates that the tagname is read-only and the property cannot change the tagname's value.
↔	Value can be set from both the tagname or the property. (Tagname takes precedence.)
|—|	The tagname and the property are both read-only.
←|	The tagname can change the property's value, but the property cannot change the tagname's value. The property cannot change the tagname's value because the property is non-bindable, or the tagname is read-only.
|→	The property can change the tagname's value, but the tagname cannot change the property's value. The tagname cannot change the property's value because the property is read-only.

4. Click **OK**.

Note You can also access or change properties through ActiveX Event scripts and/or other InTouch QuickScripts. All ActiveX script functions are qualified by the # (pound) sign. The valid syntax to access ActiveX properties is:

```
#ControlName.PropertyName
```

Examples:

```
#Calendar1.Day = 29;
Tag1 = #Calendar1.year;
```

For more information, see, "Configuring an ActiveX Control."

Note When an ActiveX property page is changed, the changes are automatically effected, whether or not you click the "Apply" or "Cancel" button on the property page. Properties updated on the control's default properties page may not be reflected in the InTouch ActiveX properties page. Switching back and forth between tabs will update the properties page.

The "SelectionChange" event is not supported by InTouch.

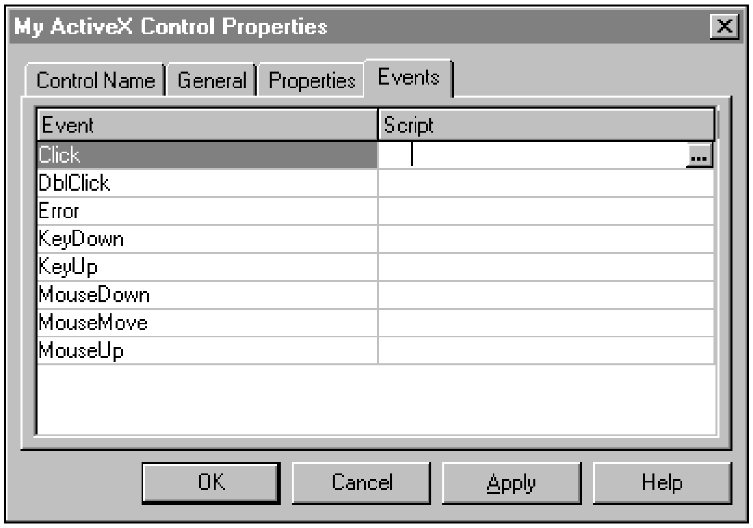
Using ActiveX Control Methods

ActiveX control methods are similar to ActiveX control properties. You activate methods in runtime (WindowViewer). ActiveX control methods are accessed through ActiveX Event scripts and/or other InTouch QuickScripts.

Note In order for an ActiveX Event script to function properly the ActiveX control for which the script was created, must be loaded into memory. If the window containing an ActiveX control is closed, its ActiveX Event scripts, or any other InTouch QuickScripts containing script functions associated with that ActiveX control, will not execute properly.

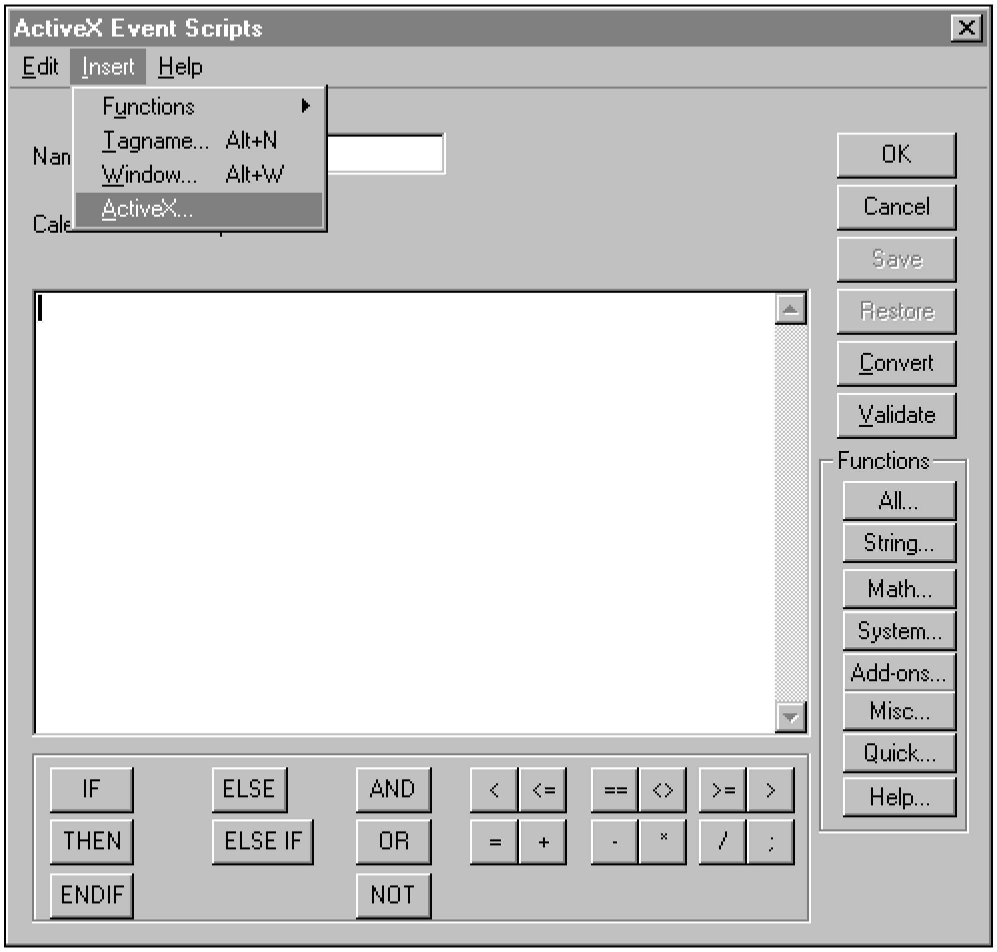
To use ActiveX methods and/or properties

1. In the ActiveX control's **Properties** dialog box, click the **Events** tab to activate the **Events** property sheet:

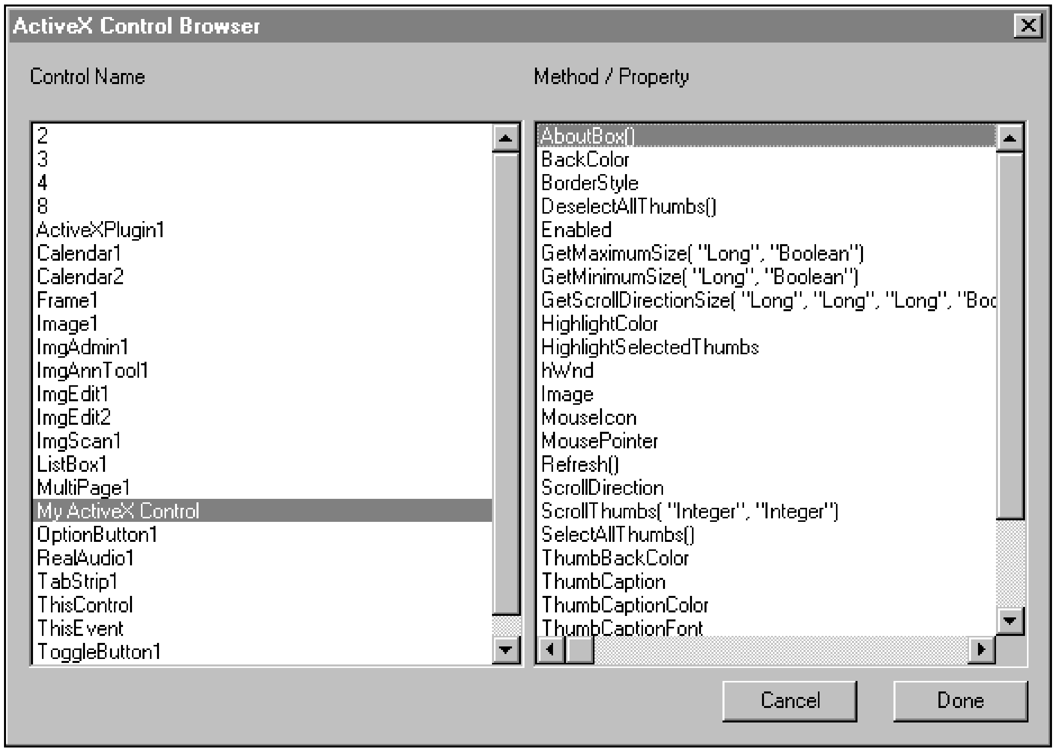


Event	Script
Click	
DblClick	
Error	
KeyDown	
KeyUp	
MouseDown	
MouseMove	
MouseUp	

2. Double-click a blank cell in the **Script** column. The **ActiveX Event Scripts** editor appears.


ActiveX Event Scripts
Edit
Insert
Help
Functions
Tagname... Alt+N
Window... Alt+W
ActiveX...
Nan
Cal
OK
Cancel
Save
Restore
Convert
Validate
Functions
All...
String...
Math...
System...
Add-ons...
Misc...
Quick...
Help...
IF
THEN
ENDIF
ELSE
ELSE IF
AND
OR
NOT
<
<=
==
<>
>=
>
=
+
-
*
/
;

3. On the **Insert** menu, click **ActiveX**. The **ActiveX Control Browser** appears.



```
Control Name          Method / Property
2                     AboutBox()
3                     BackColor
4                     BorderStyle
8                     DeselectAllThumbs()
ActiveXPlugin1        Enabled
Calendar1             GetMaximumSize( "Long", "Boolean")
Calendar2             GetMinimumSize( "Long", "Boolean")
Frame1                GetScrollDirectionSize( "Long", "Long", "Long", "Boo
Image1                HighlightColor
ImgAdmin1             HighlightSelectedThumbs
ImgAnnTool1           hWnd
ImgEdit1              Image
ImgEdit2              MouseIcon
ImgScan1              MousePointer
ListBox1              Refresh()
MultiPage1            ScrollDirection
My ActiveX Control    ScrollThumbs( "Integer", "Integer")
OptionButton1         SelectAllThumbs()
RealAudio1            ThumbBackColor
TabStrip1             ThumbCaption
ThisControl           ThumbCaptionColor
ThisEvent             ThumbCaptionFont
ToggleButton1
```

4. In the **Control Name** list, select the ActiveX control whose methods or properties you want to access.

   The names of all ActiveX controls currently being used in your application are listed.

Note If you select **ThisControl** instead of the actual Control Name, the methods and properties displayed will be those for the ActiveX control currently selected in your application. By selecting **ThisControl** instead of the actual Control Name, you can create generic ActiveX Event script functions. You can then copy and paste these functions into any other ActiveX Event script, or any other InTouch QuickScript without having to change the Control Name in the new script. For example:

```
#ThisControl.Navigate
("http:\\www.wonderware.com");
#ThisControl.Navigate(URL);   { where URL is a
tagname}
```

ThisControl is accessible only through ActiveX Event scripts. It is not accessible through any other type of InTouch QuickScript.

ThisControl is used to write generic references in event scripts that can be reused in any instance of the same control type. When **ThisControl** is selected, the same list of properties and methods appears as the list for the specific control name.

5. In the **Method / Property** list, select the method or property that you want to use in your script.

Tip Methods are the items in the list that include parenthesis. For example, **Display()**.

6. Click **Done**. The selected control name and method or property are automatically inserted into your script.

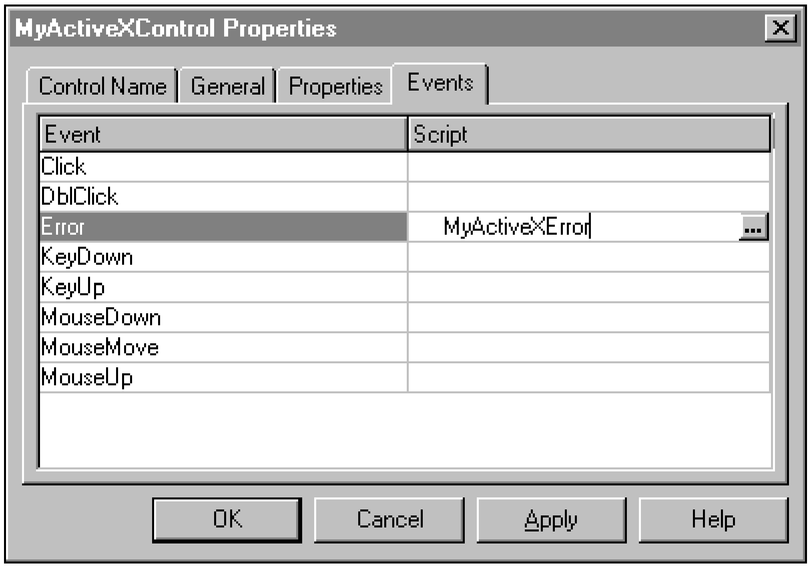
Tip ActiveX control's methods and properties are also accessed through the **Insert** menu in all other InTouch QuickScripts types.

Using ActiveX Control Event Parameters

You can execute ActiveX control events in runtime (WindowViewer) by designing a particular action and associating it to the event. For example, if your ActiveX control has an error event handler, you could create a ActiveX Event QuickScript that displays a window with an error message when an error occurs. ActiveX Event scripts are provided to support event actions. You can associate a named event script to each event..

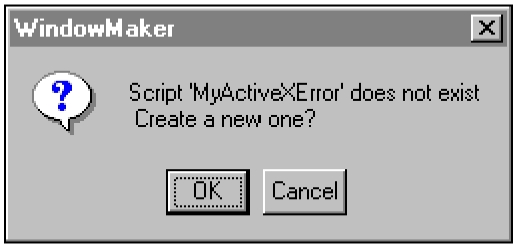
To use ActiveX Event parameters

1. Double-click the ActiveX control for which you want to create an ActiveX Event script. The selected ActiveX control's **Properties** dialog box appears.
2. Click the **Events** tab to activate the **Events** property sheet:



Event	Script
Click	
DblClick	
Error	MyActiveXError
KeyDown	
KeyUp	
MouseDown	
MouseMove	
MouseUp	

3. In the **Event** column select the event to which you want to associate an ActiveX Event Script.
4. In the respective cell in the **Script** column, type a unique name for the ActiveX Event Script that you want to create and then double-click the name, or click **OK**. The following message box appears.

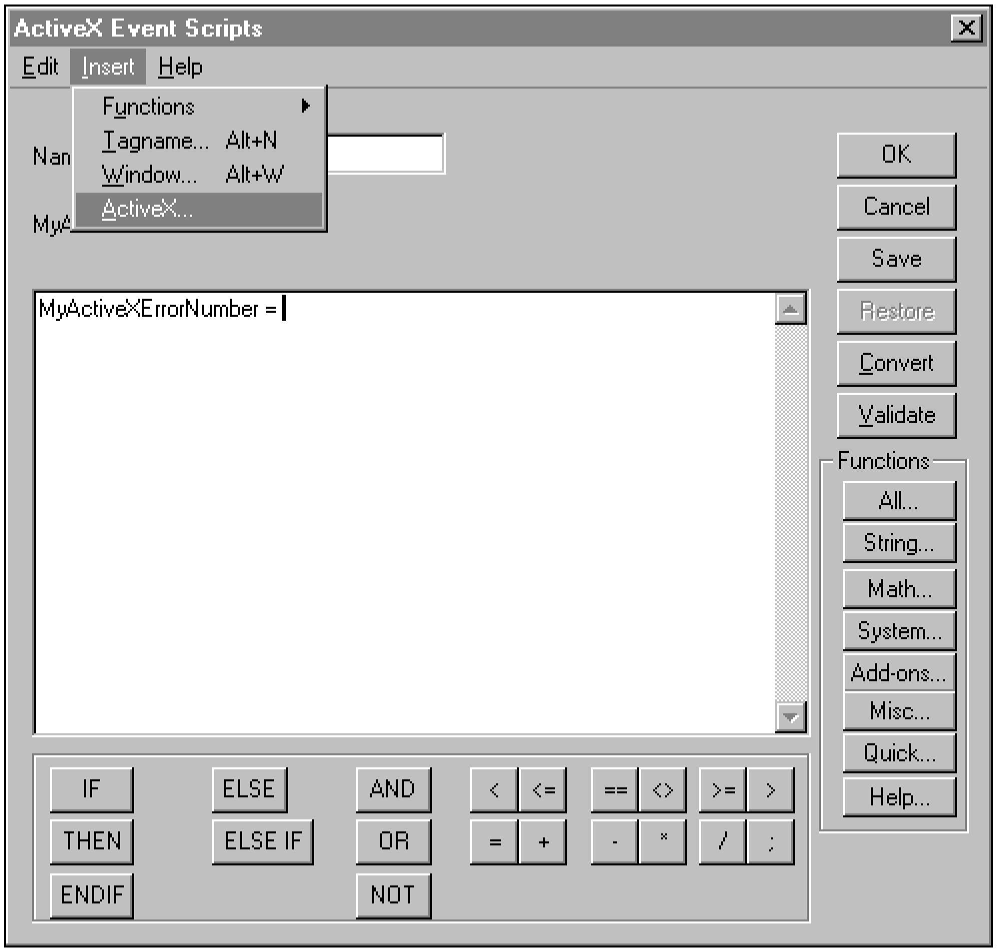


Script 'MyActiveXError' does not exist
Create a new one?

5. Click **OK**. The ActiveX Event script editor appears displaying the name that you typed in the **Name** input box (see example below). If you double-click a blank **Script** cell, when the ActiveX Event script editor appears, you must then type a name for the ActiveX Event script.

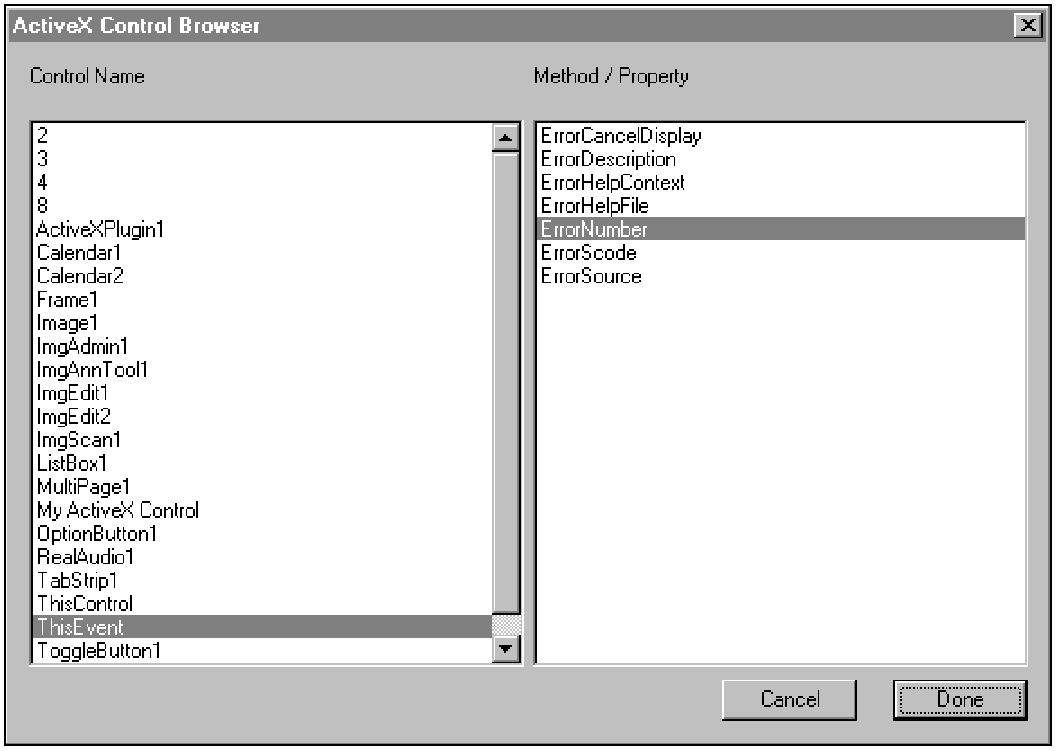
Tip If the ActiveX Event script that you want to use already exists, click the … button. The **Choose ActiveX Script** dialog box appears listing all existing ActiveX Event scripts in your application.

For more information, see "Reusing ActiveX Event Scripts."



```
MyActiveXErrorNumber =
```

6. On the **Insert** menu, click **ActiveX**. The **ActiveX Control Browser** appears.



2
3
4
8
ActiveXPlugin1
Calendar1
Calendar2
Frame1
Image1
ImgAdmin1
ImgAnnTool1
ImgEdit1
ImgEdit2
ImgScan1
ListBox1
MultiPage1
My ActiveX Control
OptionButton1
RealAudio1
TabStrip1
ThisControl
ThisEvent
ToggleButton1

ErrorCancelDisplay
ErrorDescription
ErrorHelpContext
ErrorHelpFile
ErrorNumber
ErrorScode
ErrorSource

7. In the **Control Name** list, select **This Event** to access the parameters for the selected event. In this case, the selected event is, **Error**.

Note **ThisEvent** is accessible only through ActiveX Event scripts. It is not accessible through any other type of InTouch QuickScript. You must select **ThisEvent** to access the event parameters for an ActiveX control.

The event parameters contain useful information. For example, you may have a keypressed event and you want to know "Okay, so we know a keystroke occurred but which key was pressed?" If this information is important, the developer of the control probably added an event parameter to store the key code. To access this parameter, you need to select **ThisEvent** and then select the parameter, which might be called **keypressedkeycode**. It is of special note that the majority of ActiveX events do not have parameters.

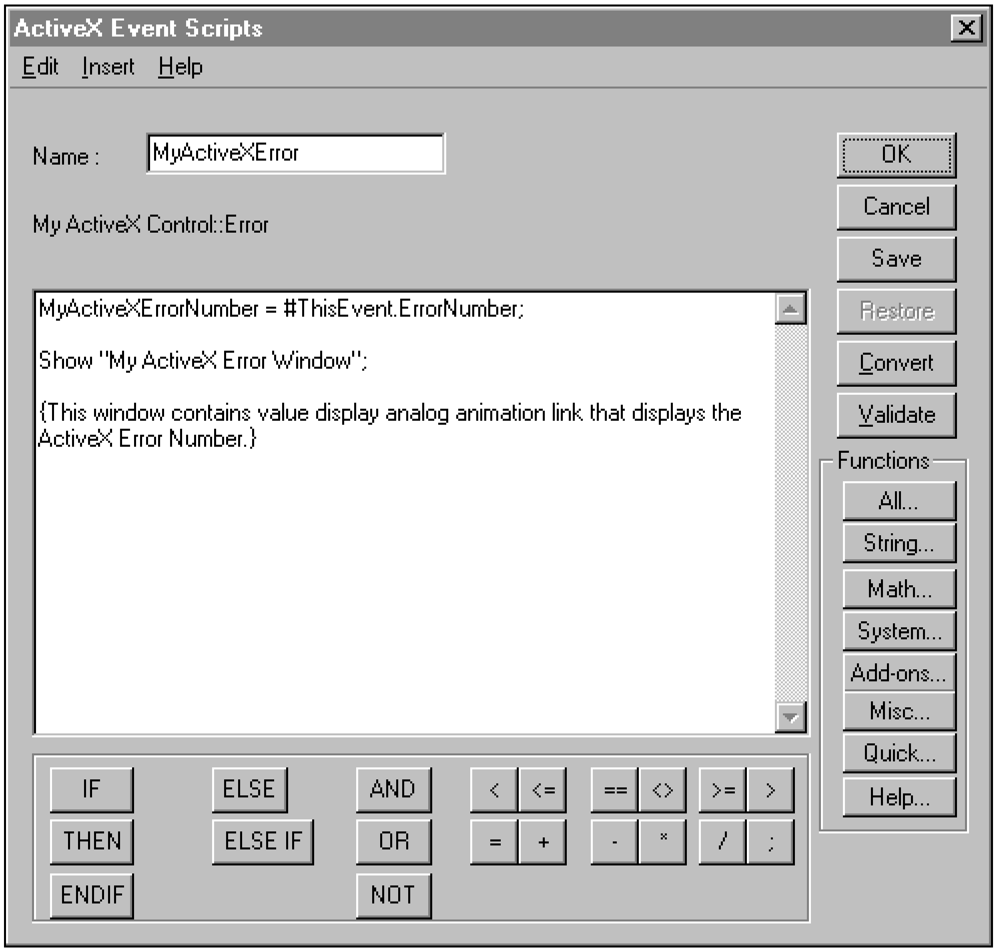
Events may or may not pass parameters in runtime. Event parameters can be accessed by using the **ThisEvent** keyword. For example:

```
MyActiveXErrorNumber = #ThisEvent.ErrorNumber;
```

Where: # indicates that this is an ActiveX script function. **ThisEvent** relates to the event selected in the ActiveX control's **Event** property sheet, and **ErrorNumber** is the parameter passed by the selected event.

8. In the **Method / Property** list, select the method or property that you want to use in your ActiveX Event script.

9. Click Done. The selected control name, in this case, ThisEvent, and selected event parameter are both automatically inserted into your script at the cursor location. For example:



Name : MyActiveXError

My ActiveX Control::Error

```
MyActiveXErrorNumber = #ThisEvent.ErrorNumber;

Show "My ActiveX Error Window";

{This window contains value display analog animation link that displays the
ActiveX Error Number.}
```

Note An ActiveX Event scripting reference consists of the following:

A number sign (#) that indicates "I'm an ActiveX scripting reference."

The name of the control.

The delimiter character (period (.)) to separate the control name from the property or method being called.

The property, method, or event parameter.

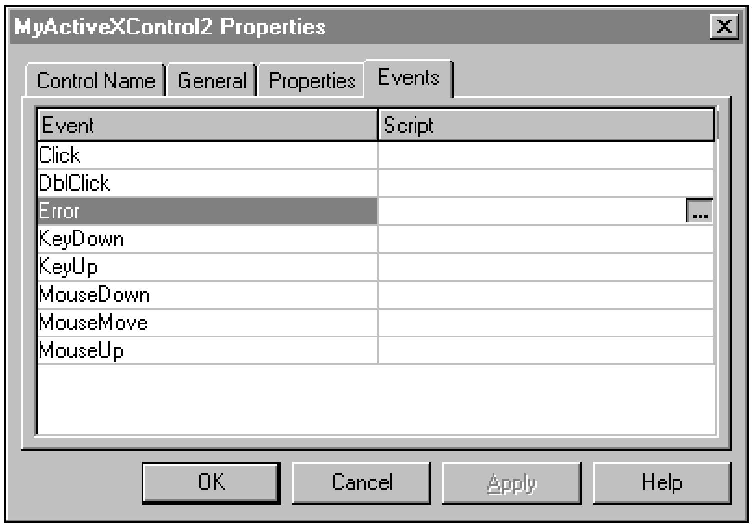
10. Click **OK** to save your ActiveX Event script and close the script editor. The ActiveX control's **Properties** dialog box reappears.
11. Click **OK** to close the **Properties** dialog box or you can continue to create ActiveX Event scripts.

Reusing ActiveX Event Scripts

ActiveX Event scripts can only be reused for the same event for the same kind of ActiveX control. For example, the mouse down event may be a stock event on hundreds of ActiveX controls. However, an ActiveX Event script written for mouse down on ActiveX ControlA cannot be reused for mouse down on ActiveX ControlB unless the two controls are the same type.

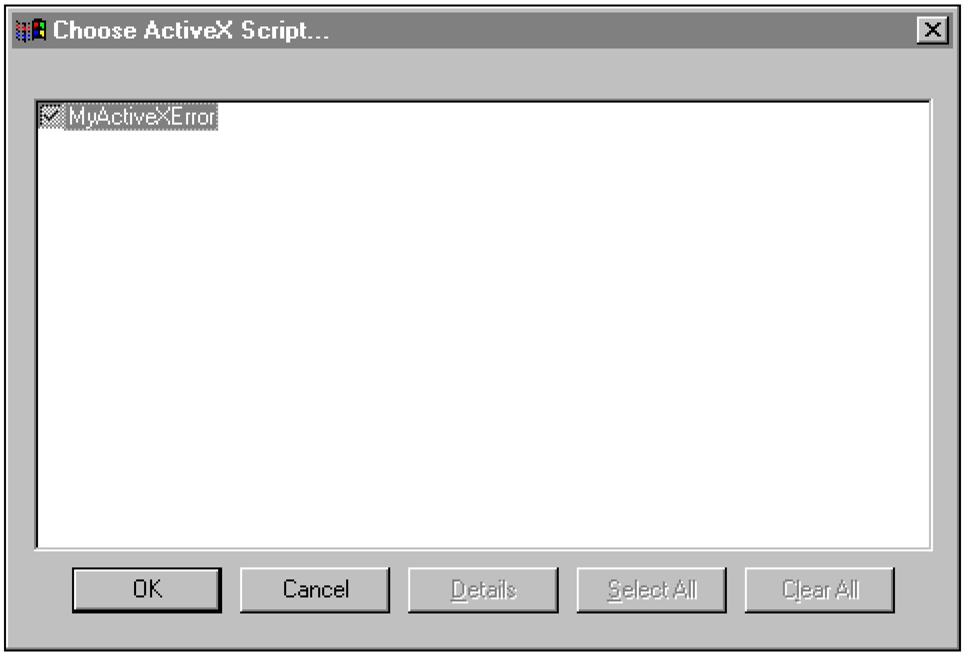
To reuse an ActiveX Event script

1. Double-click the ActiveX control for which you want to reuse an existing ActiveX Event script. The selected ActiveX control's **Properties** dialog box appears.
2. Click the **Events** tab to activate the **Events** property sheet:



Event	Script
Click	
DblClick	
Error	
KeyDown	
KeyUp	
MouseDown	
MouseMove	
MouseUp	

3. In the **Script** column for the respective event, click the ... button. The **Choose ActiveX Script** dialog box appears.



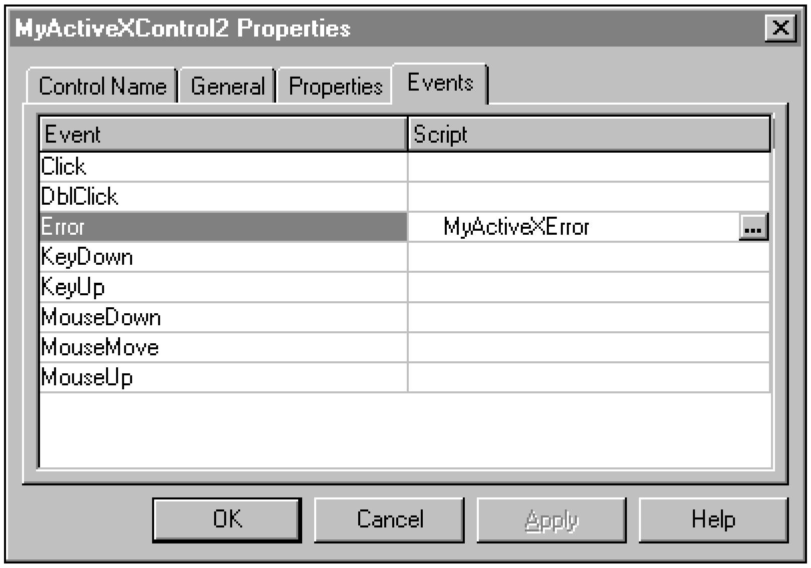
This dialog box will only display the ActiveX Event scripts that were written for the same type of ActiveX control and the same selected event.

For example, let's assume that you are creating an ActiveX Event script for a second ActiveX Calendar control's "Click" event. You have already created two other ActiveX Event scripts named Click1 and Click2 in your application. Click1 was created for a different ActiveX Calendar control's "Click" event, and Click2 was created for an ActiveX InSQLTrend

control's "Click" event. When you click the ... button and the **Choose ActiveX Script** dialog box appears, it will only display the Click1 script since was created for the same type of ActiveX control and the same event.

4. Select the ActiveX Event script that you want to use and then click **OK**.

   The name of the selected script is automatically inserted into the **Script** cell where you previously clicked the ... button. For example:



Event	Script
Click	
DblClick	
Error	MyActiveXError
KeyDown	
KeyUp	
MouseDown	
MouseMove	
MouseUp	

5. Click **OK** to close the **Properties** dialog box, or continue to create ActiveX Event scripts.

Importing ActiveX Event Scripts

Importing ActiveX Event scripts from one InTouch application to your current application can save you a considerable amount of development time. When you move ActiveX Event scripts from one InTouch application to another, you must use the **Import** command on the WindowMaker **File** menu.

Note When you import ActiveX Event scripts, from one application to another, all ActiveX Event scripts are imported. Additionally, in order for an imported ActiveX Event script to function properly in the new application, the same ActiveX control and the same event for which the script was originally created, must also be used in the new application, and it must be loaded into memory. If the window containing an ActiveX control is closed, its ActiveX Event scripts, or any other InTouch QuickScripts containing script functions associated with that ActiveX control, will not execute properly.

For more information on importing scripts, see Chapter 8, "Creating QuickScripts in InTouch."

Customizing Your Runtime Environment

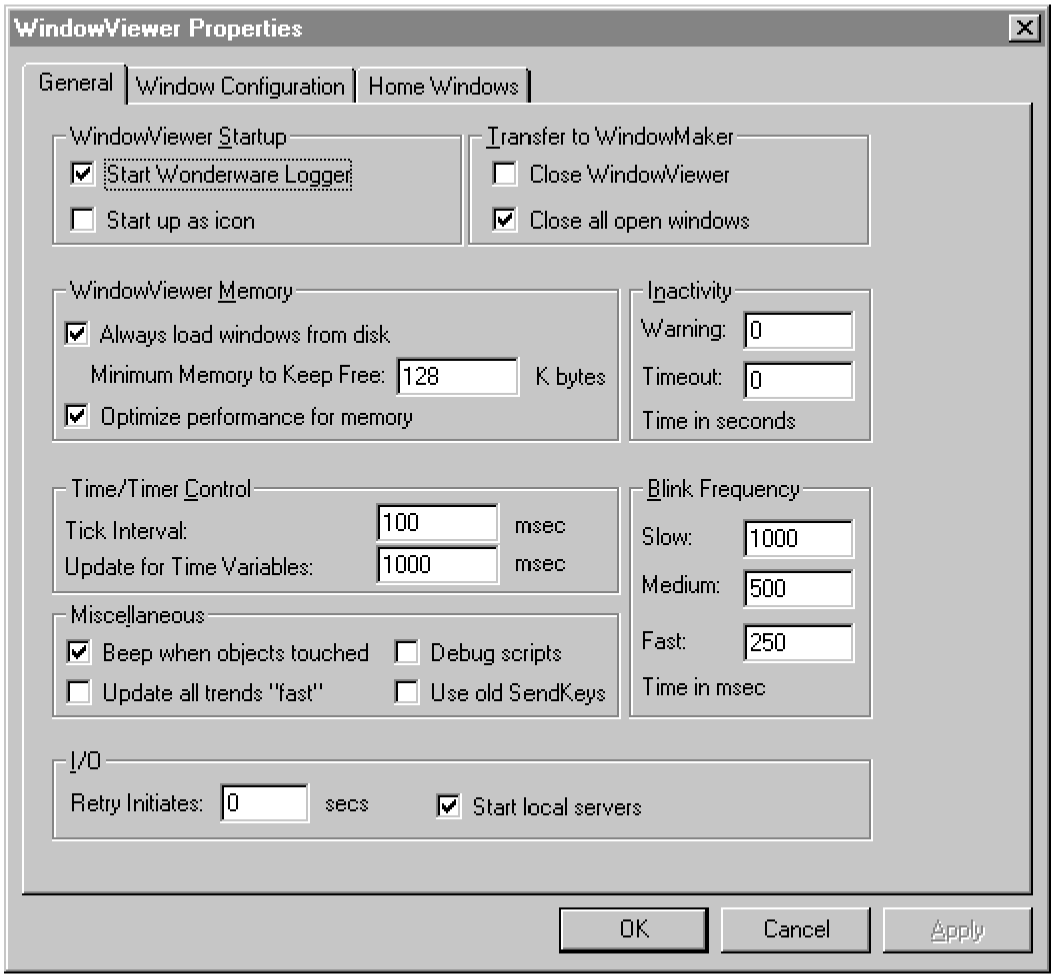
Like WindowMaker, there are many properties that you can set to customize your runtime environment (WindowViewer). For example, you can set the blinking speed for blinking objects, the system inactivity timeout and warning values, the windows that are automatically opened when WindowViewer is started from its icon or its menu command.

Setting WindowViewer's General Properties

To set the properties for WindowViewer

1. On the **Special** menu, point to **Configure** and then click **WindowViewer** or in the Application Explorer under **Configure**, double-click **WindowViewer**. The **WindowViewer Properties** dialog box appears with the **General** properties sheet active.

Tip In the Application Explorer, you can also right-click **WindowViewer** and then click **Open**.


WindowViewer Properties

General | Window Configuration | Home Windows

WindowViewer Startup
- [✓] Start Wonderware Logger
- [ ] Start up as icon

Transfer to WindowMaker
- [ ] Close WindowViewer
- [✓] Close all open windows

WindowViewer Memory
- [✓] Always load windows from disk
- Minimum Memory to Keep Free: 128 K bytes
- [✓] Optimize performance for memory

Inactivity
- Warning: 0
- Timeout: 0
- Time in seconds

Time/Timer Control
- Tick Interval: 100 msec
- Update for Time Variables: 1000 msec

Blink Frequency
- Slow: 1000
- Medium: 500
- Fast: 250
- Time in msec

Miscellaneous
- [✓] Beep when objects touched
- [ ] Debug scripts
- [ ] Update all trends "fast"
- [ ] Use old SendKeys

I/O
- Retry Initiates: 0 secs
- [✓] Start local servers

Tip If you right-click any of the text entry boxes in any dialog box, a menu appears displaying the commands that you can apply to the selected text.

2. Select **Start up as icon** if you want WindowViewer to start up as an icon instead of a window.

Tip Select this option only when you are using WindowViewer to gather data for other I/O-interconnected applications.

3. Select **Close WindowViewer** if you want WindowViewer to automatically close when you start WindowMaker.

Tip If memory is not an issue, and you are using the fast switch to move between WindowViewer and WindowMaker, this option should not be selected.

The fast switch option is selected in the **WindowMaker Properties - General** dialog box.

4. When you select this option, the **Close on Transfer to WindowViewer** option located on the **WindowMaker Properties/General** property sheet is automatically selected too.
5. Select **Close all open windows** if you want all open windows to automatically be closed when you transfer from WindowViewer to WindowMaker.

Tip Selecting this option will free up memory on your system.

6. Select **Always Load Windows from Disk** if a low memory situation exists.

Tip Selecting this option causes your application windows to be loaded from disk and not saved in RAM memory when you close them.

7. In the **Minimum Memory to keep free** box, type the number of K bytes of memory that you want to keep free for other Windows applications.
8. Select **Optimize performance for memory** to significantly increase the drawing update speed. Selecting this option significantly increases the update rate for text fields.

Tip If your system is low on memory do not enable this option.

9. In the **Warning** box, type the number of seconds that can elapse with no operator activity (mouse clicks or keystrokes) before the system discrete tagname **$InactivityWarning** is set to 1 (True).

Tip You can use **$InactivityWarning** in a Condition QuickScript to show a window warning the operator that he/she is about to be logged off the system. If the operator clicks the mouse, presses a key, or performs an action using any other pointing device before the specified timeout elapses, they are not logged off. **$InactivityWarning** and the timer are reset.

10. In the **Timeout** box, type the number of seconds that can elapse with no operator activity (mouse clicks, keystrokes, and so on) before the system discrete tagname **$InactivityTimeout** is set to 1 (True). When **$InactivityTimeout** is true, the system equates the logged on operator name to the reserved name "None" and sets the security tagname, **$AccessLevel**, to 0.

Tip You can use **$InactivityTimeout** in a Condition QuickScript to show a window telling the operator that he/she has been logged off the application.

You can use the **Timeout** feature independently of the **Warning** feature. However, the **Timeout** value must be greater than the **Warning** value for proper use of both system tagnames.

For example, **Warning** becomes true after 30 seconds of inactivity and **Timeout** becomes true after an additional 15 seconds (for a total of 45 seconds) of inactivity.

11. In the **Tick Interval** box, type the speed interval that InTouch will use to check its internal timers.

Note This setting controls how fast an Application **While Running**, Window **While Showing**, Condition **While On True/On False**, Key and Touch Pushbutton Action **While Down** QuickScripts will be executed.

Scripts cannot execute faster than every 10 milliseconds on the Windows NT operating system or every 50 milliseconds on Windows 2000.

For more information, see Chapter 8, "Creating QuickScripts in InTouch."

12. In the **Update Time Variables every** box, type the interval (in milliseconds) that you want WindowViewer to update the time-based system tagnames such as $Msec, $Second, $Minute, and so on.

Tip We recommend that you use the default setting of 1000 milliseconds. However, you can type a zero to prevent updating of all time variables.

13. Select **Beep when objects touched** if you want all touch-sensitive objects to beep when selected in WindowViewer.
14. Select **Update all trends "Fast"** if you want your trend objects to be updated faster.

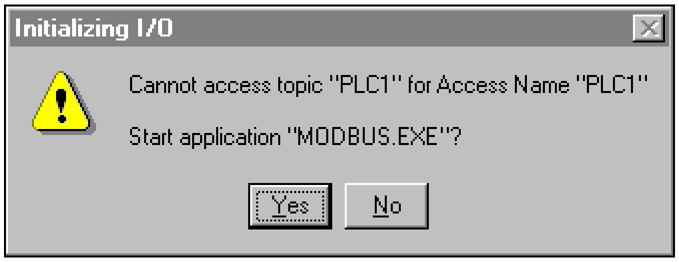
Note Select this option only when you are absolutely certain that no objects are overlapping your runtime trend objects. If you select this option and any object is overlapping the trend object, it will not be drawn properly.

15. Select **Debug Scripts** if you want a message to be written to the Logger each time a QuickScript is executed.

Tip If you select the **Debug** menu option in the **WindowViewer Properties/Window Configuration** property sheet, you will be able to turn this command on and off in runtime from WindowViewer's **Special** menu.

16. Select **Use old SendKeys** only if you are using an international application that was developed using InTouch Version 3.26 or earlier. **Use old SendKeys** is a legacy option and is not used for FactorySuite.
17. In the **Slow, Medium, Fast** boxes type the intervals (in milliseconds) that you want to use for your blink animation links.

18. In the **I/O Retry Initiates** box, type the number of seconds that you want to elapse before InTouch retries connecting to an I/O server. (The **I/O Retry Initiates** box has no effect when InTouch can successfully connect to the I/O server the first time.
19. Select the **Start Local Servers** option if you want a dialog box to display whenever you start WindowViewer and the server you are trying to communicate with is not running. For example:



Cannot access topic "PLC1" for Access Name "PLC1"

Start application "MODBUS.EXE"?

Click **Yes** to start the server or click **No** to ignore the message and close the dialog box.

20. Click **OK** to save your property settings and close the dialog box.

Note After you modify any of these parameters, you must restart WindowViewer to apply your changes.

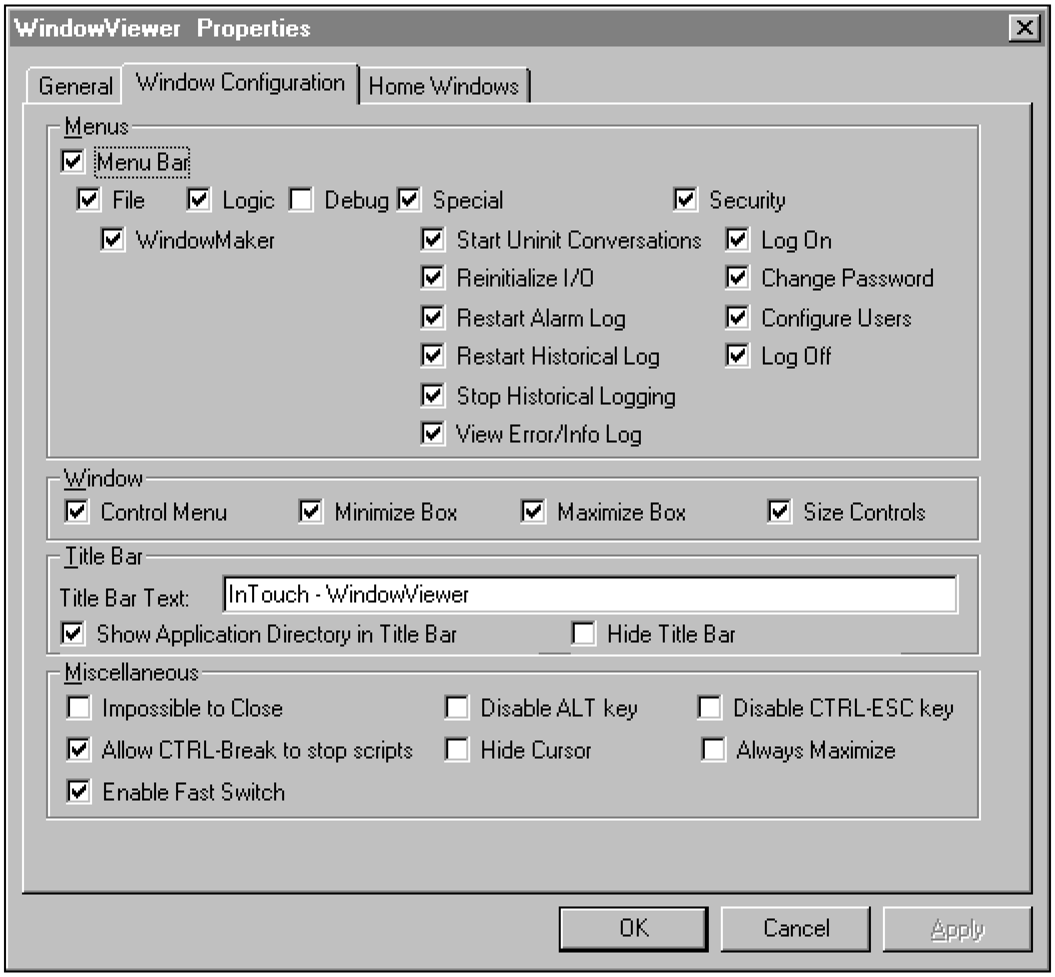
Setting WindowViewer's Window Configuration Properties

To configure the WindowViewer program window

1. On the **Special** menu, point to **Configure** and then click **WindowViewer** or in the Application Explorer under **Configure**, double-click **WindowViewer**. The **WindowViewer Properties** dialog box appears.

Tip In the Application Explorer, you can also right-click **WindowViewer** and then click **Open**.

2. Click the **Window Configuration** tab:



WindowViewer Properties

General | Window Configuration | Home Windows

Menus
- [✓] Menu Bar
- [✓] File [✓] Logic [ ] Debug [✓] Special [✓] Security
- [✓] WindowMaker
- [✓] Start Uninit Conversations [✓] Log On
- [✓] Reinitialize I/O [✓] Change Password
- [✓] Restart Alarm Log [✓] Configure Users
- [✓] Restart Historical Log [✓] Log Off
- [✓] Stop Historical Logging
- [✓] View Error/Info Log

Window
- [✓] Control Menu [✓] Minimize Box [✓] Maximize Box [✓] Size Controls

Title Bar
- Title Bar Text: InTouch - WindowViewer
- [✓] Show Application Directory in Title Bar [ ] Hide Title Bar

Miscellaneous
- [ ] Impossible to Close [ ] Disable ALT key [ ] Disable CTRL-ESC key
- [✓] Allow CTRL-Break to stop scripts [ ] Hide Cursor [ ] Always Maximize
- [✓] Enable Fast Switch

OK Cancel Apply

Tip If you right-click any of the text entry boxes in any dialog box, a menu appears displaying the commands that you can apply to the selected text.

3. In the menu**s** group, select the menus and commands that you want displayed in runtime.

Tip By default, the menu bar is displayed when WindowViewer is running. Clear the **Menu Bar** option to prevent the menu bar from showing.

4. Clear the **WindowMaker** command if you want to prevent the operator from being able to switch to the WindowMaker program. (Selecting this option does not affect the fast switch to WindowMaker.)
5. Clear the **Logic** menu if you want to prevent the operator from starting and stopping all QuickScripts from executing during runtime.

Note You can use the system tagname, **$LogicRunning** to allow the operator to start and stop all QuickScripts.

If you select **Allow CTRL-Break to stop scripts** option (described later), the operator will be able to stop all QuickScripts from executing regardless of whether the **Logic** menu is displayed or not.

Asynchronous QuickFunctions that are currently executing cannot be stopped. However, you can prevent new asynchronous QuickFunctions from executing.

6. Select the **Debug** menu only when you need to "debug" your application.
7. Select the **Window** controls that you want available in runtime.

Note You must clear the **Control Menu** option (also called the System menu) in order to hide the close (**X** button) in the upper right hand corner of the application.)

8. In the **Title Bar Text** box, type the title that you want to appear in your application's title bar in runtime. For example:

   ABC Company, Paint APP1

Note You cannot change the title bar if you are using a "Promotional License."

For more information on FactorySuite licensing, see your *FactorySuite System Administrator's Guide*.

9. Select **Show Application Directory** if you want to include the path to the application's directory in the title bar. For example:

   ABC Company, Paint APP1 - C:\DEMOAPP1
10. Select **Hide Title Bar** if you want to hide the application's title bar in runtime.
11. Select **Impossible to Close** if you want to prevent the operator from closing WindowViewer.

Note You must also clear the **Control Menu** option (also called the **System** menu) in order to hide the close (X) box in the upper right hand corner of the application.)

12. Select **Allow CTRL-Break to stop scripts** if you want to allow the operator to press the CTRL + BREAK key sequence to stop the execution of all QuickScripts whenever necessary during runtime.

Note Asynchronous QuickFunctions that are currently executing cannot be stopped. However, you can prevent new asynchronous QuickFunctions from executing.

13. Select **Disable ALT key** if you want to disable the ALT key and prevent the operator from executing menu commands by using the ALT + accelerator key. For example, ALT + F4 to exit the application.

Note You must also clear the **Control Menu** option (also called the **System** menu) in order to hide the close (X) box in the upper right hand corner of the application.)

14. Select **Hide Cursor** if you want to prevent the cursor from being displayed during runtime because a touch-screen will be used.
15. Select **Disable ESC key** and **Disable Win key** if you want to prevent the operator from accessing the Windows **Start** menu to close and/or switch applications.
16. Select **Always Maximize** if you want to keep the WindowViewer program maximized at all times.
17. Click **OK** to save your settings and close the dialog box.

Note After you modify any of these parameters, you must restart WindowViewer to apply your changes.

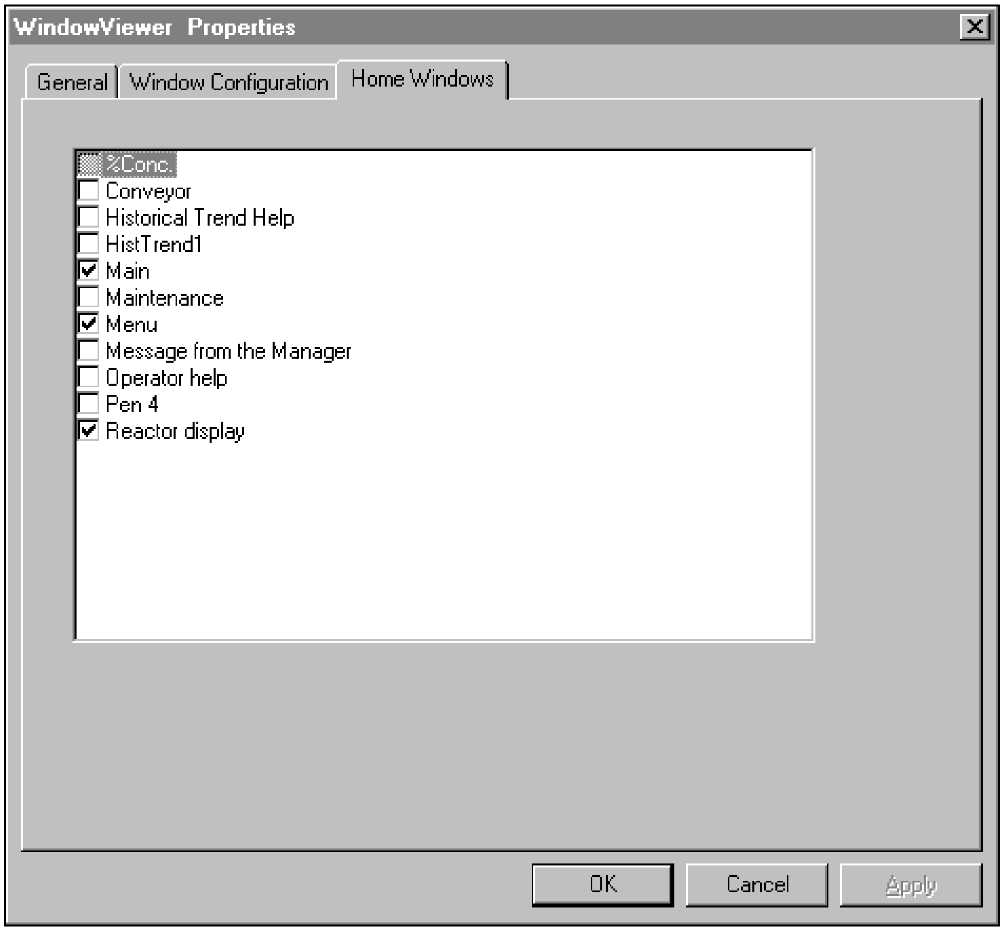
Selecting WindowViewer's Home Windows

To select the WindowViewer default start up windows

1. On the **Special** menu, point to **Configure** and then click **WindowViewer**, or in the Application Explorer under **Configure**, double-click **WindowViewer**. The **WindowViewer Properties** dialog box appears.

Tip In the Application Explorer, you can also right-click **WindowViewer** and then click **Open**.

2. Click the **Home Windows** tab:



- [ ] %Conc.
- [ ] Conveyor
- [ ] Historical Trend Help
- [ ] HistTrend1
- [✓] Main
- [ ] Maintenance
- [✓] Menu
- [ ] Message from the Manager
- [ ] Operator help
- [ ] Pen 4
- [✓] Reactor display

3. Select the window(s) that you want to automatically open when WindowViewer is started directly.

Note The home windows selections have no effect when you use the fast switch to start WindowViewer. Home windows are automatically opened when you start WindowViewer directly from either its icon on its menu command.

4. Click **OK**.

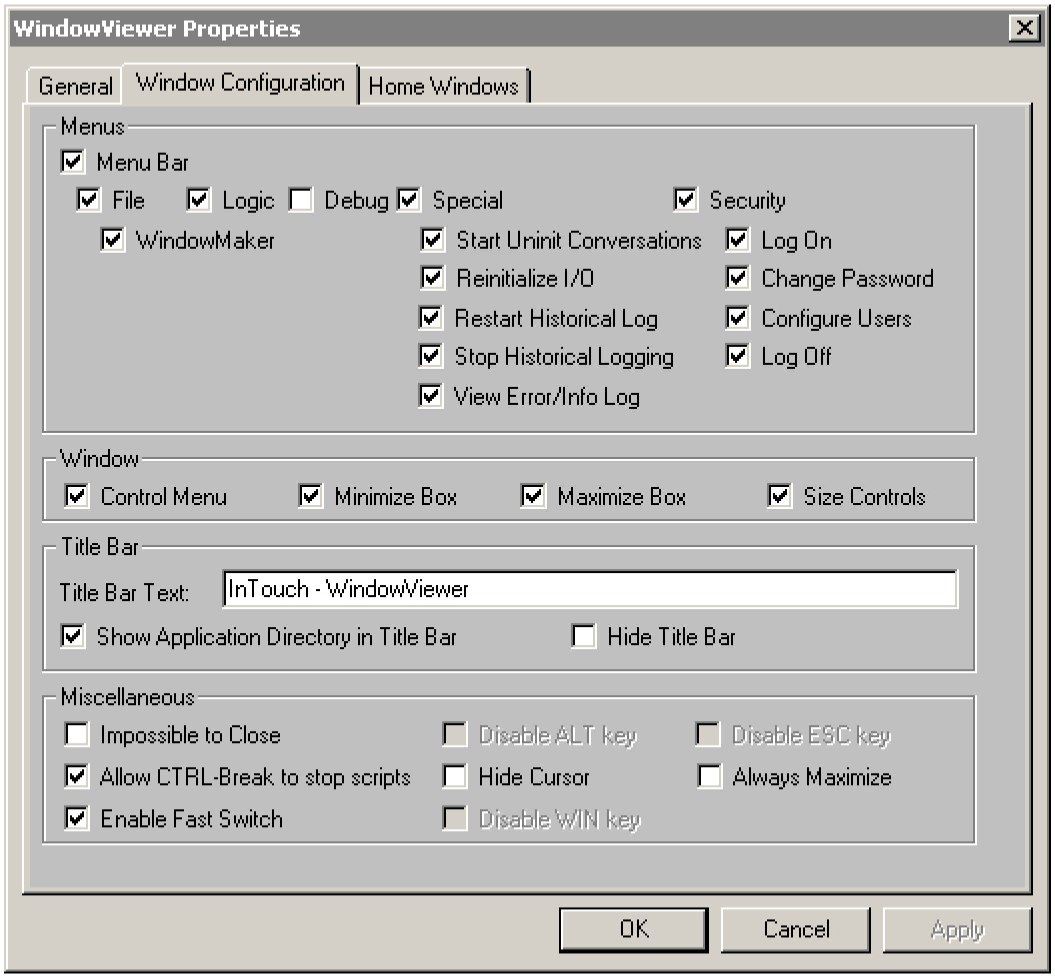
Enabling Key Filters

An application can be designed and used in a plant to allow differing levels of access to operating system functionality. If you have operators with different access levels, you can enable key filters for some users. Other users such as administrators will still be able to use the keys. For example, if an operator with no special access privileges uses the application and types Ctrl-ALT-DEL on the console, a Windows Security dialog is displayed. The only button available is **Cancel**. Therefore, the user cannot use any options such as Lock Computer, Logoff, Shutdown, Change Password or Task Manager while the InTouch application is running. The user can only click **Cancel** and return to the application with the logon screen still showing. However, if an administrator logs on using the correct administrator's logon name and password, these keys will not be disabled.

In order to enable key filters, you must ensure that these keys are inoperative on application startup.

To enable the key filters

1. On the **Special** menu, point to **Configure** and select **WindowViewer**. The WindowViewer Properties dialog box appears.


WindowViewer Properties

General | Window Configuration | Home Windows

Menus
- [✓] Menu Bar
- [✓] File [✓] Logic [ ] Debug [✓] Special [✓] Security
- [✓] WindowMaker
- [✓] Start Uninit Conversations
- [✓] Reinitialize I/O
- [✓] Restart Historical Log
- [✓] Stop Historical Logging
- [✓] View Error/Info Log
- [✓] Log On
- [✓] Change Password
- [✓] Configure Users
- [✓] Log Off

Window
- [✓] Control Menu [✓] Minimize Box [✓] Maximize Box [✓] Size Controls

Title Bar
- Title Bar Text: InTouch - WindowViewer
- [✓] Show Application Directory in Title Bar [ ] Hide Title Bar

Miscellaneous
- [ ] Impossible to Close
- [✓] Allow CTRL-Break to stop scripts
- [✓] Enable Fast Switch
- [ ] Disable ALT key
- [ ] Hide Cursor
- [ ] Disable WIN key
- [ ] Disable ESC key
- [ ] Always Maximize

2. Click the **Window Configuration** tab.
3. Uncheck the **Enable Fast Switch** box.
4. In the **Miscellaneous** field, check the **Disable ALT key**, **Disable WIN key** and **Disable ESC key** check boxes and click **Apply**.
5. Configure a script that is run on operator change. The script function to enable/disable key filters is as follows:

EnableDisableKeys()

	Enables/Disables key filters for the Alt,Escape and Windows keys	
Category	View	
Syntax	`EnableDisableKeys(int AltKey,int EscKey,int WinKey);`	
	Parameter	**Description**
	AltKey	Integer to enable or disable key filters for the Alt key, 0=disable, 1=enable
	EscKey	Integer to enable or disable key filters for the Escape key, 0=disable, 1=enable
	WinKey	Integer to enable or disable key filters for the Windows key, 0=disable, 1=enable
Remarks	Enables/Disables Alt,Escape and Windows keys	

Example(s)	`EnableDisableKeys(0,0,0); // enable all three keys`
	`EnableDisableKeys(1,1,1); // disable all three keys`
	`EnableDisableKeys(0,0,1); // enable Alt and Escape keys, disable Windows key.`	

Running WindowViewer as an NT Service

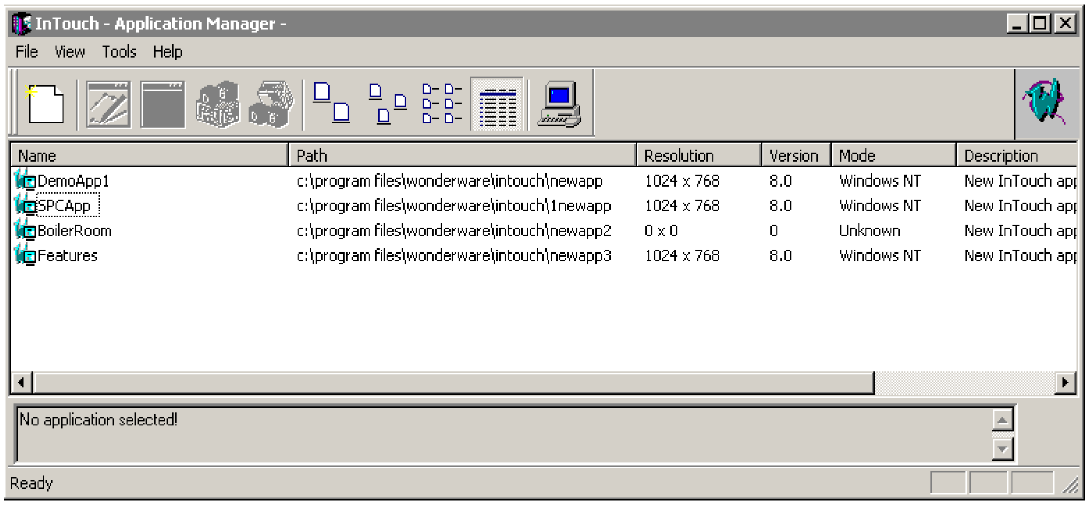
When you are using InTouch 7.0 (or later), you can create client-server configurations very easily. You can configure a node that acts as a server node. This server node can then store the Tagname Dictionary and historical log data, execute InTouch QuickScripts, provide an alarming facility and I/O data. Any client node can then retrieve this information from the server node and display graphics.

Running WindowViewer as an NT Service allows you to take advantage of all the features that an NT Service provides. For example, continuous operation after the operator logs off and automatic startup at system boot time without operator intervention. This allows unmanned station startup of WindowViewer without compromising NT operating system security.

For more information on Windows NT services, see Appendix A, "Overview of the InTouch Windows NT Services."

To configure WindowViewer to start as an NT service

1. Start the InTouch program (intouch.exe). The **InTouch Application Manager** dialog box appears.



Name	Path	Resolution	Version	Mode	Description
DemoApp1	c:\program files\wonderware\intouch\newapp	1024 x 768	8.0	Windows NT	New InTouch app
SPCApp	c:\program files\wonderware\intouch\1newapp	1024 x 768	8.0	Windows NT	New InTouch app
BoilerRoom	c:\program files\wonderware\intouch\newapp2	0 x 0	0	Unknown	New InTouch app
Features	c:\program files\wonderware\intouch\newapp3	1024 x 768	8.0	Windows NT	New InTouch app

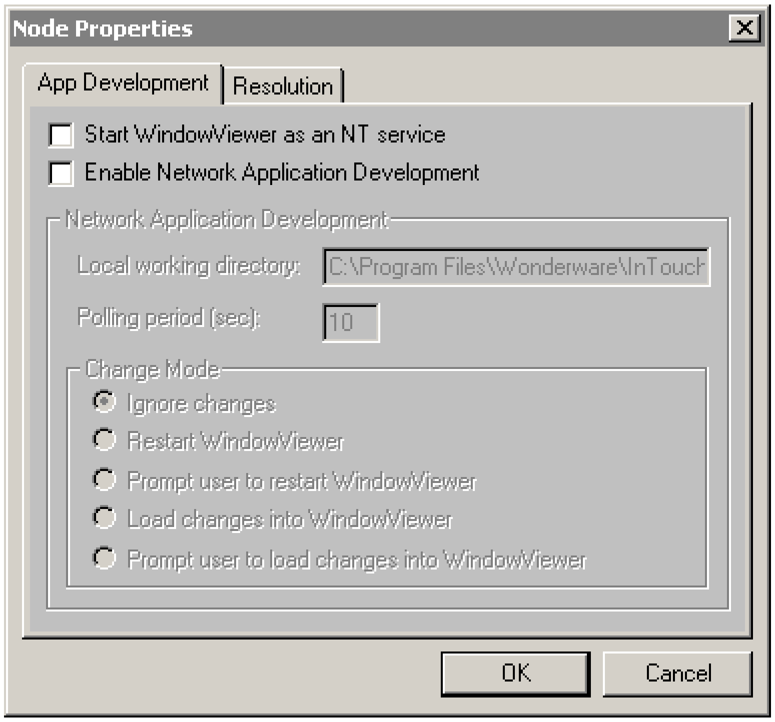
No application selected!

Ready

2. Click the **Node Properties** tool or on the **Tools** menu, click **Node Properties**. The **Node Properties** dialog box appears with the **App Development** property page active.

Tip To quickly access the **Node Properties** dialog box, right-click a blank area of the display window and then click **Node Properties**.

Note When an application is selected in the Application Manager window, selecting the **Properties** command on the **File** menu will display the **Properties** dialog box for that application.



Node Properties

App Development | Resolution

- [ ] Start WindowViewer as an NT service
- [ ] Enable Network Application Development

Network Application Development

Local working directory: C:\Program Files\Wonderware\InTouch

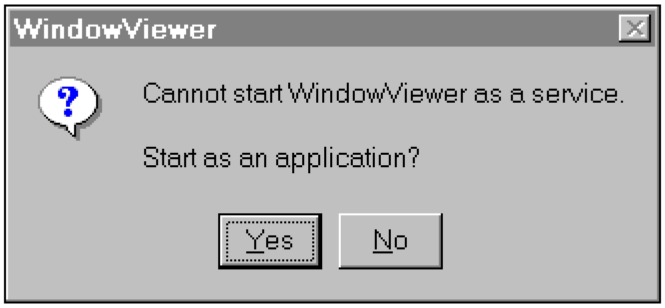
Polling period (sec): 10

Change Mode

[✓ Ignore changes] Restart WindowViewer | Prompt user to restart WindowViewer | Load changes into WindowViewer | Prompt user to load changes into WindowViewer

3. Select the **Start WindowViewer as an NT service** option to configure WindowViewer to automatically run as an NT service.
4. Click **OK**.

Note If WindowViewer is configured as an NT service and subsequently started directly (from its icon, the Windows startup menu and so on), there will be approximately a 15 second delay before WindowViewer will display a window. This delay is due to WindowViewer attempting to connect to the NT Service Control Manager. Upon failing to connect to the Service Control Manager, WindowViewer will display the following message box:


WindowViewer

Cannot start WindowViewer as a service.

Start as an application?

If you click **Yes**, WindowViewer is started as an application not an NT service. If you click **No**, the command to start WindowViewer is canceled.

If you stop WindowViewer from running as an NT service by turning off the **Start WindowViewer as an NT service** option, WindowViewer service is automatically uninstalled. However, it can be run as an application.

For more information on Windows NT services, see Appendix A, "Overview of the InTouch Windows NT Services."

Working with WindowViewer Windows

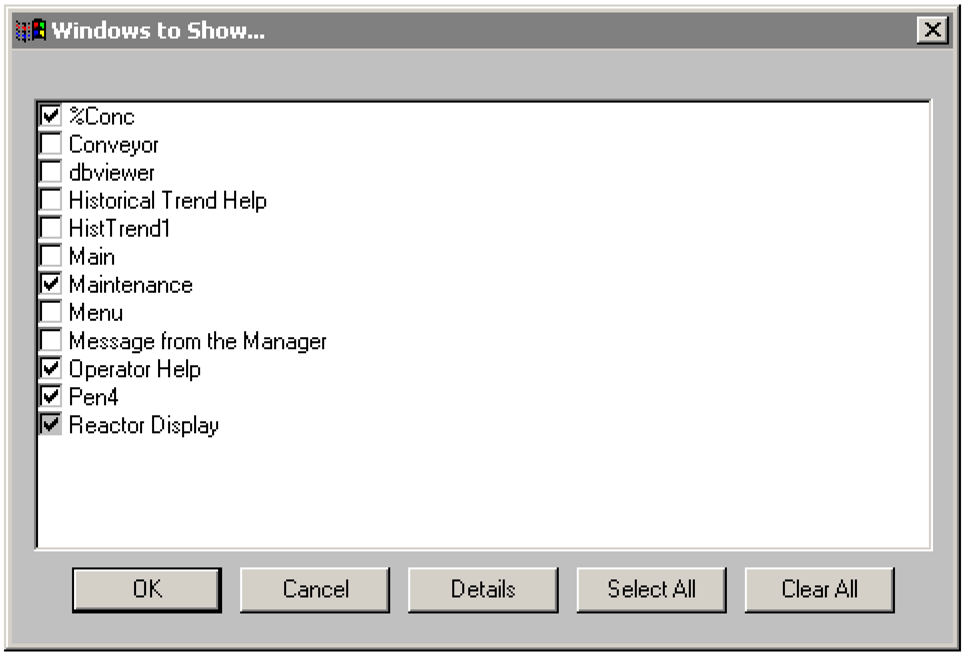
Your InTouch application will more than likely be comprised of numerous windows that display the graphics and text objects created in WindowMaker.

This section describes the procedures that you will follow to open and close the windows contained in your InTouch application.

Common Dialog Box Features

When you are opening or closing a window(s), the dialog boxes that you will use are very similar and have many common features. To avoid redundancy in the procedures describing how you perform these actions, the common features of those dialog boxes are described in this section.

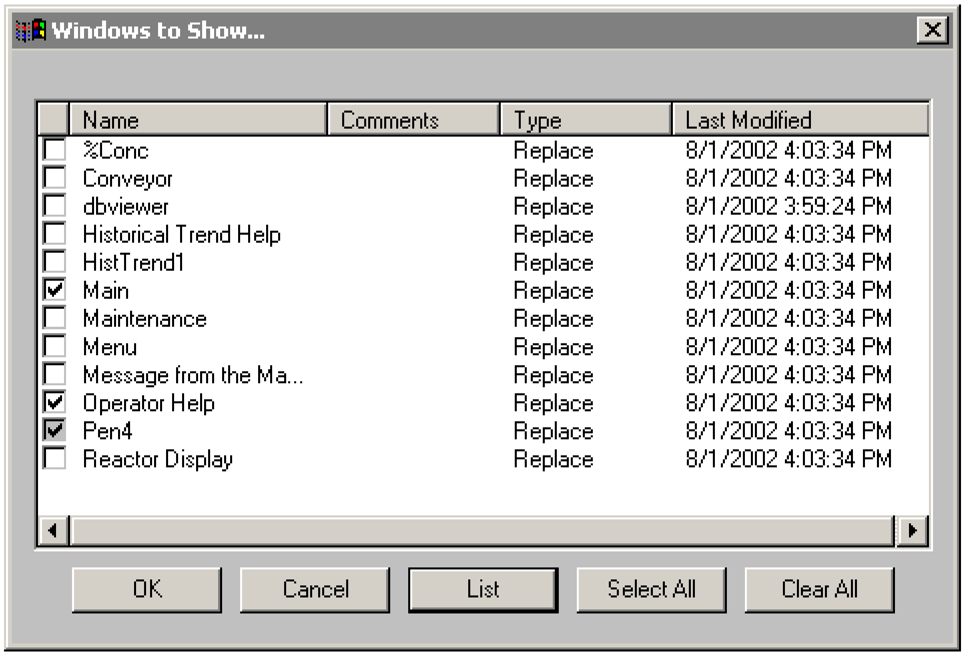
When you click either the **Open Window** or **Close Window** command on the **File** menu, by default, the respective dialog box for the command you selected appears in the "list view." Meaning that the names of all the windows that are applicable for the selected command appear in a continuous list. For example



- [✓] %Conc
- [ ] Conveyor
- [ ] dbviewer
- [ ] Historical Trend Help
- [ ] HistTrend1
- [ ] Main
- [✓] Maintenance
- [ ] Menu
- [ ] Message from the Manager
- [✓] Operator Help
- [✓] Pen4
- [✓] Reactor Display

Note A horizontal scroll bar appears when the number of window names exceeds the default list space.

Click **Details** to change from the "list view" to the details view. When you select the details view, the windows and their details are displayed in a multi-column format. The details displayed include any comments regarding the window that the application developer entered when the window was created, the window's type, the date and time it was last modified. For example:


Windows to Show...

	Name	Comments	Type	Last Modified
☐	%Conc		Replace	8/1/2002 4:03:34 PM
☐	Conveyor		Replace	8/1/2002 4:03:34 PM
☐	dbviewer		Replace	8/1/2002 3:59:24 PM
☐	Historical Trend Help		Replace	8/1/2002 4:03:34 PM
☐	HistTrend1		Replace	8/1/2002 4:03:34 PM
✓	Main		Replace	8/1/2002 4:03:34 PM
☐	Maintenance		Replace	8/1/2002 4:03:34 PM
☐	Menu		Replace	8/1/2002 4:03:34 PM
☐	Message from the Ma...		Replace	8/1/2002 4:03:34 PM
✓	Operator Help		Replace	8/1/2002 4:03:34 PM
✓	Pen4		Replace	8/1/2002 4:03:34 PM
☐	Reactor Display		Replace	8/1/2002 4:03:34 PM

1

Note In the details view, you can select any unopened window by clicking on any portion of its row, not just the check box. (The entire row will be highlighted.) You can click on a selected window a second time, to deselect it.

A vertical scroll bar will also appear when the number of window names exceeds the default list space.

To sort the list by a detail type, click the column header for that detail. The details view sort sequences:

- **Name** - Alphabetically
- **Comments** - Alphabetically
- **Type** - Overlay, Replace then Popup
- **Last Modified** - From oldest date/time (top) to most recent (bottom)

Tip Each time you click a column header, the list sort order will toggle from ascending to descending. For example, if the list is currently sorting in ascending order and you click a column header, the list will be resorted in descending order for the column selected.

To return the list to the default display, click the small box on the far left side of the column header.

To size the columns, place the cursor over the vertical lines that separate each detail header. When the cursor changes to an "I" bar, click and drag the header to the width you want for the column.

Tip To quickly auto-size a column, double-click on the column′s right vertical line separator.

To open selected window(s) click **OK**.

To cancel your selections and close the dialog box, click **Cancel**.

To return the dialog box to "list view," click **List**.

To select all listed windows, click **Select All**.

To clear all selected windows, click **Clear All**.

Opening Windows

To open windows

1. On the **File** menu, click **Open Window**. The **Windows to Show** dialog box appears.
2. Click the check box next to the name of the window(s) that you want to open.

   Tip By default, all currently opened windows will already be checked.

3. Click OK to close the dialog box and open the selected window(s).

   Note If a "Replace" type window is selected, it will cause any windows that it intersects to close.

For more information on window types, see your online *InTouch User's Guide*.

Closing Windows

To close open windows

1. On the **File** menu, click **Close Window**. The **Windows to Hide** dialog box appears.
2. Click the check box next to the name of the window(s) that you want to close.
3. Click **OK** to close the dialog box and close the selected window(s).

Transferring to WindowMaker

To transfer from the WindowViewer program to the WindowMaker program

1. On the **File** menu, click WindowMaker. The Windows to Edit dialog box appears.

Tip To quickly transfer to WindowMaker, click the **Development** fast switch in the upper right hand corner of the menu bar (or use the short cut keys ALT + !). When you transfer using the fast switch, the **Windows to Edit** dialog box does not appear in WindowViewer. The windows that are open in WindowViewer when you transfer to WindowMaker will remain open.

Note The fast switch will only be available if, during development, the application developer configured the application to use it.

2. Click the check box next to the name of the window(s) that you want to be open when you transfer to WindowMaker.
3. Click **OK** to close the dialog box and transfer to WindowMaker.

Note If the application developer selected the **Close WindowViewer** option when WindowViewer's properties were configured during development, WindowViewer will automatically close when you transfer to WindowMaker.

Executing InTouch QuickScripts

By default, when WindowViewer is initially started, the logic for all scripts will be executing.

To stop all QuickScripts from executing

On the **Logic** menu, click **Halt Logic**. The **Windows to Edit** dialog box appears.

Note During development, if the application developer selected the **Allow CTRL-Break to stop scripts** option when WindowViewer was configured, you will not be able to stop the QuickScripts from executing regardless of whether the **Logic** menu is displayed or not.

Also, the **Halt Logic** command will not stop any currently executing asynchronous QuickFunctions, but it will prevent any new asynchronous QuickFunctions from executing.

For more information on the above items, see your online *InTouch User's Guide*.

Initializing I/O Conversations

When WindowViewer is started, it automatically processes an *initiate* request to start all I/O conversations. If an I/O Server program does not respond to WindowViewer's i*nitiate* request, you can force WindowViewer to try again to establish the I/O conversation.

To start all uninitiated I/O conversations

On the **Special** menu, click **Start Uninitiated Conversations**.

Tip Executing this command will not affect existing conversations.

To restart all I/O conversations

On the **Special** menu, click **Reinitialize I/O**.

Tip This command closes all existing I/O conversations and restarts the entire process of setting up I/O conversations. All I/O points are affected by this command.

C H A P T E R 3

Using InTouch Security

InTouch gives you the option of selecting either traditional InTouch-based security, operating system-based security or ArchestrA-based security. All InTouch security methods are configurable with application granularity, meaning that you can operate two applications with different security settings on the same computer.

All three security methods are compatible with Network Application Development (NAD) distribution of applications. InTouch-based security works with NAD as it did in pervious versions of InTouch. For more information on NAD, see Network Application Development (NAD). ArchestrA-based security is centralized regardless of whether NAD is used or not.

If the authentication mode is operating system-based, then the user names will be the Windows Domain Name / User name pairs. If the mode is ArchestrA-based, then the security related activities would be configured externally in the Integrated Development Environment (IDE). For more information on the IDE, see the Wonderware ® ArchestrA™ Integrated Development Environment (IDE) Guide.

Contents

Using InTouch-Based Security

Applying security to your application is optional. The default security setting for InTouch applications is "None." However, by applying security to your application, you can control specific functions that an operator is allowed to perform by linking those functions to internal tagnames. In addition, when you establish security on your application, audit trails can be created that tie the operator to all alarms/events that occur during the time he/she is logged on to the system.

Security is based on the concept of the operator "logging on" to the application, typing his/her name and password. You must configure a user name, password, and access level for each operator.

There is no association between Microsoft operating system security and InTouch security.

When you create a new application, by default, the user name is set to "Administrator" with an access level of 9999 (which allows access to all security commands). After you add a new user name to the security list and restart WindowMaker or WindowViewer, the default user name is automatically reset to "None" with an access level of "0" (which prevents access to the **Configure Users** command in both WindowMaker and WindowViewer). However, the Administrator account and password remain and can still be used.

After an operator logs on to the application, access to any protected function will be granted upon verification of the operator's password and access level against the value specified for the internal security tagname linked to the function.

For example, you can control access to a window, or the visibility of an object and so on, by specifying that the logged on operator's "Access Level" must be greater than 2000.

The operator can log on to the application by executing the **Log on** menu command under **Security** in the WindowViewer **Special** menu (if the **Special** menu is displayed), or you can create a custom log on window with touch-sensitive input objects that are linked to internal security tagnames.

The commands used to establish security on an application are located under **Security** on the **Special** menu in both WindowMaker and WindowViewer. The security commands are used to log on and off the application, change passwords and to configure the list of valid user names, passwords and access levels.

Using the Security Internal Tagnames

After you implement security for your application, there are several internal security tagnames that you can use on buttons, in animation link expressions or QuickScripts, and so on, to control whether or not the logged on operator is allowed to perform specific functions:

Tagname	Type	Valid Values	Access
$AccessLevel	System Integer	0-9999	Read Only
$Operator	System Message	16-characters max	Read Only
$ChangePassword	System Discrete	1 or 0	Read Write
$ConfigureUsers	System Discrete	1 or 0	Read Write

Tagname	Type	Valid Values	Access
$InactivityTimeout	System Discrete	1 or 0	Read Only
$InactivityWarning	System Discrete	1 or 0	Read Only
$OperatorEntered	System Message	16 characters max	Read Write
$PasswordEntered	System Message	16 characters max	Read Write

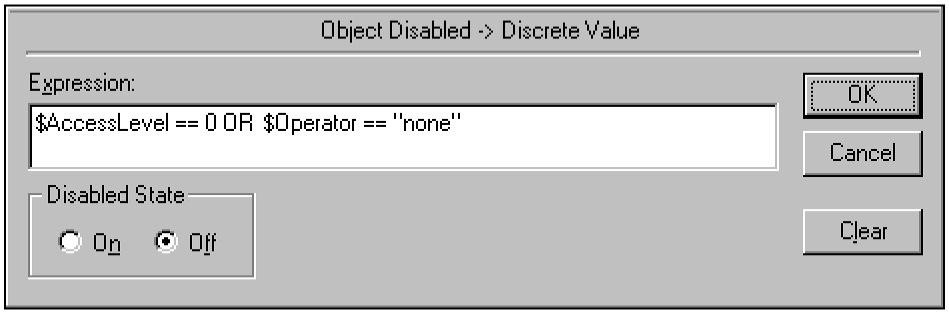
For example, to make an object become visible based on the logged on user's access level, the following statement could be used in a visibility animation link expression:

```
$AccessLevel >= 2000;
```

Or, a QuickScript can be bounded by an IF statement:

```
IF $Operator == "DayShift" THEN
   Show "Control Panel Window";
   {and other lines that only execute for the DayShift
      Operator}
ENDIF;
```

You can also control an object's touch functionality based upon the value of an internal security tagname by using the **Disable** animation link. For example:


Object Disabled -> Discrete Value

Expression:

$AccessLevel == 0 OR $Operator == "none"

Disabled State: On [✓ Off]

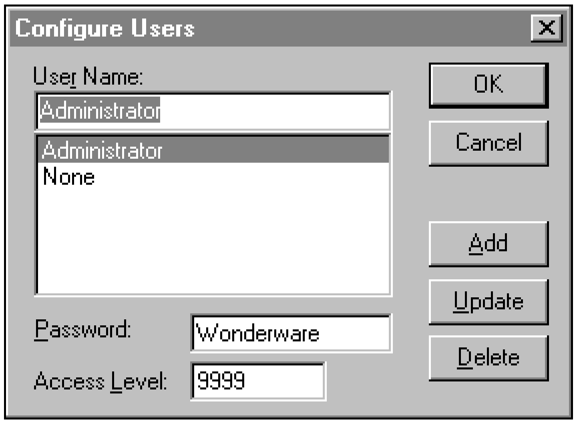
By using this expression, if no one is logged on, the object or button is secured from tampering.

For more information on the internal security tagnames, see your online *InTouch Reference Guide*.

Configuring the User and Security Levels

To configure security for the operators of your application

1. On the **Special** menu, point to **Security**, then click **Configure Users**. The **Configure Users** dialog box appears.`



User Name: Administrator

Administrator
None

Password: Wonderware

Access Level: 9999

Tip If you right-click any of the text entry boxes in any dialog box, a menu appears displaying the commands that you can apply to the selected text.

---

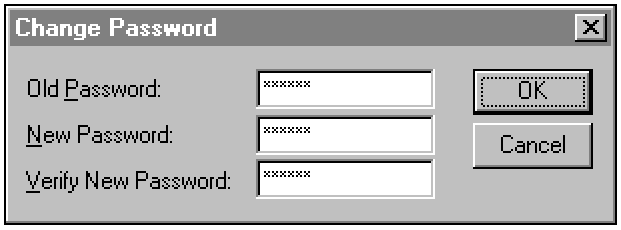
2. In the **User Name** box, type the name that you want to assign to the operator.
3. In the **Password** box, type a password (up to 16 characters).
4. In the **Access Level** box, type a value (lowest = 0 to highest = 9999).
5. Click **Add** to add the user name to the security list.
6. To **modify** an existing user name, select the desired name in the **User Name** list. Type your changes and then click **Update** to accept the changes.
7. To **delete** a user name, select it in the list and then click **Delete**.

The **None** and **Administrator** names are reserved and only the password of the Administrator may be changed. Once you have configured user names for your application, you should change the Administrator name's password since it will more than likely become commonly known to most users of the system. The Administrator default access level (9999) is the highest and allows access to everything including the Configure Users menu command.

Changing a Security Log On Password

To change the password for an operator

1. On the **Special** menu, point to **Security** and then click **Change Password**. The **Change Password** dialog box appears.`



   If you right-click any of the text entry boxes in any dialog box, a menu appears displaying the commands that you can apply to the selected text.

2. In the **Old Password** field, type the old password.
3. In the **New Password** field, type the new password (up to 16 characters).
4. In the **Verify Password** field, type the new password again.
5. Click **OK**.

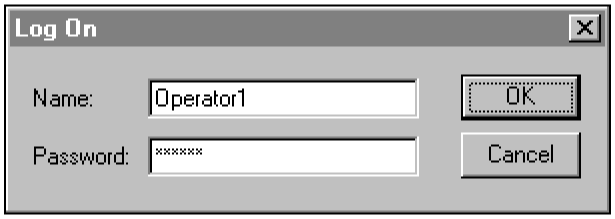
   To prevent anyone who may be watching the operator from seeing the password, the information entered is displayed on the screen as asterisks.

Tip If you do not plan on displaying the **Special** menu in WindowViewer, you can create a discrete button and link it to the **$ChangePassword** internal tagname. Do so in order to set the **$ChangePassword** tagname equal to 1 to cause the **Change Password** dialog box to be displayed. Once displayed, the operator can change his/her password.

Logging on to an InTouch-Secured Application

To log on to an application

1. On the **Special** menu, point to **Security** and then, click **Log On**. The **Log On** dialog box appears.



2. In the **Name** box, type your user name.
3. In the **Password** box, type your password.

4. Click **OK**.

   If the information is entered incorrectly or is invalid, a message box indicating that log on failed appears.

   If log on is successful, the **$AccessLevel** internal tagname will be set to its predefined value (configured in the security user list).

Note See also PostLogonDialog().

Logging Off an InTouch-Secured Application

To log off the application

- On the **Special** menu, point to **Security** and then click **Log Off**.

When this command is executed, the "User Name" is reset to "None" with an Access Level of "0."

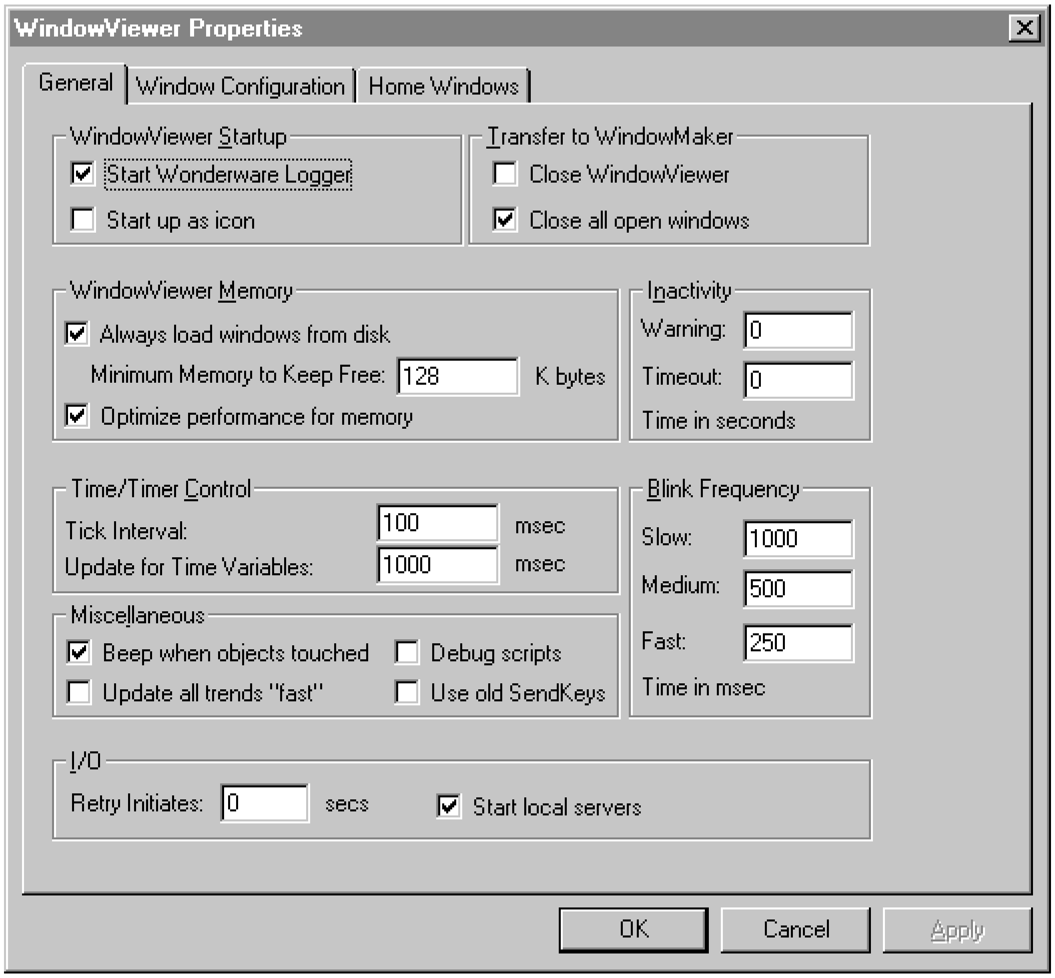
You can also configure the application to automatically log off the operator after a specified amount of time has elapsed with no activity by the operator.

Automatically Logging Off the System

You can configure your application to automatically log off the operator when there has been no activity for a specified period of time by using the warning and time out settings.

To configure inactivity

1. On the **Special** menu, point to **Configure** and then click **WindowViewer** or in the Application Explorer under **Configure**, double-click **WindowViewer**. The **WindowViewer Properties** dialog box appears with the **General** properties sheet active.In the Application Explorer, you can also right-click **WindowViewer** and then click **Open**.



WindowViewer Startup
- [✓] Start Wonderware Logger
- [ ] Start up as icon

Transfer to WindowMaker
- [ ] Close WindowViewer
- [✓] Close all open windows

WindowViewer Memory
- [✓] Always load windows from disk
- Minimum Memory to Keep Free: 128 K bytes
- [✓] Optimize performance for memory

Inactivity
- Warning: 0
- Timeout: 0
- Time in seconds

Time/Timer Control
- Tick Interval: 100 msec
- Update for Time Variables: 1000 msec

Blink Frequency
- Slow: 1000
- Medium: 500
- Fast: 250
- Time in msec

Miscellaneous
- [✓] Beep when objects touched
- [ ] Debug scripts
- [ ] Update all trends "fast"
- [ ] Use old SendKeys

I/O
- Retry Initiates: 0 secs
- [✓] Start local servers

2. In the **Warning** box, type the number of seconds that can elapse with no operator activity (mouse clicks or keystrokes) before the system discrete tagname **$InactivityWarning** is set to 1 (True). When the **Inactivity Warning** is set to zero, there will not be an inactivity warning.

Tip You can use **$InactivityWarning** in a Condition QuickScript to show a window warning the operator that he/she is about to be logged off the system. If the operator clicks the mouse, presses a key, or performs an action using any other pointing device before the specified time-out elapses, they are not logged off. **$InactivityWarning** and the timer are reset.

3. In the **Timeout** box, type the number of seconds that can elapse with no operator activity (mouse clicks, keystrokes, and so on) before the system discrete tagname **$InactivityTimeout** is set to 1 (True). When **$InactivityTimeout** is true, the system equates the logged on operator name to the reserved name "None" and sets the security tagname, **$AccessLevel**, to 0.

Tip You can use **$InactivityTimeout** in a Condition QuickScript to show a window telling the operator that he/she has been logged off the application.

You can use the **Timeout** feature independently of the **Warning** feature. However, the **Timeout** value must be greater than the **Warning** value for proper use of both system tagnames.

For example, set $InactivityWarning to 30 and $InactivityTimeout to 45. The operator will be logged off 15 seconds after the $InactivityWarning variable is set to 1.

Using Operating System-Based Security

In the operating system-based authentication scheme, user names can be chosen from the list of users associated with a Windows Network Domain\ Workgroup. Each user name has an assigned access level that determines the user's authorization for a given activity. Since the operating system manages the passwords internally, InTouch will not store passwords.

Operating system-based security uses the InTouch script function AddPermission to maintain a list of users and their corresponding access levels. This list, created after the execution of the AddPermission() call, is written to disk. The file containing the authentication details of users will not be copied to the NAD client machines.

Setting Up Operating System-Based Security

Operating system-based security can be selected from the Security Type menu in WindowMaker. This would typically be done when a new application is created. Typically, you will select security settings when you create a new InTouch application.

To set operating system-based security:

1. Open a window in WindowMaker.
2. On the **Special** menu, point to **Security**, then point to **Select Security Type** and select **OS**.

Using the Security System Tagnames

The three new security system tagnames for operating system-based security are $OperatorDomain, $OperatorDomainEntered and $OperatorName. The tables below explain the functions of these new system tags and provide examples:

$OperatorDomain

Category	**Security**
Usage	If operating system-based security is selected and an operator has successfully logged on, the $OperatorDomain tag will contain the domain or machine name that was specified at log on. If ArchestrA security is selected a user is logged on, the $OperatorDomain will contain "ArchestrA." If InTouch security is selected, the $OperatorDomain tag contains the string "InTouch." If "None" is selected, it is an empty string "".
Remarks	N/A
Data Type	String

Example(s)	`$Operator = "john";`
	`$OperatorDomain="CORPORATE_HQ";`

See Also	**$Operator**

$OperatorDomainEntered

Category	**Security**
Usage	Whenever the $PasswordEntered tag changes, a log on is attempted internally without any GUI being displayed. The log on attempt uses the $*Entered tags as input user name and the string value of $OperatorDomainEntered as the domain name (used only if the current mode is operating system-based security). If the security mode is not operating system-based, this tag is ignored.
Remarks	N/A
Data Type	String

Example(s)	`$OperatorEntered="john";`
	`$OperatorDomainEntered="Corporate_hq"`
	`$PasswordEntered="password";`

See Also	**$Operator**

$OperatorName

Category	**Security**
Usage	The $OperatorName tag will contain the full legal name of the operator if operating system-based or ArchestrA authentication is used and someone has logged on and has not logged off. Otherwise, the tag will contain the name of the user logged on (same contents as the $Operator tag).
Remarks	N/A
Data Type	String
Example(s)	`$Operator = "john";$OperatorName = "John Smith";`
See Also	**$Operator**

$VerifiedUserName

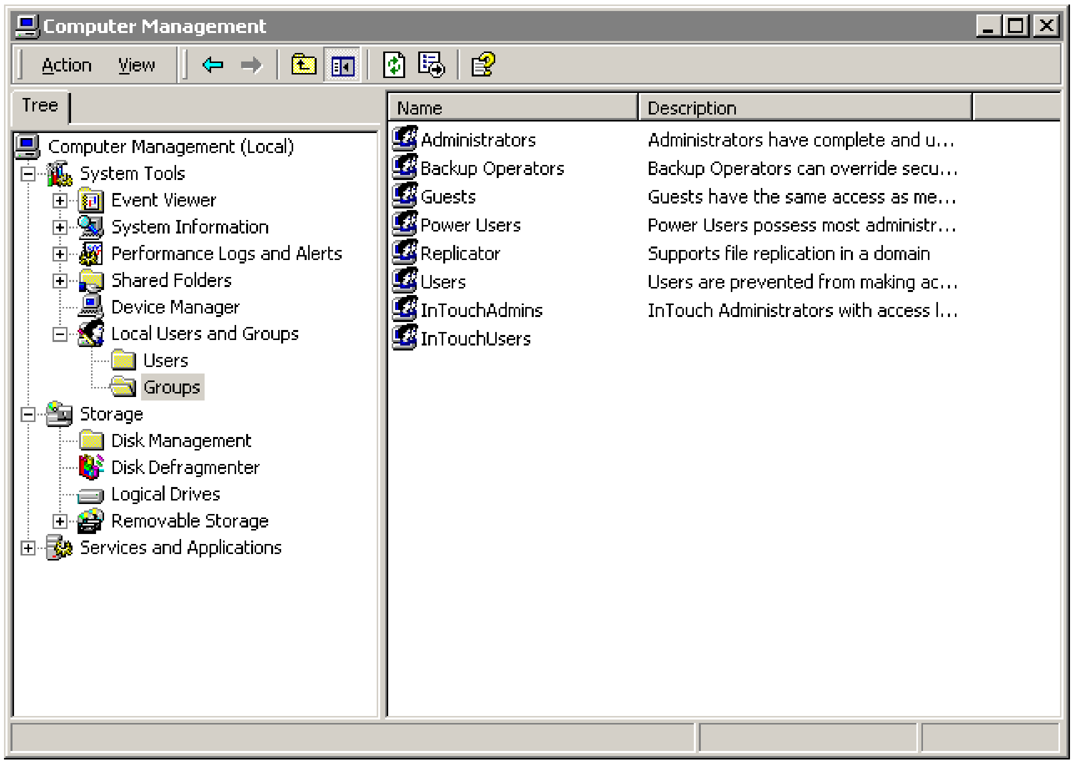
	Contains the verified user's full name if the call to InvisibleVerifyCredentials() is successful and if the security mode is set to operating system-based or ArchestrA AppServer-based security. If the call fails, then the above system tag will be set to null.
Category	security
Usage	**`$VerifiedUserName`**
Remarks	Whenever the above system tag changes (meaning whenever InvisibleVerifiCredentials is called), an event will be generated and the column "Value" will contain the verified user's full name if the call was successful. The column "Value" will contain null if the call failed. The column Name will contain the value "$VerifiedUserFullName."
Data Type	String
Valid Values	**`A User's full Name`**
Example(s)	`Tag = InvisibleVerifyCrdentials("john","password", "Plant_Floor"). If the call is successful, the $VerifiedUserName is set to "John Smith" and an Operator Event is generated. The name column indicates in the event $VerifiedUserName and the value column is set to "John Smith." If the above call is not successful, $VerifiedUserName and the value column in the event are set to "". Every time the above script function is called, the $VerifiedUserName is set to the corresponding user's full name or to null.`
See Also	**InvisibleVerifyCredentials(); $OperatorName, $Operator**

Setting Up User Groups

Operating system-based security uses a list of authorized Windows user groups. Users will create user groups either on the local computer or a domain server. The administrator must associate Windows users to groups by adding them to specific groups. In WindowMaker, the application developer must use the script function AddPermission() to set up a list of groups with the desired access levels for each group. AddPermission() will typically be called on application start-up so that view recognizes all the authorized user groups when a user is ready to log on. You must be logged on as a local administrator or have local administrator rights to set up and administrator group on a local computer.

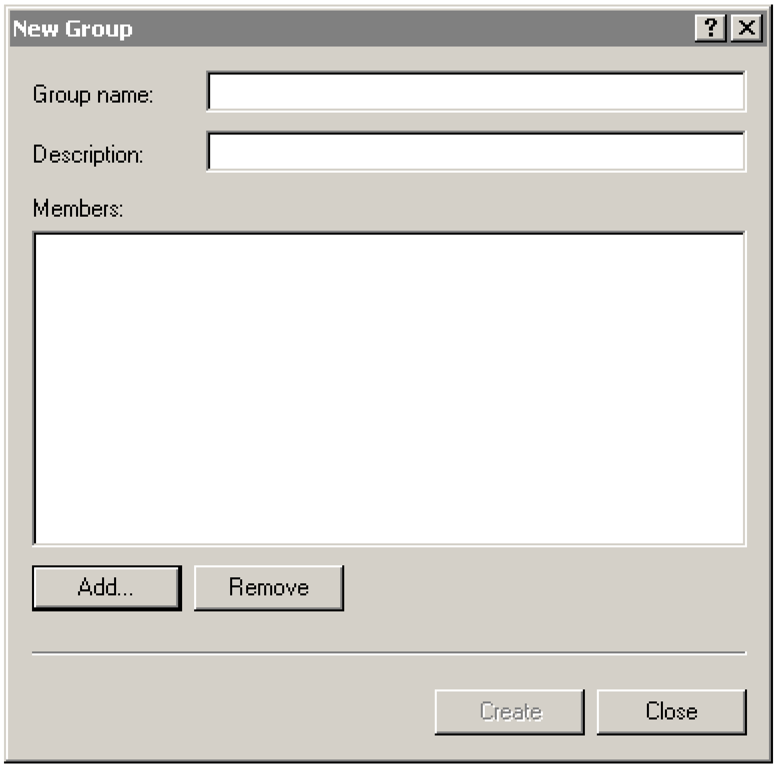
To set up an administrator group on a local computer and add users

1. On the **Start** menu, click **Settings**, then click **Control Panel**.
2. Double-click the **Administrative Tools** icon, then double-click the **Computer Management** icon.The Computer Management window appears.
3. In the operating system's Computer Management window, create a new user group for InTouch administrators (for example, InTouchAdmins).



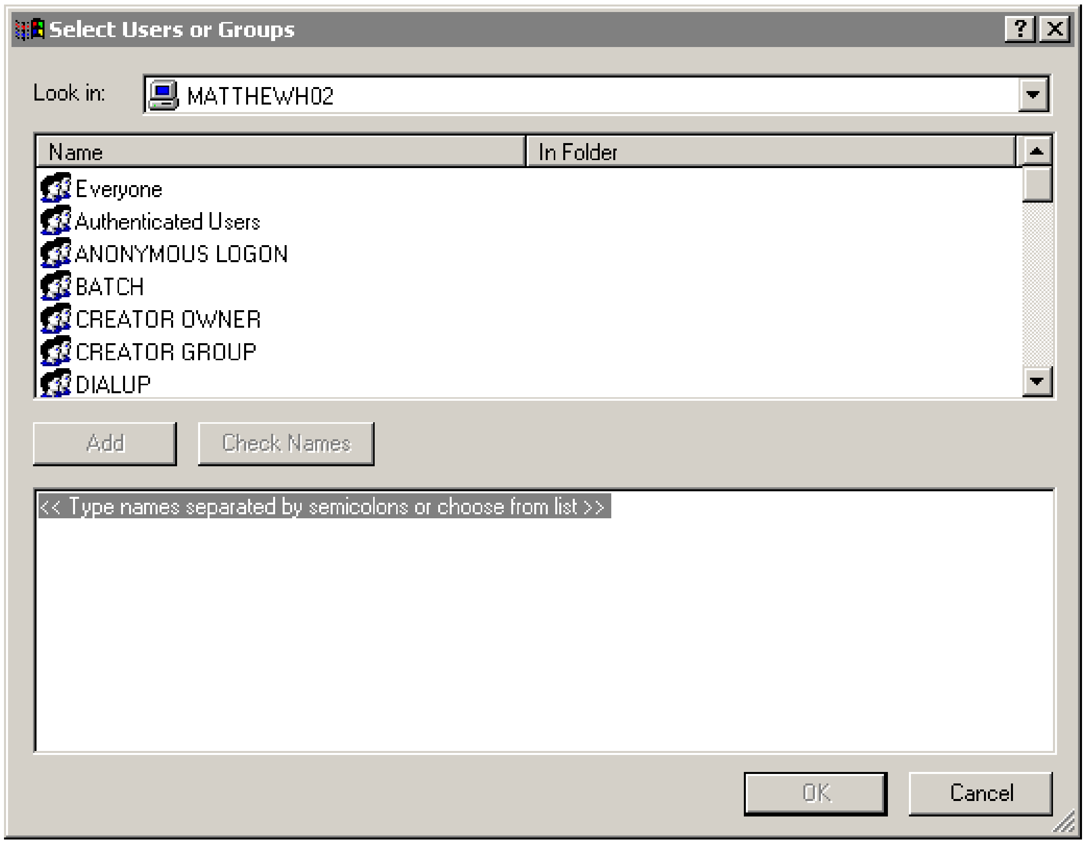
Name	Description
Administrators	Administrators have complete and u...
Backup Operators	Backup Operators can override secu...
Guests	Guests have the same access as me...
Power Users	Power Users possess most administr...
Replicator	Supports file replication in a domain
Users	Users are prevented from making ac...
InTouchAdmins	InTouch Administrators with access l...
InTouchUsers	

4. From the **Action** menu, click **New Group** or right-click in the right pane of the column management screen and click **New Group** on the shortcut menu. The **New Group** dialog box appears.



5. In the **Group Name** box, type a name for your group, then type a description, if desired, in the **Description** box.

6. Click **Add**. The **Select Users or Groups** window appears.



7. Click the name of the member to add, then click **Add**.

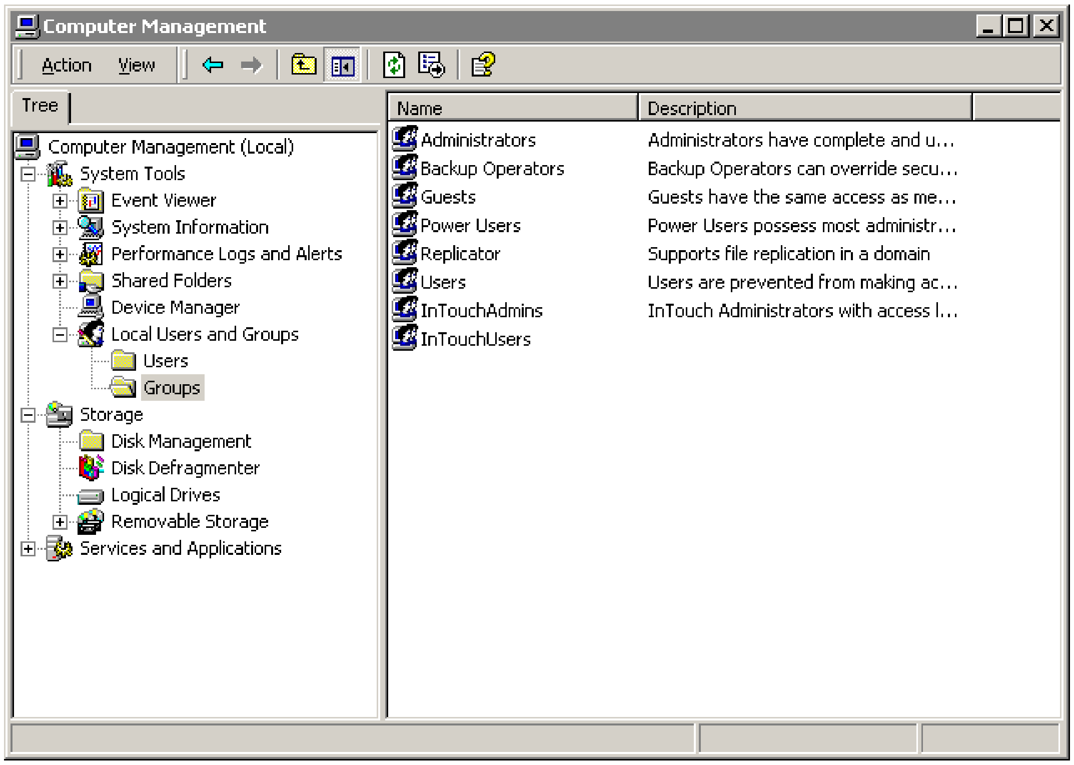
Tip To add multiple members, hold down the CTRL key and click additional member names, then click **Add**.

8. Click **OK**.

To set up a user group on a local computer and add users

1. On the **Start** menu, click **Settings**, then click **Control Panel**.
2. Double-click the **Administrative Tools** icon, then double-click the **Computer Management** icon.The Computer Management window appears.

3. In the operating system's **Computer Management** window, create a new user group for InTouch users (for example, InTouchUsers).



Name	Description
Administrators	Administrators have complete and u...
Backup Operators	Backup Operators can override secu...
Guests	Guests have the same access as me...
Power Users	Power Users possess most administr...
Replicator	Supports file replication in a domain
Users	Users are prevented from making ac...
InTouchAdmins	InTouch Administrators with access l...
InTouchUsers	

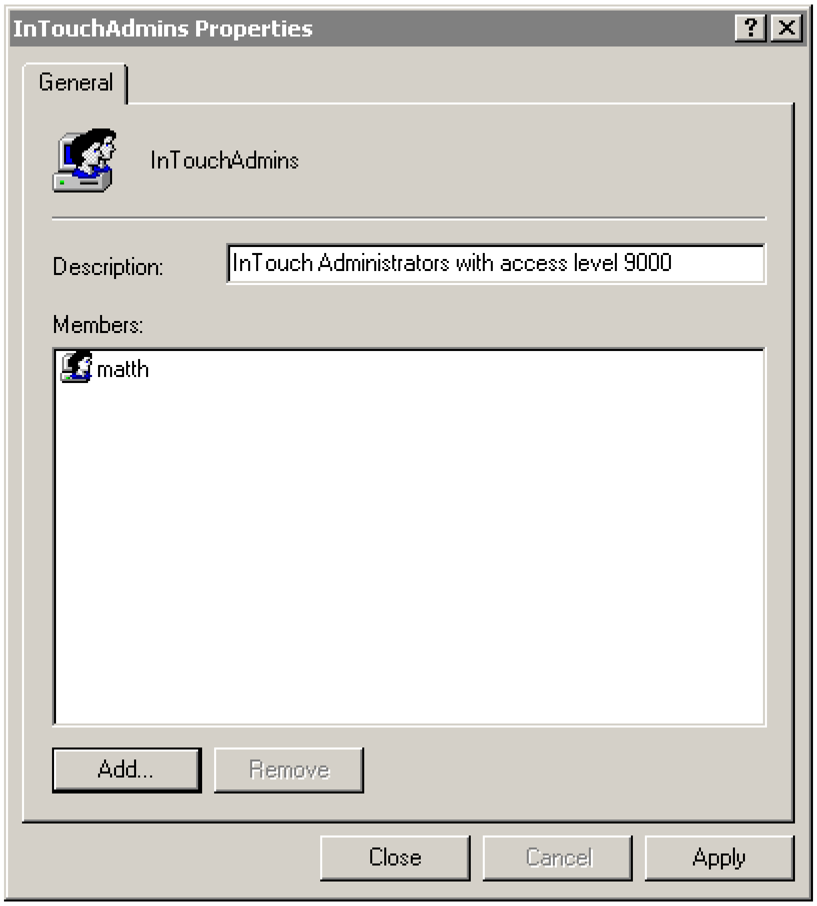
4. From the **Action** menu, click **New Group** or right-click in the right pane of the column management window and click **New Group** on the shortcut menu. The **New Group** dialog box appears.
5. In the **Group Name** box, type a name for your group, then type a description, if desired, in the Description box.
6. Click **Add**. The **Select Users or Groups** window appears.
7. Click the name of the member to add, then click **Add**.

Tip To add multiple members, hold down the CTRL key and click additional member names, then click **Add**.

8. Click **OK**.

To add users to an existing group

1. Double-click the group name to view the group properties dialog box. The group's **Properties** dialog box appears.



Description: InTouch Administrators with access level 9000

Members: matth

2. Click **Add** to add users. The **Select Users or Groups** window appears.
3. Click the name of the member to add, then click **Add**.

Tip To add multiple members, hold down the CTRL key and click additional member names, then click **Add**.

4. Click **OK**.

For more information on creating user groups, see your Windows operating system documentation.

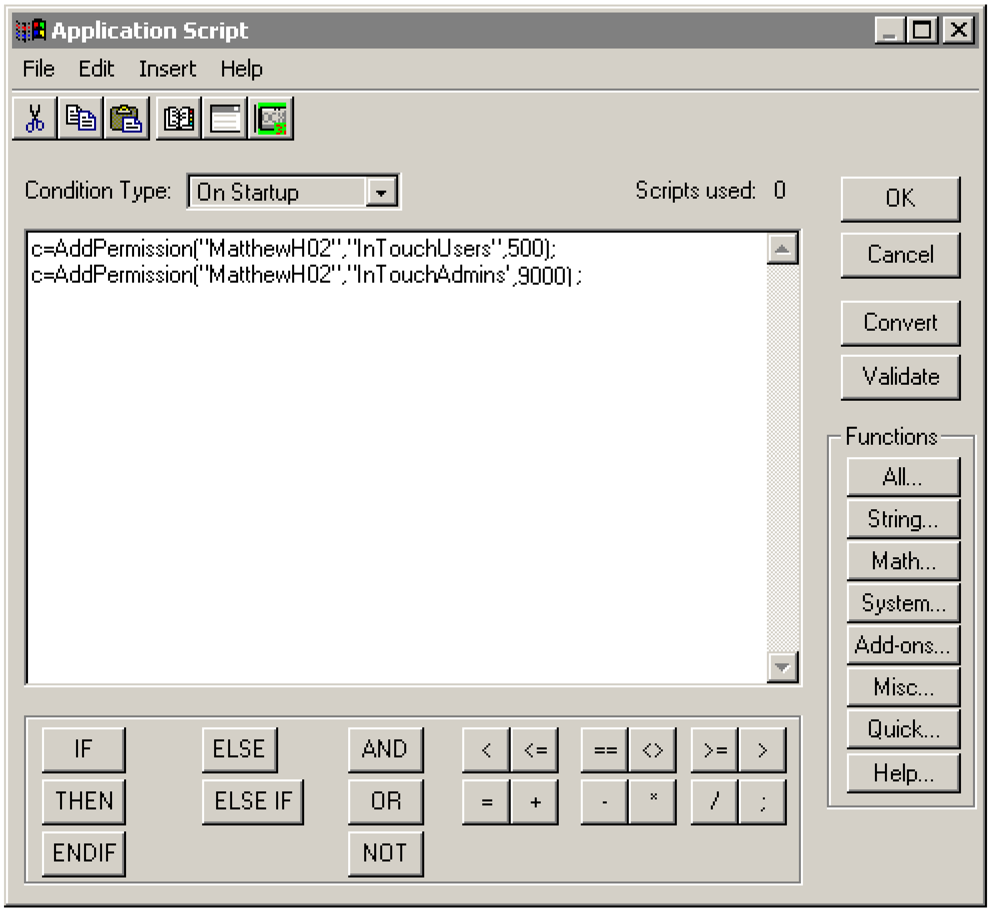
Once you configure the InTouch Application to utilize the operating system Authentication and internal InTouch Authorization, the **Change Password**, **LogOn**, **Configure Users** and **LogOff** options on the **Special**...**Security** menu will be unavailable.

Setting Up Access Levels for Groups in WindowMaker

The final step in setting up operating system-based security for InTouch is to set up access levels for groups in WindowMaker. In WindowMaker, AddPermission() is used to setup a list of groups with the desired access level.

To set up access levels for groups

1. Start WindowMaker.
2. On the **Special** menu, point to **Scripts** and click **Application Scripts**.
3. In the **Condition Type** list, click **On Startup**.
4. Using AddPermission(), enter the group names and corresponding access levels. The required and default arguments for AddPermission() are operating system or Domain, Group and Access level.



Condition Type: On Startup

Scripts used: 0

```
c=AddPermission("MatthewH02","InTouchUsers",500);
c=AddPermission("MatthewH02","InTouchAdmins',9000);
```

5. Click **OK**.

InTouch Operating System Security Functions

The operating system-based authentication scheme inherits enforcement of some account policies from the operating system, while other policies are enforced within InTouch. Password policies such as maximum and minimum password age and minimum password length are enforced by the operating system. User names used during installation act as a part of the operating system. The Windows domain must be set up with the desired account policies in order to enforce these standards. InTouch enforces the inactivity time-out.

Logging on to an Operating System-Secured Application

When a user logs on to an InTouch application, a dialog box appears requiring user name, password and domain or local computer name. The Domain/User Name combination is passed on to the operating system to authenticate the account. An attempt is made to log on with or without enabling the operating system cache. If the user cannot be logged on without the cache (due to a network outage, for example), but the user was previously authenticated with the cache enabled, then the user's full name and access level is obtained from the local InTouch cache. If all of the security checks are cleared successfully, the user is considered to be logged on to InTouch and the relevant data structures (for example, $Operator) are updated. Otherwise, an appropriate error message is displayed.

Note If the operator has never logged on successfully before and the domain is unavailable, the log on will fail.

Using ArchestrA-Based Security

When you configure a node to use ArchestrA security, InTouch invokes methods and dialog boxes from AppServer for configuring users and for Logins/Logoff. Users are configured on the AppServer Galaxy node. For more information, see the AppServer documentation.

About ArchestrA Authentication and Authorization

The ArchestrA Security system is designed to allow the system administrators to easily define the users of the system and assign the operations they are allowed to perform. The security permissions are defined in terms of the operations the users can perform using these UI tools. The basic approach consists of the following steps:

1. Define the security model.
2. Organize the automation objects according to the security model for protection.
3. Define the users according to the security model.

The system administrator defines the system users by creating corresponding user profiles. He/she then defines their roles (a user can have more than one role) by selecting from a list of user roles predefined in the security model.

InTouchView users are normally authenticated by means of password based log-in. An install time option is provided to use the Windows log-in for authentication. This of-course requires that the user be defined in the Windows operating system.

A user authentication utility is also provided as an ActiveX control that can be used by third parties to develop client applications for ArchestrA with support for ArchestrA security.

Setting Up ArchestrA-Based Security

After the administrator has defined user profiles for users of the InTouch or InTouchView application, the administrator can set up ArchestrA

To set ArchestrA-based security:

1. Open a window in WindowMaker.
2. On the **Special** menu, point to **Security,** then point to **Select Security Type** and select **ArchestrA**.

Note Once you configure the InTouch application to use ArchestrA authentication and authorization, the **Change Password**, **LogOn**, **Configure Users** and **LogOff** options on the **Special**...**Security** menu will be unavailable.

InTouch ArchestrA Security Functions

The ArchestrA security system is a global function that applies to every object in the Galaxy Database. It is a relationship-based system between users and the objects and functions of the Galaxy. This system is based on security roles (configuration, system administration, and runtime permissions) and security groups, which determine a particular security role's runtime permissions on an object-level basis. Configuration of the security system is done in the GalaxyObject's editor and applied to every object through its own editor.

Logging on to an ArchestrA-Secured Application

Users typically log on and log off of an ArchestrA-secured InTouch application by entering a valid user name and password.

If your system has been configured with open security, the log in credentials of the default user are used and you are not prompted to log in. The following procedure assumes your system has been configured for authentication mode security.

To log on, do the following:

1. Launch the ArchestrA-secured InTouch application. A log in dialog box is displayed.

2. Type a valid user name and password. If the system cannot authenticate you, you will be prompted again to log on.

After the system authenticates your log in data, access to all future operations is granted based on your associated roles/permissions in the Security Model.

Creating a Custom Security Log on Window

If the **Special** menu will not be displayed in WindowViewer, you can create a custom logon window that the operator uses to log on to the application.

To create a custom log on window

- Link the **$OperatorEntered**, **$PasswordEntered** and **$OperatorDomainEntered** system tagnames to user input objects or use them in a QuickScript to set the "User Name," "Password," and Domain Name (these are internal message type tagnames that are intended for write operation only.) The **$OperatorDomainEntered** is required only if the security mode is operating system-based. Otherwise, this tag will be ignored. If the security mode is operating system-based and the **$OperatorDomainEntered** is null, it is treated as pointing to local machine.

For example:

```
Set the User Name string into ->$OperatorEntered

Set the User Domain Name string into ->
    $OperatorDomainEntered

Set the User Password string into ->$PasswordEntered
```

Unlike $OperatorEntered and $PasswordEntered, a change in the value of $OperatorDomainEntered does not trigger a log on.

If the entries are valid, the **$AccessLevel** and **$Operator** internal tagnames are set to their predefined values (configured in the security user list).

Also, when you are not displaying the **Special** menu in WindowViewer, you can link a **User Input - Discrete** button to the **$ChangePassword** tagname to show the **Change Password** dialog box and allow the operator to change his/her password. When the operator clicks the button, the value of the **$ChangePassword** tagname is set to 1 and the **Change Password** dialog box appears. When the operator closes the dialog box, the system resets the value to 0. (This is a system discrete tagname intended for write operation only.)

You can also link a **User Input - Discrete** button to the **$ConfigureUsers** tagname to allow an authorized operator with an access level of equal to or greater than 9000 to access the **Configure Users** dialog box to edit the security user name list. When the operator clicks the button, the value of the **$ConfigureUsers** tagname is set to 1 and the **Configure Users** dialog box appears. When the operator closes the dialog box, the system resets the value to 0. (This is a system discrete tagname intended for write operation only.)

Security and Alarms

When an InTouch alarm provider is configured to use either operating system or ArchestrA authentication and an alarm occurs, the alarm display object will contain the full name of the operator in the operator column, assuming the operator is logged on. For example if a user is registered in the PLANT_FLOOR domain with a UserID of JohnS and a full name of John Smith, the operator column will contain John Smith. If the alarm is subsequently ACKed, and the node performing the ACK is set to use operating system or ArchestA security, the operator column will be updated to show the full name of the ACK operator. Otherwise the alarm display object will show a computer name concatenated with whatever is in the $Operator tag

Full Name Expansion in Alarm Records

InTouch security can provide an operator’s full name on alarm acknowledgements. This is also possible on records pertaining to alarm detection. In most organizations, a Login ID is not a person's full name, but rather an abbreviation or role classification.

When operating system authentication is chosen at provider and consumer InTouch nodes:

- The alarm display object will display full names when alarms are generated and when acknowledgements are performed.
- The alarm print object will print full names when alarms are generated and when acknowledgements are performed.
- The Alarm DB Logger will record domain name, login userID, and full user name with each alarm record for both operator and AckOperator fields. This will allow for unique identification even if an organization has two employees with identical full names.
- The domain name and login userID will be concatenated to the existing operator name field when alarm and ack packets are sent on the network.

InTouch Security Script Functions

InTouch features new security related script functions. The new functions are described in the section below.

InvisibleVerifyCredentials()

	Checks to verify the credentials of the given user without logging the user on to InTouch.	
Category	**security**	
Syntax	`AnalogTag=InvisibleVerifyCredentials( "UserId", "Password", "Domain" );`	
	Parameter	**Description**
	UserId	Windows operating system user account name that is part of local machine, workgroup or domain.

Remarks	If the supplied combination of user, password and domain are valid then the corresponding access level associated with the user is returned as an integer, in all other cases -1 is returned. This call does not change the currently logged on user. The domain field is only valid for the operating system-based security mode. If ArchestrA security mode is in use and if ArchestrA security is in turn using operating system-based security, the UserId should contain the fully qualified user name with domain name or computer name.
Example(s)	`AnalogTag=InvisibleVerifyCredentials( "john", "Password", "corporate_hq" );`
See Also	**PostLogonDialog(), AttemptInvisibleLogon(), IsAssignedRole(), QueryGroupMembership(), AddPermission().**

PostLogonDialog()

	Brings up the InTouch Logon Dialog and returns TRUE.	
Category	**security**	
Syntax	`DiscreteTag=PostLogonDialog();`	
	Parameter	**Description**
	N/A	N/A
Remarks	Brings up the InTouch Logon Dialog and returns TRUE.	
Example(s)	`DiscreteTag=PostLogonDialog();`	
See Also	**InvisibleVerifyCredentials(), AttemptInvisibleLogon(), IsAssignedRole(), QueryGroupMembership(), AddPermission().**	

AttemptInvisibleLogon()

	Attempts to logon to InTouch using the supplied credentials.	
Category	**security**	
Syntax	`DiscreteTag=AttemptInvisibleLogon( "UserId", "Password", "Domain" );`	
	Parameter	**Description**
	UserId	A valid user account name.
	Password	Password of the user.
	Domain	Name of the local machine, workgroup or domain to which the user belongs. This column applies only if the current security type is operating system-based.
	DiscreteTag	Return value: returns TRUE if authentication is successful. Otherwise, it returns FALSE.

Remarks	An attempt is made to logon to InTouch using the supplied credentials (Domain is ignored if the security mode is not operating system-based). If the logon attempt succeeds, then TRUE is returned and the system tags $OperatorDomain, $OperatorName, $AccessLevel and $Operator are updated accordingly. If the Logon attempt fails, then FALSE is returned and the currently logged on user (if any) continues to be the current user. The domain field is only valid for the operating system-based security mode. If ArchestrA security mode is in use and if ArchestrA security is in turn using operating system-based security, the UserId should contain the fully qualified user name with domain name or computer name.
Example(s)	`AnalogTag=AttemptInvisibleLogon( "john", "Password", "corporate_hq" ); \\ security is operating system-based AnalogTag=AttemptInvisibleLogon( "john", "Password", "" ); \\ security is either InTouch or ArchestrA-based.`
See Also	**PostLogonDialog(), InvisibleVerifyCredentials(), IsAssignedRole(), QueryGroupMembership(), AddPermission().**

IsAssignedRole()

Category	**security**	
Syntax	`DiscreteTag=IsAssignedRole( "RoleName" );`	
	Parameter	**Description**
	RoleName	The role associated with a AppServer user.
Remarks	Valid for ArchestrA security mode only and applies to the currently logged on user. If a user is currently logged on and if he has the role RoleName assigned to him in Galaxy IDE, then a TRUE is returned. In all other cases a FALSE is returned.	
Example(s)	`DiscreteTag=IsAssignedRole( "Administrator" );`	
See Also	**AttemptInvisibleLogon(), PostLogonDialog(), InvisibleVerifyCredentials(), QueryGroupMembership(), AddPermission().**	

QueryGroupMembership()

Category	**security**	
Syntax	`DiscreteTag=QueryGroupMembership( "Domain", "Group" );`	
	Parameter	**Description**
	Domain	Name of the domain or local machine in which the group is located
	Group	Name of the domain or local machine in which the group is located.

Remarks	Valid for operating system security mode only and applies to the currently logged on user. If a user is currently logged on and if he part of the group Group which is located on the domain Domain then a TRUE is returned and in all other cases a FALSE is returned. QueryGroupMembership will work with InTouch OS security and with InTouch ArchestrA security only when the ArchestrA security is set to OS Group based security
Example(s)	`DiscreteTag=QueryGroupMembership( "corporate_hq", "InTouchAdmins" ); DiscreteTag=QueryGroupMembership( "JohnS01", "InTouchUsers" );`
See Also	**BOOL PostLogonDialog(), InvisibleVerifyCredentials(), BOOL IsAssignedRole(), AttemptInvisibleLogon(), AddPermission().**

AddPermission()

	Attempts to reach the account Account located on domain Domain	
Category	**security**	
Syntax	`DiscreteTag=AddPermission( "Domain", "Group", AccessLevel);`	
	Parameter	**Description**
	Domain	Name of the domain or local machine in which the group is located.
	Group	Windows user group.
	AccessLevel	InTouch AccessLevel that is associated with the given group
Remarks	Valid for operating system security mode only. An attempt is made to reach the account Account located on domain Domain. If successful, a TRUE is returned and the access level AccessLevel is assigned to the account in the internal records in InTouch for use during authorization when a user logs on. In all other cases, a FALSE is returned.	
Example(s)	`DiscreteTag=AddPermission( "corporate_hq", "InTouchAdmins", 9000);DiscreteTag=AddPermission( "johns01", "InTouchUsers", 5000);`	
See Also	**PostLogonDialog(), InvisibleVerifyCredentials(), IsAssignedRole(), AttemptInvisibleLogon(), QueryGroupMembership().**	

ChangePassword()

	Displays the **Change Password** dialog box allowing the logged on operator to change his/her password.	
Category	**security**	
Syntax	`[Result=]ChangePassword();`	
	Parameter	**Description**
	[Result]	Returns one of the following integer values:
		0 = Cancel was pressed.
		1 = OK was pressed.

Remarks	If using touch screen applications, there is an option to use the alphanumeric keyboard.
Example(s)	`Errmsg=ChangePassword();`
	This QuickScript, if placed on a button or called based on a Condition or Data Change QuickScript, will open a dialog box (with optional keyboard) prompting the user to enter the current password, the new password and verification of the new password.

Logoff()

	Logs the user off of InTouch.	
Category	**security**	
Syntax	`DiscreteTag = LogOff();`	
	Parameter	**Description**
	N/A	
Remarks	Logs off the currently logged on user and sets the current user status to the default none operator.	
Example(s)	`DiscreteTag = LogOff();`	
See Also	**PostLogonDialog(), InvisibleVerifyCredentials(), IsAssignedRole(), AttemptInvisibleLogon(), QueryGroupMembership(), AddPermission().**	

InTouch Security System Tags

$OperatorName

Category	**Security**
Usage	The $OperatorName tag will contain the full name of the operator if operating system-based or ArchestrA authentication is used and someone has logged on and has not logged off. Otherwise, the tag will contain the name of the user logged on (same contents as the $Operator tag).
Remarks	N/A
Data Type	String

Example(s)	`$Operator = "john";`
	`$OperatorName = "John Smith";`

See Also	**$Operator**

$OperatorDomain

Category	**Security**
Usage	If operating system-based security is selected and an operator has successfully logged on, the $OperatorDomain tag will contain the domain or machine name that was specified at log on. If ArchestrA security is selected a user is logged on, the $OperatorDomain will contain "ArchestrA." If InTouch security is selected, the $OperatorDomain tag contains the string "InTouch." If "None" is selected, it is a empty string "".
Remarks	N/A
Data Type	String

Example(s)	`$Operator = "john";`
	`$OperatorDomain="CORPORATE_HQ";`

See Also	**$Operator**

$OperatorDomainEntered

Category	**Security**
Usage	Whenever the $PasswordEntered tag changes, a log on is attempted internally without any GUI being displayed. The log on attempt uses the $*Entered tags as input user name and the string value of $OperatorDomainEntered as the domain name (used only if the current mode is operating system-based security). If the security mode is not operating system-based, this tag is ignored.
Remarks	N/A
Data Type	String

Example(s)	`$OperatorEntered="john";`
	`$OperatorDomainEntered="Corporate_HQ";`
	`$PasswordEntered="password";'`

See Also	**$Operator**

C H A P T E R 4

Using InTouchView

InTouchView is designed as a visualization and interface product specifically for use in an ArchestrA Application Server environment. InTouchView does not use a different code base and does not require a different installation procedure than InTouch. Application developers specify whether an application will use InTouch or InTouchView on a case by case basis. The developer uses a setting in WindowMaker to configure each application as either a full-featured InTouch or an InTouchView application.

Contents

About InTouchView

InTouchView allows you to develop ArchestrA-enabled applications which provide operators with the ability to perform tasks when connected to an ArchestrA Application Server system. The section below explains the differences between InTouch and InTouchView.

Differences between InTouchView and InTouch

The main differences for InTouchView applications are that they do not have the ability to connect to I/O sources other than an ArchestrA Application Server Galaxy. InTouchView applications also do not generate alarms, but they support the display and acknowledgment of alarms from other alarm providers. InTouchView applications use the ArchestrA security model.

Starting Up InTouchView

When you start WindowMaker, WindowMaker checks for file information regarding the application type. If the information indicates that the application is an InTouchView application, WindowMaker will restrict the menus and functions of the application accordingly.

The InTouchView License

If the application is an InTouchView type, WindowViewer requests a license from the License Manager to run an InTouchView application. Otherwise, it requests a license to run a full-featured InTouch application.

Acquiring a License for InTouchView

- Start the InTouch application via InTouch or InTouch TSE.

During the start-up, the InTouch software assesses the Application that is starting. It checks to see whether it is an InTouchView application and if so, it requests an InTouchView license from the License Manager.

Note If the application is not an InTouchView application, the software requests an InTouch license from the License Manager.

If the InTouchView license line is successfully read for this application, InTouch continues its start-up process. If an InTouchView license is not available, the InTouch software requests an InTouch license. If the InTouch line is successfully read, InTouch continues its startup process. Otherwise, InTouch indicates that the InTouchView license is unavailable, provides a brief description of the problem, and offers the options to quit, retry or run in demo mode.

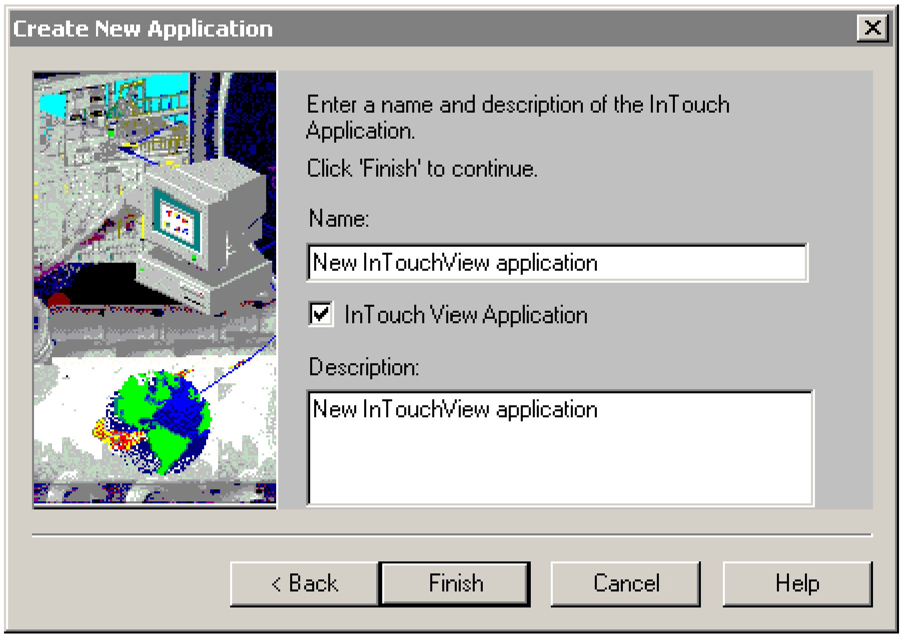
Creating a New InTouchView Application

To run WindowMaker, you must have an InTouch license, whether the application you are creating is an InTouchView or full-featured InTouch application.

To create a new InTouchView application

1. Point to **Start**, **Programs** and **Wonderware** and launch InTouch.
2. In the InTouch Application Manager, point to **File** and click **New**. You can also click the **New** icon on the toolbar.
3. Follow the new application wizard.

4. Enter the name of the application and its description and check the **InTouchView Application** check box.



Enter a name and description of the InTouch Application.

Click 'Finish' to continue.

Name:

New InTouchView application

☑ InTouch View Application

Description:

New InTouchView application

5. Click **Finish**. The system creates a new InTouchView application and sets the security type to ArchestrA.

Running an InTouchView Application

You can run an InTouchView application using the InTouchView or InTouch runtime license.

To run an application in InTouchView

- Open an InTouchView application in WindowViewer.

The InTouchView application does not generate historical logs, alarms or connect to I/O clients other than the Application Server Galaxy. Only SYS and USER-related events are generated.

Using the InTouchView Menus

You can use the InTouchView menus and functions to perform many of the functions available in InTouch. However, some functions are not supported by InTouchView. These functions will appear on menus, but will be grayed out to indicate that they are not enabled. The list below specifically explains which menus and functions are not available in InTouchView.

InTouchView WindowMaker Menu Differences

The following **Special** menu options are unavailable in an InTouchView application:

- Access Names
- Alarm Groups
- Configure..Alarms
- Configure..Historical Logging
- Configure..Distributed Name Manager

The following options are unavailable in the application frame and will be unavailable in an InTouchView application:

- Configure..Access Names
- Configure..Alarm Groups
- Configure..Alarms
- Configure..Historical Logging
- Configure..Distributed Name Manager

The following **Tagname Dictionary** options are not available and will be grayed out in an InTouchView application:

- Alarms
- Details & Alarms
- Log Data
- Log Events
- Priority (for Log Events)

Converting InTouchView Applications

InTouch allows you to convert applications from InTouchView to InTouch and vice versa. For example, if you created an InTouchView application but you need the application to access tags from data sources other than the Application Server, you can convert the application to an InTouch application. If you created an application in InTouch, but you want to access only an Application Server, you can convert the application to InTouchView.

Converting an InTouchView Application to an InTouch Application

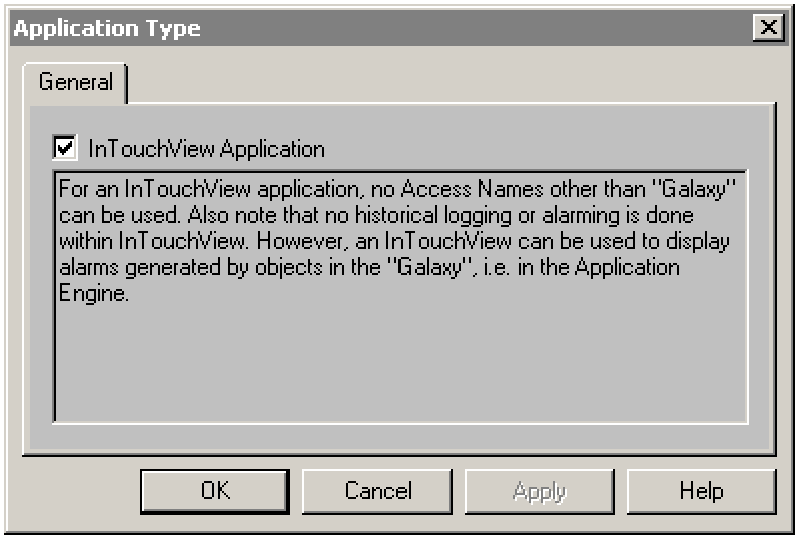
You can convert an InTouchView application to a standard InTouch application if you need the application to access tags from data sources other that the Application Server. A full InTouch license is required to run applications converted from InTouchView to InTouch.

Note You cannot change the Application Type while WindowViewer is active.

To convert an InTouchView application to an InTouch application:

1. Open the InTouchView application in WindowMaker.

2. From the **Special** menu, select **Application Type**. A dialog box appears with the InTouchView box checked.



[✓] InTouchView Application

For an InTouchView application, no Access Names other than "Galaxy" can be used. Also note that no historical logging or alarming is done within InTouchView. However, an InTouchView can be used to display alarms generated by objects in the "Galaxy", i.e. in the Application Engine.

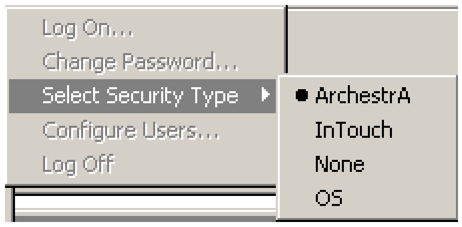
3. Uncheck the InTouchView box and click **OK**.

Once you convert the application to InTouch, you can use the functions that were disabled in the InTouchView application.

Note You may need to change the security type in your application once you have converted it from InTouchView to InTouch.

To change the security type:

From the **Special** menu, point to **Security**, then point to **Select Security Type**.:



4. Select the security type for your application.

   For more information on security, see Using InTouch Security.

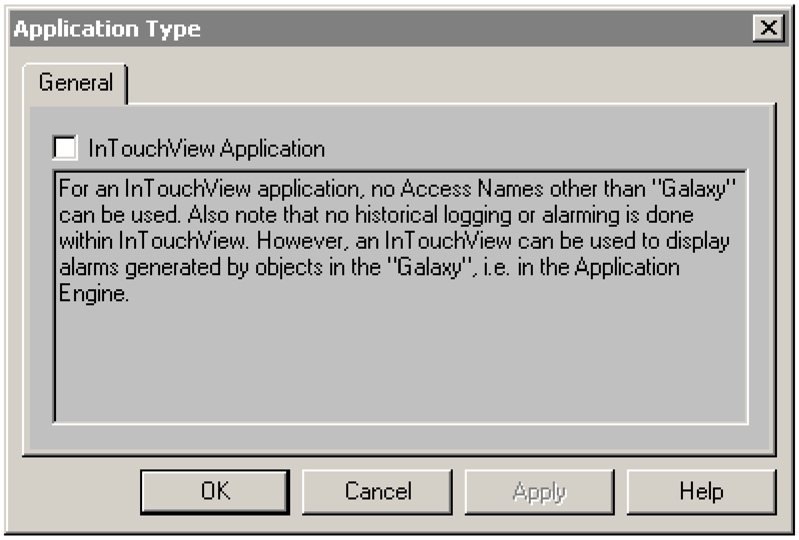
Converting an InTouch Application to an InTouchView Application

You can convert an InTouch application to InTouchView if you only want to connect to an Application Server.

To convert an InTouch application to an InTouchView application:

1. Open the InTouch application in WindowMaker.

   From the **Special** menu, select **Application Type**. A dialog box appears with the InTouchView box unchecked:

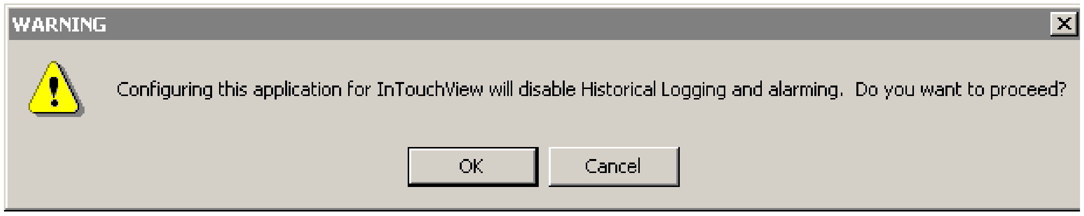


☐ InTouchView Application

For an InTouchView application, no Access Names other than "Galaxy" can be used. Also note that no historical logging or alarming is done within InTouchView. However, an InTouchView can be used to display alarms generated by objects in the "Galaxy", i.e. in the Application Engine.

2. Check the InTouchView box and click **OK**. The system then checks to see if the application uses any Access Names other than Galaxy. If so, a dialog box appears to notify you that you need to remove any non-Galaxy Access Names prior to switching to InTouchView. The system also notifies you that InTouchView does not support historical logging or alarming.

Important! You must remove all Access Names other than Galaxy prior to changing an application to InTouchView.



Configuring this application for InTouchView will disable Historical Logging and alarming. Do you want to proceed?

3. Click **OK**.

Once you convert the application to InTouchView, some menu functions will be disabled.

C H A P T E R 5

Building a Distributed Application

InTouch is designed to support both stand-alone and distributed applications. Stand-alone applications are those that use just one Operator Interface (OI) for each monitored system, such as in a boiler package control. Stand-alone applications are generally easier to configure, with minimal to no networking, and require only simple maintenance. Distributed applications, conversely, are much more complex, often with several layers of networks. Distributed applications, typically, have a central development station, central data storage, with many *client* stations which interact with the central station and each other.

InTouch provides many features that greatly ease the building and maintenance of distributed applications. One of the most powerful is "Network Application Development" (NAD). NAD allows many *client* stations to maintain a copy of a single application without restricting the development of that application. InTouch NAD also provides automatic notification to these *client* stations when the application changes.

This chapter describes how to use the distributed features of InTouch, the different architectures you can use, and the advantages and disadvantages of each.

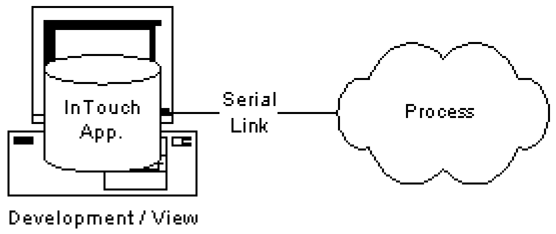
Contents

Network Architectures

InTouch is a highly configurable package that can be set up in many different ways, depending on your application's needs. This section provides an overview of the various architectures available with InTouch, and the relative advantages and disadvantages of each. While mention is made of various application components such as alarms and history, these systems are covered in greater depth later in their respective chapters.

Stand-alone Application

Stand-alone applications are defined as those with a single operator interface for each monitored process. These typically consist of one non-networked personal computer (PC) that functions as the primary operator interface (OI). This computer is connected to the industrial process via a direct connection, such as a serial cable



In this architecture, a single InTouch application is installed on the computer. If development work is required, the application can be developed directly on this computer. It can also be copied to another computer, and modified, and then copied back to the original computer. While not a network architecture, the stand-alone architecture is covered for completeness.

Advantages

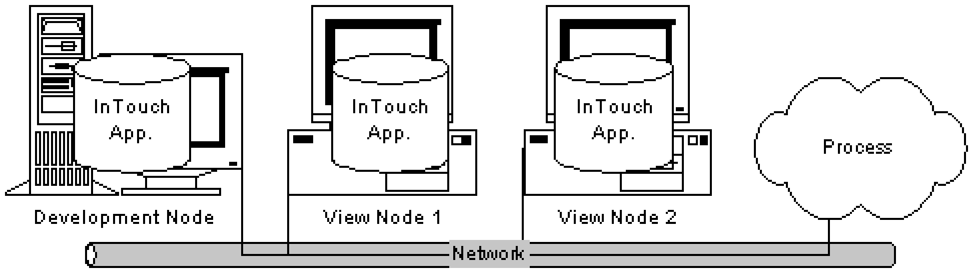
- Easy to maintain

Disadvantages

- Limited to single node

Client-Based Architecture

The client-based architecture is the first of the networked architectures, and is a direct outgrowth of the stand-alone. It provides a unique copy of one InTouch application for each computer running WindowViewer and NetDDE (View node). This application can be installed on each computer's hard drive, or in a unique location on a network server. In the example below, an application would be developed and tested on the development node, and then copied to each View node.



As each View node has an identical copy of the application, each must also have identical access to any data sources referred to by the application. These sources may be I/O Servers, SQL databases, DOS files, an so on. If a central data source is used, for example a network-shared I/O Server, each View node will maintain a separate conversation with the shared server, which can result in increased network loading. Therefore, it is a good idea to consider individual I/O Servers on each node if heavy network use is expected.

The client-based architecture has several tradeoffs when it comes to application maintenance. Since each node has its own copy of the application, the development node has unrestricted editing capability for that application. Modifications can be made and tested on that node without affecting the running process. The drawback of this approach is in the effort required to distribute the modified application to the View nodes. Each View node must be shut down locally, the new application copied to it, and then View must be restarted.

Advantages

- Unrestricted development of the application
- Inherent redundancy since each node can be self-sufficient
- No limit to the number of View nodes you can use

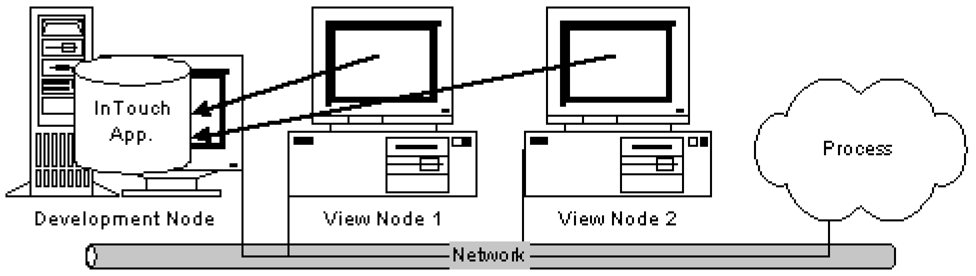
Disadvantages

- Distributing applications is difficult
- All nodes must have identical access to the same data sources

Note This architecture has been superseded by the NAD architecture, which is discussed later in this chapter. It is described here only for the purposes of presenting a complete overview of networking architectures.

Server-Based Architecture

The server-based architecture allows several View nodes to share a common InTouch application. In the example below, the two View nodes are accessing the same application from the development node. Each View node must create a logical drive in the networking software and map it to the shared network drive of the development node. Each View node must also have the shared application registered with the InTouch program.



As in the client-based architecture, each View node must have identical access to any data sources referred to by the application. There are also ways to tailor the data source locations by using a combination of scripts to get the node name and change each data source location based on that name.

For more information, see "Configuring InTouch for Common Data Sources."

This architecture does allow the View nodes to be updated when the application changes and WindowViewer is restarted

Advantages

- Single application to maintain
- View nodes automatically updated when application changes

Disadvantages

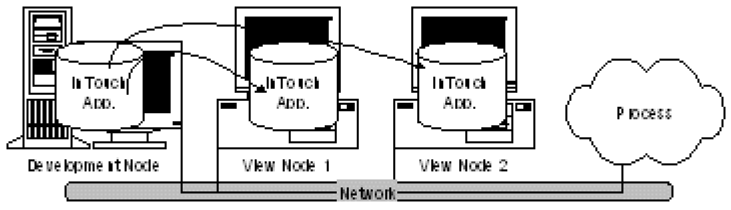
- Development of application is restricted
- No redundancy if the Development station goes down
- All nodes must have the same screen resolution

Note This architecture has been superseded by the NAD architecture, which is discussed later in this chapter. It is described here only for the purposes of presenting a complete overview of networking architectures.

Network Application Development (NAD)

Network Application Development or NAD is an architecture that combines the best of the client-based and server-based architectures. NAD provides automatic notification of application changes and automatic distribution of the updated applications to View nodes

In the NAD architecture, a master copy of an application is maintained on a central network location. Each View node loads that network application as they would in a server-based architecture but, instead of running the application from the server, the application is copied to and run from a user-defined location. This provides the client-based advantage of redundancy. In the example below, the two View nodes both have the master application registered from the development node, but actually run it from their own hard drives.



When a View node copies and runs a master application, it automatically monitors for changes in the master copy. These changes are indicated by a flag in the master application directory. This flag is set manually when the application developer uses the **Notify Clients** command on the WindowMaker **Special** menu while editing that application. When this flag changes, each View node has a user-definable action that specifies the response of that node. This can range from ignoring the flag, to automatically shutting down and restarting the View node, which reloads the master application.

Note If configured to write historical data to the master application node's "Application Directory", all NAD nodes will attempt to write their historical data to the master application. To avoid this, on each NAD node, configure historical data to write to a local directory, **not** the master application node.

Advantages

- Single application to maintain
- View nodes automatically notified when application changes
- Each View node has user-definable action for application updates
- Unrestricted development of the application

Disadvantages

- If distributing a large, complex application to numerous nodes, slowed system response time may be apparent on the initial down load, updates are optimized
- Limits flexibility of having different applications running on different nodes
- Application transfer may be a problem for slow networks or over serial connections

Configuring Network Resources

InTouch provides many configurable options for distributed use. This section contains hands-on examples of how to setup and use these options to distribute your InTouch applications.

Configuring UNC Paths for Files

InTouch supports the use of Universal Naming Conventions (UNCs) for application directory entries, configuration items, and distributed alarms. UNC allows direct access to network-based files without needing to create a mapped drive letter. Each UNC address may consist of three parts: Node, Share, and Path in the form \\Node\Share\Path. Node refers to the computer node name that contains the file share. Share refers to the logical name assigned to the shared directory on that computer. Path refers to the normal DOS path to that file with respect to the share.

Note When using Wonderware's SuiteLink protocol, **NodeNames** are limited to a maximum of 15 characters.

Before you can access a file through UNC, you must create a file share on the computer you want to access. You can share an entire drive, or just a directory of it; and even customize the security of that share. Either way, you must create a share name that will be used in the UNC address. For more information on creating a file share, refer to the manuals for your Windows operating system.

Once the share is created, you can use a UNC address to refer to that drive anywhere that you would normally enter a file path. For example, the InTouch program allows either a UNC path or a standard DOS path to be entered for the location of InTouch applications.

For example, assume that you have a computer with the network name of "EngineRm" that you have shared the root drive "C:\" with the share name of "Root". To set up a UNC path to the "C:\InTouch.32\Apps\Boiler" application you must use the following UNC:

\\EngineRm\Root\InTouch.32\Apps\Boiler

If the "Boiler" directory itself was shared as "Boiler," the UNC could be shortened to:

\\EnginerRm\Boiler

No path is required if the share is the path.

Note If you need to write to a file referred to by a UNC address, the share must be a read/write share, even on a local node. If you create a share that is password-protected, you will not be able to access the share with a UNC unless you first set up a network drive mapping. You can set up a drive mapping from the remote node by using Windows Explorer.

Wonderware SuiteLink Communication Protocol

Wonderware FactorySuite is shipped with Wonderware's communications protocol SuiteLink. Wonderware SuiteLink uses a TCP/IP based protocol. SuiteLink is designed specifically to meet industrial needs, such as data integrity, high-throughput, and easier diagnostics. This protocol standard is supported for both Microsoft Windows NT and Windows 2000.

SuiteLink is not a replacement for DDE, FastDDE, or NetDDE. Each connection between a client and a server depends on your network situation. SuiteLink was designed specifically for high speed industrial applications and provides the following features:

Value Time Quality (VTQ) places a timestamp and quality indicator on all data values delivered to VTQ-aware clients.

Extensive diagnostics of the data throughput, the server loading, computer resource consumption, and network transport are made accessible through the Microsoft Windows NT operating system performance monitor. This feature is critical for the scheme and maintenance of distributed industrial networks.

Consistent high data volumes can be maintained between applications regardless if the applications are on a single node or distributed over a large node count.

Note The network transport protocol is TCP/IP using Microsoft's standard Winsock interface.

For more information on using SuiteLink, see Chapter 13, "I/O Communications."

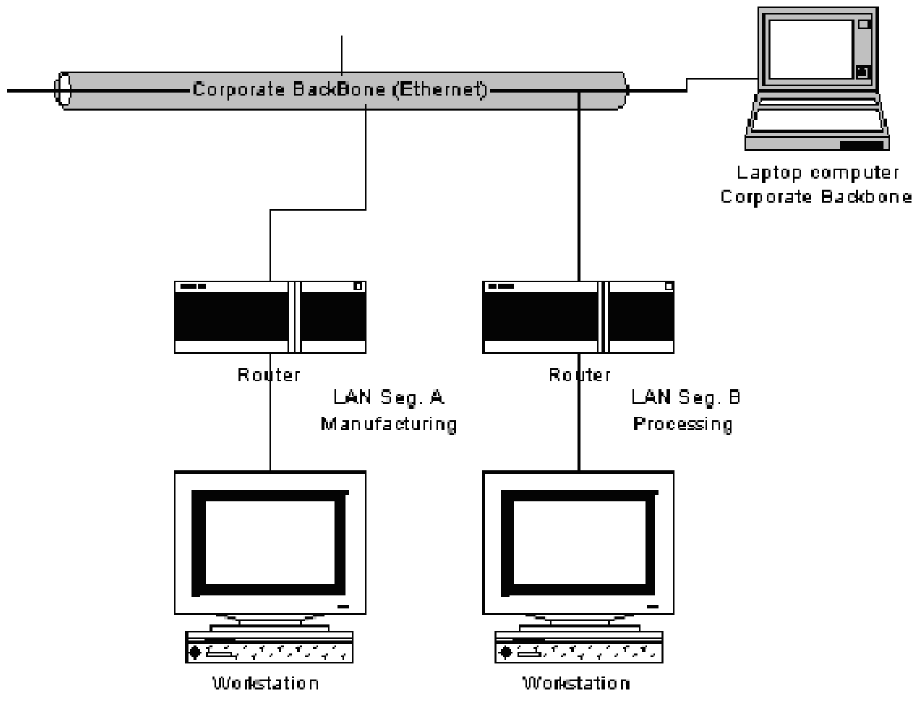
Troubleshooting Networks

There are some key networking architecture constraints that must be considered when using InTouch's distributed network features. One of the first things to recognize when implementing a large network of InTouch nodes is that all Ethernet connections are not the same. Many MIS departments segment their networks using devices called "routers." These routers act as traffic cops to regulate the flow of traffic from one Ethernet segment to another. Routers have the ability to "filter" out certain types of network traffic and addresses.

If you are using NetBEUI, it can be frustrating trying to connect to a remote I/O Server in another building or different city and find that it does not work, even though the networks are connected. If this happens it usually means the router has been set up to filter out NetBEUI traffic. One way around this is to reprogram the router to allow this traffic. However, reprogramming the router does have one drawback in that the NetBEUI broadcasts will now reach a larger audience and add traffic to the other segments on the network.

Note Wonderware's SuiteLink protocol cannot be used with NetBEUI. The SuiteLink protocol requires that all computers being referenced have a computer name that is not longer than 15 characters.

Network Segmentation Example



To prevent this, switch to a TCP/IP protocol and set up the router to route from one router to the next automatically. The router must be configured to allow bi-directional data flow (A to B, and B to A) to minimize network traffic crossing the router onto other segments. By using TCP/IP, you will improve network performance while allowing conversations over very long distances using a large variety of network services (such as, Internet, Frame Relay, ISDN, and so on). Another key reason for you to use TCP/IP is the quick adoption of TCP/IP as the protocol of choice for the PLC suppliers.

Microsoft makes switching to TCP/IP a very easy task. The Windows NT Server comes with the Network Client for Windows 2000 or Windows NT 4.x (or later) and has TCP/IP support built in. The Windows NT Server also has the ability to administer the allocation of TCP/IP addresses and names dynamically.

Microsoft's main networking product, Windows NT Server, also poses some challenges for networking InTouch nodes.

For example, whenever you try to retrieve InTouch files such as historical log files, from a different Domain, you must supply a password and have an account in that domain to make the connection. The creation of these accounts and constant interruptions for passwords make this almost unworkable. There are two solutions to avoid this:

1. You can make all the computers join the same domain; however, there can be administrative problems if there are too many computers in the domain.
2. You could setup a "Trust Relationship" between the domains to allow computers in one domain to share the resources of another domain without having to create additional accounts and supply passwords each time you connect.

The latter setup is definitely preferable, as well as being easier to setup and manage. Other, more advanced models of NT Domain Architecture are available that you may have to implement which have similar constraints.

It is recommended that you consult a Microsoft NT systems specialist on the design and topology of your whole network implementation in an instance such as this.

Configuring InTouch for Common Data Sources

InTouch allows you to build your application using several different architectures. Regardless of the architecture you choose, it is important to consider the data sources that application will access and how it will access them.

Each architecture shares a common trait in that the application is run by each View node as if it owned it. While this may not seem problematic at first, it's important when you consider the data source references that an InTouch application may contain. Typical data sources are Access Names, SQL connect strings, or Recipe files.

Each of these sources are retrieved through a reference address, such as D:\PROCESS\RECIPE.CSV in the case of a Recipe file, or DSN=PROCDB in the case of a SQL connect. While these addresses may make sense from the perspective of the computer they were developed on, they may be meaningless when that application is copied to, and run from, a View node that has no D: drive or registered ODBC data source name called PROCDB.

If you plan to distribute an application to more than one node, you need to consider the impact of your data source addresses. There are two basic ways to do this:

1. Create identical copies of the data sources on each View node, or
2. Use only global addresses for the data sources.

The following sections discuss these options as they pertain to the two major data sources: remote data and file access.

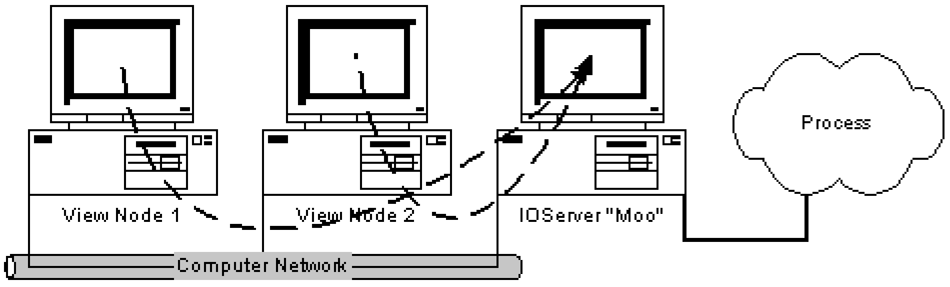
InTouch Access Names

InTouch uses Access Names to reference real-time I/O data. Each Access Name equates to an I/O address, which can contain a Node, Application, and Topic. In a distributed application, I/O references can be set up as global addresses to a network I/O Server or local addresses to a local I/O Server.

For more information on Access Names, see Chapter 13, "I/O Communications.""

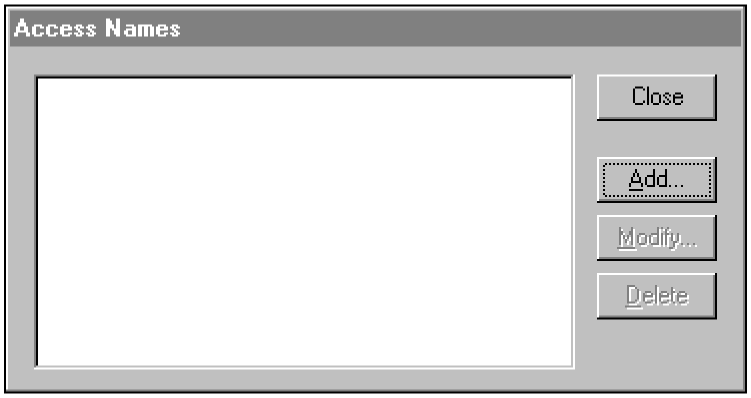
Global Addresses to I/O Data Sources

Global addresses to I/O data allow all of the View nodes to share a common network-based I/O Server. This prevents the cost of multiple I/O Servers, but is less fault-tolerant and can result in lower overall performance. In the example below, two View nodes, each running a copy of the same application, are referencing the same I/O data source. Since each application uses a fully qualified I/O address for the data source, all references point to the same I/O Server.

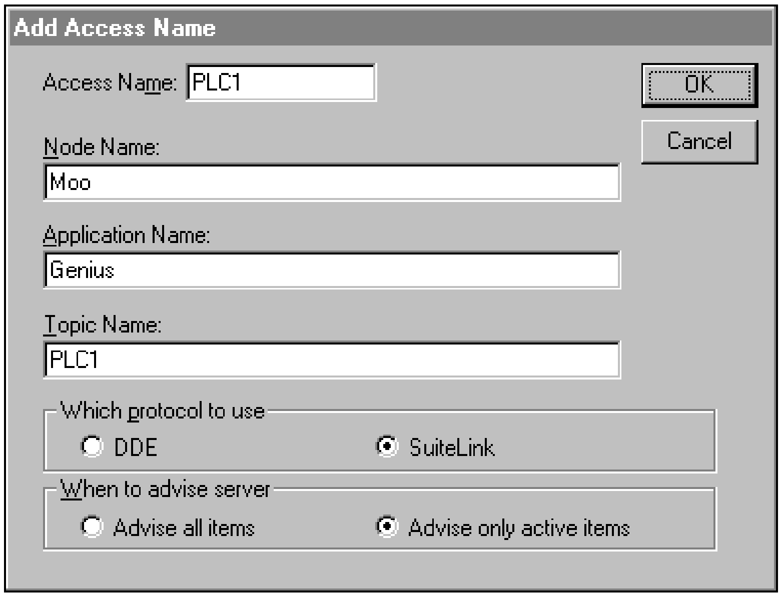


To set up this configuration

1. On the **Special** menu, click **Access Name** or in the Application Explorer under **Configure**, double-click **Access Names**. The **Access Names** dialog box appears.



2. Click **Add**. The **Add Access Name** dialog box appears.



Access Name: PLC1

Node Name: Moo

Application Name: Genius

Topic Name: PLC1

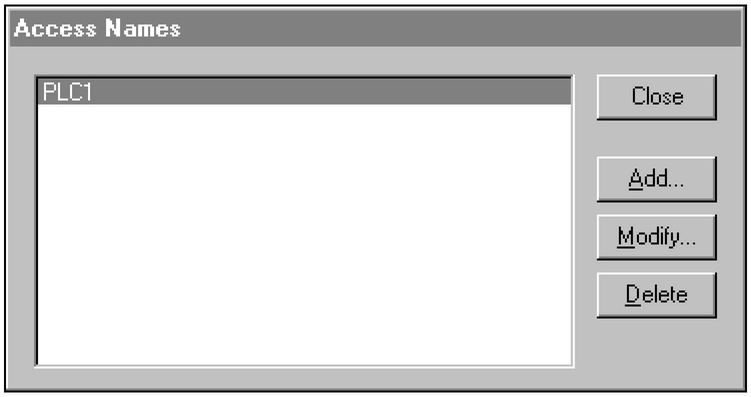
Which protocol to use: ( ) DDE [✓ SuiteLink]

When to advise server: ( ) Advise all items [✓ Advise only active items]

3. In the **Access Name** box, type **PLC1**.
4. In the **Node Name** box, type **Moo**. (Do not prefix the node name with \\.)
5. In the **Application Name** box, type **Genius**.
6. In the **Topic Name** box, type **PLC1**.

Note You can define any arbitrary name for the **Access Name**, but the **Node Name**, **Application Name** and **Topic Name** must reference the computer with the I/O Server. However, using the same name as the Topic Name is recommended for simplicity.

7. Select the protocol that you are using.
8. Click **OK**. The **Access Names** dialog box reappears showing the new Access Name in its list:



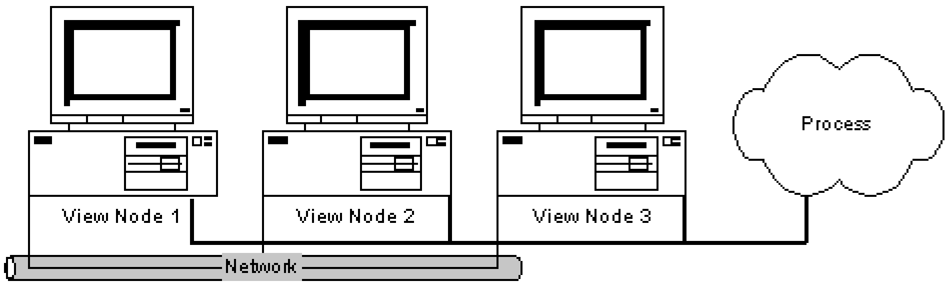
PLC1

9. Click **Close**.

Local Addresses to I/O Data Sources

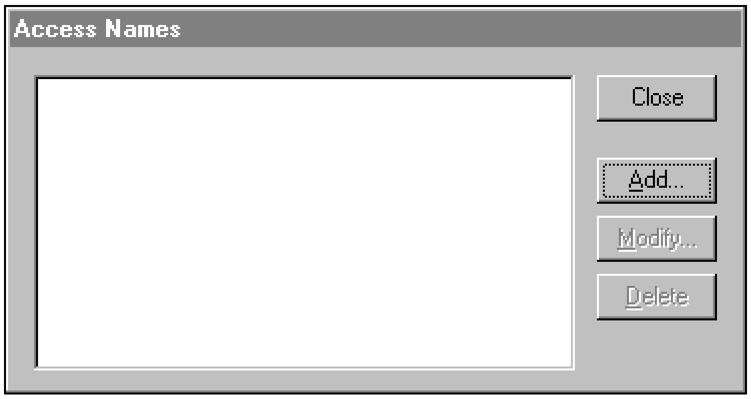
Local addresses to I/O data can be used when each View node has it's own I/O Server. This architecture provides fault-tolerant operation, as each View node can independently run if the network goes down. In the example below, two View nodes, each running a copy of the same application, are referencing their own I/O data source. Since each application uses a local I/O address for the data source, each reference points to the local I/O Server.

This method however, significantly increases the load on the process connection network. That is, three nodes triples the traffic created by one node, as each node's requests must be separately processed.

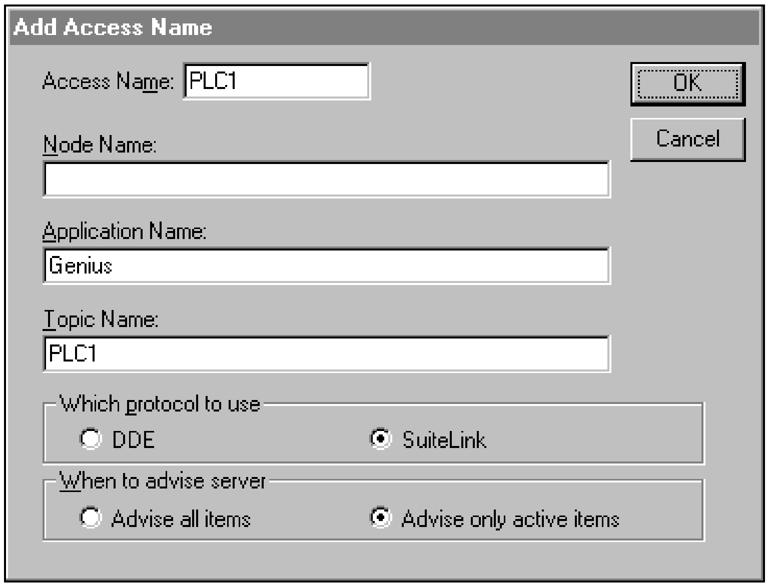


To set up this configuration

1. On the **Special** menu, click **Access Name** or in the Application Explorer under **Configure**, double-click **Access Names**. The **Access Names** dialog box appears.



2. Click **Add**. The **Add Access Name** dialog box appears.



Access Name: PLC1

Node Name:

Application Name: Genius

Topic Name: PLC1

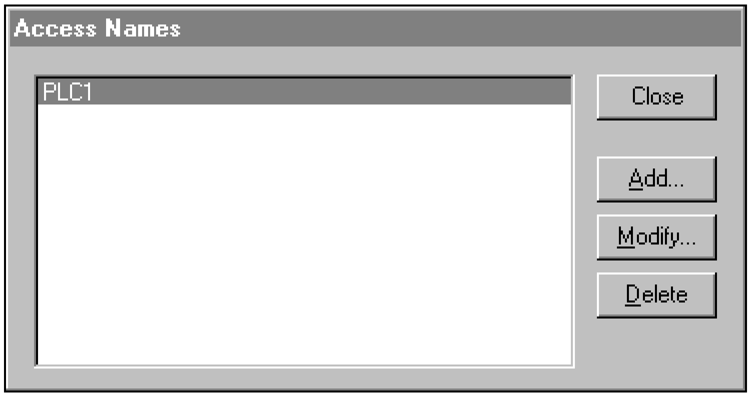
Which protocol to use: DDE [✓ SuiteLink]

When to advise server: Advise all items [✓ Advise only active items]

3. In the **Access Name** box, type **PLC1**.
4. Leave the **Node Name** box blank.
5. In the **Application Name** box, type **Genius**.
6. In the **Topic Name** box, type **PLC1**.

Note You can define any arbitrary name for the **Access Name**, but the **Node Name**, **Application Name** and **Topic Name** must reference the computer with the I/O Server. However, using the same name as the Topic Name is recommended for simplicity.

7. Select the protocol that you are using.
8. Click **OK**. The **Access Names** dialog box reappears showing the new Access Name in its list:



PLC1

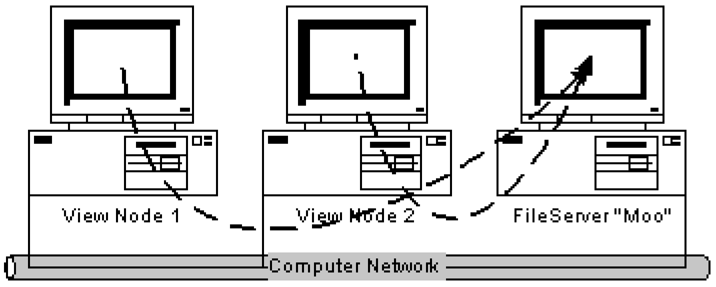
9. Click **Close**.

File Access

InTouch uses DOS files, FAT or NTFS to read and write reference data. Certain programs such as Recipe Manager, utilize files very heavily. In a distributed application, file references can be set up as global addresses to a network file server or as local addresses to a local file.

Global Addresses to File Data Sources

Global addresses to file data allow all of the View nodes to share a common network-based set of files. This provides single-source maintenance of the files, but is less fault-tolerant than local copies. In the example below, two View nodes, each running a copy of the same application, can reference the same Recipe file. Since each application uses a drive letter mapped to a fully-qualified network path for the file, all references point to the same file.



To set up this configuration

Map a network drive to the shared path containing the referenced files. In a script to retrieve a Recipe file, type the following:

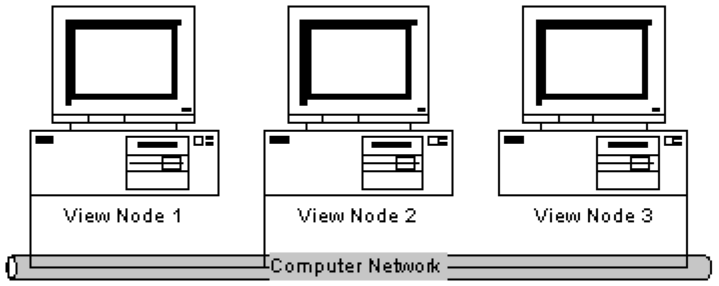
```
RecipeSelectRecipe("G:\Directory\Recipe.CSV", "review",
    "RecipeName");
```

where "G:\" is the mapped drive letter that refers to \\Moo\Share. Every View node must be independently configured to have this same "G:\" drive mapped.

For more information, see "Configuring UNC Paths for Files."

Local Addresses to File Data Sources

Local addresses to file data can be used when each View node has it's own copy of the file. This architecture provides fault-tolerant operation, as each View node can independently run if the network goes down, but requires that any changes to the files be copied to all the View nodes. In the example below, three View nodes, each running a copy of the same application, can reference their own copies of the Recipe file. Since each application uses a local address for the file, each reference points to the local file.



To set up this configuration

Use a local address (for example, **C:\Directory**) to directly reference files. In a script to retrieve a Recipe file, type the following:

```
RecipeSelectRecipe("C:\Directory\Recipe.CSV", "review",
    "RecipeName");
```

where "**C:**\" is a local drive.

A copy of the file "Recipe.csv" must be stored on each machine in its local directory "**C:**\directory." If the file is modified, it must be copied again to each machine. Because of the difficulty in maintaining this configuration any file access should be "Read Only" and modification to the local file should not be permitted.

Configuring an InTouch Application for NAD

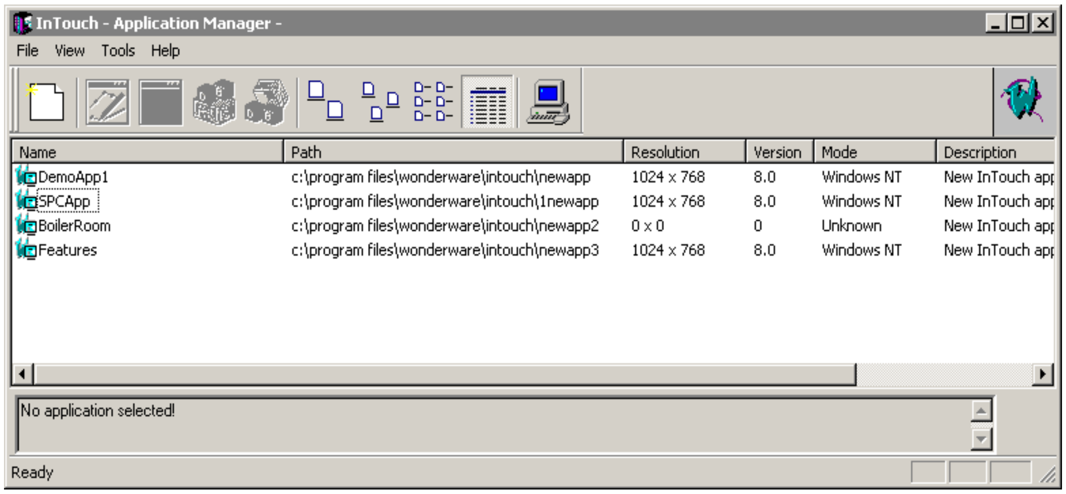
Network Application Development or NAD is an architecture that combines the best of the client-based and server-based architectures. NAD provides automatic notification of application changes and automatic distribution of the updated applications to View nodes

Note You cannot use the NAD features if you are using WindowViewer as an NT Service.

If configured to write historical data to the master application node's "Application Directory", all NAD nodes will attempt to write their historical data to the master application. To avoid this, on each NAD node, configure historical data to write to a local directory, **not** the master application node.

To configure an application for NAD

1. Start the InTouch program (intouch.exe). The **InTouch Application Manager** dialog box appears.


InTouch - Application Manager -

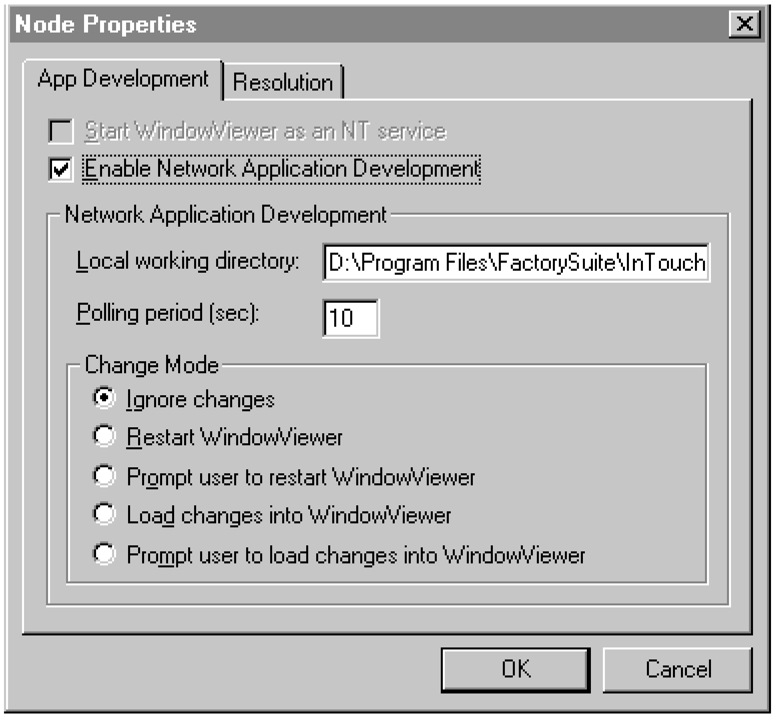
Name	Path	Resolution	Version	Mode	Description
DemoApp1	c:\program files\wonderware\intouch\newapp	1024 x 768	8.0	Windows NT	New InTouch app
SPCApp	c:\program files\wonderware\intouch\1newapp	1024 x 768	8.0	Windows NT	New InTouch app
BoilerRoom	c:\program files\wonderware\intouch\newapp2	0 x 0	0	Unknown	New InTouch app
Features	c:\program files\wonderware\intouch\newapp3	1024 x 768	8.0	Windows NT	New InTouch app

No application selected!

Ready

2. Click the **Node Properties** tool or on the **Tools** menu, click **Node Properties**. The **Node Properties** dialog box appears with the **App Development** property page active.

Tip To quickly access the **Node Properties** dialog box, right-click a blank area of the display window and then, click **Node Properties**.


Node Properties

App Development | Resolution

[ ] Start WindowViewer as an NT service
[✓] Enable Network Application Development

Network Application Development

Local working directory: D:\Program Files\FactorySuite\InTouch

Polling period (sec): 10

Change Mode

● Ignore changes
○ Restart WindowViewer
○ Prompt user to restart WindowViewer
○ Load changes into WindowViewer
○ Prompt user to load changes into WindowViewer

Note The **App Development** property sheet provides several options that allow you to specify how NAD will function. These settings are configured on each View node, NOT ON THE DEVELOPMENT NODE. This allows unique configurations for each View node.

Additionally, when you run WindowViewer as an NT service, it allows continuous operation of WindowViewer through operating system log-ins, log-outs, for example, operator shift changes. By selecting this option you also allow automatic start up of InTouch following power failure or when the machine is turned off and on. By doing this, you provide unmanned station startup of WindowViewer without compromising NT operating system security. However, you cannot use NAD features if you are using WindowViewer as an NT service.

For more information on running Windows NT services, see Appendix A, "Overview of the InTouch Windows NT Services."

3. Select **Enable Network Application Development** to enable NAD. The five **Change Mode** options become active. Only one change mode may be selected for each node.

Note The initial copying of the master application may take longer than subsequent updates.

4. In the **Local working directory** box, type the directory that you want WindowViewer to copy the master application to.

   If this is the development node, you can type a local directory path, such as **c:\InTouch\NAD**. You can also type a networked remote UNC path, such as **\\node\share\path**. This is convenient for file server based networks where most file storage is kept in a central location. If this is a client node (runtime only), it will likely use a local directory path. If you do not specify a directory, WindowViewer automatically creates a local subdirectory named "NAD" in the directory from which WindowViewer is launched.

   It is recommended that you use a local directory whenever possible to prevent network delays and failures from affecting the operation of WindowViewer.

Caution! Do not use a "root" directory or a UNC pathname that points to a root directory. The View node will recursively delete all files and subdirectories in the specified destination application directory before copying the master application directory. Therefore, never use the path of the master application directory or a UNC to the master application directory.

   This directory should be considered a temporary directory and no files should be saved to it except those copied by NAD itself.

   For more information on UNC paths, see "Configuring UNC Paths for Files."

5. Set the **Polling period (sec):** time. This value sets the interval (in seconds) at which the View node will poll the development node to check for changes.

Note Caution is advised when specifying this setting. If you set the value too small, WindowViewer will waste time checking for master application changes. This can interfere with WindowViewer servicing the running application.

6. In the **Change Mode** group, select the option for the action that you want WindowViewer to take when the master application changes.

Ignore changes	Causes the runtime (WindowViewer) node to ignore any changes made on the development node.
Restart Window Viewer	The runtime node copies the updated master application (if configured to do so), then restarts WindowViewer on the runtime node.
Prompt user to Restart WindowViewer	Causes an interactive message box to appear notifying the operator that the application has changed and asks the operator if he wants to restart WindowViewer.
Load Changes into WindowViewer	Causes changes made in the development node to dynamically be loaded into WindowViewer. This may affect performance for large updates.
Prompt user to load changes into WindowViewer	Causes an interactive message box to appear notifying the operator that the application has changed and asks the operator if he wants to load the changes dynamically into WindowViewer.

For more information, see "Customizing NAD Update Function."

7. Click **OK**.

Customizing NAD Update Function

In addition to the five update options described above, NAD provides the following tools that you can also use to customize the update behavior of an application:

Tool	Description
$ApplicationChanged	Provides an indication when a master application has changed. This tagname could be used to cause a message to appear telling the operator that the master application has changed. You can also use the **$ApplicationChanged** system tagname in a data change script to build a node update notification script. This script could include launching your own dialog boxes or closing down certain processes. Then, you could use the **ReloadWindowViewer()**to initiate the update process.
ReloadWindowViewer()	When this QuickScript function executes, it dynamically updates WindowViewer with the updated master application without any interruption in service. For example, you could use this function on a button to allow an operator to choose an appropriate time for updating the application. The function can also be used in a QuickScript to automatically update at a specified time or between shifts.
RestartWindowViewer()	When this QuickScript function executes, it automatically shuts down WindowViewer, copies the updated master application (if configured to do so), and then restarts WindowViewer. **Note** This function has been superceded by the **ReloadWindowViwer()** function described above.

Note To use these functions, the **Change Mode** option must be set to **Ignore Changes** in the **Node Properties** dialog box. Setting this option prevents the system from interfering with customized NAD functions.

For additional information on these functions, see your online *InTouch Reference Guide*.

Manually Notifying Clients of Application Changes

During application development, you can execute the **Notify Clients** command on the WindowMaker **Special** menu to automatically update InTouch client applications.

When you execute this command a flag is set to notify all remote View nodes that the master application has changed. These clients in turn, may automatically start an update process based on the parameters defined for each node.

Tip You can also use InTouch QuickScripting to notify clients.

For more information, see Chapter 12, "Real-time and Historical Trending."

The Application Copying Process

When the WindowViewer node copies an application, it makes every attempt to retain the attributes (read-only, system, hidden, and so on.) of the master application during the copy process. WindowViewer also copies all files and subdirectories of the master application. The copy process does not copy the following files: ***.WVW**, ***.DAT, *.LGH, *.IDX, *.LOG, *.LOK, *.FSM, *.STG, *.DBK, *.CBK, *.HBK, *.KBK, *.LBK, *.NBK, *.OBK, *.TBK, *.WBK, *.XBK, *.$$$, RETENTIV.X, RETENTIV.D, RETENTIV.A, RETENTIV.S, RETENTIV.H, RETENTIV.T, WM.INI, DB.INI, LINKDEFS.INI, TBOX.INI, GROUP.DEF,** and **ITOCX.CFG.**

Note WindowViewer will recursively delete all files and subdirectories in the destination application directory. This directory should be considered a temporary directory (no files should be placed into it).

Application Editing Locks

InTouch applications can only be edited by one developer at a time. To prevent multiple developers from trying to edit an application, WindowMaker locks an application during the edit session. If you try to load an application into WindowMaker that has a lock created, you will receive a message telling you that the application cannot be edited because it is being edited by another computer. The name of the node editing the application will also be stated in the message.

Note If WindowMaker is abnormally shut down with an application loaded, the appedit.lok file is automatically deleted. However, you can manually remove the lock by deleting the appedit.lok file from the application directory.

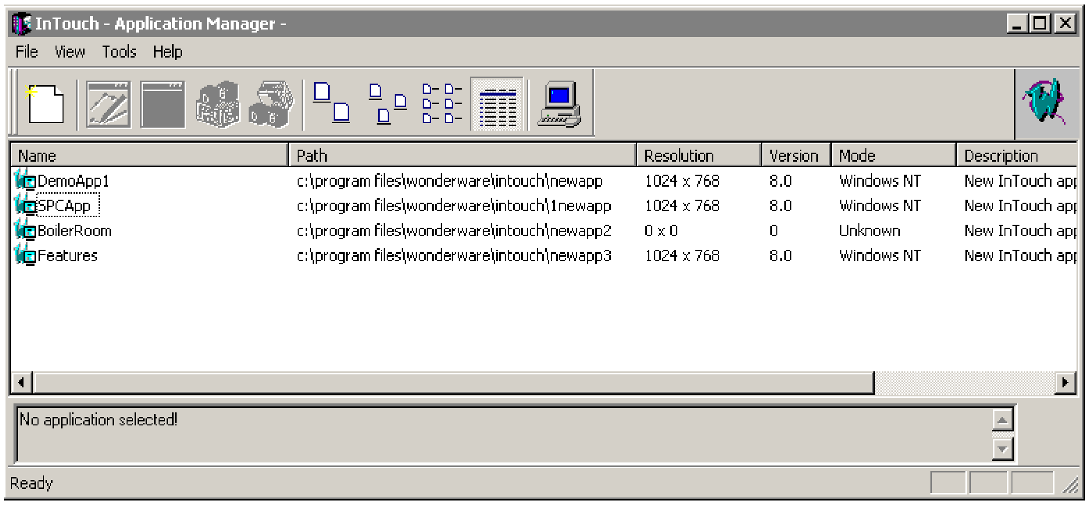
Dynamic Resolution Conversion (DRC)

Dynamic Resolution Conversion (DRC) works with other distributed features to provide independence from screen resolution restrictions. In a NAD architecture, an InTouch application is created and maintained on a development node, and then copied to several View nodes. DRC allows all of these nodes to view the application, even if they are running at different screen resolutions.

DRC enables each View node to scale the application to a number of user-defined options, including a custom resolution. This scaling takes place while WindowViewer compiles the application, and does not require WindowMaker. Since each View node can use a different DRC setting, each View node must have its own settings configured.

To configure an application for DRC

1. Start the InTouch program (intouch.exe). The **InTouch Application Manager** dialog box appears.



Name	Path	Resolution	Version	Mode	Description
DemoApp1	c:\program files\wonderware\intouch\newapp	1024 x 768	8.0	Windows NT	New InTouch apr
SPCApp	c:\program files\wonderware\intouch\1newapp	1024 x 768	8.0	Windows NT	New InTouch apr
BoilerRoom	c:\program files\wonderware\intouch\newapp2	0 x 0	0	Unknown	New InTouch apr
Features	c:\program files\wonderware\intouch\newapp3	1024 x 768	8.0	Windows NT	New InTouch apr

No application selected!

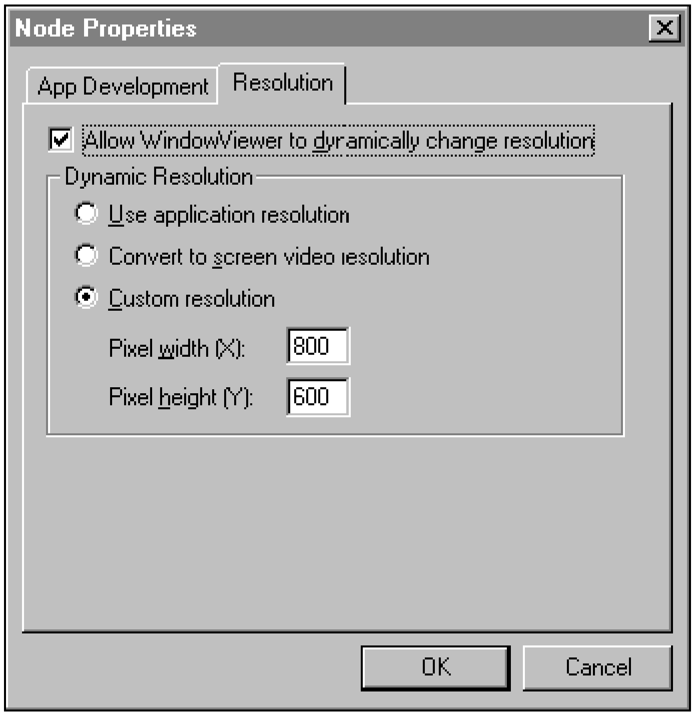
Ready

2. Click the **Node Properties** tool or on the **Tools** menu, click **Node Properties**. The **Node Properties** dialog box appears with the **App Development** property page active.

Tip To quickly access the **Node Properties** dialog box, right-click a blank area of the display window and then, click **Node Properties**.

Note When an application is selected in the Application Manager window, selecting the **Properties** command on the **File** menu displays the **Properties** dialog box for the selected application.

3. Click the **Resolution** tab.



[✓] Allow WindowViewer to dynamically change resolution

Dynamic Resolution

○ Use application resolution

○ Convert to screen video resolution

● Custom resolution

Pixel width (X): 800

Pixel height (Y): 600

4. Select **Allow WindowViewer to dynamically change resolution** if you want WindowViewer to locally scale the master application, based on the resolution option you select. (The three resolution options are described below.)

Note If you do not select this option, WindowViewer will only run the application if the node's screen resolution is identical to the screen resolution of the application development node. If the resolutions are different, WindowViewer prompts the operator to run WindowMaker to convert the application to the node's resolution. Use caution when doing this if you have set up a UNC path to the master application directory as this will only modify the original application.

5. Select **Use Application resolution** if you want WindowViewer to run the application at the resolution it was developed for and ignore the node's resolution. For example, if the application was developed at 640x480 and the node's resolution is 1024x768, WindowViewer will not dynamically scale the application. Instead, the application will be displayed at 640x480.

6. Select **Convert to screen video resolution** if you want WindowViewer to run the application at the node's resolution and ignore the resolution the application was developed at. For example, if the node is running at 640x480 and the application was developed at 1280x1024, WindowViewer will dynamically scale the application (smaller) to fit the node's 640x480 display. (This will more than likely be the most commonly used setting.)

7. Select **Custom Resolution** if you want WindowViewer to run the application at the resolution you specified in the **Pixel width (X)** and **Pixel height (Y)** (must be integer values) boxes. The application's resolution and the node's resolution are both ignored. For example, if **Pixel width (X)** and **Pixel height (Y)** are set to 512 and 384, respectively, the application will dynamically be scaled to fit in a 512x384-pixel area on the node's display.
8. Click **OK**.

Working with Multiple Monitor Systems

There are several advanced graphics adapter cards on the market today that allow you to have more than one VGA monitor connected to your system at a time. These monitors act in tandem, creating a virtual screen which can be very large. As an example, a popular system connects four 17" monitors stacked as a cube: two on the bottom and two on the top. Since each screen has a resolution of 800x600, the virtual screen created is 1600x1200 pixels.

Dynamic Resolution Conversion (DRC) makes it easy to support these multi-monitor systems. Simply select from the DRC resolution conversion options, and you can take full advantage of all the virtual display or just a portion of it.

If an application is scaled to run on an even number of the monitors, a problem exists when certain dialogs are displayed over the span of the monitors. One of these dialogs, the **Keypad**, can cause particular problems as certain keys may not be accessible. To solve the problem, InTouch provides several multi-monitor configuration options.

To configure the multi-monitor settings on a node

1. Using a suitable text editor, for example, Windows Notepad open the **WIN.INI** file located in your Windows directory.
2. Locate the **[InTouch]** section and add the following parameters:
3. [InTouch]

Parameter	Description
MultiScreen=1	turns on multi-screen mode
MultiScreenWidth=640	width in pixels of a single screen
MultiScreenHeight=480	height in pixels of a single screen

For example, if your computer's resolution is 2560 x 1024 split on two horizontal screens, enter the following:

[InTouch]

MultiScreen=1

MultiScreenWidth=1280

MultiScreenHeight=1024

Note The above entries affect the numeric keypad and the QWERTY keyboard. Other InTouch dialog boxes and option boxes are not affected.

Distributed Applications and Time Zones

InTouch provides services that ease the use of applications across multiple time-zones. These services are used by both the alarm and history systems to permit values to be viewed at the local time they occurred. For example, if an engineer in California is viewing an alarm that occurred in a manufacturing plant in Kansas at 10AM, that engineer would see the local California time that the alarm occurred; 8AM. The same is true if the engineer is viewing historical data from that plant.

The key to these services is the use of UCT (Universal Coordinated Time), also known as Greenwich Mean Time or GMT, as the time reference. Each computer is configured with both the local time and a UCT offset for the time zone where it resides. In the above example, the time zone for the computer in California is set at GMT eight hours, while the time zone for the computer in Kansas is set at GMT six hours.

InTouch uses these GMT offsets as the basis for retrieving all alarm and historical data. In the above example, when the InTouch application in California received the alarm from the application in Kansas, it also looked at the GMT offset of both computers to determine the local California time that the alarm occurred. Thus, a 10AM alarm in a UCT six time zone equals an 8AM alarm in a GMT eight time zone. To use this feature, the GMT offset must be configured for each computer.

For more information, see Chapter 8, "Historical Trending and Daylight Savings Time."

Distributed Alarms

InTouch supports a Distributed Alarm System. That allows the display of alarms and events generated by the local InTouch application and by alarm systems of other networked InTouch applications. Alarms can be acknowledged on the local InTouch node or from a remote node on the network.

For more information on setting up and configuring the distributed alarm system, see Chapter 9, "Alarms/Events."

Distributed History

InTouch provides a distributed history system that allows retrieval of historical data from any InTouch 5.6 (or later) application, even those across a network. This system extends the capabilities of the standard InTouch history by allowing remote retrieval of data from multiple historical databases simultaneously. These databases are referred to as history providers. Up to eight history providers can be displayed simultaneously, one for each historical trend chart pen.

Note History providers can be configured as native InTouch history or IndustrialSQL (InSQL) history providers.

For more information on setting up and configuring the distributed history system, see Chapter 12, "Real-time and Historical Trending."

C H A P T E R 6

Tagname Dictionary

The Tagname Dictionary (runtime database) is the heart of InTouch. At runtime, the database contains the current value of all of the items in the database. In order to create the runtime database, InTouch requires information about all of the variables being created. Each variable must be assigned a tagname and type. InTouch also requires additional information for some variable types. For instance, for I/O type tagnames, InTouch requires more information in order to be able to acquire the value and convert it for internal use. The Tagname Dictionary is the mechanism used to enter this information.

The two database utility programs, DBDump and DBLoad are also described in this chapter. DBDump allows you to export an InTouch application Tagname Dictionary as a text file that can be accessed from another package such as Microsoft Excel for modifying, storing, etc. DBLoad allows a database of tagnames created in another package such as Excel or a DBDump file from another InTouch application to be loaded into an existing InTouch application.

Contents

- Substituting Tagnames
- Converting Placeholder Tagnames
- Scaling I/O Tagnames
- Internal System $Tagnames
- Tagname Dotfields
- Tagname Dictionary Utilities

Tagname Dictionary Special Features

The Tagname Dictionary provides you with the following special features:

Feature	Description
Tag Browser	The Tag Browser is used for selecting tagnames and tagname **dotfields**, remote tagname references and SuperTag member tagnames from FactorySuite applications, or any other tag source that supports the Tagname Dictionary interface.
Tagname Cross Referencing	Tagname Cross Referencing allows you to cross-reference a tagname to the locations where it is used in your application including windows, scripts, SQL configuration, SPC triggers, and so on. You can print the cross-reference information or save it to a file.
SuperTags	InTouch supports a SuperTag structure that allows you to define composite tagname types. You can define SuperTag templates with up to 64 member tagnames and 2 nesting levels. Member tagnames behave exactly like normal tagnames and they support trending and alarming and all tagname **dotfields**.
References	Remote tagname references allow InTouch to access data from an I/O server without creating a tagname in the local Tagname Dictionary. Remote references allow you to import or export a window or QuickScript without requiring conversion of the tagnames from placeholders.
Extended Tagname Support	InTouch can support up to 61,405 tagnames in its Tagname Dictionary. (The number of tagnames that your system supports is determined by your software license.)

Tagname Types

When you are defining tagnames in the InTouch database, you must assign a specific type to each tagname according to its usage. For example, if the tagname is to read or write values coming to or from another Windows application such as an I/O Server, it must be an I/O type tagname. The following describes each InTouch tagname type and its usage.

Memory Type Tagnames

Memory type tagnames exist internally within your InTouch application. You use them to create system constants and simulations. You can also use them to create calculated variables that are accessed by other Windows programs. For example, you can define a memory tagname with the initial value of 3.1416 or, you could store recipes in groups of memory tagnames. In simulations, you can use memory tagnames to control the actions of a background QuickScript. For example, you could define a memory tagname "COUNT" that is changed in an action QuickScript to cause various animation effects to occur for the current STEP of a process. There are four Memory types:

Memory Discrete

Internal discrete tagname with a value of either 0 (False, Off) or 1 (True, On).

Memory Integer

A 32-bit signed integer value between -2,147,483,648 and 2,147,483,647.

Memory Real

Floating (decimal) point memory tagname. The floating point value may be between $-3.4e^{38}$ and $3.4e^{38}$. All floating point calculations are performed with 64-bit resolution, but the result is stored in 32-bit.

Memory Message

Text string tagname that can be up to 131 characters long.

I/O Type Tagnames

All tagnames that read or write their values to or from another Windows program are I/O type tagnames. This includes all inputs and outputs from programmable controllers, process computers and data from network nodes. I/O tagnames are accessed either through the Microsoft Dynamic Data Exchange (DDE) or Wonderware SuiteLink communication protocols.

When the value of a read/write I/O type tagname changes, it is immediately written to the remote application. The tagname may also be updated from the remote application whenever the item to which the tagname is linked changes in the remote application. By default, all I/O tagnames are set to Read/Write. However, you can restrict them to read only by selecting the Read Only option in the **Tagname Dictionary** dialog box. There are four I/O types:

I/O Discrete

Discrete input/output tagname with a value of either 0 (False, Off) or 1 (True, On).

I/O Integer

A 32-bit signed integer value between -2,147,483,648 and 2,147,483,647.

I/O Real

Floating (decimal) point tagname. The floating point value may be between $\underline{+}3.4e^{38}$. All floating point calculations are performed with 64-bit resolution, but the result is stored in 32-bit.

I/O Message

Text string input/output tagname that can be up to 131 characters long.

For more information on using I/O tagnames, see Chapter 13, "I/O Communications."

Miscellaneous Type Tagnames

There are several special tagname types that you can assign to tagnames to perform complex functions such as creating dynamic alarm displays, historical trends, monitoring or controlling the tagname each historical trend pen is plotting. There are also indirect tagname types that you can use to reassign a tagname to multiple sources. These special tagname types are described below.

Group Var

The **Group Var** type is used for a tagname with an assigned Alarm Group to create dynamic alarm displays, disk logs and print logs. You use **Group Var** type tagnames to create alarm windows or alarm logs that display all alarms associated with a specific group variable. You can also control the alarms that are displayed or logged by assigning a different Alarm Group to the **Group Var** tagname.

You can also use a **Group Var** type tagname to create buttons that the operator clicks to selectively display alarms for different areas of a plant in the same alarm window. All of the **dotfields** associated with Alarm Groups can be applied to **Group Var** tagnames.

Note The Group Var type is still supported in InTouch 8.0 but it serves no purpose. It is no longer used to dynamically change Alarm Groups, because the Standard Alarm System has been retired.

For more information on Alarms, see Chapter 9, "Alarms/Events."

Hist Trend

InTouch requires a **Hist Trend** type tagname when you create a historical trend. All of the **dotfields** associated with historical trends can be applied to **Hist Trend** tagnames.

Tag ID

This is a special type that is used with historical trend objects. You use **Tag ID** type tagnames to retrieve information about tagnames being plotted in a historical trend. In most cases, you would use **Tag ID** tagnames to display the name of the tagname assigned to a specific pen, or to change the tagname assigned to the pen.

You can process a statement in a QuickScript to assign a new tagname to any pen in any historical trend. For example, the following statement could be used in your QuickScript:

```
MyHistTrendTag.Pen1=MyLoggedTag.TagID;
```

When this QuickScript executed, Pen1 in the historical trend associated with the **Hist Trend** tagname "MyHistTrendTag," would begin trending the historically logged data for the "MyLoggedTag."

For more information on using **Tag ID** tagnames, see your *InTouch Reference Guide*.

Indirect Discrete, Indirect Analog, Indirect Message

Indirect type tagnames allow you to create one window and reassign the tagnames in that window to multiple source tagnames. For example, let's say you have fifty identical pumps that you want an operator to monitor for alarm conditions. Instead of creating fifty different windows (one for each pump) you can use indirect tagnames in one window that will call the source tagnames associated with individual pumps based on a QuickScript that points the indirect tagnames to the source tagnames of a pump that has entered an alarm state. You could also use a Touch Pushbutton QuickScript that allows the operator to manually select which pump to display. This method reduces both development time and application size.

Note InTouch also supports indirect SuperTags. All indirect SuperTags are displayed in the Tagname Dictionary **Tagname Types** dialog box. Indirect SuperTags can also be used in InTouch QuickScripts.

For more information, see "Creating Indirect SuperTags."

When you equate an indirect tagname to another source tagname, both the indirect tagname and the source tagname become exact duplicates of each other in every aspect including dotfields, scripts, and so on. If the value of the source tagname changes, the indirect tagname reflects the change. If the indirect tagname's value changes, the source tagname changes accordingly. You can define indirect tagname values in the database as retentive and reset them to take on their last tagname assignment on startup.

Indirect tagnames are assigned by using the **.Name** field. For example, if you created an indirect analog tagname called "Setpoint" and used the expression below in a QuickScript, "Setpoint1" would become the source for the value of "Setpoint":

```
Setpoint.Name = "Setpoint1";  or  Setpoint.Name =
    Setpoint1.Name;
```

You can also concatenate tagnames for use in indirect tagnames. For example, if you created a Data Change QuickScript that executes each time the value of the tagname "Number" changes, the indirect tagname, "Setpoint," would change accordingly:

```
Number=1;
```

```
Setpoint.Name = "Setpoint" + Text(Number, "#" );
```

When this QuickScript executes, the value of the analog tagname "Number" is converted to text and concatenated to the analog tagname "Setpoint," making "Setpoint.Name" equal to "Setpoint1." (Indirect analog type tagnames are used for both integer (whole numbers) and real (floating point) tagnames.

When you call a source tagname and the source tagname is enclosed in quotes:

```
Indirect.name= "mytag"
```

Or, the source tagname is defined by concatenating a text string with a variable:

```
Indirect.Name = "mytag" + Text(Number, "#");
```

In this case, the source tagname is not active for an indirect. The indirect will not take on the characteristics of the source tagname until the second time the QuickScript executes. We recommend you verify that all indirect tagnames reference active tagnames.

The source tagname(s) is activated when it is:

- displayed in an open application window
- used in a Window or Action QuickScript and the application window that is associated with the script is opened
- actively being used by a Real-time Trend
- used for alarming purposes
- event logged and Event Logging is enabled
- logged in an Historical Log and Historical Logging is enabled
- used in a Key, Condition, Data Change or Application QuickScript
- an Auto Collection tagname in InTouch SPCPro
- currently accessed by a client application (such as, Microsoft Excel) using DDE

- referenced by a local client using Point Access (PTACC.DLL)
- assigned to (referenced) by an Indirect Tag (a split-second delay will occur since the script has to request the correct value of the I/O point)

or

- an InTouch Access Name is configured for the DDE protocol to **Advise All Items**

SuperTags

InTouch SuperTags allow you to define composite tagname types. You can define SuperTags with up to 64 member tagnames and 2 nesting levels. Member tagnames behave exactly like normal tagnames. They support trending, alarming and all tagname **dotfields**.

For more information on SuperTags see, "Creating InTouch SuperTags."

Extended Tagname Support

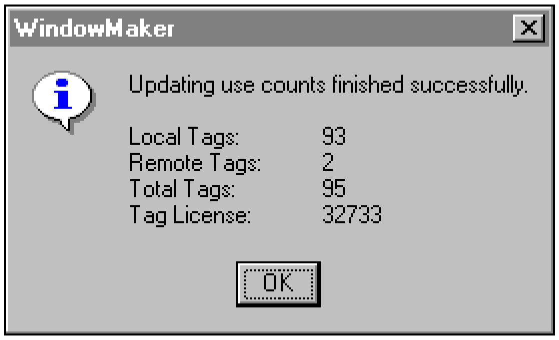
InTouch can support up to 61,405 tagnames in its Tagname Dictionary. The number of tagnames that your system supports is determined by your software license.

To determine the tagname support for your system

1. Close all your windows.
2. On the **Special** menu, click **Update Use Counts**.

Tip A message box appears telling you that updating use counts can take quite a while. You may at that time cancel the command or continue.

3. Click **Yes** to continue updating the use counts.
4. Once the system has completed updating the use counts, the following dialog box appears.



Updating use counts finished successfully.

```
Local Tags:      93
Remote Tags:     2
Total Tags:      95
Tag License:     32733
```

5. The **Tag License** line will display the number of tagnames supported by your license.

Note Any tagname used in SPCPro as a Collection Tag, Scooter Tag or Trigger Tag will not be included in the count.

6. Click **OK**

Defining a New Tagname

Tagnames can be up to 32 characters long and must begin with an alpha character (**A-Z** or **a-z**). The remaining characters can be A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &.

Note Backslash character (\) is valid for SuperTag only.

Tagnames are also auto-indexed. For example, if you enter and save tagname R4001, and then click **New**, the tagname will automatically be indexed to R4002. If an tagname contains a character separating numbers, it is auto-indexed by the first whole number InTouch finds. For example, N7-0 would be indexed as N7-1. Positive changes only are permitted. For example, R4002 to R4003, R4003 to R4004 and so on.

You need to be careful when you use dashes (-) in tagnames. They are valid for use in tagnames but, they are also used as the negation and subtraction "operator" in expressions or logic. Therefore, some ambiguity arises.

For example, if you use `A=B-C` in an expression, do you mean that `A=B minus C` or do you mean that you simply want to assign a tagname named **B-C** to a tagname named **A**? InTouch will assume the latter. You can prevent this by separating the tagnames from the operators with blank space(s). For example, `A = B - C`.

Consider this example: `X-101=FT-101*SP-101`

Can you see where **FT-101** is being multiplied by **SP-101** and assigned to **X-101** due to the fact that no spaces were used?

For more information on the Tag Browser, see "The Tag Browser."

The options at the top of the **Tagname Dictionary** dialog box are used to display the various tagname detail level dialog boxes as follows:

Dialog Box	Description
Main	Displays the main tagname dictionary dialog box. In the case of SuperTags, **Main** shows only the parent or root tagname. Any changes made to the parent or root tagname can overwrite Member tagname information. After making a change, click **Save**. A message box appears asking if you want to overwrite member tagnames with the root tagname changes.
Details	Displays the respective details level dialog box for the tagname type selected.
Alarms	Displays the respective alarms configuration dialog box for the tagname type selected.

Dialog Box	Description
Details & Alarms	Displays the both respective details and alarm configuration dialog boxes for the tagname type selected.
Members	Displays the member details dialog box for a SuperTag type tagname.

Tip If you right-click any of the text entry boxes in any of the Tagname Dictionary dialog boxes, a menu appears displaying the commands that you can apply to the selected text.

Select Tag

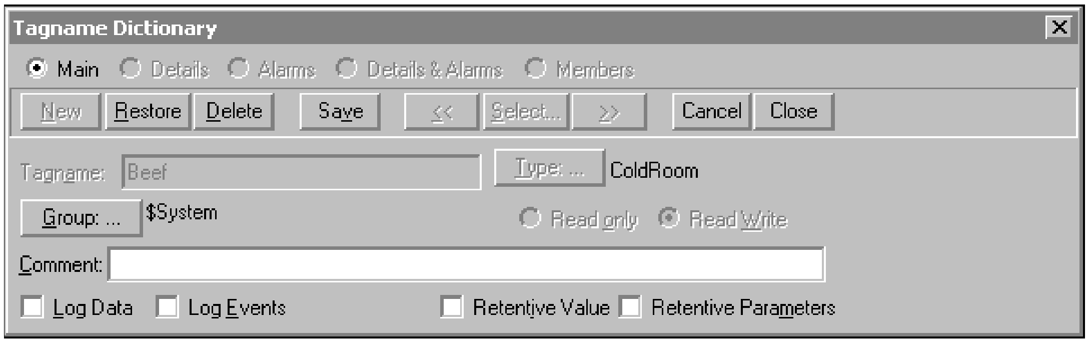
The first time you access the Tagname Dictionary, the definition for the internal system tagname **$AccessLevel** is displayed. Once you define tagnames in the Tagname Dictionary, when you access it again, the last edited tagname's definition is displayed.

Click << or >> to browse through the tagname definitions currently stored in your Tagname Dictionary. (The browse buttons will be inactive when there are no previous or next tagnames to display.)

Click **Select** to quickly locate a specific tagname definition. The **Select Tag** dialog box appears in the selection mode.

To define a new tagname

1. On the **Special** menu, click **Tagname Dictionary** or, in the Application Explorer, double-click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears.



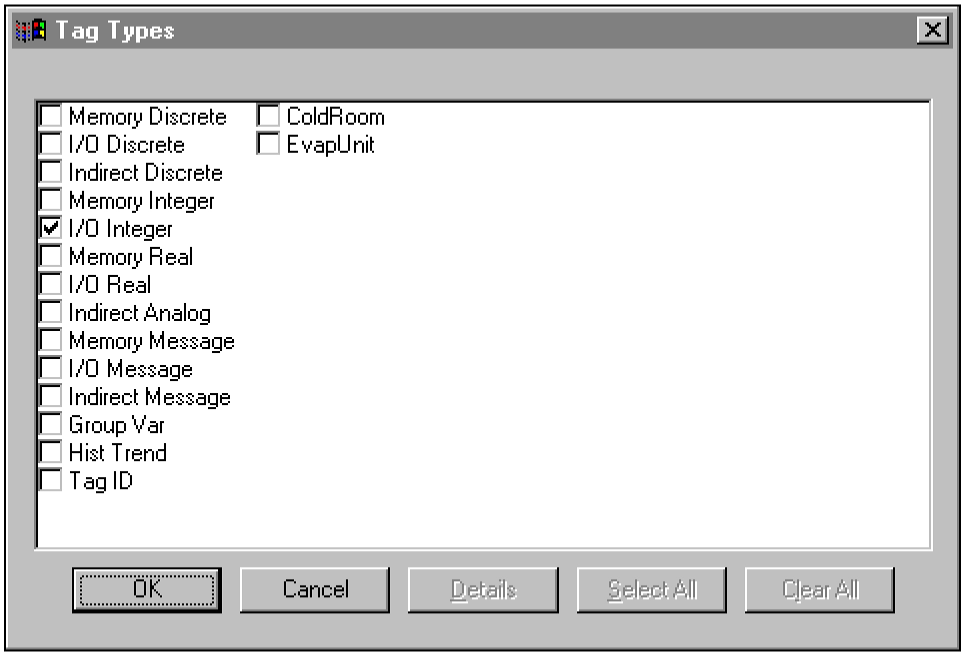
Tagname: Beef
Type: ColdRoom
Group: $System
Comment:

2. Click **New**. (The **Tagname** box clears.)
3. In the **Tagname** box, type the name you want to use for the new tagname.

Tip Tagnames can be up to 32 characters long and must begin with an alpha character (**A-Z** or **a-z**). The remaining characters can be A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &.

You cannot use the word **RetVal** for a tagname. This is a reserved word. If you attempt to use this word, and then try to edit a QuickFunction you will receive the error message "A variable cannot have that name. Tag exists."

4. Click **Type**. The **Tag Types** dialog box appears.



- [ ] Memory Discrete
- [ ] I/O Discrete
- [ ] Indirect Discrete
- [ ] Memory Integer
- [x] I/O Integer
- [ ] Memory Real
- [ ] I/O Real
- [ ] Indirect Analog
- [ ] Memory Message
- [ ] I/O Message
- [ ] Indirect Message
- [ ] Group Var
- [ ] Hist Trend
- [ ] Tag ID
- [ ] ColdRoom
- [ ] EvapUnit

5. Select the type for the tagname, then click **OK**. The respective details dialog box for the selected type appears. (The detail dialog boxes are described later in this chapter.)

Tip The names of any SuperTag created in the TemplateMaker will also appear in this dialog box and can be selected as the tag type. For example, ColdRoom and EvapUnit above. For SuperTags not created using the TemplateMaker, the name "SuperTag" appears. For example, SuperTags created in an animation link tagname or expression input box, a QuickScript, or created in an external file and then loaded the DBLoad utility.

For more information on tagname types, see "Tagname Types."

For more information on creating InTouch SuperTags see, "Creating InTouch SuperTags."

Note If a tagname is currently linked to an object or used in a QuickScript, its type can only be changed when WindowViewer is not running.

6. Click **Group** to assign the tagname to a specific Alarm Group. The **Alarm Groups** dialog box appears. Select the Alarm Group that you want to assign to the tagname, and then click **Done**.

   Note If you do not assign the tagname to a specific Alarm Group, by default, InTouch will assign it to the root group, **$System**.

   Once you create a tagname and assign it to an Alarm Group, if you do not close the dialog box, all subsequent tagnames that you define will be assigned to the same Alarm Group, unless you change it.

   For more information on defining Alarm Groups, see Chapter 9, "Alarms/Events."

7. For I/O type tagnames, select **Read Only** to restrict the tagname to read only capabilities in runtime.
8. For I/O type tagnames, select **Read Write** to grant the tagname read and write capabilities in runtime.
9. In the **Comment** box, type any miscellaneous comment you want the system to store regarding your tagname (up to 50 characters).

   Tip The first time you access the **Tagname Dictionary** dialog box, the default comment for the internal system tagname **$AccessLevel** will be displayed in the **Comment** box. You should delete this comment to prevent it from being associated with any tagnames that you define. To delete the comment, select it and press the DEL key.

   For more information on using the Distributed Alarm System, see Chapter 9, "Alarms/Events."

10. Select **Log Data** if you want the tagname to be logged to the historical log file during runtime whenever its engineering unit value changes more than the **Log Deadband** value you specify or, by default once an hour, regardless of change.

    Note In order for your tagnames to actually be logged, you must enable historical logging through the **Configure Historical Logging** command on the **Special** menu.

    If you decide to later clear this option so the tagname will not be logged, the data previously logged for the tagname will not be accessible. Also, if you make changes in WindowMaker to logging while WindowViewer is running, they will not take affect until WindowViewer is restarted.

11. Select **Log Events** if you want to log all data value changes to the tagname that are initiated by the operator, I/O, a QuickScript or by the system.

Tip When you define a tagname to do event monitoring, an event message is logged to the alarm system each time the tagname's value changes. The event message logs how the value changed. For example, whether the operator, I/O, QuickScripts or the system initiated the change.

When you select **Log Events**, the **Priority** field becomes active. The value you type for the **Priority** determines the event priority level for the tagname. Valid entries in this field are 1 to 999, where 1 is the highest and 999 is the lowest priority.

For more information on events and priorities, see Chapter 9, "Alarms/Events."

12. Select **Retentive Value** if you want to retain the current value of the tagname whenever WindowViewer is exited. This value will be used as the initial value for the tagname whenever WindowViewer is restarted.

Note Retentive values are not written to I/O devices when WindowViewer is restarted. I/O values update the first time the I/O server scans the device.

Tip Retentive values cannot be selected or cleared for new or existing tagnames if WindowViewer is running. When you select this option, the initial value of the tagname will constantly be updated to reflect the current value of the tagname. When WindowViewer is exited, the initial value is set based on the last retained value. If this option is later cleared, the initial value of the tagname will be set to the last retained value.

13. Select **Retentive Parameters** if you want to retain any changes the operator makes to the value of any alarm limit fields for the tagname. This value will be used as the initial value for the alarms when WindowViewer is restarted.

Note Since changes are logged immediately, we strongly recommended that you only select the above two retentive options for values that do not change often.

14. Define the details for the type of tagname and then, click **Close**.

Defining Tagname Details

The initially displayed **Tagname Dictionary** dialog box is used to input basic tagname information. Many points, especially inputs and outputs, require greater detail to be properly handled. For each type of tagname specified, a specific details dialog box exists that you use to define the details for the tagname type.

Most of the tagname types have their own specific detail level dialog boxes and alarm condition dialog boxes. By default, when you select the type for your tagname, its respective details level dialog box appears

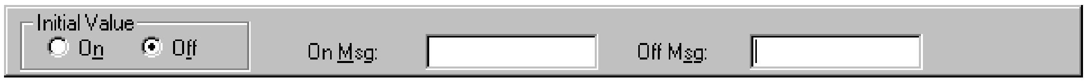
Once you have completed defining the basic tagname, you will need to define the details for the tagname and, if required, the alarm conditions. The steps that you need to follow to define the details for each tagname type are described in the following sections.

Defining Memory Discrete Tagname Details

Memory discrete type tagnames exist internally within your InTouch application. You define a **Memory Discrete** type tagname when you need an internal tagname with a value of either 0 (False, Off) or 1 (True, On).

To define the details for a memory discrete tagname

1. When you select **Memory Discrete** as the type for your tagname, the following details dialog box appears.



Tip If it does not appear, click **Details** at the top of the **Tagname Dictionary** dialog box.

2. Click the **Initial Value** that you want stored in the tagname when the runtime database is first loaded.
3. If you define a discrete alarm state for this tagname that is "on" when the tagname's value is equal to 1 (On, True), type the message in the **On Msg** box that you want to be displayed in your alarm window's value/limit field.
4. If you define a discrete alarm state for this tagname that is "on" when the tagname's value is equal to 0 (Off, False), type the message in the **Off Msg** box that you want to be displayed in your alarm window's value/limit field.
5. If you want to define alarm conditions for the tagname, click either **Alarms** or **Details & Alarms** at the top of the **Tagname Dictionary** dialog to display the respective alarm conditions dialog box for the type of tagname you are defining.

   For more information on alarms, see Chapter 9, "Alarms/Events."
6. Once you have completed defining your tagname, click **Close** to save your tagname definition and close the tagname dialog boxes.

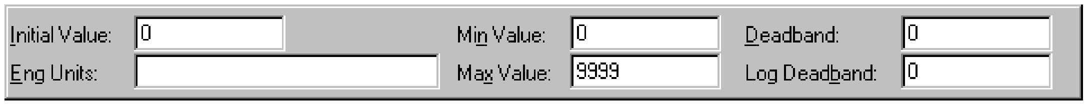
Defining Memory Analog Tagname Details

Memory analog type tagnames exist internally within your InTouch application. There are two memory analog types: **Memory Integer** and **Memory Real**. You define a **Memory Integer** type tagname when you need an internal tagname with a 32-bit signed integer value between -2,147,483,648 and 2,147,483,647.

You define a **Memory Real** type tagname when you need an internal tagname with a floating point value between -3.4e^{38} and 3.4e^{38}. (All floating point calculations are performed with 64-bit resolution, but the result is stored in 32-bit.)

To define the details for a memory analog tagname

1. When you select **Memory Integer** or **Memory Real** as the type for your tagname, the following details dialog box appears.



Initial Value:	0	Min Value:	0	Deadband:	0
Eng Units:		Max Value:	9999	Log Deadband:	0

Tip If it does not appear, click **Details** at the top of the **Tagname Dictionary** dialog box.

2. In the **Initial Value** box, type the value you want stored in the tagname when the runtime database is first loaded.
3. In the **Min Value** box, type the minimum value for the tagname. This is used for Historical Trend charts, I/O and the **.Min EU** tagname dotfields.
4. In the **Max Value** box, type the maximum value for the tagname. This is used for Historical Trend charts, I/O and the **.Max EU** tagname dotfields.
5. In the **Eng Units** box, enter the label you want to use for the tagname's engineering units.
6. In the **Deadband** box, type the amount the tagname's engineering units must change before the database is updated.
7. In the **Log Deadband** box, type the amount the tagname's engineering units must change before the tagname is logged to the historical log file. The default of zero means every change is logged.

Note You must select **Log Data** for the tagname if you want it to be logged to disk when its engineering units change more than the **Log Deadband** value.

If you change the **Log Deadband** value while WindowViewer is running, your changes will not take effect until historical logging has been stopped and restarted.

For more information on historical logging, see Chapter 12, "Real-time and Historical Trending."

8. If you want to define alarm conditions for the tagname, click either **Alarms** or **Details & Alarms** at the top of the **Tagname Dictionary** dialog to display the respective alarm conditions dialog box for the tagname type you are defining.

   For more information on alarm conditions, see "Defining Tagname Alarm Conditions."

9. Once you have completed defining your tagname, click **Close** to save your tagname definition and close the tagname dialog boxes.

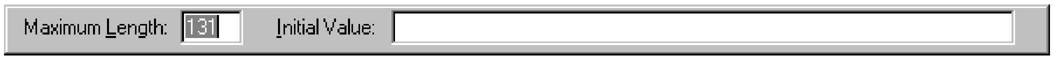
Defining Memory Message Tagname Details

Memory message type tagnames exist internally within your InTouch application. You define a **Memory Message** type tagname when you need an internal text string tagname that can be up to 131 characters long.

To define the details for a memory message tagname

1. When you select **Memory Message** as the type for your tagname, the following details dialog box appears.

Note If it does not appear, click **Details** at the top of the **Tagname Dictionary** dialog box.


Maximum Length: 131 Initial Value:

2. In the **Maximum Length** box, type the maximum number of characters to be allowed in the tagname's message. (InTouch allows a maximum of 131, which is displayed as the default.)
3. In the **Initial Value** box, type the text string that you want displayed for the tagname when WindowViewer is initially started.
4. Once you have completed defining your tagname, click **Close** to save your tagname definition and close the tagname dialog boxes.

Defining I/O Discrete Tagname Details

All tagnames that read or write their values to or from another Windows program are I/O type tagnames. This includes all inputs and outputs from programmable controllers, process computers, other Windows programs and data from network nodes.

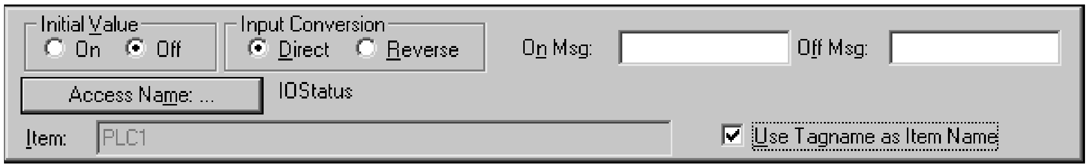
You define an **I/O Discrete** type tagname when you need an I/O tagname with a value of either 0 (False, Off) or 1 (True, On).

To define the details for a I/O discrete tagname

1. When you select **I/O Discrete** as the type for your tagname, the following details dialog box appears.

Tip If it does not appear, click **Details** at the top of the **Tagname Dictionary** dialog box.

Tip If it does not appear, click **Details** at the top of the **Tagname Dictionary** dialog box.


Initial Value: On [✓ Off]
Input Conversion: [✓ Direct] Reverse
On Msg: Off Msg:
Access Name: ... IOStatus
Item: PLC1
[✓ Use Tagname as Item Name]

2. Click the **Initial Value** that you want stored in the tagname when the runtime database is first loaded. (**Off** equals **0**, **On** equals **1**.) This value is not written to the I/O device.
3. Click the **Input Conversion** that you want applied to the value when the runtime database is updated:

Input Conversion	Description
Direct	The I/O input value is read unchanged directly from the server program.
Reverse	The I/O input value is reversed when read from the server program. For example, if the I/O input value in the server program is 0, InTouch will automatically reverse it, save it and display a 1.

4. If you define a discrete alarm state for this tagname that is "on" when the tagname's value is equal to 1 (On, True), type the message in the **On Msg** box that you want to be displayed in your alarm window's value/limit field.
5. If you define a discrete alarm state for this tagname that is "on" when the tagname's value is equal to 0 (Off, False), type the message in the **Off Msg** box that you want to be displayed in your alarm window's value/limit field.
6. Click **Access Name** to define or select the Access Name that you want to assign to this tagname. (If an Access Name already appears to the right of this button, and you do not define or select a different one, it will be assigned to the tagname.

   For more information on Access Names, see Chapter 13, "I/O Communications."
7. In the **Item** box, type the valid item name for the data point in the server program that the tagname will read/write its value from/to. For example, if you want to read a value from a register in a PLC, enter the valid identification for that register as the item name.

   Tip Item names are auto-indexed. For example, if you enter and save item name R4001, then click **New** (to define a new tagname), the item name will automatically be indexed to R4002. If an item name contains a character separating numbers, it is auto-indexed by the first whole number InTouch finds. For example, N7-0 would be indexed as N7-1. Positive changes only are permitted. For example, R4002 to R4003, R4003 to R4004 and so on.
8. Select the **Use Tagname as Item Name** option if you want to use the tagname for the item name.
9. If you want to define alarm conditions for the tagname, click either **Alarms** or **Details & Alarms** at the top of the **Tagname Dictionary** dialog to display the respective alarm conditions dialog box for the tagname type you are defining.

   For more information on alarm conditions, see Defining Tagname Alarm Conditions."

10. Once you have completed defining your tagname, click **Close** to save your tagname definition and close the tagname dialog boxes.

Defining I/O Analog Tagname Details

All tagnames that read or write their values to or from another Windows program are I/O type tagnames. This includes all inputs and outputs from programmable controllers, process computers, other Windows programs and data from network nodes. There are two memory analog types: **I/O Integer** and **I/O Real**.

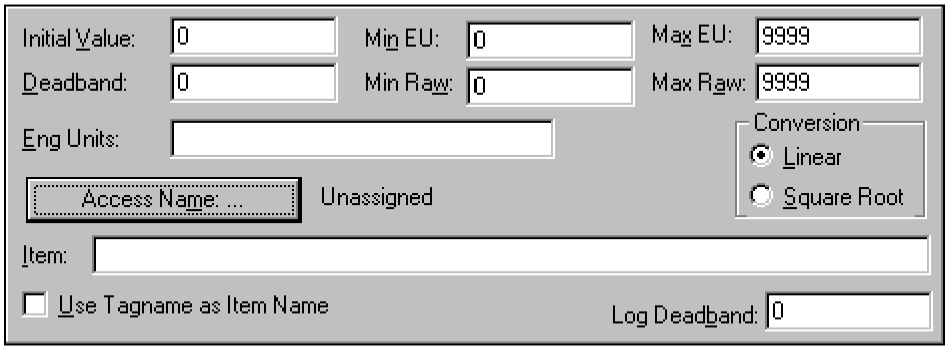
You define an **I/O Integer** type tagname when you need an I/O tagname with a 32-bit signed integer value between -2,147,483,648 and 2,147,483,647.

You define an **I/O Real** type tagname when you need an I/O tagname with a floating point value between -3.4e^{38} and 3.4e^{38}. (All floating point calculations are performed with 64-bit resolution, but the result is stored in 32-bit.)

To define the details for an I/O analog tagname

1. When you select **I/O Integer** or **I/O Real** as the type for your tagname, the following details dialog box appears.

Tip If it does not appear, click **Details** at the top of the **Tagname Dictionary** dialog box.


Initial Value: 0
Min EU: 0
Max EU: 9999
Deadband: 0
Min Raw: 0
Max Raw: 9999
Eng Units:
Conversion: [✓ Linear] Square Root
Access Name: ... Unassigned
Item:
Use Tagname as Item Name
Log Deadband: 0

2. In the **Initial Value** box, type the value you want stored in the tagname when the runtime database is first loaded. This value is not written to the I/O device.

3. In the **Deadband** box, type the amount the engineering units for the tagname can change before the database is updated.

4. In the **Min EU** box, type the engineering units value for the tagname when the minimum raw count value is received.

5. In the **Min Raw** box, type the minimum value of the low clamp on the raw I/O integer values.

6. In the **Max EU** box, type the engineering units value for the tagname when the maximum raw count value is received.

7. In the **Max Raw** box, type the maximum value of the high clamp on the raw I/O integer values.

Tip You can use the **Min EU**, **Min Raw**, **Max EU** and **Max Raw** values to scale your I/O tagnames.

For more information on scaling tagnames, see "Scaling I/O Tagnames."

8. In the **Eng Units** box, enter the label you want to use for your tagname's engineering units.
9. Select the type of **Conversion** that you want the database to use to scale the raw counts when calculating the engineering units as follows:

   If you select **Linear**, the result is calculated using linear interpolation between the end points. The algorithm for linear scaling of input is:

   ```
   EUValue = (RawValue - MinRaw) * ((MaxEU - MinEU) /
       (MaxRaw - MinRaw)) + MinEU
   ```

   The algorithm for linear scaling of output is:

   ```
   RawValue = (EUValue - MinEU) * ((MaxRaw - MinRaw) /
       (MaxEU - MinEU)) + MinRaw
   ```

   If you select **Square Root**, the raw counts values are used for interpolation. This is useful for scaling inputs from nonlinear devices such as pressure transducers. The algorithm for square root scaling of input is:

   ```
   EUValue = sqrt(RawValue - MinRaw) * ((MaxEU - MinEU) /
       sqrt(MaxRaw - MinRaw)) + MinEU
   ```

   The algorithm for square root scaling of output is:

   ```
   RawValue = square((EUValue - MinEU) * (sqrt(MaxRaw –
       MinRaw) / (MaxEU -MinEU))) + MinRaw
   ```

10. Click **Access Name** to define or select the Access Name that you want to assign to this tagname. (If an Access Name already appears to the right of this button, and you do not define or select a different one, it will be assigned to the tagname.)

    For more information on Access Names, see Chapter 13, "I/O Communications."

11. In the **Item** box, type the valid item name for the data point in the server program that the tagname will read/write its value to/from. For example, if you want to read a value from a register in a PLC, enter the valid identification for that register as the item name.

Tip Item names are auto-indexed. For example, if you enter and save item name R4001, then click **New** (to define a new tagname), the item name will automatically be indexed to R4002. If an item name contains a character separating numbers, it is auto-indexed by the first whole number InTouch finds. For example, N7-0 would be indexed as N7-1. Positive changes only are permitted. For example, R4002 to R4003, R4003 to R4004 and so on.

12. Select the **Use Tagname as Item Name** option if you want to use the tagname for the item name.

13. If you want to define alarm conditions for the tagname, click either **Alarms** or **Details & Alarms** at the top of the **Tagname Dictionary** dialog to display the respective alarm conditions dialog box for the tagname type you are defining.

    For more information on alarm conditions, see "Defining Tagname Alarm Conditions."

14. In the **Log Deadband** box, type the amount the tagname's engineering units must change before the tagname is logged to the historical log file.

    Note You must select **Log Data** for the tagname if you want the tagname to be logged to disk when its Engineering units change more than the **Log Deadband** value.

    If you change the **Log Deadband** value while WindowViewer is running, your changes will not take effect until historical logging has been stopped and restarted.

    For more information on historical logging, see Chapter 12, "Real-time and Historical Trending."

15. Once you have completed defining your tagname, click **Close** to save your tagname definition and close the tagname dialog boxes.

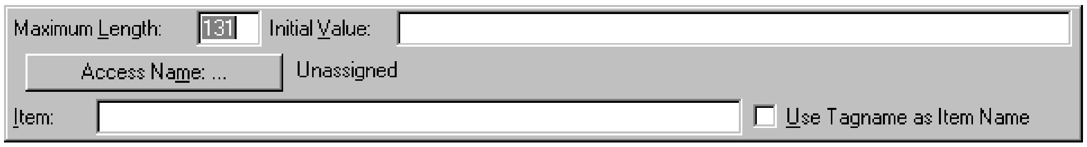
Defining I/O Message Tagname Details

All tagnames that read or write their values to or from another Windows program are I/O type tagnames. This includes all inputs and outputs from programmable controllers, process computers, other Windows programs and data from network nodes. You define an **I/O Message** type tagname when you need to gather text strings from an I/O device. I/O message tagnames are limited to 131 characters in length.

To define the details for a I/O message tagname

1. When you select **I/O Message** as the type for your tagname, the following details dialog box appears.

   Tip If it does not appear, click **Details** at the top of the **Tagname Dictionary** dialog box.


Maximum Length: 131
Initial Value:
Unassigned
Item:
☐ Use Tagname as Item Name

2. In the **Maximum Length** box, type the maximum number of characters to be allowed in the tagname's message. (InTouch allows a maximum of 131, which is displayed as the default.)

3. In the **Initial Value** box, type the text string that you want displayed for the tagname when WindowViewer is initially started. This value is not written to the I/O device.

4. Click **Access Name** to define or select the Access Name that you want to assign to this tagname. (If an Access Name already appears to the right of this button, and you do not define or select a different one, it will be assigned to the tagname.)

   For more information on Access Names, see Chapter 13, "I/O Communications."

5. In the **Item** box, type the valid item name for the data point in the server program that the tagname will read/write its value to/from. For example, if you want to read a value from a register in a PLC, enter the valid identification for that register as the item name.

   Tip Item names are auto-indexed. For example, if you enter and save item name R4001, then click **New** (to define a new tagname), the item name will automatically be indexed to R4002. If an item name contains a character separating numbers, it is auto-indexed by the first whole number InTouch finds. For example, N7-0 would be indexed as N7-1. Positive changes only are permitted. For example, R4002 to R4003, R4003 to R4004 and so on.

6. Select the **Use Tagname as Item Name** option if you want to use the tagname for the item name.

7. Once you have completed defining your tagname, click **Close** to save your tagname definition and close the tagname dialog boxes.

Defining SuperTag Member Tagname Details

Member tagnames are defined in SuperTag Templates. Member tagnames behave exactly like normal tagnames and can be of type Discrete, Integer, Real, Message or, another SuperTag. Like normal InTouch tagnames, member tagnames support trending, alarming and all tagname **dotfields**.

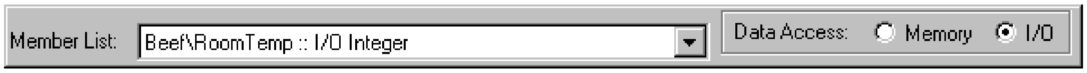
For more information on member tagnames see, "Creating InTouch SuperTags."

When you define a tagname and select a SuperTag template for its tagname type, by default, all member tagnames defined in SuperTag templates will be set to the data access type "Memory." If this is the type that you need for them to be, no special configuration is necessary. However, if you need any of the member tagnames in the SuperTag template to be defined as I/O types, you must do some additional configuring.

To define I/O SuperTag member tagnames

1. When you select a SuperTag template as the type for your tagname, the following details dialog box appears.

Tip If it does not appear, click **Members** at the top of the **Tagname Dictionary** dialog box.


Member List: Beef\RoomTemp :: I/O Integer

Data Access: Memory [✓ I/O]

Notice that the new tagname that you typed in the **Tagname** box becomes the "parent" for all the member tagnames in the **Member List**.

2. Click the **Member List** arrow, and then select the member tagname in the list that you want to define as an I/O data access type
3. In the **Data Access** group, select **I/O**. The respective I/O details dialog box for the member tagname's type (Discrete, Analog (Real or Integer) or Message) appears.
4. Enter the required I/O details just as you would for a normal InTouch I/O type tagname.
5. To save your changes, select another member tagname in the list and configure it, or click **Close**.

Defining Tagname Alarm Conditions

You can define alarm conditions for tagnames at the same time that you define the tagname. There are two types of alarm detail dialog boxes. One for discrete type tagnames and one for analog (integer or real) type tagnames.

Alarm Inhibitor Tagnames

You can optionally assign to each alarm (or alarm sub-state) a tagname that can inhibit the alarm. When the tagname is TRUE (non-zero or non-NULL), the alarm is actively inhibited. Likewise, when the tagname is FALSE (zero or NULL), the alarm is not inhibited. You assign an **Alarm Inhibitor Tag** through the Tagname Dictionary in WindowMaker. **Alarm Inhibitor Tag**s cannot be changed during runtime. However, the value of the **Alarm Inhibitor Tag** can be changed during runtime.

An alarm can be actively inhibited or inhibited or both. If it is either inhibited or actively inhibited, it is "turned off." Meaning that if you attempt to enable an alarm that is actively inhibited it will not "turn on."

Note **Alarm Inhibitor Tags** are regular tagnames that are included in use counts and license limitations.

There are read-only tagname dot fields that you can also use to return a string containing the name of the **Alarm Inhibitor Tag**. These dot fields are:

- AlarmDscInhibitor
- AlarmLoLoInhibitor
- AlarmLoInhibitor
- AlarmHiHiInhibitor

- AlarmHiInhibitor
- AlarmMajDevInhibitor
- AlarmMinDevInhibitor
- AlarmRocInhibitor

These fields return the name of a tagname. Therefore, you can use the name in an indirect tagname reference in an InTouch QuickScript to find out the current value of the **Alarm Inhibitor Tag,** or to change the value of the **Alarm Inhibitor Tag**. By doing this, you can force groups of alarms to be enabled or actively inhibited during runtime.

For more information on these alarm dotfields, see your online *InTouch Reference Guide*.

Inhibiting an alarm is a two-stage process:

1. Assignment of the inhibitor tagname.
2. A change in the state of the inhibitor tagname from FALSE to TRUE or vice-versa.

As with disablement, for an alarm that has sub-states, each sub-state can be inhibited individually. Each sub-state can be inhibited by a different tagname. As with disablement, an alarm that is inhibited (and for which the tagname is TRUE) is not waiting for an ***acknowledgment***. If the alarm has sub-states, it can only be waiting for an ***acknowledgment*** on sub-states that are still available.

Whenever the transition causes an alarm to change from being **actively inhibited**, the checking logic is executed to determine whether InTouch should put the item in the ***alarmed*** state.

If an alarm becomes **actively inhibited** while the item is in an ***alarmed*** state, the item must be forced to a different (valid) state. What that state should be depends upon which states are available and whether they have also been disabled or actively inhibited. This activity is handled by InTouch according to the type of alarm, limit values, and so on.

If an alarm (or an alarm sub-state) becomes **actively inhibited** while waiting for an ***acknowledgment***, the item must be forced to a different (valid) state. As with whether the item is alarmed, InTouch must determine what this state should be.

For more information on alarms, see Chapter 9, "Alarms/Events."

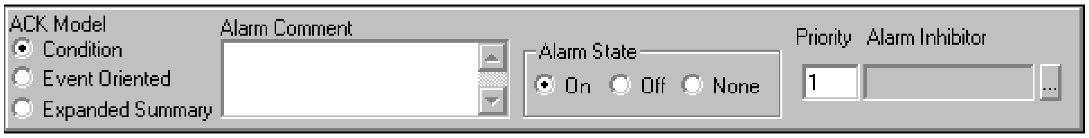
Defining Discrete Tagname Alarm Conditions

A discrete alarm corresponds to a discrete tagname. You can configure whether the ***alarmed*** state corresponds to the TRUE (On, Yes, 1) state or the FALSE (Off, No, 0) state of the discrete tagname, and the associated ***priority*** of the alarm.

To define alarm conditions for a discrete tagname

1. On the **Special** menu, click **Tagname Dictionary** or, in the Application Explorer, double-click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears.

2. Click either **Alarms** or **Details & Alarms** at the top of the **Tagname Dictionary** dialog box to display the discrete alarm details dialog box:



ACK Model: [● Condition] ○ Event Oriented ○ Expanded Summary
Alarm Comment:
Alarm State: [● On] ○ Off ○ None
Priority: 1
Alarm Inhibitor:

3. Select the **ACK Model** option that you want to use:
   - **Condition:** An acknowledgment counts against all transitions into the alarmed state or a sub-state up to the time of the acknowledgement. (This is the default and functions the same as previous versions.)
   - **Event Oriented:** An acknowledgment is only for a particular transition to the alarmed state or a sub-state; an acknowledgment is accepted only if it refers to the most recent such transaction.
   - **Expanded Summary:** An acknowledgment is only for a particular transition, whether to an alarmed state, to a sub-state, or a return to normal. Each transition from the normal state marks the beginning of a new RTN group. All transitions in an RTN group must be acknowledged individually before the overall RTN group is considered acknowledged.

Note When you define a tagname with the **Expanded Summary** ACK Model, the **RTN Implies Ack** option in the **Alarm Properties** dialog box does not apply to that tagname.

Any individual alarm can be configured to use any one of these acknowledgment models.

For more information, see the "Alarm Acknowledgment Models" section of Chapter 7.

1. In the **Alarm Comment** box, type the default comments (up to 131 characters) that you want to use for the **.AlarmComment** dot field. (This is an optional field.)

   For more information, see "Tagname Alarm Comments."
2. Click the **Alarm State** that you want the tagname to be in when in alarm.
3. In the **Priority** box, type a value between 1 and 999 (1 is the highest priority and 999 is the lowest). You can use this priority value to select the alarms that you want to be displayed in a window, logged to disk or printed.
4. To associate an inhibitor tagname with an alarm condition, click the alarm condition's respective **Alarm Inhibitor Tag** ellipse (…) button. The **Select Tag** dialog box appears.

Note Each **Alarm Inhibitor Tag** ellipse (…) button is activated when its respective alarm condition is selected. You cannot type in the **Alarm Inhibitor Tag** entry box.

5. In the **Select Tag** dialog box, double-click the tagname that you want to use to suppress the alarm condition. The dialog box will close and the tagname you selected will appear in the **Alarm Inhibitor Tag** box.

   For more information, see "Alarm Inhibitor Tagnames."

6. Once you have completed all entries, click **Close** to save your tagname definition and close all of the tagname dialog boxes.

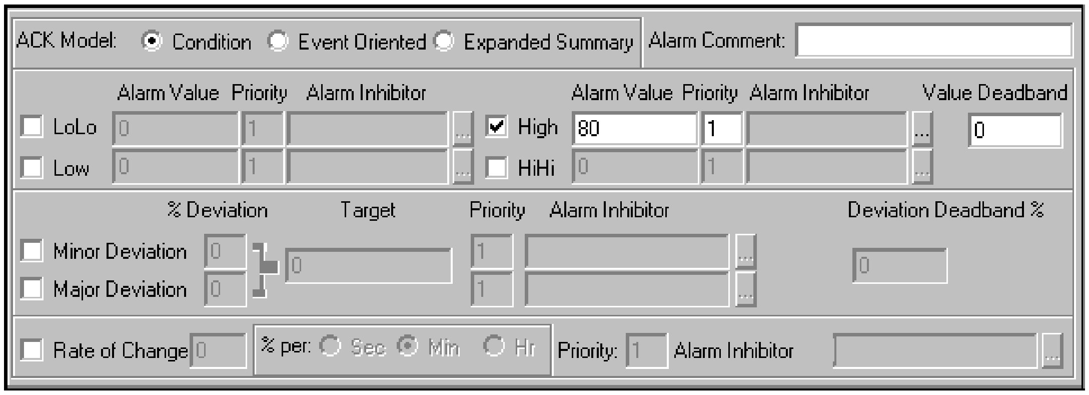
Defining Analog Tagname Alarm Conditions

An analog alarm corresponds to an analog tagname. You can configure whether the ***alarmed*** state corresponds to any analog value of the tagname and the associated priority of that alarm.

To define alarm conditions for an analog tagname

1. On the **Special** menu, click **Tagname Dictionary** or, in the Application Explorer, double-click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears.

2. Click either **Alarms** or **Details & Alarms** at the top of the **Tagname Dictionary** dialog box to display the analog alarm details dialog box:



ACK Model: [✓ Condition] Event Oriented Expanded Summary
Alarm Comment:

	Alarm Value	Priority	Alarm Inhibitor		Alarm Value	Priority	Alarm Inhibitor	Value Deadband
LoLo	0	1		[✓ High]	80	1		0
Low	0	1		HiHi	0	1		

	% Deviation	Target	Priority	Alarm Inhibitor	Deviation Deadband %
Minor Deviation	0	0	1		0
Major Deviation	0		1		

Rate of Change 0 % per: Sec [✓ Min] Hr Priority: 1 Alarm Inhibitor

3. Select the **ACK Model** option that you want to use:

   - **Condition:** An acknowledgment counts against all transitions into the alarmed state or a sub-state up to the time of the acknowledgement. (This is the default and functions the same as previous versions.)

   - **Event Oriented:** An acknowledgment is only for a particular transition to the alarmed state or a sub-state; an acknowledgment is accepted only if it refers to the most recent such transaction.

   - **Expanded Summary:** An acknowledgment is only for a particular transition, whether to an alarmed state, to a sub-state, or a return to normal. Each transition from the normal state marks the beginning of a new RTN group. All transitions in an RTN group must be acknowledged individually before the overall RTN group is considered acknowledged.

Note When you define a tagname with the **Expanded Summary** ACK model, the **RTN Implies Ack** option in the **Alarm Properties** dialog box does not apply to that tagname.

Any individual tagname can be configured to use any one of these acknowledgment models. The selected ACK model applies to all types of alarms on that tagname.

For more information, see the "Alarm Acknowledgment Models" section of Chapter 7.

4. In the **Alarm Comment** box, type the default comments (up to 131 characters) that you want to use for the **.AlarmComment** dot field.

   For more information, see "Tagname Alarm Comments."

5. Select the alarm types (**LoLo**, **Low**, **High**, **HiHi**) that you want to use to detect when the value of an analog type tagname is beyond an absolute limit.

6. In the **Alarm Value** box, type the limit value for the alarm.

   For example, in the case of **LoLo** and **Low** alarms, an alarm condition exists whenever the value of the tagname is less than the **Alarm Value**. In the case of **High** and **HiHi** alarms, an alarm condition exists whenever the value of the tagname is greater than the **Alarm Value**. These fields support the use of real numbers (i.e., 100.75).

7. In the **Value Deadband** box, type the number of engineering units the tagname value must drop below the alarm value before it is taken out of alarm.

   For example, to return-to-normal from an alarm condition, a tagname value must not only return inside its alarm limit, but also return through your specified **Value Deadband**. The **Value Deadband** prevents "nuisance" alarms caused by repetitive re-annunciation of an alarm (where the tagname value hovers around the limit, continually hopping in and out of alarm).

8. Select the deviation (**Minor** and **Major Deviation**) alarm types you want to use to detect when the value of an analog type tagname is in a major or minor deviation from the specified **Target** value.

9. In the **%Deviation** box, type the percentage that the analog tagname can deviate from the **Target** value to produce a minor or major deviation alarm condition. It is expressed as a percentage of the range of the tagname. The MIN EU and MAX EU values entered in the tagname's details dialog box define the range.

10. In the **Target** box, type the desired or reference value of the tagname from which minor and/or major deviation percentages are based.

    For example, let's assume the following setup for an integer tagname:

    Minimum Value = -1000

    Maximum Value = 1000

    Minor Deviation % = 10

    Major Deviation % = 15

    Target = 500

    To calculate at what value the Minor or Major Deviation alarm will take place if the total range of the tagname is **-1000 to +1000 or 2000**, multiply **2000** by either the Minor or Major Deviation percentage **(2000 x .10 (Minor) =200)**. If the Target is **500**, a Minor Deviation will occur whenever the tagname's value drops below **300** or rises above 700.

11. In the **Deviation Deadband %** box, type the deviation percentage the tagname value must drop below the limit before it is taken out of alarm.

12. Select **Rate-of-Change** if you want to detect when the value of an alarm changes an excessive amount for a specified time interval. The tagname is tested for a **Rate-of-Change** alarm whenever its value changes. At this time, the change rate is calculated using the previous value, the time of the last update, the current value, and the current time. This is compared to the rate-of-change allowed in the alarm definition. If the rate-of-change is greater than the alarm limit, the **Rate-of-Change** alarm condition is set for the tagname. A **Rate-of-Change** alarm remains in effect until the next change in the tagname is less than the excessive change amount for the time interval.

13. In the **% per** box, type the maximum allowable percentage change.

14. Select **Sec**, **Min**, or **Hr** for the time interval units of the change.

15. In the respective **Pri** (priority) box type a number between 1 and 999 (1 is the highest priority and 999 is the lowest). You can use the priority value to select the alarms you want to be displayed in a window, logged to disk or printed.

16. To associate an **Alarm Inhibitor Tag** with an alarm condition, click the alarm condition's respective **Alarm Inhibitor Tag** ellipse (…) button. The **Select Tag** dialog box appears.

    Note Each **Alarm Inhibitor Tag** ellipse (…) button is activated when its respective alarm condition is selected. You cannot type in the **Alarm Inhibitor Tag** entry box.

17. In the **Select Tag** dialog box, double-click the tagname that you want to use to suppress the alarm condition. The dialog box will close and the tagname you selected will appear in the **Alarm Inhibitor Tag** box.

    For more information, see "Alarm Inhibitor Tagnames."

18. Once you have completed all entries, click **Close** to save your tagname definition and close all of the tagname dialog boxes.

Tagname Alarm Comments

In the past, InTouch allowed you to set a configuration so that when an alarm was generated, the tagname comment was attached to the alarm. This tagname comment could be displayed in the alarm wizard or in the alarm log. InTouch also provided a configuration setting that allowed the operator to enter a comment when he ACKed an alarm. That ACK comment updated the tagname comment in the Tagname Data Dictionary.

To accommodate the additional configuration options for the Distributed Alarm System, the amount of information stored for each tagname in the Tagname Data Dictionary has been expanded. Consequently, it is now possible to create a NEW comment field specifically for alarms, make it longer, and completely separate it from the tagname comment.

The alarm comment is a new dot field, of type string, that can be set or read via InTouch QuickScripts. The name of the dot field is **.AlarmComment**. As an InTouch string, it can hold up to 131 characters. The Alarm Comment dialog box enables the user to enter the default setting for this dot field.

As a functional change, alarming no longer uses the tagname comment as the alarm comment. Instead, **.AlarmComment** is used. Also, if the operator enters a new comment when ACKing an alarm and the application is configured to copy this comment to the Tagname Data Dictionary **.AlarmComment** is updated – NOT the tagname comment.

Since these changes involve a change to the Tagname Data Dictionary, the version number in file tagname.x has been bumped a revision level. Consequently, if Window Maker loads an application with a previous version of the Tagname Data Dictionary, it automatically converts the old Tagname Data Dictionary to the current version level. As part of this conversion, the tagname comment is copied to the alarm comment as the default setting.

Each alarm acknowledgment can have a comment attached to it -- whether the ACK is done via the Distributed Alarm Object, a script function, or any other means. The operator acknowledging the alarm can use this comment to add information about the alarm.

When an alarm becomes active, the Distributed Alarm System creates an Alarm Record to track that instance of the alarm. For the comment relating to the onset of the alarm, InTouch uses the Alarm Comment, which is entered in the Tagname Dictionary. If the operator provides a comment when ACKing an alarm, InTouch adds this to the Alarm Record as the ACK Comment for that instance of the alarm. The comments for the onset of the alarm and for the acknowledgment are both kept in the Distributed Alarm System, and both are logged in the alarm database. The Distributed Alarm Object and the Alarm Printer show the Alarm Comment or the ACK Comment, according to whether the instance of the alarm has been ACKed. The next time an alarm occurs on the same tag, the Alarm Comment is again used for the onset of a new alarm instance, and the operator can enter a different ACK Comment when acknowledging the new instance.

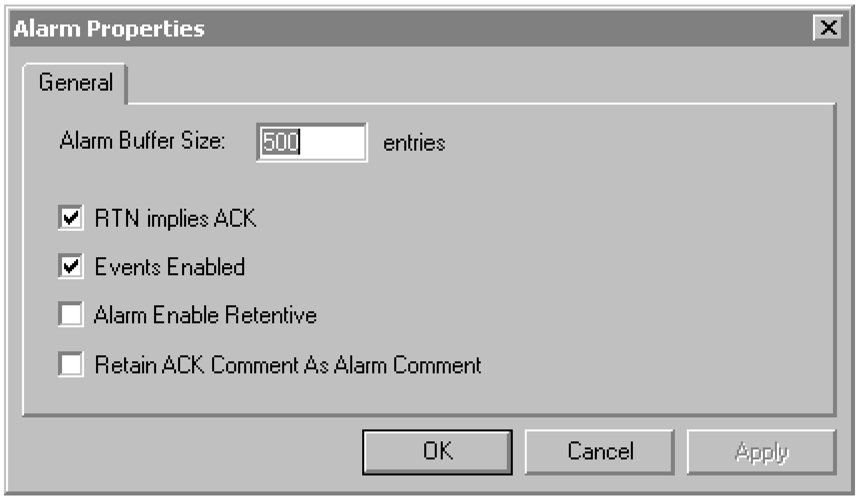
You can also choose to use the ACK Comment to update the Alarm Comment in the tagname database. If you enable this feature, the AlarmComment dot fields will be overwritten during runtime including the Alarm Comment entries in the Tagname Data Dictionary.

To write runtime changes to the **Alarm Comment** field in the tagname database, add the following line to the INTOUCH.INI file for the current application (located in the directory where the current application is stored):

```
CommentRetentive=1
```

You may also configure this feature in the WindowMaker GUI as follows:

1. On the **Special** menu, point to **Configure** and select **Alarms**. The Alarm Properties dialog box appears with the General properties sheet active:



Alarm Properties

General

Alarm Buffer Size: 500 entries

[✓] RTN implies ACK

[✓] Events Enabled

[ ] Alarm Enable Retentive

[ ] Retain ACK Comment As Alarm Comment

Tip If you right-click a text box in any alarm configuration dialog box, a menu appears displaying the commands that you can apply to the selected text.

2. Select **Retain ACK Comment** if you want comments entered with alarm acknowledgements to be kept as updates to the corresponding tagname's AlarmComment dot field and be copied to the Tagname Dictionary. If this box is not checked, the ACK Comment will display with the ACKed alarm (in the database, printouts and displays), but the Alarm comment will not change.
3. Click **OK**.

Creating InTouch SuperTags

InTouch supports a template structure that allows you to define composite tagname types called SuperTags. SuperTag templates can contain up to 64 member tagnames and 2 nesting levels. Meaning, a SuperTag parent can contain up to 64 embedded child members and each child member can contain up to 64 sub-member tagnames for a total of 4095 member tagnames. (When one SuperTag template parent is embedded into another SuperTag template it becomes a "child member.") All SuperTag template sub-member tagnames behave exactly like normal tagnames. They support trending, alarming and all tagname **dotfields**.

For convenience, InTouch provides you with a "TemplateMaker" that you can use to create your SuperTags. The TemplateMaker allows you to create, edit and delete SuperTag templates and member tagnames. InTouch saves all SuperTag templates in the file supertag.dat in your InTouch installation directory (not the application directory). This allows the templates that you create to be used in any application.

InTouch also provides you with the ability to create SuperTags in several alternative ways. For example, you can create SuperTags directly from the Tagname Dictionary, in animation link tagname or expression input boxes, InTouch QuickScripts, or in an external file that you load into your application by using the InTouch DBLoad utility.

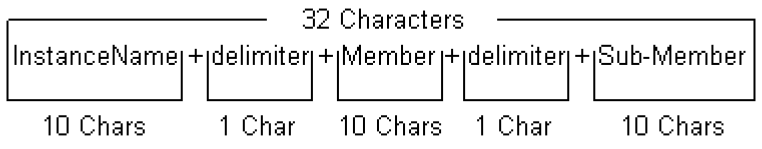
For more information, see "Alternative Methods for Creating SuperTags."

When you create a SuperTag parent template, its name is automatically added to the tagname **Tagname Types** dialog box in the Tagname Dictionary and is immediately available for selection when you create a new tagname. You do not need to restart WindowMaker to define tagnames that use a newly created SuperTag type.

Caution! If you modify an existing SuperTag template, all existing instances of that SuperTag are not affected. (Instances are tagnames defined in the Tagname Dictionary that use a SuperTag for their type.) In other words, modifications that you make to a SuperTag are not retroactive. However, all new instances that you define using the modified SuperTag will use the new composition. Similarly, if you add a member tagname to a SuperTag instance through an alternative method, its template is not updated.

InTouch SuperTag Syntax

Since InTouch tagnames are limited to 32 characters, each SuperTag ParentInstance\ChildMember\Sub-member is restricted to a maximum of 32 characters. A SuperTag reference can only be a maximum of two templates (ParentInstance\ChildMember) and one member deep as illustrated below:



Each member in a SuperTag template is accessible in the standard format that you currently use to access the **dotfields** of normal InTouch tagname types. The SuperTag reference syntax is supported throughout InTouch where normal tagnames can be used. For example, a valid SuperTag reference would be:

```
ColdRoom4\EvapUnit1\FanMotor2.MaxEU
```

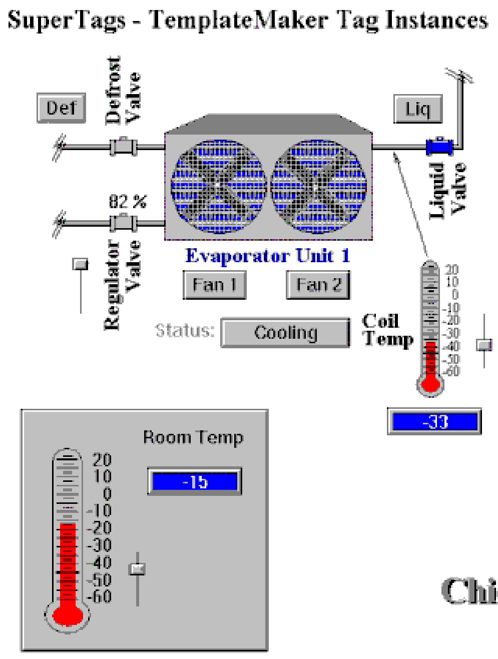
Remote tagname references also support SuperTags. Syntax example:

```
PLC1:"Turkey\EvapUnit2\PrsRegVlv.EngUnits"
```

For more information on using remote tagname references, see "Remote Tagname Referencing."

Creating a SuperTag Template Structure

To realistically illustrate the SuperTag concept in a factory environment, let's assume that we have four identical refrigerated storage rooms in which we store beef, pork, chicken and turkey. Each of these cold rooms has a room temperature, and two evaporator units. Each evaporator unit has seven data values that we need to monitor or control in runtime. For example:



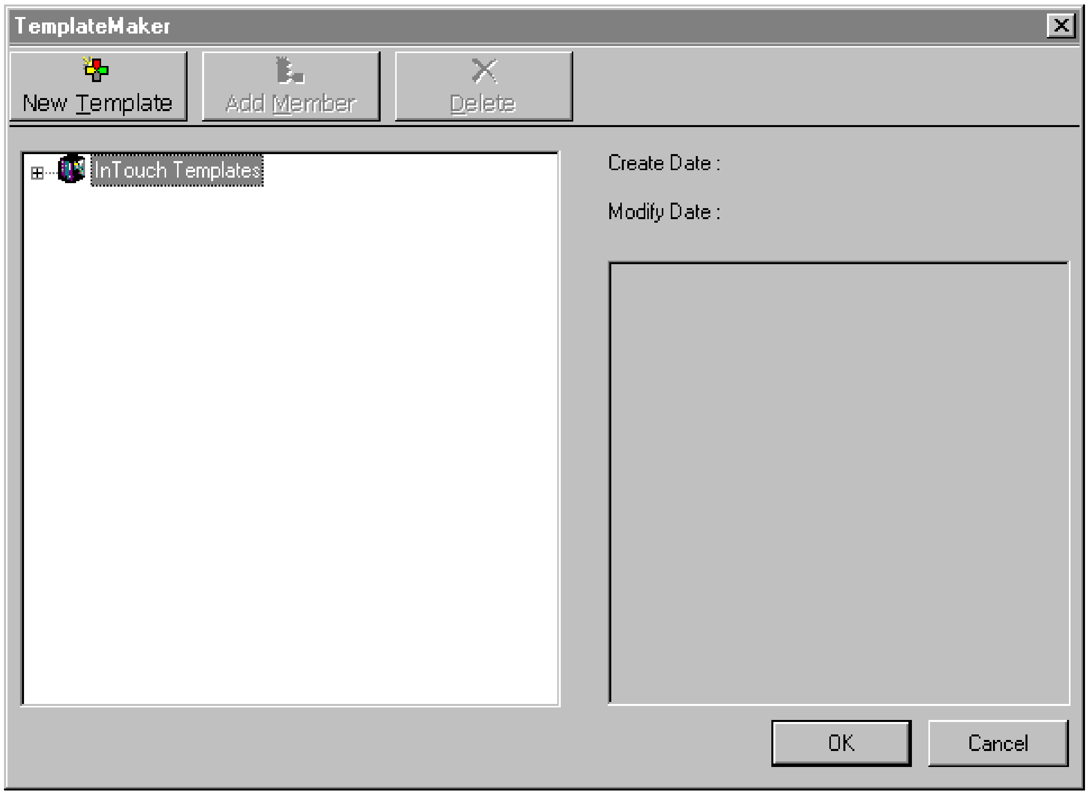
If we do not create SuperTag templates to accomplish this, we would need to manually define each individual tagname for every data value in each cold room multiplied by our total number of cold rooms. In other words, we would have to organize and define dozens of tagnames in the Tagname Dictionary!

By using SuperTags we can save hours of development time and minimize our chance for errors. Using our Cold Room scenario described above, we will create one SuperTag parent template called "EvapUnit." (This EvapUnit will later become a child member of the ColdRoom parent template. This is a "detail-up" design concept.) EvapUnit will be defined with seven sub-member tagnames:

Member Tag	Type	Description
FanMotor1	Discrete	Motor Starter for Fan 1
FanMotor2	Discrete	Motor Starter for Fan 2
DefrostVlv	Discrete	Defrost Gas Valve State
LiquidVlv	Discrete	Liquid Refrigerant Valve State
CoilTemp	Real	Temperature of the refrigerant
PrsRegVlv	Integer	Pressure Regulator Valve (0-100%)
EvapStatus	Message	Evaporator Unit Status String

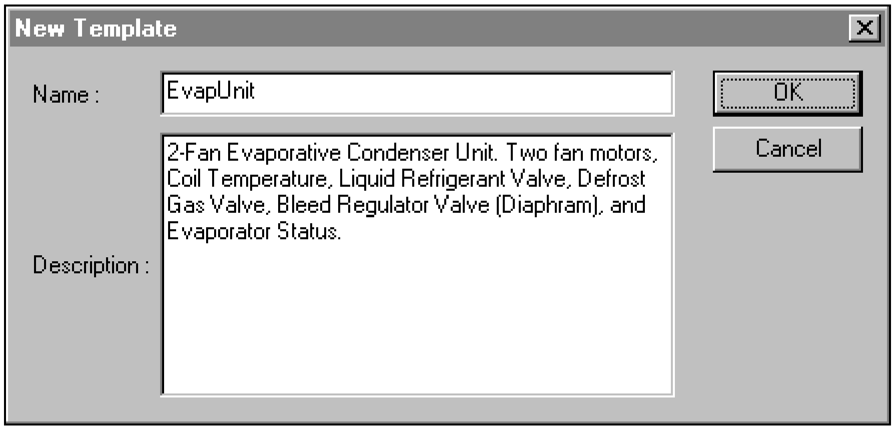
To create a SuperTag parent template

1. On the **Special** menu, click **TemplateMaker** or, in the Application Explorer, double-click **TemplateMaker**. The **TemplateMaker** dialog box appears.



2. In the TemplateMaker window, select **InTouch Templates**, and then click **New Template** or, right-click **InTouch Templates**, and then select **New Template**. The **New Template** dialog box appears.

Tip You can also select **InTouch Templates**, and then right-click a blank area of the window.



Name : EvapUnit

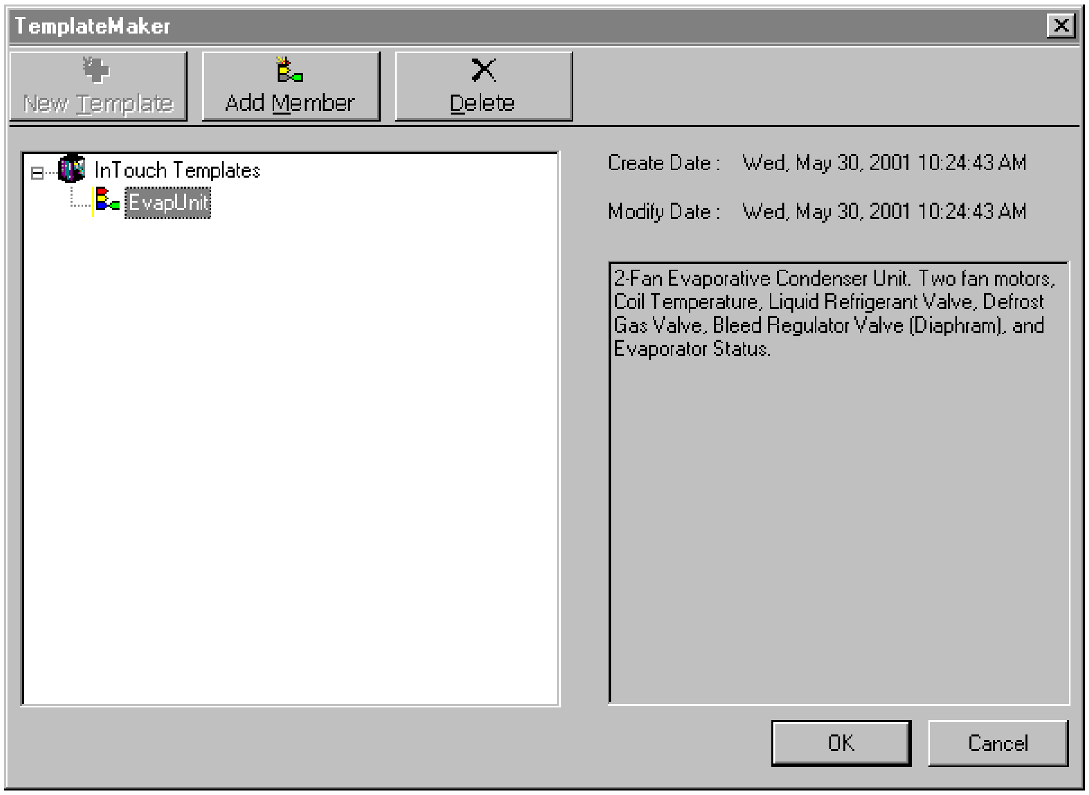
Description : 2-Fan Evaporative Condenser Unit. Two fan motors, Coil Temperature, Liquid Refrigerant Valve, Defrost Gas Valve, Bleed Regulator Valve (Diaphram), and Evaporator Status.

Tip If you right-click any of the text entry boxes in any of the TemplateMaker dialog boxes, a menu appears displaying the commands that you can apply to the selected text.

3. In the **Name** field, type a unique name for the new template (maximum of 10 characters.)

Tip As you add new parent templates, their names immediately appear as a tagname type in the **Tagname Types** dialog box in the Tagname Dictionary and are immediately available for selection. You do not need to restart WindowMaker to define new tagnames and assign them to the SuperTag type.

4. In the **Description** field, type any information that you want to describe the template.
5. Click **OK**. The **TemplateMaker** dialog reappears displaying the new template name in its window.



Create Date : Wed, May 30, 2001 10:24:43 AM

Modify Date : Wed, May 30, 2001 10:24:43 AM

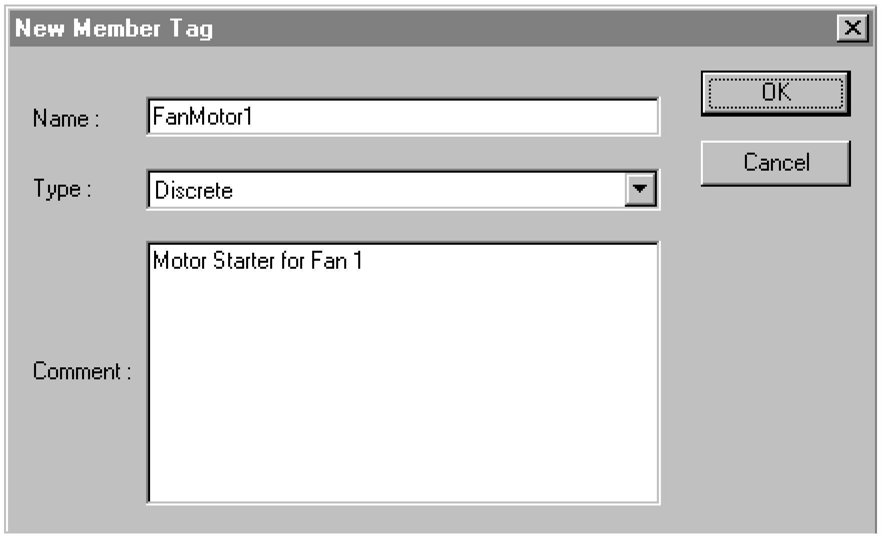
2-Fan Evaporative Condenser Unit. Two fan motors, Coil Temperature, Liquid Refrigerant Valve, Defrost Gas Valve, Bleed Regulator Valve (Diaphram), and Evaporator Status.

Once a template is created, the **New Member** and **Delete** buttons become active. The day, date and time the template was created and/or last modified, and the template's description are also now displayed when the template name is selected.

Note The TemplateMaker window displays all your currently defined SuperTag parent templates and their child members in a hierarchical list. To expand a template's view, click the left mouse button on the ⊞ next to the template name. All member tagnames defined for the parent template name will be displayed. To collapse a view, click the left mouse button on the ⊟.

To create SuperTag member tagnames

1. In the TemplateMaker window, select the SuperTag template (in this case, EvapUnit), and then click **Add Member**, or right-click the SuperTag template name, and then click **Add Member**. The **New Member Tag** dialog box appears.



Name : FanMotor1

Type : Discrete

Comment : Motor Starter for Fan 1

2. In the **Name** box, type the name that you want to use for the member tagname.
3. In the **Type** box, type the tagname type for the member or click the **Type** arrow and select the type in the list. A type can be Discrete, Integer, Real, Message or another SuperTag template.

Tip If you type the first letter of a type, the first type in the list box beginning with that letter will automatically be displayed in the box. If multiple types exist that begin with the same letter, you can continuously type the letter to cycle through the names.

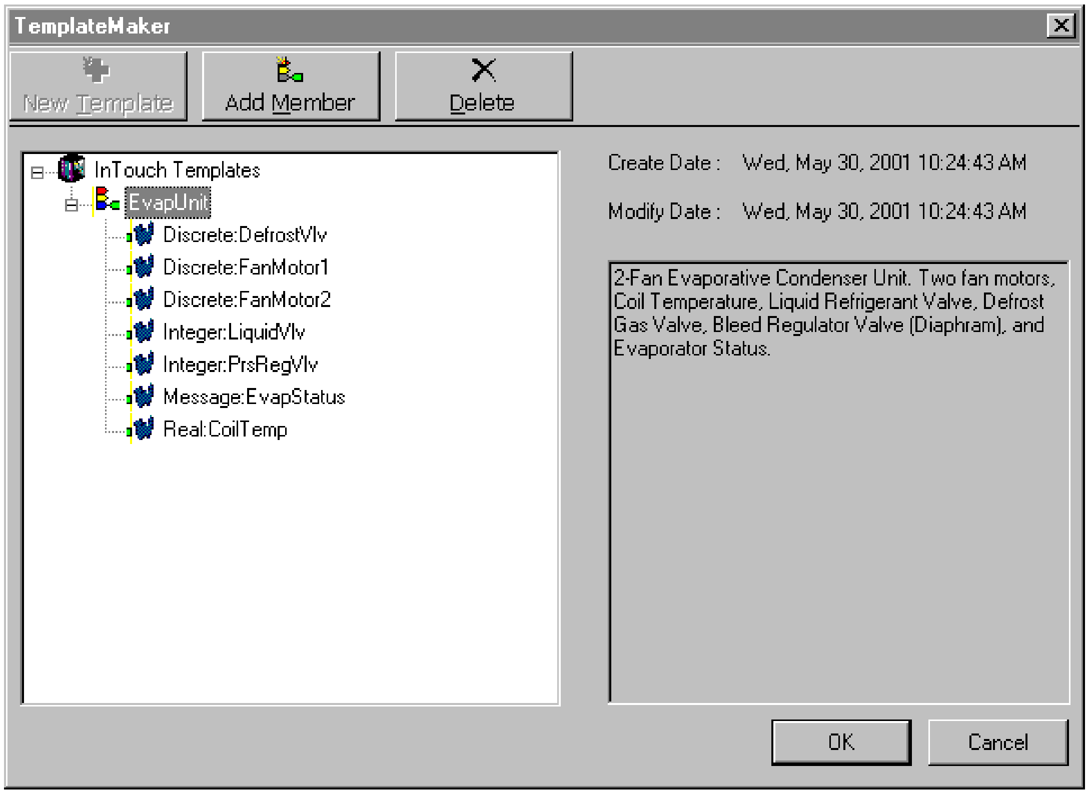
Note The type you specify here is only a placeholder for the SuperTag template. By default, all member tagnames are set to "Memory" types when you define them in the TemplateMaker. However, when you define a template instance in the Tagname Dictionary, you will need to specify whether they are truly "Memory" or "I/O" type tagnames.

For more information on the **Members** dialog box, "Defining SuperTag Member Tagname Details."

4. In the **Comment** field, type any information that you want to describe the member tagname.
5. Click **OK**.

Tip Repeat this procedure to add additional member tagnames to the SuperTag template.

The new member tagnames are added beneath the SuperTag parent template in the TemplateMaker window.



Create Date : Wed, May 30, 2001 10:24:43 AM

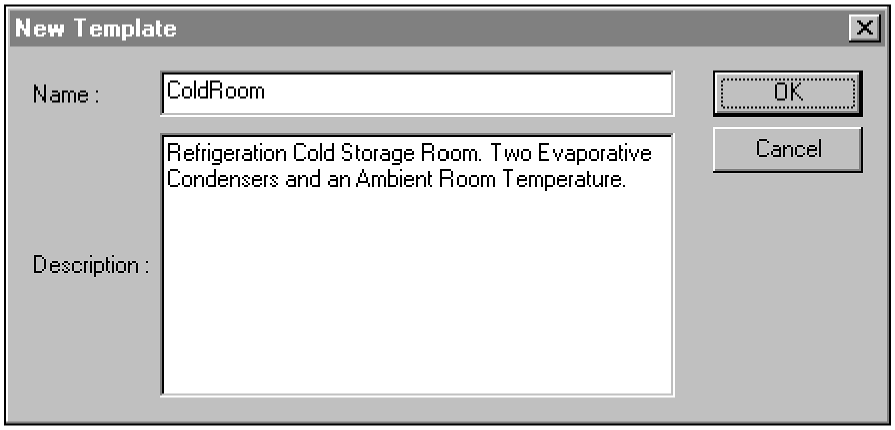
Modify Date : Wed, May 30, 2001 10:24:43 AM

2-Fan Evaporative Condenser Unit. Two fan motors, Coil Temperature, Liquid Refrigerant Valve, Defrost Gas Valve, Bleed Regulator Valve (Diaphram), and Evaporator Status.

Tip Notice that if a member tagname is selected, the **New Member** button is no longer active since members can only be created for existing SuperTag parent templates. The day, date and time the member tagname was created and/or last modified, and the template's description are displayed when the member is selected.

6. You will now create another parent template called ColdRoom. ColdRoom will have one member tagname called RoomTemp and two EvapUnit child member templates (EvapUnit1 and EvapUnit2). The two child member templates will use the parent, EvapUnit, SuperTag template for their type.

7. In the TemplateMaker window, select **InTouch Templates**, and then click **New Template**, or right-click **InTouch Templates**, and then click **New Template**. The **New Template** dialog box appears.



Name : ColdRoom

Description : Refrigeration Cold Storage Room. Two Evaporative Condensers and an Ambient Room Temperature.

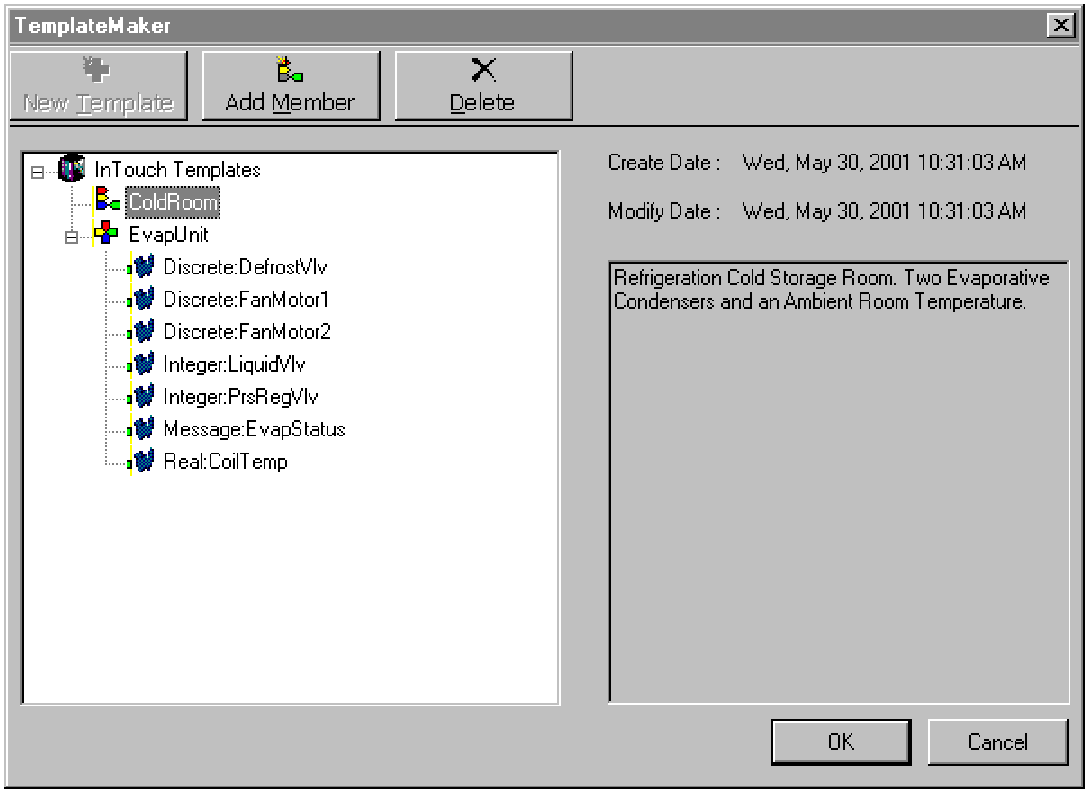
8. In the **Name** field, type a unique name for the new parent template (maximum of 10 characters.)

Tip As you add new parent templates, their names immediately appear as a tagname type in the **Tagname Types** dialog box in the Tagname Dictionary and are immediately available for selection. You do not need to restart WindowMaker to define new tagnames and assign them to the SuperTag type.

9. In the **Description** field, type any information that you want to describe the template.
10. Click **OK**. The **TemplateMaker** dialog reappears displaying the new template name in its window:

11. Click **OK**.

    The parent template is added to the list of **InTouch Templates** in the TemplateMaker window.



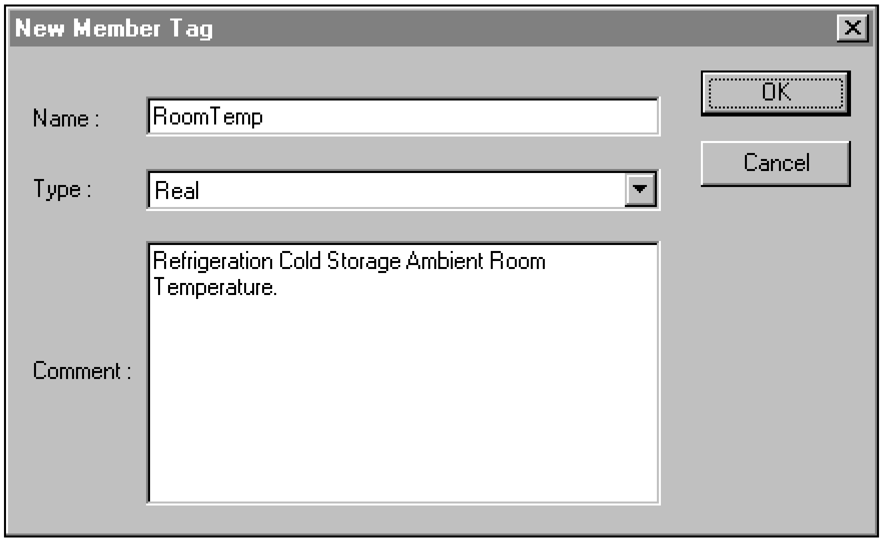
Create Date : Wed, May 30, 2001 10:31:03 AM

Modify Date : Wed, May 30, 2001 10:31:03 AM

Refrigeration Cold Storage Room. Two Evaporative Condensers and an Ambient Room Temperature.

12. We now need to create three members for our ColdRoom parent template; two EvapUnit child members and one member tagname called RoomTemp.

13. In the TemplateMaker window, select the SuperTag parent template (in this case, ColdRoom), and then click **Add Member**, or right-click the SuperTag parent template name, and then click **Add Member**. The **New Member Tag** dialog box appears.



Name : RoomTemp

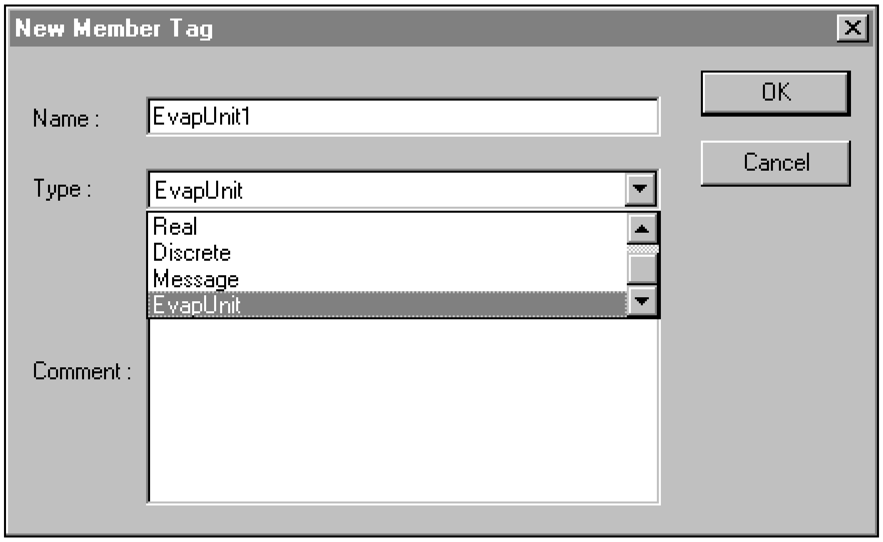
Type : Real

Comment : Refrigeration Cold Storage Ambient Room Temperature.

14. In the **Name** box, type the name that you want to use for the member tagname.
15. In the **Type** box, type the tagname type for the member or, click the **Type** arrow and select the type in the list. A type can be Discrete, Integer, Real, Message or another SuperTag template.
16. In the **Comment** field, type any information that you want to describe the member tagname.
17. Click **OK**.

Next we will create our two child member templates, EvapUnit1 and EvapUnit2 which use the EvapUnit template type.

1. In the TemplateMaker window, select the SuperTag parent template (in this case, ColdRoom), and then click **Add Member**, or right-click the SuperTag parent template name, and then click **Add Member**. The **New Member Tag** dialog box appears.



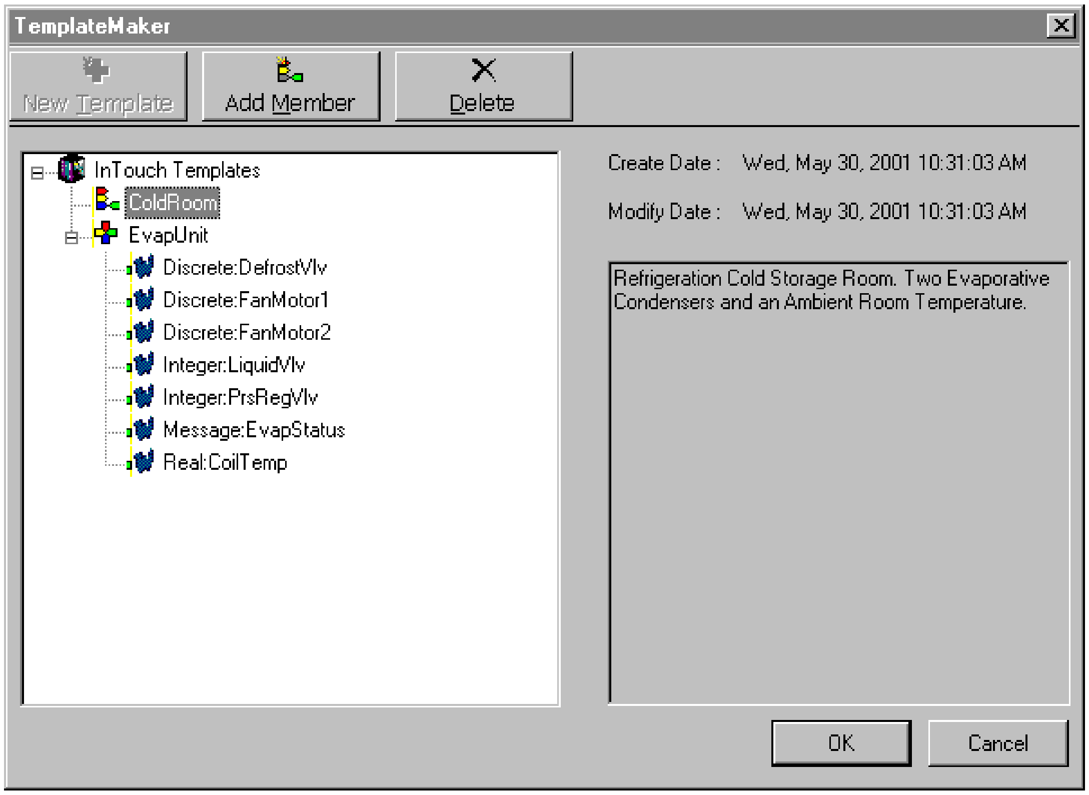
Name : EvapUnit1

Type : EvapUnit

2. In the **Name** box, type the name that you want to use for the member tagname.
3. In the **Type** box, type the tagname type for the member or, click the **Type** arrow and select the type in the list. In this case, we are using the special template type EvapUnit.
4. In the **Comment** field, type any information that you want to describe the member tagname.
5. Click **OK**.

Tip Repeat this procedure for EvapUnit2.

Once we have completed the ColdRoom parent template, the TemplateMaker window will display the following template structure hierarchy.



Create Date : Wed, May 30, 2001 10:31:03 AM

Modify Date : Wed, May 30, 2001 10:31:03 AM

Refrigeration Cold Storage Room. Two Evaporative Condensers and an Ambient Room Temperature.

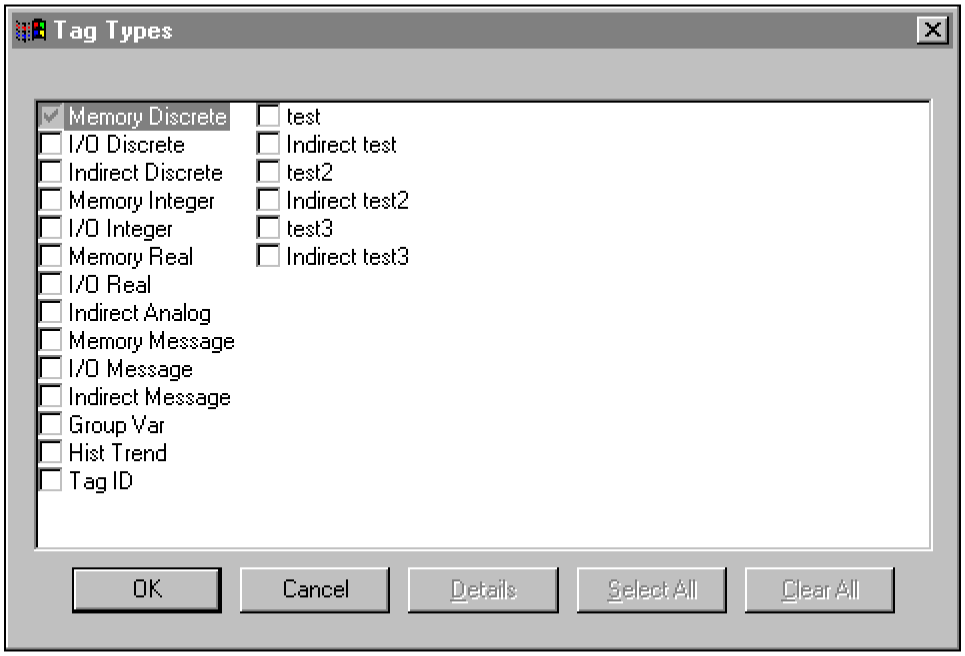
Click **OK**.

Now that we have completed the ColdRoom SuperTag template, we can immediately create tagname instances that use the template for their tagname type.

For more information, see Defining SuperTag Template Instances."

Creating Indirect SuperTags

When you define a new SuperTag tag type, InTouch automatically defines a corresponding Indirect SuperTag tag type. This corresponding indirect SuperTag is an exact duplicate, including all member tagnames, of the original.



The tag types for all members in the duplicate are automatically set to indirect. For example, suppose you define the following SuperTag tagname type:

MySuperTagType

Discrete	MyDiscrete
Integer	MyInteger
Real	MyReal
Message	MyMessage

InTouch defines an Indirect SuperTag tagname type with the following structure:

Indirect MySuperTagType

Indirect	Discrete	MyDiscrete
Indirect	Analog	MyInteger
Indirect	Analog	MyReal
Indirect	Message	MyMessage

Indirect SuperTag tag types behave in scripts similar to Indirect Discrete, Indirect Analog or Indirect Message tag types. For example, consider the following statement:

```
MyIndirectSuperTag.Name = "MySuperTag";
```

If the name on the right-hand side refers to a SuperTag parent, InTouch will loop through its children and attempt to assign a value to an equivalent member of the Indirect SuperTag on the left hand side. If InTouch is unable to find an equivalent member, InTouch will not make an assignment of that particular child.

Assume that the SuperTags mentioned above have the following structures:

MyIndirectSuperTag	MySuperTag
Value1	Value1
Value2	Value2
Value3	Value3
ValueX	Value4
	Value5

The statement listed above will have the following effect:

```
MyIndirectSuperTag.Value1 = MySuperTag.Value1
MyIndirectSuperTag.Value2 = MySuperTag.Value2
MyIndirectSuperTag.Value3 = MySuperTag.Value3
```

Furthermore, Indirect SuperTag tag types function just as Indirect Discrete, Indirect Analog or Indirect Message tag types in animation links. For example, a Pushbutton Action script can execute the following statement:

```
MyIndirectSuperTag.Name = "MySuperTag";
```

Then, an Analog Value Display link can use the following expression to show ten times the value of MySuperTag.Value1:

```
MyIndirectSuperTag.Value1 * 10
```

Editing SuperTag Templates and Member Tagnames

You can modify SuperTag templates or member tagnames at any time. However, if you modify an existing SuperTag template or its members, all existing instances of that template are not affected. (Instances are tagnames defined in the Tagname Dictionary that use a SuperTag for their type.) In other words, modifications that you make to a SuperTag are not retroactive. However, all new instances that you define using the modified SuperTag will use the new composition.

To edit an existing SuperTag template or member tagname

1. In the TemplateMaker window, double-click the SuperTag template name (or member name), or right-click it, and then click **Edit**. The **Edit Template** (or **Edit Member Tag**) dialog box appears displaying the SuperTag template's (or member's) definition.
2. Make your required edits, and then click **OK**.

To delete a SuperTag template or member

1. In the TemplateMaker window, select the SuperTag template name (or member name) that you want to delete, or right-click it, and then click **Delete**. A message box appears asking you to confirm the deletion.
2. Click **Yes** to delete the selected name, or click **No** to cancel the deletion.

Note If you press the ESC key to close the TemplateMaker instead **OK**, the template is not deleted. When you delete a template, all its associated member tagnames are also deleted.

Defining SuperTag Template Instances

An important concept in TemplateMaker is distinguishing a SuperTag template from a template instance. A template instance is a specific instantiation of a SuperTag template. The most important difference between a template and an instance is that the parent template name is replaced by the instance tagname. The child template name and the sub-member tagnames do not change.

For example, this could be equated to a literal template that you use for drafting such as a stencil that you use to produce actual drawings. The drawings themselves, in this metaphor, are "template instances" that are patterned after the template or stencil used to create them.

Once again referring to our ColdRoom template scenario, after we have created our template, from it we could create SuperTag instances of "Beef," "Pork," "Chicken" and "Turkey." To do so, we will simply create four tagnames that use ColdRoom for their types. Thus with our one time effort, we will quickly create 60 tagnames in the Tagname Dictionary. A huge time saver!

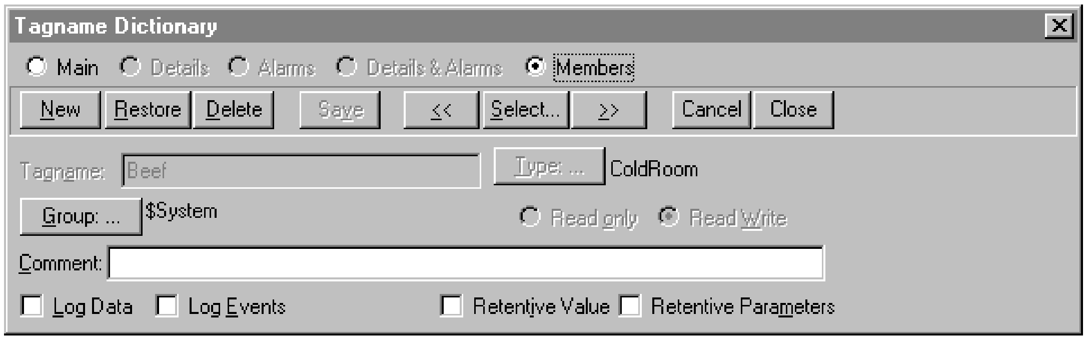
After we have created the "ColdRoom" SuperTag template and the instances, we can refer to any of its members by using valid SuperTag references in animation link expressions or QuickScripts. For example:

```
Beef\RoomTemp

Chicken\RoomTemp.RawValue

Chicken\EvapUnit1\FanMotor1.OnMsg

Pork\EvapUnit2\EvapStatus

Turkey\EvapUnit2\PrsRegVlv.EngUnits
```

For more information on defining template instances see "Defining SuperTag Member Tagname Details."

To create a SuperTag from the newly created template

1. On the **Special** menu, click **Tagname Dictionary** or, in the Application Explorer, double-click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears.



Main | Details | Alarms | Details & Alarms | [✓ Members]

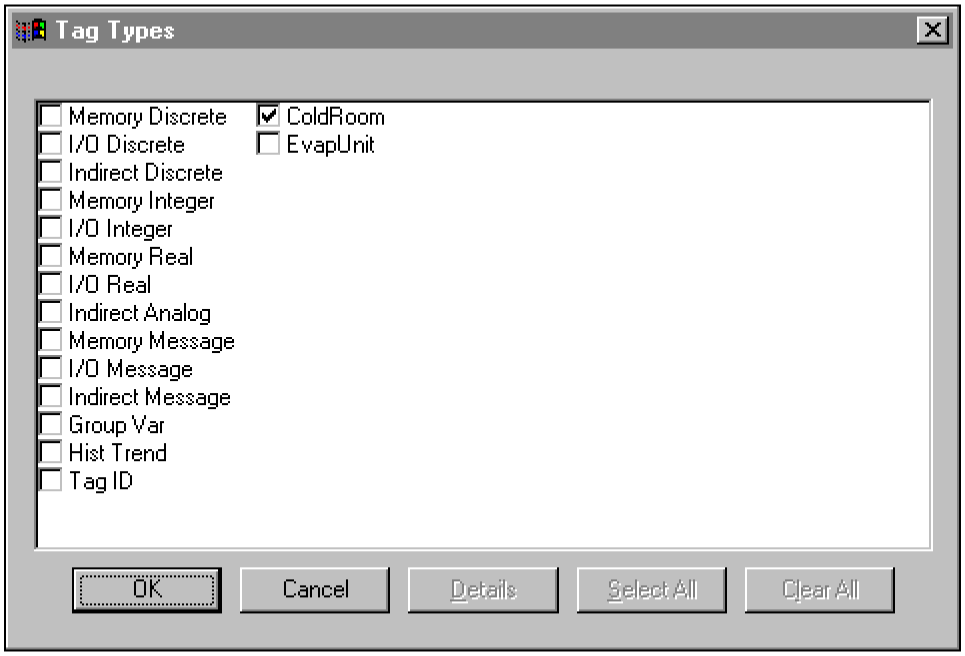
Tagname: Beef
Type: ColdRoom
Group: $System
Read only [✓ Read Write]
Comment:
☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

2. Click **New**. (The **Tagname** box clears.)
3. In the **Tagname** box, type the name you want to use for the new tagname.

Tip Tagnames can be up to 32 characters long and must begin with an alpha character (**A-Z** or **a-z**). The remaining characters can be A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &.

You cannot use the word **RetVal** for a tagname. This is a reserved word. If you attempt to use this word, and then try to edit a QuickFunction you will receive the error message "A variable cannot have that name. Tag exists."

4. Click **Type**. The **Tagname Types** dialog box appears.

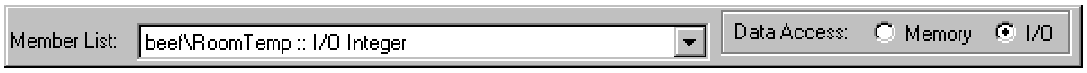


☐ Memory Discrete
☐ I/O Discrete
☐ Indirect Discrete
☐ Memory Integer
☐ I/O Integer
☐ Memory Real
☐ I/O Real
☐ Indirect Analog
☐ Memory Message
☐ I/O Message
☐ Indirect Message
☐ Group Var
☐ Hist Trend
☐ Tag ID
☑ ColdRoom
☐ EvapUnit

5. Select **ColdRoom** for the tagname, then click **OK**.

6. When you select a SuperTag template as the type for your tagname, the following details dialog box appears.

Tip If it does not appear, click **Members** at the top of the **Tagname Dictionary** dialog box.



Notice that the new tagname that you typed in the **Tagname** box becomes the "parent" for all the member tagnames in the **Member List**.

7. Click the **Member List** arrow, and then select the member tagname in the list that you want to define as an I/O data access type
8. In the **Data Access** group, select **I/O**. The respective I/O details dialog box for the member tagname's type (Discrete, Analog (Real or Integer) or Message) appears.
9. Enter the required I/O details just as you would for a normal InTouch I/O type tagname.
10. To save your changes, select another member tagname in the list and configure it, or click **Close**.

Tip For more information on tagname types, see "Tagname Types."

For more information on creating InTouch SuperTags see, "Creating InTouch SuperTags."

To replicate a SuperTag instance which is created from a template

1. Select a SuperTag in the tagname dictionary then **New**.
2. A dialog appears asking if you wish to replicate. If you click **Yes** a dialog appears prompting for the name of the new SuperTag. Type a 10 character name and click **OK**.

Note This functionality does not work for SuperTags created through animation link expressions, InTouch QuickScripts, and external .csv (Comma Separated Variable) files that you upload into the Tagname Dictionary through the DBLoad utility.

You will have created a new SuperTag instance modeled after the original. The only difference is the root or parent name.

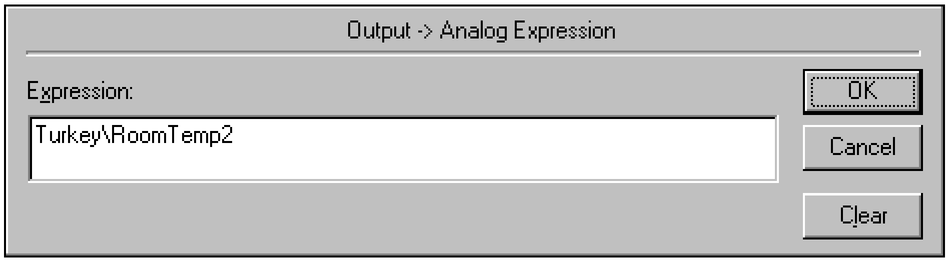
Alternative Methods for Creating SuperTags

In addition to the TemplateMaker, InTouch supports the creation of SuperTags through animation link expressions, InTouch QuickScripts, and external .csv (Comma Separated Variable) files that you upload into the Tagname Dictionary through the DBLoad utility. However, you can also add a member or sub-member to an existing SuperTag through the Tagname Dictionary, which is the easiest method to use.

For more information on creating SuperTags using DBLoad, see "Creating SuperTag Instances."

Note When you use one of the alternative methods to create the member, it is not reflected in the SuperTag template definition in the TemplateMaker.

When you create a SuperTag through an animation expression or InTouch QuickScript you must use the valid SuperTag format. For example:



```
Turkey\RoomTemp2
```

The following syntax examples are valid:

```
ParentInstance\ChildMember
   ParentInstance\ChildMember\Submember
```

The following syntax examples are invalid:

```
ParentInstance\
```

```
ParentInstance\ChildMember\
```

If an invalid format is used, the an error message box appears informing you that the syntax is invalid.

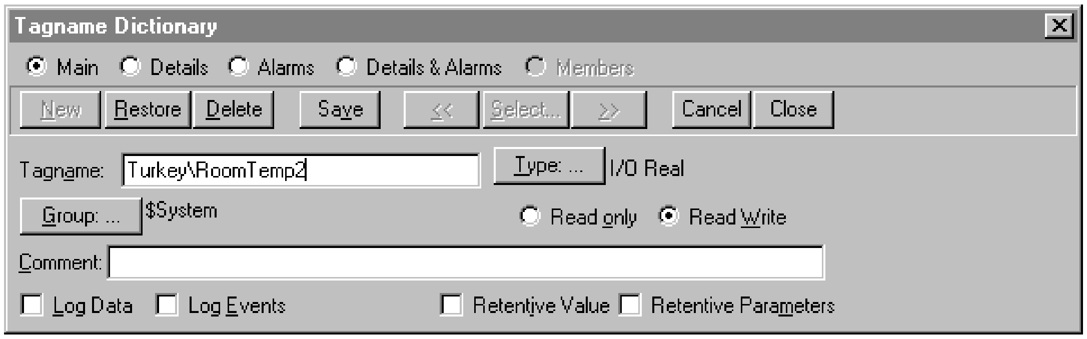
Tip If the SuperTag instance and member tagname you specify in an animation expression or QuickScript are currently not defined, a message box appears asking you if you want to define it now. Click **OK**. The Tagname Dictionary appears displaying the SuperTag instance and member tagname that you specified.

Using the Tagname Dictionary to Create SuperTags

The Tagname Dictionary is the easiest alternative method to use for creating a SuperTag instance or member tagnames.

To create a SuperTag in the Tagname Dictionary

1. On the **Special** menu, click **Tagname Dictionary**, or in the Application Explorer, double-click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears. Click **New**.


Tagname Dictionary
[✓ Main] Details Alarms Details & Alarms Members
New Restore Delete Save << Select... >> Cancel Close

Tagname: Turkey\RoomTemp2
Type: ... I/O Real
Group: ... $System
Read only [✓ Read Write]
Comment:
☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

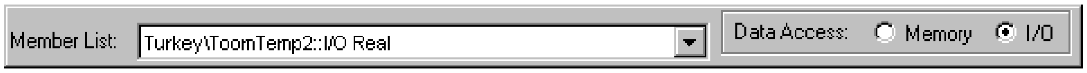
2. In the **Tagname** box, type the exact name of your SuperTag instance followed by the backslash (\) delimiter and the name of the new member tagname. In this case, we would type **Turkey\RoomTemp2**.

Note When you are adding a new member tagname to an existing SuperTag instance, the spelling of the instance name must match exactly. Otherwise, a brand new SuperTag instance and member will be added.

Tip Tagnames can be up to 32 characters long and must begin with an alpha character (**A-Z** or **a-z**). The remaining characters can be A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &.

You cannot use the word **RetVal** for a tagname. This is a reserved word. If you attempt to use this word, and then try to edit a QuickFunction you will receive the error message "A variable cannot have that name. Tag exists."

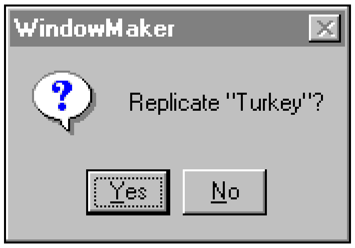
3. Click **Type** and select the type for the SuperTag member. (Selection of the remaining options is not required in this context.) In this case, we have selected I/O Real.
4. Click **Save** or **Close** to add the member.
5. To view the member tagname in the Turkey SuperTag without exiting the Tagname Dictionary, click either the left or right double-arrow buttons. The **Members** details dialog box appears.



Member List: Turkey\ToomTemp2::I/O Real
Data Access: Memory [✓ I/O]

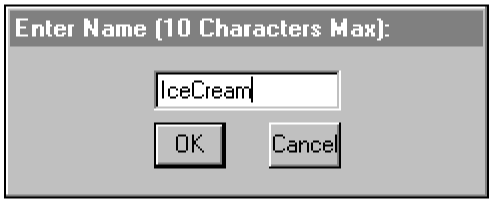
6. Click **Close** to close the Tagname Dictionary.
7. If you click **New** when a SuperTag is displayed in the Tagname Dictionary, the following dialog box appears asking you if you want to make an identical copy of the displayed SuperTag instance:

Note When you are adding a new member tagname to an existing SuperTag instance, the spelling of the instance name must match exactly. Otherwise, a brand new SuperTag instance and member will be added.


WindowMaker

Replicate "Turkey"?

8. Click **Yes** to create another SuperTag instance that is an exact duplicate of the displayed SuperTag instance. The **Enter Name** dialog box appears.


Enter Name (10 Characters Max):

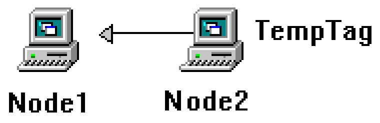
IceCream

9. Type a new SuperTag instance name.
10. Click **OK**.

Tip The Tagname Dictionary automatically creates all member tagnames and sub-member tagnames for the new SuperTag instance, and they are immediately available for usage in animation links and InTouch QuickScripts.

Remote Tagname Referencing

InTouch allows true client-server architecture for factory automation applications. Client applications can be designed without using any tagnames in the local Tagname Dictionary. This can be achieved by the using "Remote Tagname Referencing." capability of InTouch. For example:



In this example, you can retrieve the value of the tagname "TempTag" on Node2 in two ways:

1. Create an I/O type tagname in Node1's Tagname Dictionary that uses "Node2" as the **Node Name** in the Access Name associated with the I/O tagname.

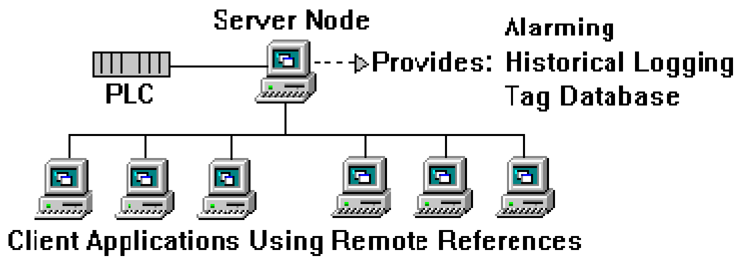
2. Use a remote reference directly to "TempTag." For example, **PLC1:"TempTag"**

In other words, in a window or QuickScript, you can either reference the local tagname or, use *AccessName:"item"* to reference a remote tagname.

For more information on remote tagname reference syntax, see "Remote Tagname Reference Syntax."

When you want to directly reference a remote tagname in any other FactorySuite application, only *AccessName:"item"* is required. You do not have to define the remote tagname in your local Tagname Dictionary. Remote references can also access data from any I/O data source such as, a Wonderware I/O Server or Microsoft Excel. In addition, they support SuperTags (SuperTags). The valid syntax for a remote tagname referencing a SuperTag is *Accessname:"ParentInstance\ChildMember\SubMember"*.

Also, when you use remote tagname references, and you import a window or QuickScript, all you have to do is convert the placeholder tagnames to remote tagname references. You do not have to define tagnames in your local Tagname Dictionary. The remote references are accessible from any FactorySuite application on the network as shown in the following illustration:



Remote Tagname Reference Syntax

The valid syntax for a remote tagname reference is *AccessName:"item"*. The characters that you can use in a remote reference are the same characters that are valid for a tagname. The valid characters are: A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &. If you are sure that you do not use any invalid characters in your remote tagname reference then you do not need to enclose the *item* portion in quotation marks.

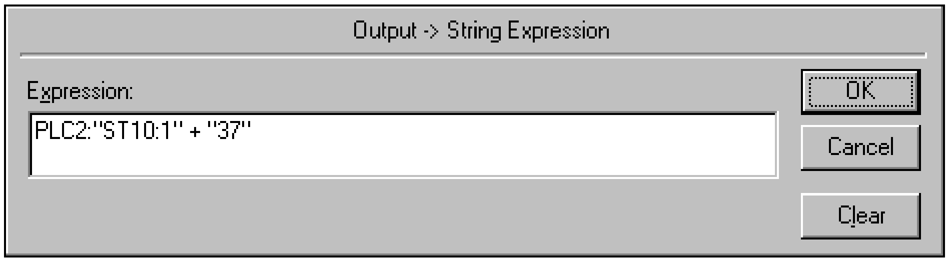
Tip Tagname **dotfields** can also be used in the "item" portion of the remote tagname reference. For example, "MyAlarm.HiHi".

In order to use any other characters, you must enclose the *"item"* in quotation marks. For example, if you use; ~, *, /, +, =, ^, |, **, <, >, <=, =>, ==, and <> you must enclose the *"item"* in quotation marks.

However, since some general ASCII I/O Servers accept any character as valid for an item name, we highly recommend that you make a practice of always surrounding the *"item"* portion with quotation marks.

For example, let's assume that you want to get a bit from an Allen-Bradley® PLC integer register and you use **N10:7/3** (third bit from integer 10) as the item name. The system will see **N10:** as an Access Name because the forward slash (/) is not a valid character. However, if you enclose the item name in quotations, **"N10:7/3"**, the system will read entire entry as the item name.

You also cannot string concatenate item names or remote tagname reference item names. For example, let's assume that you created a string output link using the following expression:



```
PLC2:"ST10:1" + "37"
```

When the system executes the above expression it will use the Access Name **PLC2** and go through the Allen-Bradley I/O Server to retrieve the string stored in the string file **ST10:1**. Then it will append the string **37** to the end of the string it retrieved in **ST10:1**. If "Green Paint" is stored in **ST10:1**, the string output object linked to the expression will display **Green Paint37**. Therefore, the operator would not see the contents of **ST10:137** as they had intended.

Whenever you use a remote reference (*accessname:"item"*), InTouch validates the Access Name that you specify. If it determines that the Access Name is not defined, you will be prompted to define it. If you select **Yes** when prompted, the **Access Names** dialog box appears and you can add the new Access Name.

The Access Name is also validated when the remote tagname is activated. If errors are encountered, they will be written to the Logger.

You can delete an Access Name that is used by in a remote reference, as long as a local tagname is not using it.

For more information on defining Access Names, see Chapter 5, "Building a Distributed Application."

Logging Remote Referenced Tagnames

By default, remote referenced tagnames are not logged to the Historical Log file. To log remote referenced tagnames, you must enable Historical Logging and then, add the following line to the INTOUCH.INI file in the application directory:

```
RemoteTagsLogEvents=1
```

To exclude I/O tagnames from being logged, add the following line to the INTOUCH.INI file in the application directory:

```
RemoteTagsNoIOEvents=1
```

Note The **RemoteTagsNoIOEvents** setting only applies if **RemoteTagsLogEvents** is set to **1**.

Remote Tagname License Enforcement

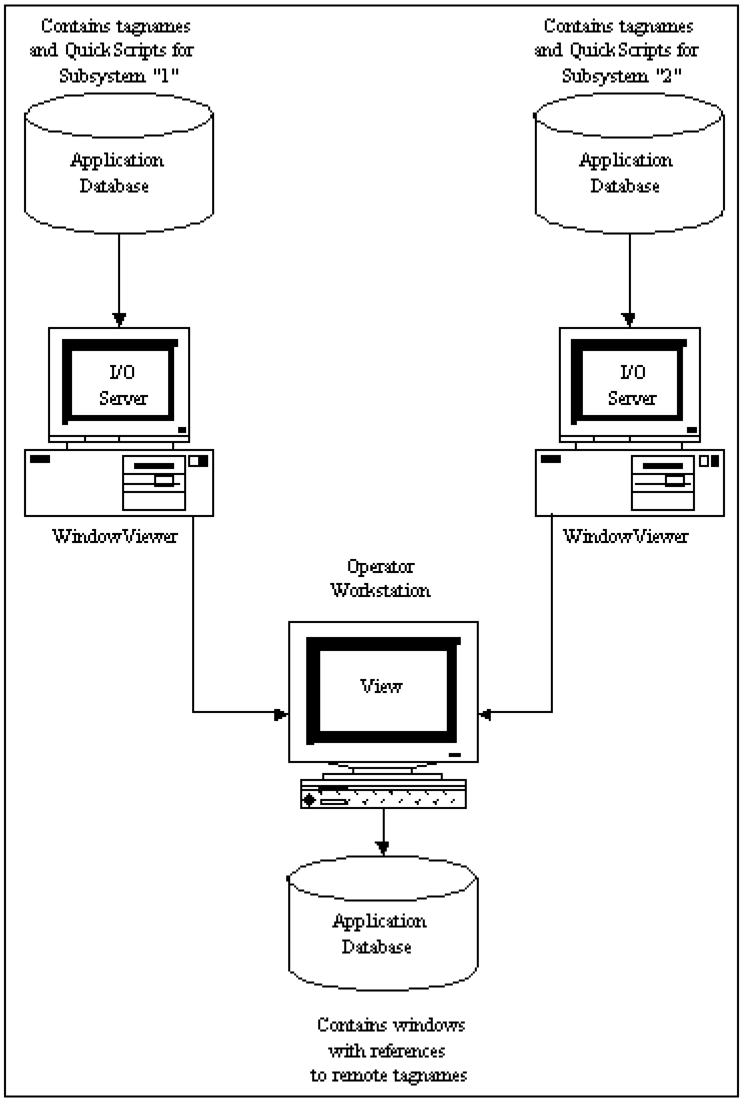
The InTouch Tagname Data Dictionary supports up to 61,405 tags. The InTouch 60K tagname license allows for the maximum number of tagname references. When a 60K tagname license is installed, the tagname reference count used by the license enforcement logic is incremented whenever a tagname is activated within a window or QuickScript, and decremented whenever the referencing window or QuickScript is closed and/or terminated.

Prior to InTouch 7.11, while within an application session, the total number of design time declared tagnames plus the number of in scope remote tagname references had to always be less than or equal to 60K. InTouch 7.11 60K tagname licenses allow more than 60K design time plus remote tagname references to be declared by the application designer. The system dynamically increments and decrements the total tagname reference count as remote tagnames come into and out of scope respectively over the life of the application session.

Note In the case of licenses that allow less than the maximum number of tag name references, once a tagname is referenced and counted against the licensed quota, the count is never decremented from the quota even if the tagname is never used again.

Creating a Tagname Server Application

By creating an application that contains only InTouch QuickScripts and tagnames, you can establish an instance of WindowViewer that functions as a tagname server. You can create another application that contains only windows (and memory tagnames for window logic processing). If these windows contain only remote tagname references, this application can serve as a repository for all the process windows for a facility. In this case, the remote tagname references are to tagnames in other WindowViewer instances that function as tagname servers. An instance of WindowViewer that connects to this database functions as an operator workstation. This WindowViewer instance can open any window and view data from anywhere on the plant floor. For example:


Contains tagnames and QuickScripts for Subsystem "1"
Contains tagnames and QuickScripts for Subsystem "2"
Application Database
Application Database
I/O Server
I/O Server
WindowViewer
WindowViewer
Operator Workstation
View
Application Database
Contains windows with references to remote tagnames

Remote references are valid for the following:

Item	Valid For
Input Links	Discrete User Input, Analog User Input, String User Input, Vertical Slider, Horizontal Slider, and Discrete Value Button
Discrete Alarm Links	Line Color, Fill Color, and Text Color
Analog Alarm Links	Line Color, Fill Color, and Text Color
Expressions	Links and Scripts where a discrete, analog or string tagname can be specified Wizards
Data Change Scripts	"Tagname[dotfields]"
ActiveX Controls	Events, Properties, and Methods
QuickScripts	All types

Remote references are not valid for the following:

Item	InValid For
Historical Trend Display	"Pen1" through "Pen 8"
Acknowledging an alarm	(Since you cannot see a remote tagname go into alarm, you cannot acknowledge it.)

Note Implementation of remote tagname references does not require conversion of applications that were created with earlier versions of InTouch that do not support this feature. However, once implemented, the applications will not be backwards compatible with the earlier versions.

WindowViewer supports 32767 references to local tagnames and *x* references to active remote references, where *x* = 61,405 minus the number of tagnames defined in the local Tagname Dictionary.

Remote references can be a maximum of 95 characters long.

The **IOSetAccessName** (**SetDdeTopic** in versions prior to InTouch 7.0) function is also supported for remote references and works that same as it does for local tagnames.

Using Remote Tagname References

There are actually three ways that you can specify a remote tagname reference in a client application:

1. Using the *AccessName:"item"* reference in any animation link tagname or expression or, in an InTouch QuickScript.

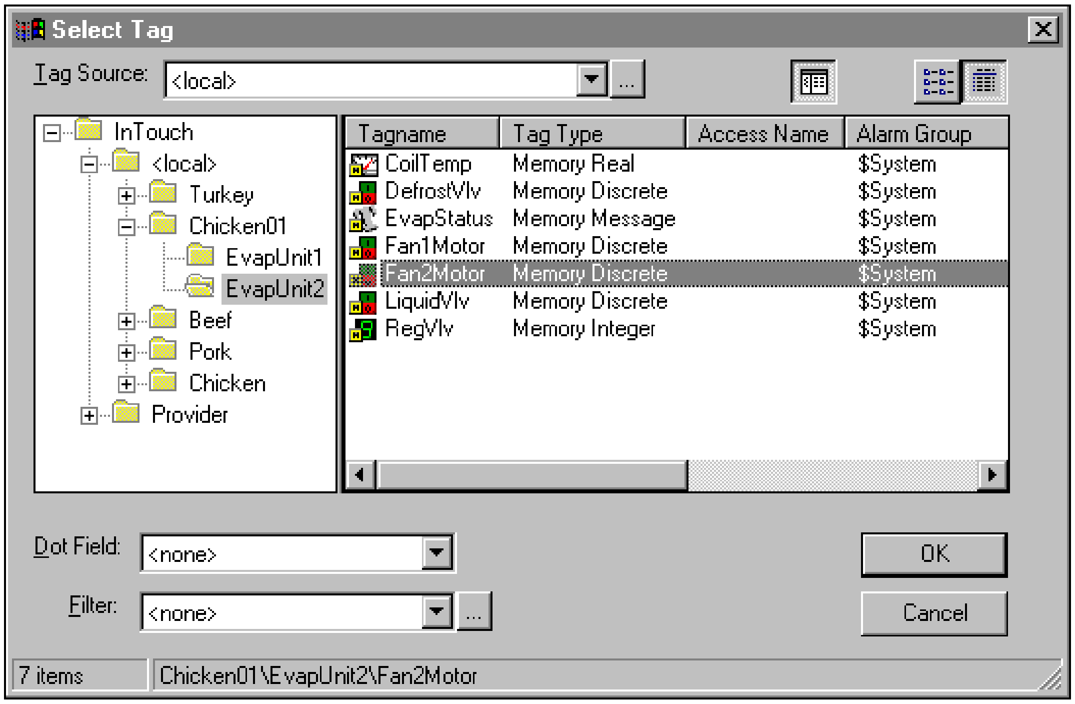
2. Importing a window or QuickScript and converting the placeholder tagnames to remote tagname references by using the **Substitute Tags** command on the **Special** menu in WindowMaker.

   For more information on converting placeholder tagnames, see "Converting Tagnames to Remote References."

   Tip One of the powerful features of InTouch is the ability to import a window from another application. When you import a window, all of its scripts and animation links are imported with it. However, all of the tagnames used in the animation links and scripts are automatically converted to placeholders. You can convert all the placeholder tagnames to remote tagname references and, if desired, design an application with no local tagnames.

   For more information on importing windows or scripts, see Chapter 2, "Using WindowMaker."

3. Selecting the remote tagname that you want to use for an object or QuickScript by configuring the remote application as the tag source in the Tag Browser. For example:



Tag Source: <local>

Tagname	Tag Type	Access Name	Alarm Group
CoilTemp	Memory Real		$System
DefrostVlv	Memory Discrete		$System
EvapStatus	Memory Message		$System
Fan1Motor	Memory Discrete		$System
Fan2Motor	Memory Discrete		$System
LiquidVlv	Memory Discrete		$System
RegVlv	Memory Integer		$System

Dot Field: <none>

Filter: <none>

7 items | Chicken01\EvapUnit2\Fan2Motor

For more information on selecting remote tagnames from the Tag Browser, see "Defining Tag Sources"

Dynamic Reference Addressing (DRA)

Dynamic Reference Addressing allows you to address multiple data sources with a single tagname. By assigning a valid reference to the **.Reference** field of an I/O type tagname, you can dynamically change the address of the data source for the tagname.

Each I/O type tagname has a reference associated with for the address of its data source. The valid syntax for the **.Reference** field includes:

Valid Syntax	Description
`Tagname.Reference="accessname.item"`	Changes Access Name and item.
`Tagname.Reference="[.]item"`	Same Access Name, different item.
`Tagname.Reference="accessname."`	Changes Access Name.
`Tagname.Reference=""`	Deactivates the tagname. If the Access Name or Item is not specified, the current value for that field is assumed.

Note Dynamic Reference Addressing is not valid for remote tagname references.

For more information on **.Reference**, see your online *InTouch Reference Guide.*

Using Dynamic References

Dynamic references are used to view data points whose values are only needed temporarily, such as in diagnostic applications. Since the data source of a tagname can be changed, dynamic references should not be used for any data that needs to be permanently stored or continuously monitored for alarm conditions.

A good example of a traditional use of dynamic references is the diagnostic application. In this application, a single tagname is used to view the input value of any analog point in a PLC. This allows a maintenance person to immediately view the status of any point for trouble-shooting purposes.

To create a diagnostic application

1. Create an I/O Integer type tagname. In this example, the tagname is called "AnalogSpy." It has an initial reference to **PLC1** for the Access Name and **WX001** for the item name.
2. Create a text object by typing a # sign.
3. Double-click the # sign to open the animation links dialog box.
4. Click **String** in the **User Inputs** section. The **Input -> String Tagname** dialog box appears.
5. In the **Tagname** box, type **AnalogSpy.Reference**
6. Click **OK**.
7. Start WindowViewer to compile and run the application.

8. Click on the text object, and enter a new value for the Access Name and item name assigned to the tagname.

   For example, to view item **WX031** from Access Name **PLC6**, enter **PLC6.WX031** as the reference.

9. If you want to confirm that the new reference is valid, use the**.ReferenceComplete** field described in the next section.

   For more information on **.ReferenceComplete** see your online *InTouch Reference Guide.*

Using IOSetItem Function to Change References

The **IOSetItem** (**SetDdeItem** in versions prior to InTouch 7.0), function is used to set an I/O type tagname's **.Reference** field. The basic format of this function is:

```
IOSetItem(TagName, AccessName, Item)
```

The tagname, Access Name, and item values can be specified as literal strings, or they can be string values provided by other InTouch tagnames or functions. For example, the **.Reference** field of tagname "MyTag1" can be changed to point to the "Excel" Access Name and the "R1C1" item by:

```
IOSetItem("MyTag1", "Excel", "R1C1");
```

or by,

```
Number = 1;

TagNameString = "MyTag" + Text(Number, "#");

IOSetItem(TagNameString, "Excel", "R1C1");
```

If an empty string ("") is specified for both the Access Name and item values, then the tagname is deactivated. For example, the tagname "MyTag2" is deactivated by:

```
IOSetItem("MyTag2", "", "");
```

If an empty string is specified only for an Access Name value, then the tagname's current Item value is retained and its Access Name value is updated. For example, the following changes the Access Name for tagname "MyTag3" to "Excel2" without affecting its current Item value:

```
IOSetItem("MyTag3", "Excel2", "");
```

Likewise, if an empty string is specified only for an Item, then the tagname's current Item value is retained and its Access Name value is updated. For example, the following changes the Item for tagname "MyTag3" to "R1C2" without affecting its current Access Name value:

```
IOSetItem("MyTag4", "", "R1C2");
```

For more information on **IOSetItem**, see your online *InTouch Reference Guide.*

Using the .ReferenceComplete to Verify References

Each I/O type tagname has a **.ReferenceComplete** field. This discrete field provides confirmation that the item requested in the reference field is reflected in the **.Value** field.

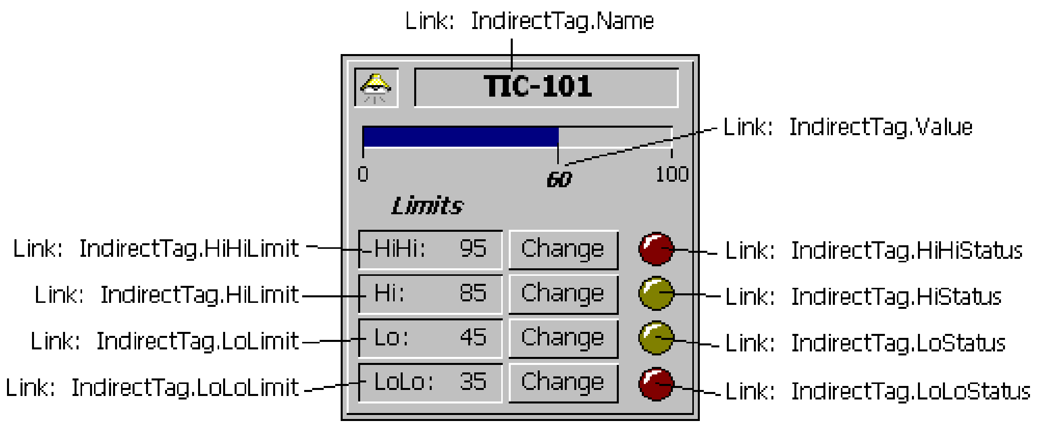
The **.ReferenceComplete** field initializes to false (0) at startup of WindowViewer. When it is confirmed that the **.Value** field is being updated by the source specified in the **.Reference** field, the **.ReferenceComplete** value is set to true (1). If the **.Reference** field is changed, the **.ReferenceComplete** field is automatically set to false (0), and then updated to true(1) when the new value is updated.

For more information on **.ReferenceComplete** see your online *InTouch Reference Guide.*

Using Indirect Tags With Local Tagnames

This section describes how Indirect tagnames are typically used with local Tagname Dictionary references.

Indirect tagnames allow you to create "generic" tagnames that you use with multiple data sets. For example, you may use a generic faceplate for modifying alarm limits that is linked to several local tagnames.


Link: IndirectTag.Name
TIC-101
Link: IndirectTag.Value
0
60
100
Limits
Link: IndirectTag.HiHiLimit
Link: IndirectTag.HiLimit
Link: IndirectTag.LoLimit
Link: IndirectTag.LoLoLimit
HiHi: 95
Hi: 85
Lo: 45
LoLo: 35
Change
Change
Change
Change
Link: IndirectTag.HiHiStatus
Link: IndirectTag.HiStatus
Link: IndirectTag.LoStatus
Link: IndirectTag.LoLoStatus

You may have many different tagnames that use your alarm limit faceplate. To redirect the faceplate to the appropriate tagname, you execute the following QuickScript:

```
IndirectTagname.dotfield = "tagname";
```

Where, *tagname* is an actual tagname defined in the local Tagname Dictionary.

When this script executes, all of the **dotfields** associated with the local tagname become accessible through the Indirect tagname.

For more information on tagname dotfields, see your online *InTouch Reference Guide.*

Using Indirect Tagnames With Remote References

Remote reference tagnames differ from local tagnames in many ways. The syntax for a remote reference is:

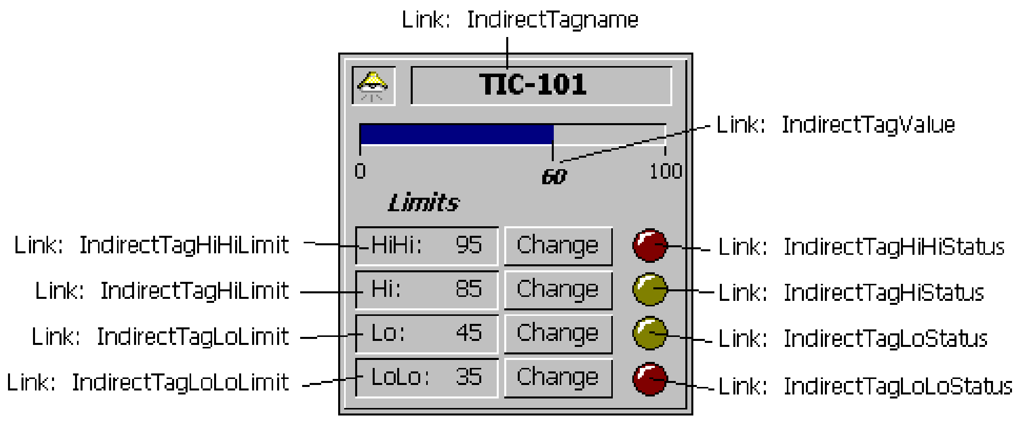
AccessName:Item

Where, ***AccessName*** is any valid InTouch Access Name and ***Item*** is any valid item name that is supported by the I/O Server defined in the Access Name.

For more information on remote references, see "Remote Tagname Referencing"

When you use remote references, the server returns a value to the client, not a tagname structure. The value includes a time stamp and a quality stamp. Thus, an Indirect tagname assigned to a remote reference cannot access any tagname **dotfields** other than those relating to value, time and quality. For example, an Indirect tagname cannot access alarm limits through a remote reference.

To solve this problem, you can create a faceplate with several Indirect tagnames. For example:



Notice that this faceplate uses ten Indirect tagnames that all use the implied **.Value** reference.

For more information on **.Value** see your online *InTouch Reference Guide.*

Let's assume this alarm faceplate is being redirected to the remote reference tagname, TIC-101, on a remote InTouch node named TagServer1. An InTouch Access Name has been configured as follows:

Access Name:	TagServer1
Node Name:	TagServer1
Application Name:	View
Topic Name:	Tagname

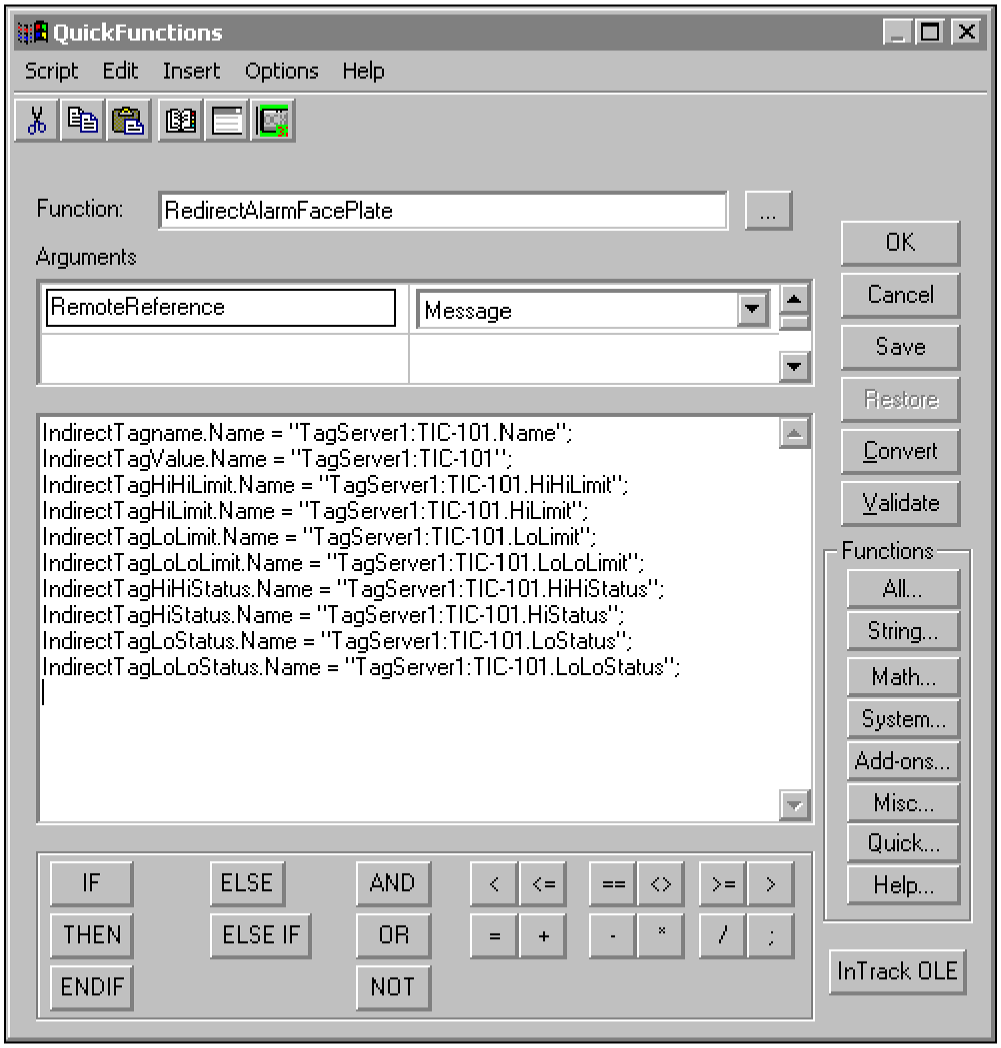
To redirect the faceplate to the remote reference tagname TIC-101, the following QuickScript is executed:

```
IndirectTagname.Name = "TagServer1:TIC-101.Name";
```

```
IndirectTagValue.Name = "TagServer1:TIC-101";

IndirectTagHiHiLimit.Name = "TagServer1:TIC-
   101.HiHiLimit";

IndirectTagHiLimit.Name = "TagServer1:TIC-101.HiLimit";

IndirectTagLoLimit.Name = "TagServer1:TIC-101.LoLimit";

IndirectTagLoLoLimit.Name = "TagServer1:TIC-
   101.LoLoLimit";

IndirectTagHiHiStatus.Name = "TagServer1:TIC-
   101.HiHiStatus";

IndirectTagHiStatus.Name = "TagServer1:TIC-101.HiStatus";

IndirectTagLoStatus.Name = "TagServer1:TIC-101.LoStatus";

IndirectTagLoLoStatus.Name = "TagServer1:TIC-
   101.LoLoStatus";
```

This script must execute each time the faceplate is redirected, which could be tedious and prone to errors. Therefore, a better solution is to create an InTouch QuickFunction that allows you to write a script once and pass it the name of the remote reference.

For example, using the set of script commands above, you could define a QuickFunction called **RedirectAlarmFacePlate()**:



Function: RedirectAlarmFacePlate

Arguments

RemoteReference	Message

```
IndirectTagname.Name = "TagServer1:TIC-101.Name";
IndirectTagValue.Name = "TagServer1:TIC-101";
IndirectTagHiHiLimit.Name = "TagServer1:TIC-101.HiHiLimit";
IndirectTagHiLimit.Name = "TagServer1:TIC-101.HiLimit";
IndirectTagLoLimit.Name = "TagServer1:TIC-101.LoLimit";
IndirectTagLoLoLimit.Name = "TagServer1:TIC-101.LoLoLimit";
IndirectTagHiHiStatus.Name = "TagServer1:TIC-101.HiHiStatus";
IndirectTagHiStatus.Name = "TagServer1:TIC-101.HiStatus";
IndirectTagLoStatus.Name = "TagServer1:TIC-101.LoStatus";
IndirectTagLoLoStatus.Name = "TagServer1:TIC-101.LoLoStatus";
```

You can now call just the QuickFunction, **RedirectAlarmFacePlate(),** to handle the entire redirection. To do this, QuickFunction must be called by another InTouch QuickScript. For example:

```
CALL RedirectAlarmFacePlate ("TagServer1:TIC-101");
```

Generally, you should develop your new applications using remote references exclusively. (It is possible to remote reference a local Tagname Dictionary in InTouch.) This abstracts the data source (local or remote) from the graphics. By using remote references exclusively in your InTouch application, you will significantly reduce your application maintenance when you later add new windows or graphical objects to your distributed FactorySuite system.

The Tag Browser

The Tag Browser is your primary tool for viewing and selecting local and remote tagnames and tagname **dotfields** from FactorySuite applications, or any other tag source that supports the InTouch Tagname Dictionary interface. It allows you to select existing tagnames, add new tagnames and view basic Tagname Dictionary information. You also use the Tag Browser to access the dialog boxes that allow you to perform tagname editing, replication, and to select tagnames (remote references) in remote tag sources.

The first time you access the Tag Browser, by default, **<local>** will be selected for the tag source. Meaning that the tagnames in the local application's Tagname Dictionary will be displayed. Thereafter, the last accessed tag source's tagnames will be displayed.

The Tag Browser operates in two modes; "Filtered Selection Mode" and "Unlimited Selection Mode." The mode for the Tag Browser is determined by the method you use to access it. The following lists the primary methods that you can use to access the Tag Browser in each mode:

Unlimited Selection Mode

- Double-clicking an animation link tagname or expression input box.
- Double-clicking an ActiveX or wizard tagname or expression input box.
- Double-clicking a blank area in any InTouch QuickScript window.
- In the InTouch QuickScript editor, selecting the **Tagname** command on the **Insert** menu.
- Pressing the **Alt+N** keys in the InTouch QuickScript editor.
- Double-clicking a blank **New Name** box in the **Substitute Tagnames** dialog box.
- Double-clicking the **TagnamedotfieldsName** input box in the SQL Access **Bind List Configuration** dialog box.

Filtered Selection Mode

- Clicking the **Select** button in the Tagname Dictionary.
- When WindowMaker is running, double-clicking a cell in the **Unit#** column in a Recipe Manager **Unit** definition.
- In runtime, clicking any Pen# button in the Historical Trend Setup dialog box. In this instance, the Tag Browser will only display the tagnames that are defined with the Log Data option selected in the Tagname Dictionary.

  Tip This functionality is only supported when the **Allow Runtime Changes** option has been selected for the historical trend during development.

- In runtime, clicking any object linked to the **HTSelectTag()** function.

For more information on the Tag Browser modes, see "Tag Browser Selection Modes."

The Tag Browser's status bar provides status on the following items for the currently displayed tag source:

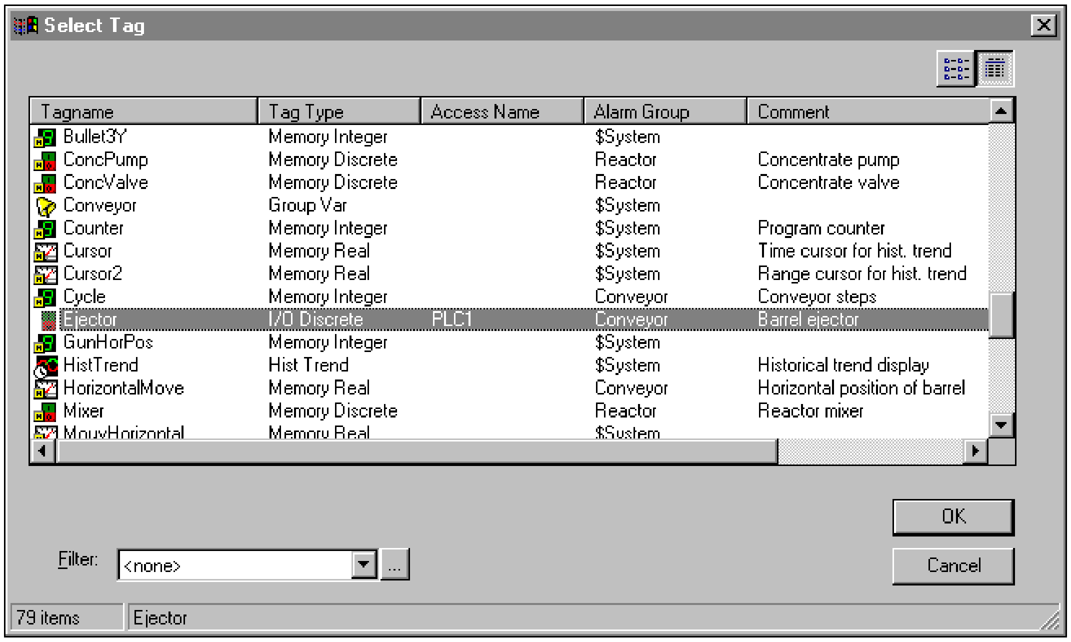
- Total number of items in the application.
- The name of the currently selected item.
- Tagname **dotfields** selected, if any.
- The Access Name associated with the tag source.

Tag Browser Selection Modes

The Tag Browser operates in two selection modes: Filtered Selection Mode and Unlimited Selection Mode.

Filtered Selection Mode

This mode is activated when you click **Select** in the **Tagname Dictionary** dialog box or during runtime (when the operator is allowed to make runtime changes to a historical trend) when selecting a new tagname for a historical trend pen. The tagnames displayed (and available for selecting) will be limited to the current InTouch application. For example:



Tagname	Tag Type	Access Name	Alarm Group	Comment
Bullet3Y	Memory Integer		$System	
ConcPump	Memory Discrete		Reactor	Concentrate pump
ConcValve	Memory Discrete		Reactor	Concentrate valve
Conveyor	Group Var		$System	
Counter	Memory Integer		$System	Program counter
Cursor	Memory Real		$System	Time cursor for hist. trend
Cursor2	Memory Real		$System	Range cursor for hist. trend
Cycle	Memory Integer		Conveyor	Conveyor steps
Ejector	I/O Discrete	PLC1	Conveyor	Barrel ejector
GunHorPos	Memory Integer		$System	
HistTrend	Hist Trend		$System	Historical trend display
HorizontalMove	Memory Real		Conveyor	Horizontal position of barrel
Mixer	Memory Discrete		Reactor	Reactor mixer
MouvHorizontal	Memory Real		$System	

Filter: <none>

79 items | Ejector

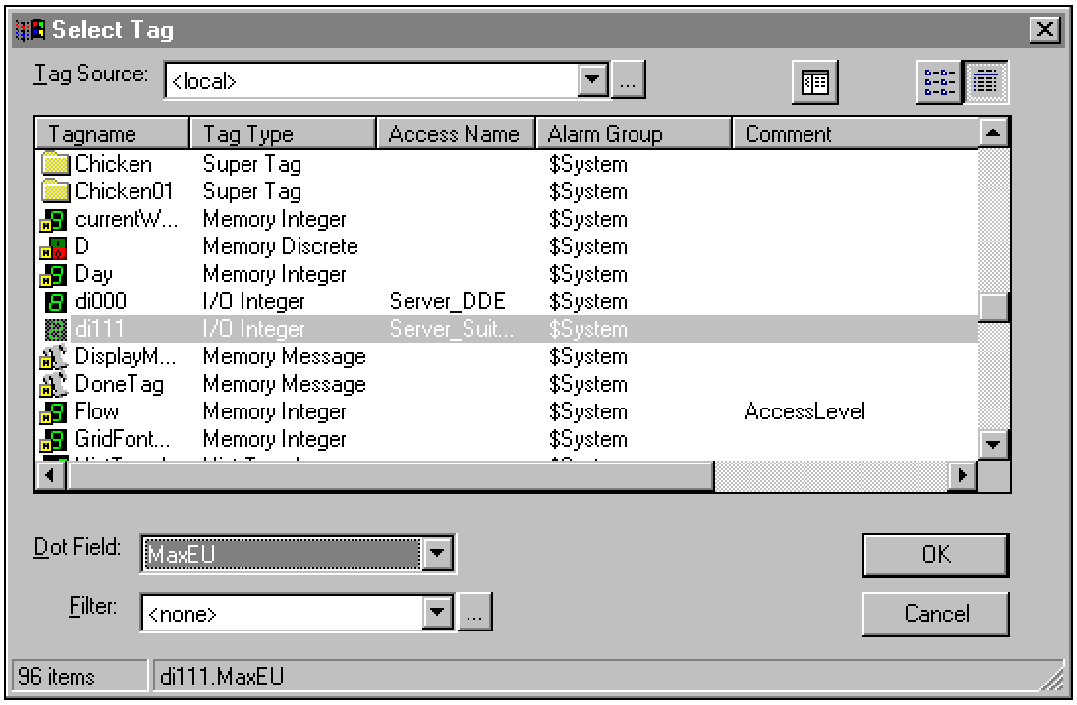
Tip When you access the Tag Browser from the Tagname Dictionary and select a tagname in this view, its Tagname Dictionary definition appears after you click **OK**.

Note Tagname **dotfields** cannot be selected in this mode.

Unlimited Selection Mode

The unlimited selection mode is accessed by double-clicking in a blank area in any InTouch QuickScript window, animation link tagname or expression box or, a blank **New Name** box in the **Substitute Tagnames** dialog box. The tagnames defined in a local or remote tag source can be displayed and selected in this mode.

Tagname **dotfields** can also be selected for the tagname in this mode. When you select a tagname and/or tagname**dotfield** in this mode, it is automatically entered into the InTouch QuickScript, animation link tagname or expression box or, other location from which you accessed the Tag Browser. For example:



Tagname	Tag Type	Access Name	Alarm Group	Comment
Chicken	Super Tag		$System	
Chicken01	Super Tag		$System	
currentW...	Memory Integer		$System	
D	Memory Discrete		$System	
Day	Memory Integer		$System	
di000	I/O Integer	Server_DDE	$System	
di111	I/O Integer	Server_Suit...	$System	
DisplayM...	Memory Message		$System	
DoneTag	Memory Message		$System	
Flow	Memory Integer		$System	AccessLevel
GridFont...	Memory Integer		$System	

Dot Field: MaxEU

Filter: <none>

96 items | di111.MaxEU

To select a dotfields

1. Click the **Dot Field** arrow to open the list of **dotfields** that you can associate with the type of tagname currently selected.

Tip By default, **<none>** will initially be displayed for all types of tagnames.

Note Dot Field is not available when you access the Tag Browser from the Tagname Dictionary or, during runtime, when selecting a tagname for a historical trend pen from the Historical Trend Setup dialog box. (The historical trend must be configured with the Allow runtime changes option selected.)

2. Click the **dotfields** in the list that you want to append to the selected tagname.

Tip Not every tagname type has the same **dotfields**. For example, a **Discrete** type tagname has **.OnMessage**, whereas an analog does not. If you select a **Discrete** type tagname and you assign **.OnMessage** to it, and then you select another **Discrete** type tagname, the displayed **dotfields** list will not change. But, if you select another type of tagname in the control view list, for example an analog, the displayed **dotfields** will revert to **<none>**.

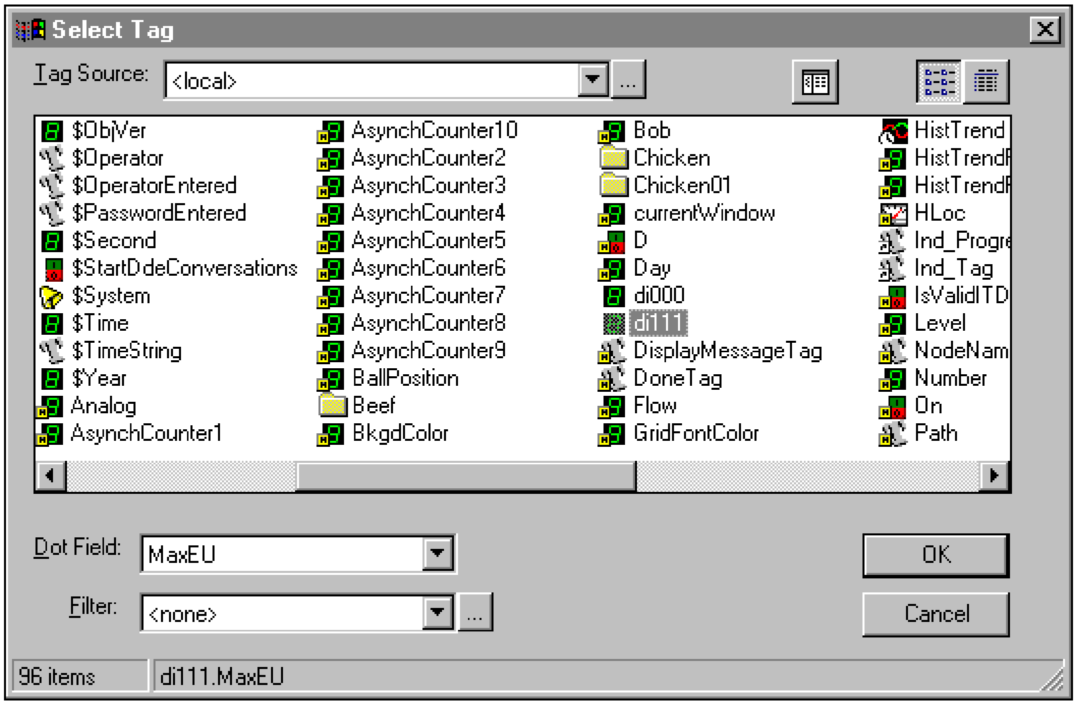
Tag Browser Views

The Tag Browser supports three control views; Tagname List Control, Tagname Details Control and Tagname Tree View Control.

List View

The list view is used to display and select tagnames within the current selection mode (described above). The Tagname List Control view displays the tagnames in two views depending upon the state of the **List View** and **Details View** buttons:

When you select list view, small icons will be displayed next to the tagnames with icons displayed according to the type of each tagname. No other fields will be displayed in the list view. For example:



Tag Source: <local>

$ObjVer	AsynchCounter10	Bob	HistTrend
$Operator	AsynchCounter2	Chicken	HistTrend
$OperatorEntered	AsynchCounter3	Chicken01	HistTrend
$PasswordEntered	AsynchCounter4	currentWindow	HLoc
$Second	AsynchCounter5	D	Ind_Progr
$StartDdeConversations	AsynchCounter6	Day	Ind_Tag
$System	AsynchCounter7	di000	IsValidTD
$Time	AsynchCounter8	di111	Level
$TimeString	AsynchCounter9	DisplayMessageTag	NodeNam
$Year	BallPosition	DoneTag	Number
Analog	Beef	Flow	On
AsynchCounter1	BkgdColor	GridFontColor	Path

Dot Field: MaxEU

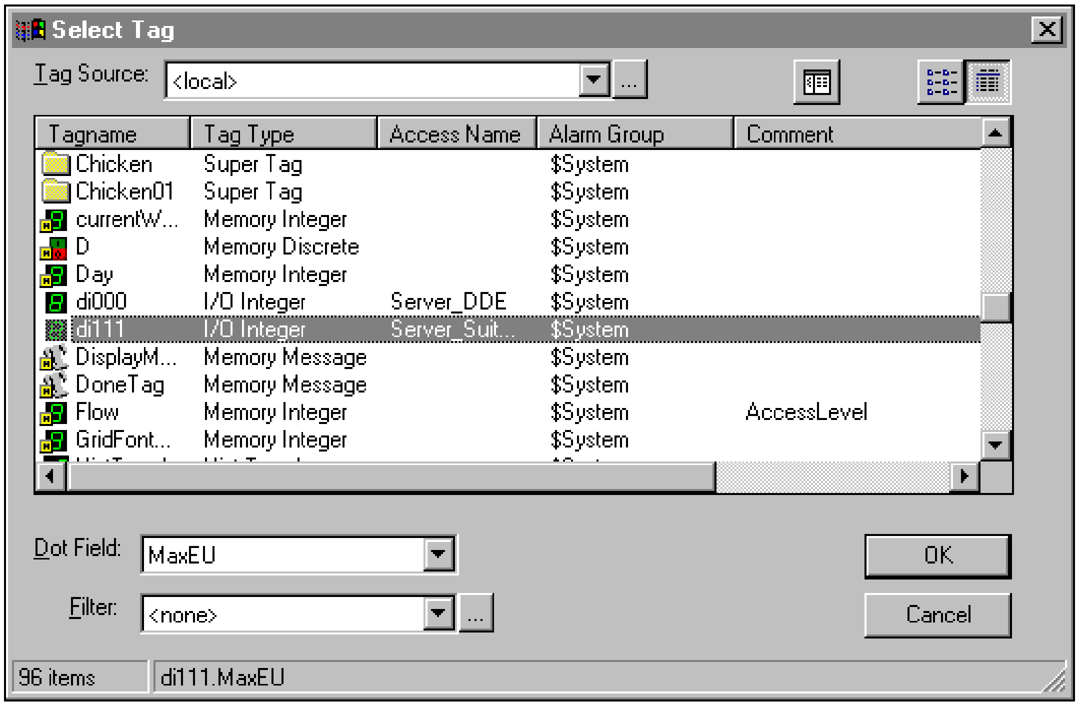
Filter: <none>

96 items di111.MaxEU

Tip To refresh the display in the Select Tag dialog box, press F5.

Details View

When you select details view, the tagnames and their details are displayed in a multi-column format. The details displayed are Tagname Name, Tagname Type, Access Name, Alarm Group and Comment. You can sort the list by each detail type by clicking on its column header name. An item can be selected by clicking on any portion of its display, not just the tagname. (The entire row will be highlighted.) For example:



Tag Source: <local>

Tagname	Tag Type	Access Name	Alarm Group	Comment
Chicken	Super Tag		$System	
Chicken01	Super Tag		$System	
currentW...	Memory Integer		$System	
D	Memory Discrete		$System	
Day	Memory Integer		$System	
di000	I/O Integer	Server_DDE	$System	
di111	I/O Integer	Server_Suit...	$System	
DisplayM...	Memory Message		$System	
DoneTag	Memory Message		$System	
Flow	Memory Integer		$System	AccessLevel
GridFont...	Memory Integer		$System	

Dot Field: MaxEU

Filter: <none>

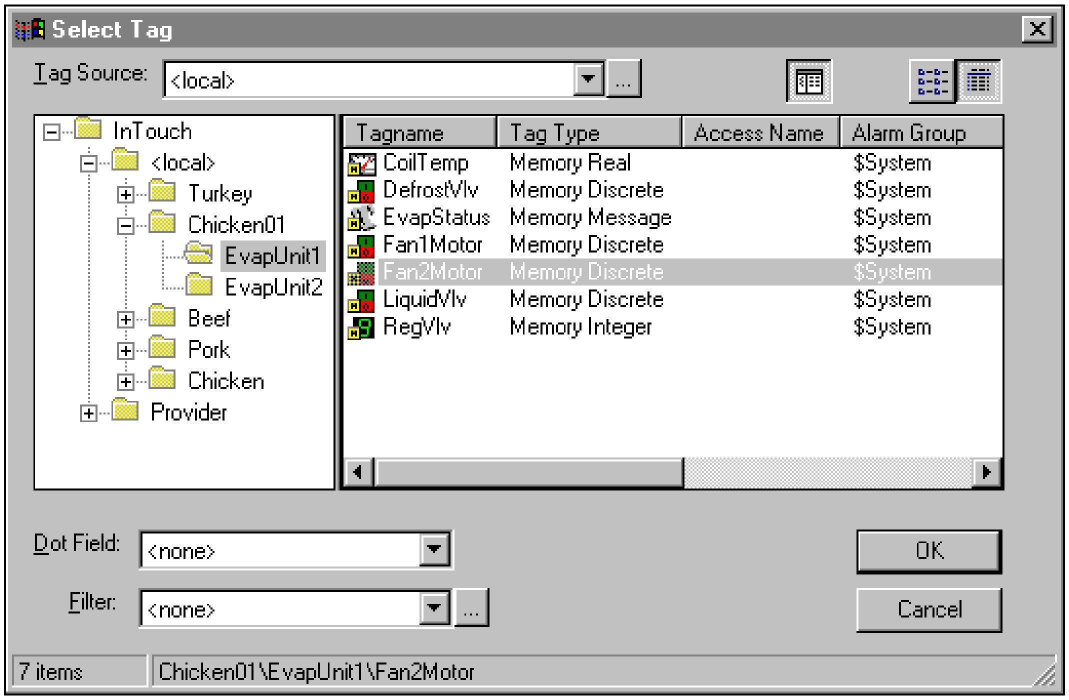
96 items di111.MaxEU

Tip When you switch views, the selected tagname will remain visible and highlighted in the new view.

Tree View

The Tree View displays the tagnames in two views depending upon the state of the **List View** and **Details View** buttons. When you select the tree view, a pane appears on the left side of the dialog box. By using the Tree View you can also access the member tagnames in any SuperTag template.

If the **Details View** mode is active when you select the Tree View, Tag Browser appear as follows:



Tagname	Tag Type	Access Name	Alarm Group
CoilTemp	Memory Real		$System
DefrostVlv	Memory Discrete		$System
EvapStatus	Memory Message		$System
Fan1Motor	Memory Discrete		$System
Fan2Motor	Memory Discrete		$System
LiquidVlv	Memory Discrete		$System
RegVlv	Memory Integer		$System

7 items | Chicken01\EvapUnit1\Fan2Motor

Tip To expand a listing in the Tree View, double-click the application name or, click the ⊞ . To collapse a listing, double-click the application name again or, click the ⊟. Double-clicking an application in the tree view pane is the same as selecting it in the **Tag Source** list.

Note When you "drill down" through different levels in the Tag Browser, you can use the BACKSPACE key to "back up" to the previous level.

Defining Tag Sources

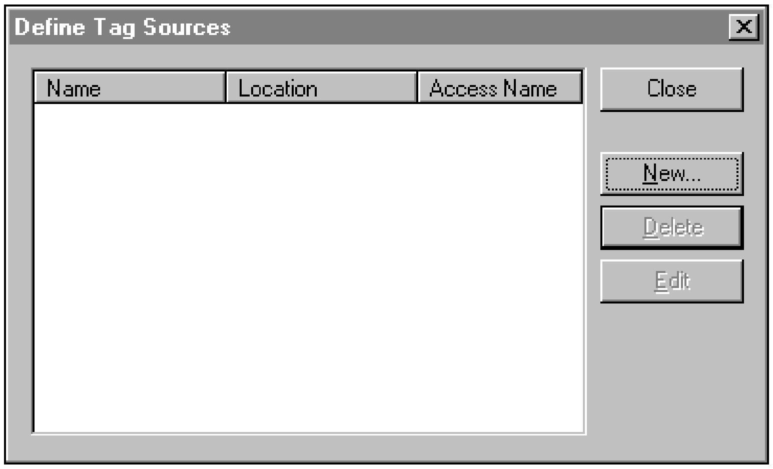
You must define the tag sources for viewing in the Tag Browser. The procedures for adding, deleting or editing tag sources are described in this section. When you add or edit a tag source definition, you will enter information such as the local network Access Name you want to associate with the tag source's tagnames, a user-defined application name, and the data source for the tag source.

Note You will also use these procedures when you are converting placeholder tagnames to remote tagname references.

For more information on remote tagname references, see "Converting Tagnames to Remote References."

To define a tag source

1. Open the Tag Browser, and then click the Define Tag Sources button. The **Define Tag Sources** dialog box appears.



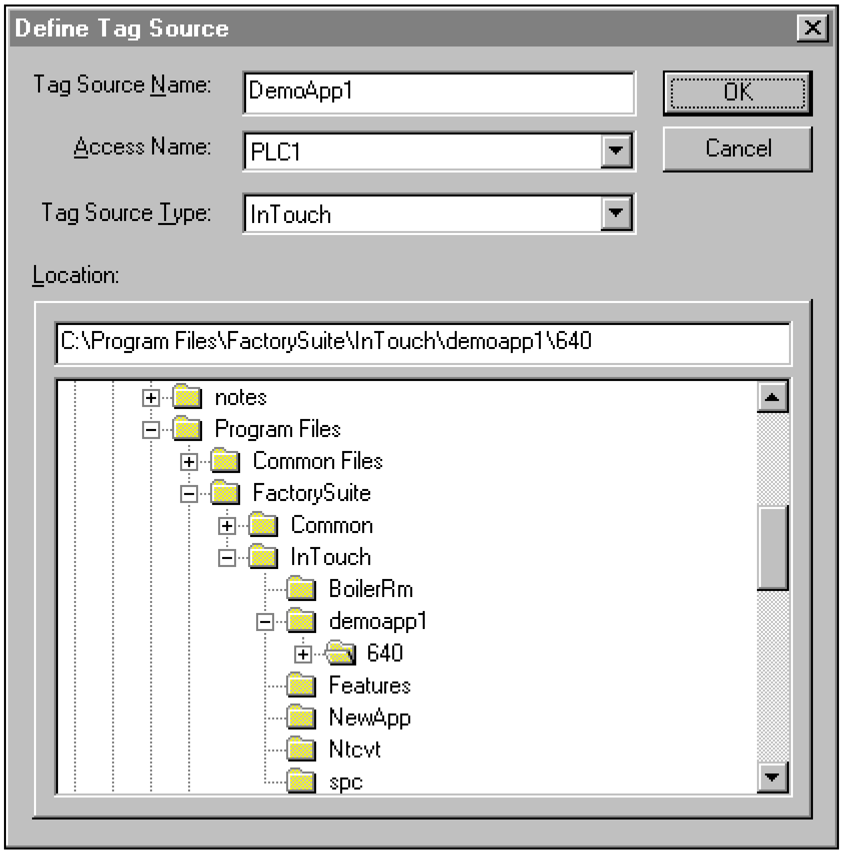
Note If tag sources are already defined, they will be listed when the dialog box appears. The list will include the user-defined Name for the tag source, the Location of the tag source (path) and the local network Access Name associated with the application.

Tip To select multiple tag sources, hold down the SHIFT key as you click each name. To select multiple tag sources that are not consecutive in the list, hold down the CTRL key as you click each name.

Note When the **Define Tag Sources** dialog box closes, you must click the **Tag Source** arrow in the Tag Browser and select the new tag source in the list. The Tag Browser is then refreshed and tagnames for the selected tag source are displayed.

2. To remove a tag source(s) from the **Tag Source** list in the Tag Browser, click the Define Tag Sources button. The **Define Tag Sources** dialog box appears. Select the tag source in the list, and then click **Delete**.
3. To edit a defined tag source, select it in the list, and then click **Edit**. The **Define Tag Source** dialog box appears displaying the configuration for the selected tag source.
4. To define a new tag source, click **New**. The **Define Tag Source** dialog box appears.

Note When you click New, if no Access Name is defined in your local application, a message box appears telling you there are no Access Names defined and you will not be allowed to define a new tag source. (Tag sources must be associated with a local network Access Name.)



Tag Source Name: DemoApp1

Access Name: PLC1

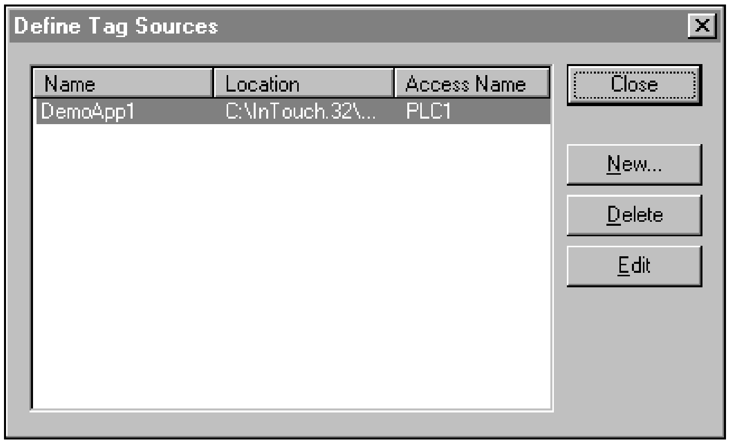
Tag Source Type: InTouch

Location:

C:\Program Files\FactorySuite\InTouch\demoapp1\640

5. In the **Tag Source Name** box, type a name to identify the tag source.
6. Click the **Access Name** arrow and select the Access Name in the local application that you want to associate with the tagnames in the tag source.
7. Click the **Tag Source Type** arrow and select the source for the tag source's tagname database. (By default, **InTouch** is displayed.)
8. The **Location** box displays the full path to the tag source.

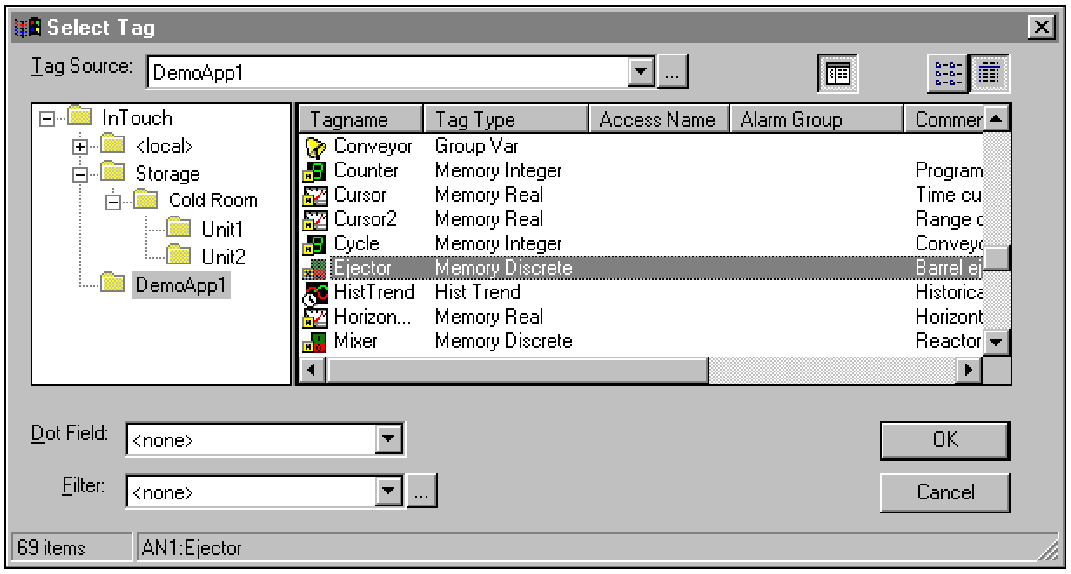
9. In the directory tree pane, locate the tag source, and then click **OK**. The **Define Tag Sources** dialog box reappears displaying the selected tag source:



Name	Location	Access Name
DemoApp1	C:\InTouch.32\...	PLC1

10. Click **Close**. The Tag Browser reappears.

11. Click the tool to display the tree view pane to display all defined tag sources:



Tag Source: DemoApp1

Tagname	Tag Type	Access Name	Alarm Group	Commer
Conveyor	Group Var			
Counter	Memory Integer			Program
Cursor	Memory Real			Time cu
Cursor2	Memory Real			Range c
Cycle	Memory Integer			Conveyo
Ejector	Memory Discrete			Barrel ej
HistTrend	Hist Trend			Historica
Horizon...	Memory Real			Horizont
Mixer	Memory Discrete			Reactor

Dot Field: <none>

Filter: <none>

69 items AN1:Ejector

Tip If you are not using the tree view mode, click the **Tag Source** arrow and select the name for the tag source that you want to display in the list. The Tag Browser will refresh and the tag sources' tagnames will be displayed.

Note The first time you access the Tag Browser, by default, **<local>** will be selected for the **Tag Source**. Thereafter, the tagnames for previously accessed tag source will be displayed.

12. Click **OK**.

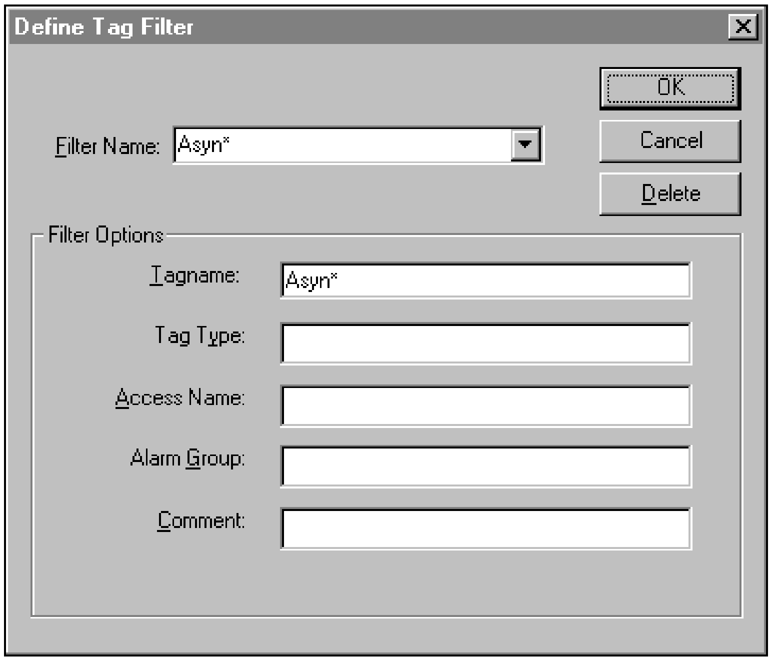
Defining Tag Browser Filters

You will use the procedures described in this section to define the filters (search criteria) you want to use to populate the Tag Browser. By creating filters, you can sort any tagname list and display only the tagnames that meet the criteria you specify. You can sort the tagnames based on **Tagname**, **Tagname Type**, **Access Name**, **Alarm Groups** and tagname **Comments**. You can use one or a combination of any of these items to set the criteria for your display. You can also save each filter instance and reuse it at any time.

Tip For example, if you have 40,000 tagnames defined in your Tagname Dictionary and you only need to deal with the 20 or so that are assigned to a particular Access Name or Alarm Group, you can create a filter and specify the Access Name and/or Alarm Group as the criteria that the tagnames must meet in order to be displayed in the Tag Browser.

To define a search filter

1. Click the Define Filter button. The **Define Tag Filter** dialog box appears.


Define Tag Filter

Filter Name: Asyn*

Filter Options

Tagname: Asyn*

Tag Type:

Access Name:

Alarm Group:

Comment:

Tip If you right click the mouse in any of the text entry boxes, a menu appears displaying the commands that you can apply to the selected text.

2. In the **Filter Name** box, type a unique name to identify the filter that you are defining, or click the **Filter Name** arrow to select a previously defined filter name from the list. (As you define filters, the **Filter Name** you type is added to the list.)

Tip All of the **Filter Option** controls (**Tagname**, **Tagname Type**, **Access Name**, **Alarm Group** and **Comment**) allow you to enter a wildcard expression to limit the scope of your search. If no filter is used, all of the tagnames in the currently displayed tag source will be displayed.

The multiple wildcard is the asterisk (*). For example, "Asyn*" would search for all tagnames beginning with the character "Asyn".

The single character wildcard is the question mark (?). For example, the filter, "Tag?" would search for all four character tagnames that begin with "Tag". The filter, "Tag??", would search for all five character tagnames that begin with "Tag", and so on.

Any sequence of valid InTouch tagname characters, together with the two wild card characters, is acceptable in a filter. The valid tagname characters are: A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &.

3. In the **Tagname** box, type the tagname expression that you want to use as a filter. If left blank, the system will ignore this field in the filter definition.
4. In the **Access Name** box, type the local Access Name expression that you want to use as a filter. If left blank, the system will ignore this field in the filter definition.
5. In the **Alarm Group** box, type the name of the Alarm Group expression that you want to use as a filter. If left blank, the system will ignore this field in the filter definition.
6. In the **Comment** box, type the comment expression you want to use as a filter. If left blank, the system will ignore this field in the filter definition.
7. Click **OK** to close dialog box.

Tip The **Filter Name** will now appear in the **Filter** list in the Tag Browser and you can select it to display only the tagnames meeting the criteria specified in the filter.

To delete a search filter

1. Click the **Filter** arrow and select the filter name in the list that you want to delete.
2. Click **Delete**. The filter is immediately deleted.

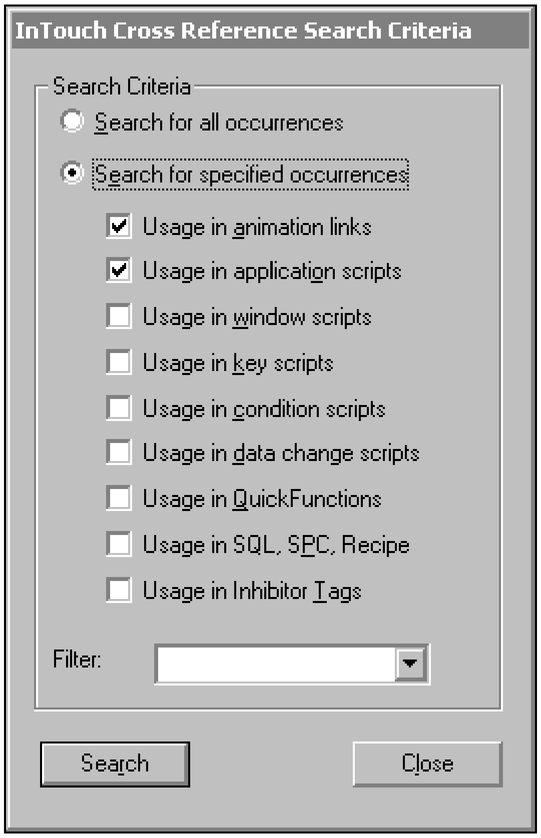
InTouch Cross Reference Utility

The Tagname Cross Referencing utility allows you to determine your tagname, remote tagname and SuperTag usage in animation links, wizards, InTouch QuickScripts, QuickFunctions, ActiveX controls, scripts and the following InTouch add-on programs: SPC Pro, SQL Access Manager, and Recipe Manager. For all objects such as wizards, ActiveX controls and animation links, it displays the window name and the coordinates of all objects linked to the tagname. It also allows you to view any QuickScript or QuickFunction where a tagname is found.

Tip For convenience, the Tagname Cross Reference utility can remain open in WindowMaker while you perform other tasks.

To use the InTouch Cross Reference utility

1. On the **Special** menu, click **Cross Reference** or, in the Application Explorer double-click **Cross Reference**. The **InTouch Cross Reference Search Criteria** dialog box appears.



InTouch Cross Reference Search Criteria

Search Criteria

- ○ Search for all occurrences
- ● Search for specified occurrences
  - ☑ Usage in animation links
  - ☑ Usage in application scripts
  - ☐ Usage in window scripts
  - ☐ Usage in key scripts
  - ☐ Usage in condition scripts
  - ☐ Usage in data change scripts
  - ☐ Usage in QuickFunctions
  - ☐ Usage in SQL, SPC, Recipe
  - ☐ Usage in Inhibitor Tags

Filter:

2. The **Search Criteria** group allows you to limit the scope of your search. You can easily determine the scope by selecting only the options required.

Search for all occurrences	Search for all uses of the tagname or SuperTag in animation links, InTouch QuickScripts and all add-on programs such as SPC, SQL Access Manager, Recipe Manager, and so on.
Search for specific occurrences	Search for only the tagname or SuperTag only in the specified options. For example, if you only want to search for the usage in window scripts, only select Usage in window scripts.

3. In the **Filter** box, type a unique name to identify the filter that you are defining or, click the **Filter** arrow to select a previously defined filter from the list. (As you define filters, the name you type is added to the **Filter** list.)

Tip The filter editor control allows you to enter a wildcard expression to limit the scope of the tagnames in your search. If no filter is used, the information for all tagnames in the current application will be acquired.

The multiple wildcard is the asterisk symbol (*****). For example, "Asyn*" would search for all tagnames beginning with the characters "Asyn".

The single character wildcard is the question mark symbol (**?**). For example, the filter, "Tag?" would search for all four character tagnames that begin with "Tag". The filter, "Tag??", would search for all five character tagnames that begin with "Tag", and so on.

Any sequence of valid InTouch tagname characters, together with the two wildcard characters, is acceptable in a filter. The valid tagname characters are: A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &.

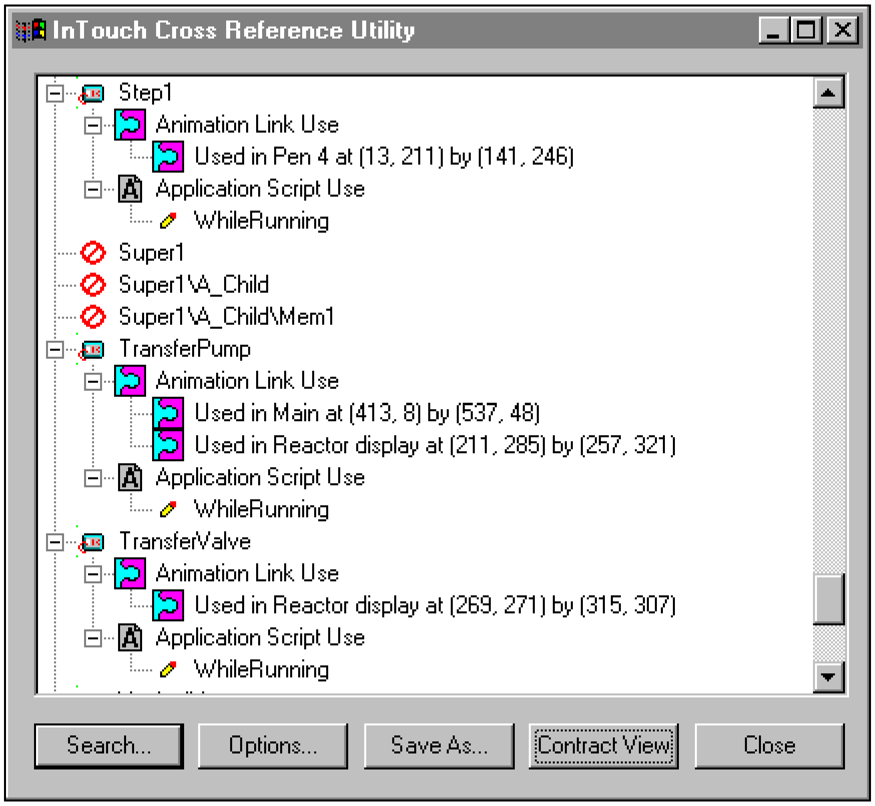
If you right click the mouse in the **Filter** box, a menu appears displaying the commands that you can apply to the selected text.

4. Click **Search** to begin the cross-reference search based upon your specified view criteria.

Viewing the Cross Reference Search Results

When you perform a cross-reference search, the **InTouch Cross Reference Utility** dialog box appears listing all instances of usage found for the **Filter** that you specified.

If no filter is used, all tagnames defined in the current application's Tagname Dictionary are displayed. For example:



```
Step1
    Animation Link Use
        Used in Pen 4 at (13, 211) by (141, 246)
    Application Script Use
        WhileRunning
Super1
Super1\A_Child
Super1\A_Child\Mem1
TransferPump
    Animation Link Use
        Used in Main at (413, 8) by (537, 48)
        Used in Reactor display at (211, 285) by (257, 321)
    Application Script Use
        WhileRunning
TransferValve
    Animation Link Use
        Used in Reactor display at (269, 271) by (315, 307)
    Application Script Use
        WhileRunning
```

Cross Reference Utility Icons

The following briefly describes the various icons that may appear in the InTouch Cross Reference Utility:

Icon Name	Icon	Description
Expand Level View	⊞	Tagname or SuperTag is assigned to an InTouch object, or used to store a value in an InTouch QuickScript, wizard, or add-on program. Click to expand the level view.
Collapse or Expand Level View	⊟	Click to collapse an expanded level view.

Icon Name	Icon	Description
Not Assigned to an Object	⊘	Displayed tagname or SuperTag is defined in the application's Tagname Dictionary, but it is not assigned to an object.
Double-click to Expand View	▭	Displayed tagname or SuperTag is used in either an animation link or InTouch QuickScript. Double-click or click ⊞ to expand the view.
Double-click to Display Window Name	▣	Displayed tagname or SuperTag is assigned to an animation link. Double-click or click ⊞ to display the window name and the coordinates for object(s) in the window assigned to the animation link.
Double-click to Expand View and Display Type	🄰	Displayed tagname or SuperTag is used in an Application script. Double-click or click ⊞ to expand the view and display the type of Application script that uses the tagname or SuperTag.
Displayed for All	✎	Displayed for all Application **On Startup**, **While Running**, and **On Shutdown** scripts; Window **On Show**, **While Showing**, and **On Hide** scripts; and Key **On Key Down**, **While Down**, and **On Key Up** scripts. Double-click the script to view it.
Double- click to Display Window Name and Script	▭	Displayed tagname or SuperTag is used in a Window script. Double-click or click ⊞ to expand the view to display the name of the window with the script. Double-click any listed window name to view the script.
Double-click to Expand View of Data Change Script	≠	Displayed tagname or SuperTag is used in a Data Change script. Double-click or click ⊞ to expand the view, and then double-click any listed script to view it.
Double-click to Expand the View to Display the Script's Condition/Type	?	Displayed tagname or SuperTag is used in a Condition script. Double-click or click ⊞ to expand the view to display the script's condition and its type. For example, **$Hour==12 On True**. Double-click any listed script to view it.

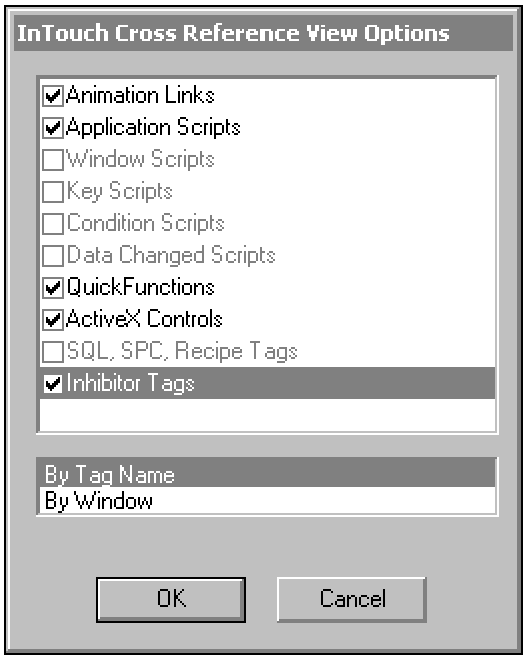
Icon Name	Icon	Description
Double-click to Expand View and Display Key Assigned to Script/Script Type		Displayed tagname or SuperTag is used in a Key script. Double-click or click ⊞ to expand the view and display the key assigned to the script and the script type. For example, **F2 On Key Down**. Double-click any listed script to view it.
Double-click to Expand the View and Display the QuickFunction		Displayed tagname or SuperTag is used in a QuickFunction. Double-click or click ⊞ to expand the view and display the QuickFunction that uses the tagname or SuperTag. Click to expand the view to display the name(s) of the QuickFunctions in which the tagname or SuperTag is used. Double-click any listed script to view it.
Double-click to expand the view and display the ActiveX Event script		Displayed tagname or SuperTag is used in an ActiveX Event script. Double-click or click ⊞ to expand the view and display the ActiveX Event script.
Used when Cross-referencing by Window		When cross-referencing by **Window**, this icon precedes the window name in which the displayed tagname or SuperTag is used. Double-click or click ⊞ to view all tagnames used in the window.
Displayed tagname or SuperTag is used in a SPC Pro application		Displayed tagname or SuperTag is used in a SPC Pro application. Double-click or click ⊞ to view the name of the SPC Dataset in which the tagname or SuperTag is used.
Displayed tagname or SuperTag is used in a SQL application		Displayed tagname or SuperTag is used in a SQL application. Double-click or click ⊞ to view the name of the SQL Bind List in which the tagname or SuperTag is used.
Displayed tagname or SuperTag is used in a Recipe Manager application		Displayed tagname or SuperTag is used in a Recipe Manager application.
Displayed tag is used as an alarm inhibitor.		Displayed tag is used as an alarm inhibitor. Double-click or click ⊞ to view the names of all tags for which it is an inhibitor.

Changing the Cross Reference Search Criteria

If desired, after you have performed your initial cross-reference search, you can narrow your search by modifying your original search options.

To change the search options

1. In the **InTouch Cross Reference Utility** dialog box, (displayed after you have performed your initial search), click **Options**. The **InTouch Cross Reference View Options** dialog box appears.



InTouch Cross Reference View Options

- [✓] Animation Links
- [✓] Application Scripts
- [ ] Window Scripts
- [ ] Key Scripts
- [ ] Condition Scripts
- [ ] Data Changed Scripts
- [✓] QuickFunctions
- [✓] ActiveX Controls
- [ ] SQL, SPC, Recipe Tags
- [✓] Inhibitor Tags

By Tag Name
By Window

2. Select the search criteria options that you want to modify for your new search.

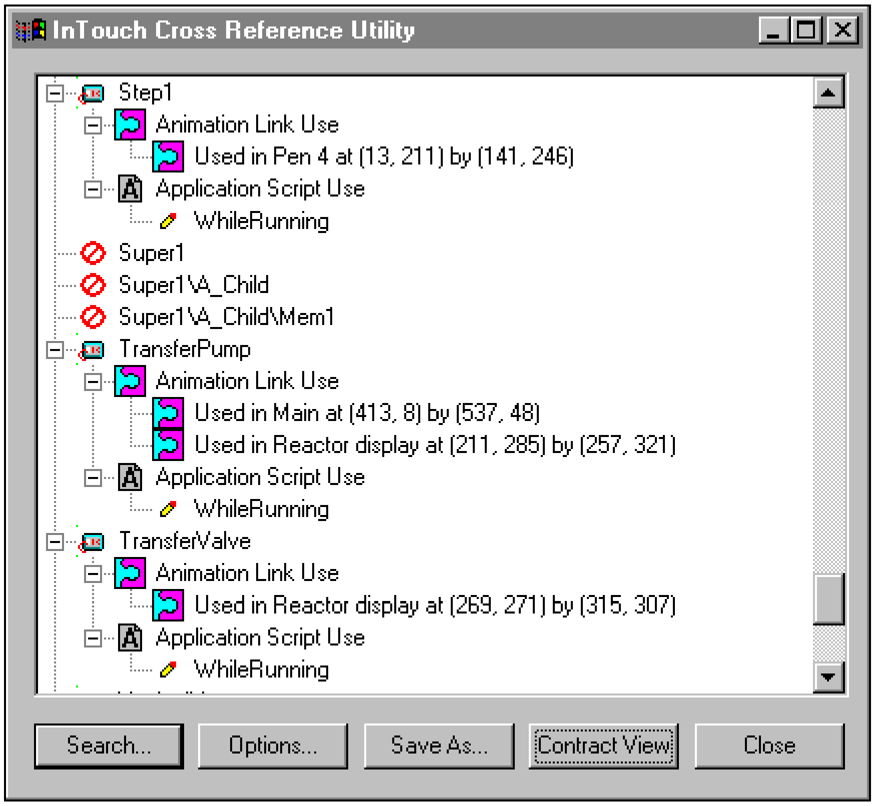
Tip The options available here are based upon the **Search Criteria** you originally selected in the **InTouch Cross Reference Search Criteria** dialog box. If you selected **Search for all occurrences**, all search criteria options will be available. If you selected **Search for specific occurrences**, only the specific occurrences you originally selected will be available. To change your **Search Criteria** selection, click **Cancel**. The **InTouch Cross Reference Utility** dialog box reappears. Click **Search**, and select the new **Search Criteria** option.

3. In the list at the bottom of the dialog box, select whether you want the tree view populated by tagname or window name, and then click **OK**.

Cross Referencing by Tagname

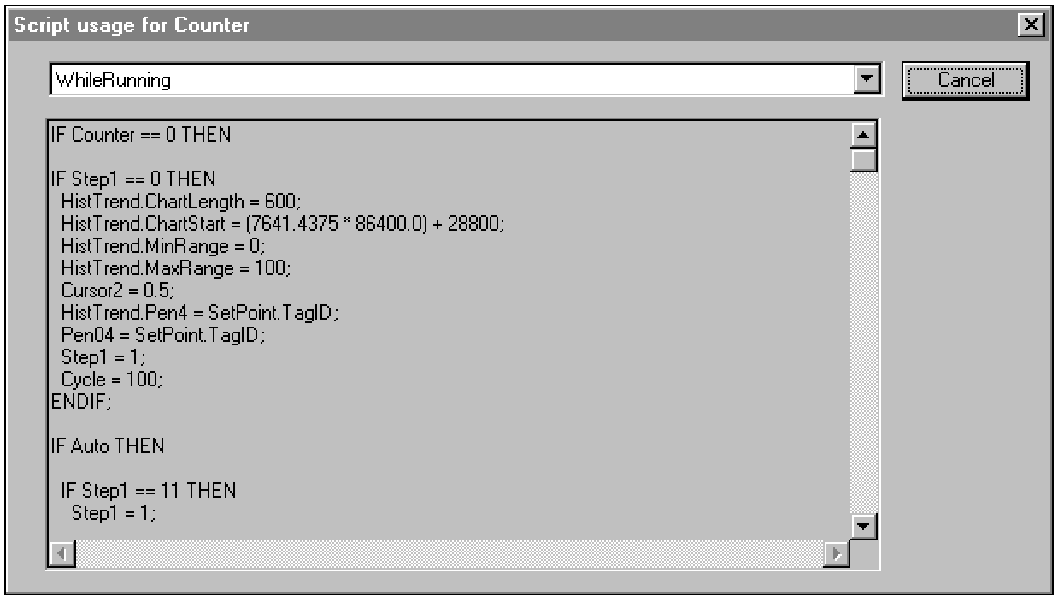
Alphabetically lists all tagnames found for your specified search criteria (default view). Based upon your specified search criteria, this view allows you to view the usage of all tagname found in windows, animation links, scripts and add-on applications.

Tip You can double-click a displayed tagname, and then double-click **Animation Link Use** to expand the view. When you expand the view, the window name and the location (coordinates) of the object(s) linked to the tagname are displayed. For example:



```
Step1
    Animation Link Use
        Used in Pen 4 at (13, 211) by (141, 246)
    Application Script Use
        WhileRunning
Super1
Super1\A_Child
Super1\A_Child\Mem1
TransferPump
    Animation Link Use
        Used in Main at (413, 8) by (537, 48)
        Used in Reactor display at (211, 285) by (257, 321)
    Application Script Use
        WhileRunning
TransferValve
    Animation Link Use
        Used in Reactor display at (269, 271) by (315, 307)
    Application Script Use
        WhileRunning
```

Additionally, you can double-click a tagname, and then double-click any of its associated scripts to open it in the **Script usage for *Tagname*** dialog box:



```
IF Counter == 0 THEN

IF Step1 == 0 THEN
  HistTrend.ChartLength = 600;
  HistTrend.ChartStart = (7641.4375 * 86400.0) + 28800;
  HistTrend.MinRange = 0;
  HistTrend.MaxRange = 100;
  Cursor2 = 0.5;
  HistTrend.Pen4 = SetPoint.TagID;
  Pen04 = SetPoint.TagID;
  Step1 = 1;
  Cycle = 100;
ENDIF;

IF Auto THEN

  IF Step1 == 11 THEN
    Step1 = 1;
```

The list box at the top of the screen shows all scripts associated with the selected tagname. Click the arrow to open the list to select another script for viewing. For Application, Window, Key and Condition scripts, the list will contain the names of all scripts that use this tagname. For Data Change scripts, only the tagname is listed. For QuickFunctions, the list will contain the names of all the QuickFunctions.

Note This is a read only view of the QuickScript. You cannot edit the QuickScript text in this dialog box. However, you can copy any portion or all of the QuickScript, and then paste it into any InTouch QuickScript editor window.

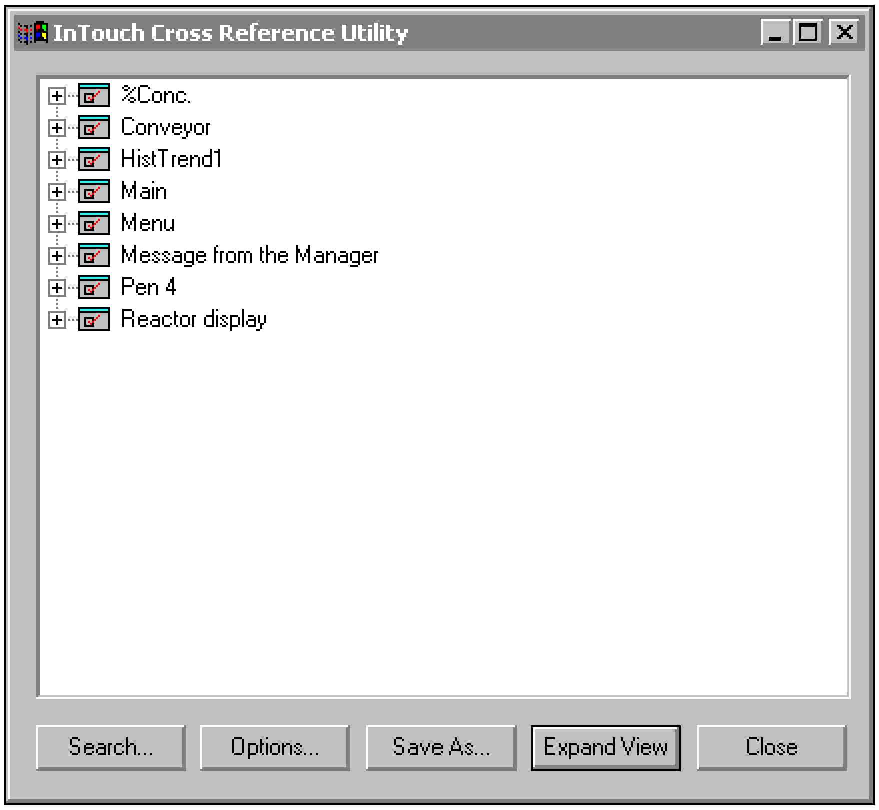
To copy the QuickScript to the Windows Clipboard, right-click the script, then click **Select All**. Right-click the script again and click **Copy**. You can also execute the Windows copy command (**Ctrl+C**).

To paste the copied script into another InTouch QuickScript, in the Application Explorer under **Scripts**, double-click the type of script that you want to create. The QuickScript editor appears. On the **Edit** menu, click **Paste**, or right-click the script window, and then click **Paste**. You can also execute the Windows paste command (**Ctrl+V**).

Click **Cancel** to close the dialog box.

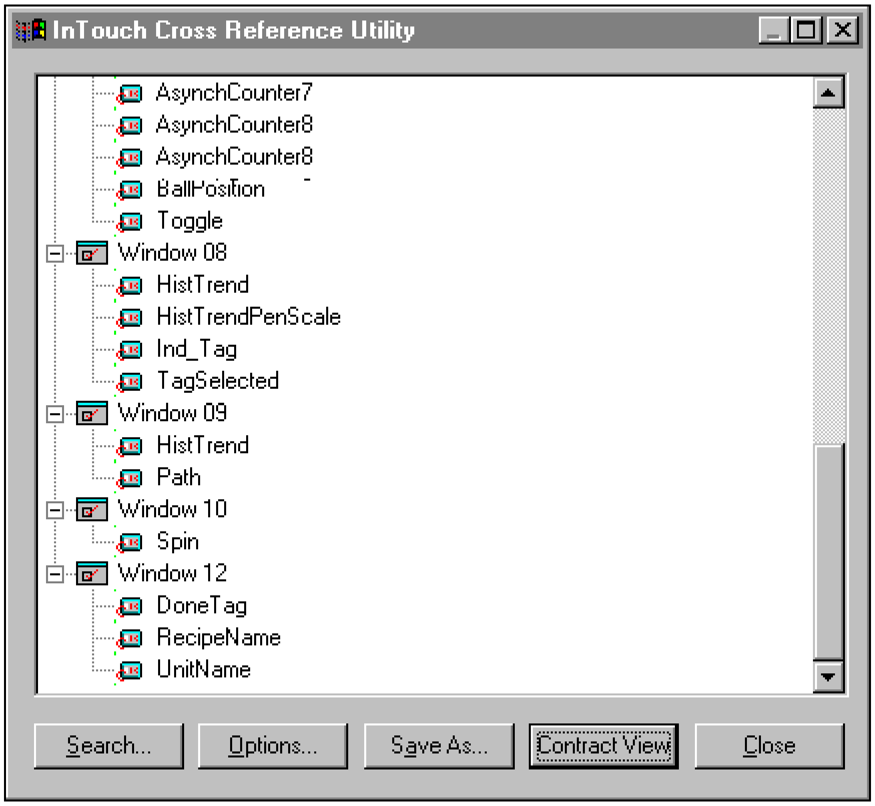
Cross Referencing by Window Name

Sorts the display by window name then the tagnames used in the window. For example:



Note This view only displays the tagnames used in the window, it does not include usage in animation links, scripts, and so on.

Click **Expand View** to display all view levels available for the displayed tagnames or windows. For example:



AsynchCounter7
AsynchCounter8
AsynchCounter8
BallPosition
Toggle
Window 08
HistTrend
HistTrendPenScale
Ind_Tag
TagSelected
Window 09
HistTrend
Path
Window 10
Spin
Window 12
DoneTag
RecipeName
UnitName

Click **Contract View** to return the dialog box to its default mode.

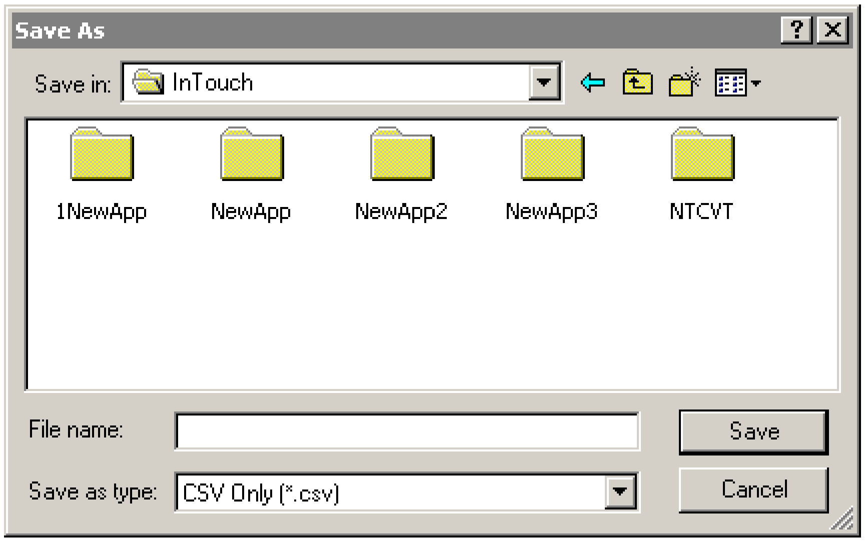
Click **Close** to exit the cross-reference utility.

Saving Cross Reference Files

Your cross reference files can be saved and viewed later in any text editor program that supports the comma separated variable (.csv) file. The information stored in a cross-reference file corresponds to the information currently displayed in the **InTouch Cross Reference Utility** dialog box.

To save a cross-reference file

1. In the **InTouch Cross Reference Utility** dialog box, click **Save As**. The **Save As** dialog box appears.



2. In the **File name** box, type the name that you want to save the cross-reference file under.

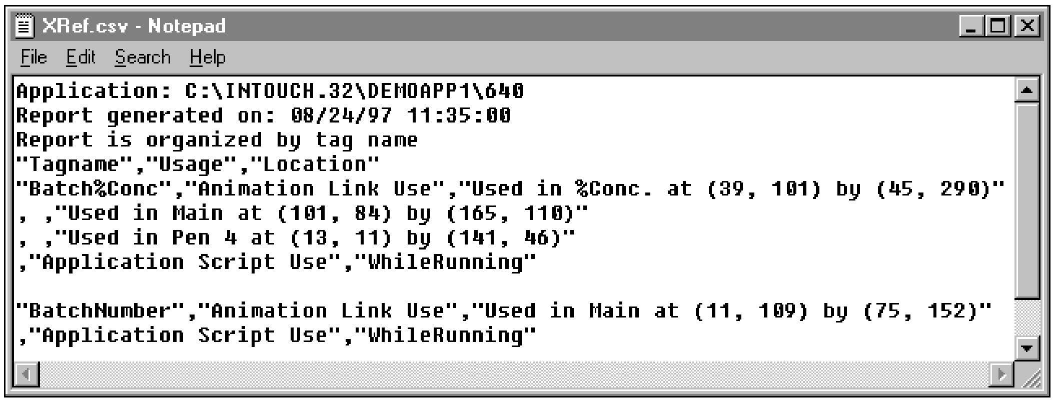
Note The file must be saved as a .csv file.

3. Click **Save**.

Printing Cross Reference Files

You can open a cross-reference .csv file in any text editor program that supports the .csv file format and print the cross-reference file as a report.

For example, if you open the file in Notepad, it appears as follows:



```
Application: C:\INTOUCH.32\DEMOAPP1\640
Report generated on: 08/24/97 11:35:00
Report is organized by tag name
"Tagname","Usage","Location"
"Batch%Conc","Animation Link Use","Used in %Conc. at (39, 101) by (45, 290)"
, ,"Used in Main at (101, 84) by (165, 110)"
, ,"Used in Pen 4 at (13, 11) by (141, 46)"
,"Application Script Use","WhileRunning"

"BatchNumber","Animation Link Use","Used in Main at (11, 109) by (75, 152)"
,"Application Script Use","WhileRunning"
```

To print the file in Notepad, on the **File** menu, click **Print**.

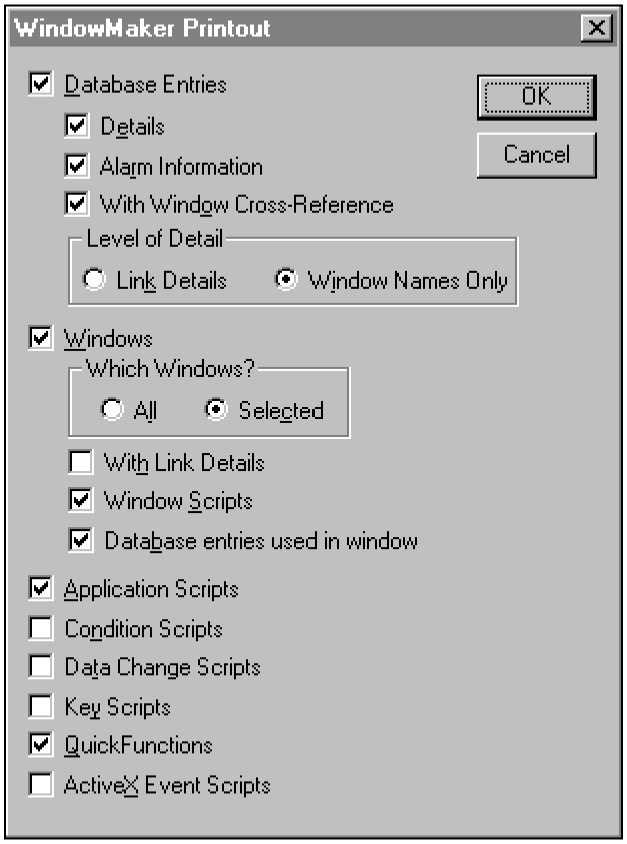
Printing Tagname Dictionary Details

In addition to printing a saved cross-reference .csv file, you can print listings of the Tagname Dictionary details, alarm information, link details and scripts. Printing the Tagname Dictionary details can help you to determine the usage of tagnames.

Note Your Windows default printer will be used to produce the printout that will be 80 columns wide. Your default printer is selected and setup through the Windows Control Panel.

To print Tagname Dictionary details

1. On the **File** menu, click **Print**. The **WindowMaker Printout** dialog box appears.



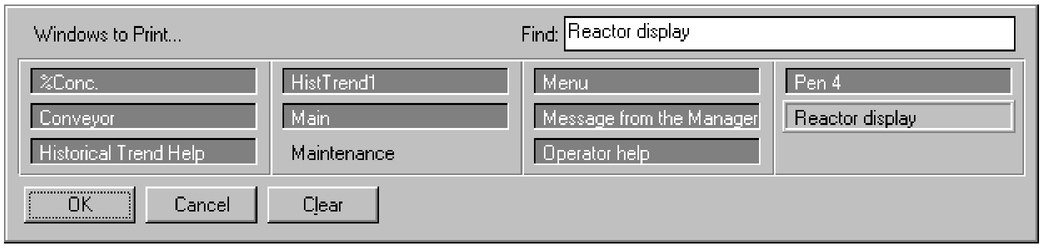
[✓] Database Entries
[✓] Details
[✓] Alarm Information
[✓] With Window Cross-Reference
Level of Detail: Link Details [✓ Window Names Only]
[✓] Windows
Which Windows?: All [✓ Selected]
[ ] With Link Details
[✓] Window Scripts
[✓] Database entries used in window
[✓] Application Scripts
[ ] Condition Scripts
[ ] Data Change Scripts
[ ] Key Scripts
[✓] QuickFunctions
[ ] ActiveX Event Scripts

2. Select **Database Entries** if you want to print all database information. If you select **Database Entries**, the following three options become active:

   - Select **Details** to include the database details in your report.
   - Select **Alarm Information** to include the database alarm information in your report.
   - Select **With Window Cross Reference** to print all database entries with window cross-references. Selecting this option will activate **Level of Detail** options:

- Select **Link Details** to print the location and animation link details where the tagname was used.
- Select **Window Names Only** to print only the name of the cross-referenced windows(s).

3. Select **Windows** to print a listing of the database entries used in the application windows. If you select **Windows**, the following three options become active:
   - Select **All** to print the database entries for all windows in the application.
   - Select **Selected** to print only the database entries for specific windows. The **Windows to Print** dialog box appears.



4. Select the windows you want to print, then click **OK**. (By default, all window names will be selected when the dialog box appears.)
   - Select **With Link Details** to print the link details for the window(s).
   - Select **Window Scripts** to print the scripts associated with the window(s).
   - Select **Database entries used in window** to print the tagnames used in the window(s).
   - Select **Application Scripts** to print the application scripts.
   - Select **Condition Scripts** to print the condition scripts associated with the window(s).
   - Select **Data Change Scripts** to print the data change scripts associated with the window(s).
   - Select **Key Scripts** to print the key scripts associated with the window(s).
   - Select **Quick Functions** to print your QuickFunctions.
5. Click **OK** to begin printing your report.

Deleting Tagnames from the Dictionary

InTouch maintains a use count for each item in the database. This count is not updated automatically for certain operations, such as deleting a window, changing tagnames in links or scripts, and so on. In these cases, InTouch continues to consider the tagname as being used in the application and will not allow you to delete it. Therefore, you may need to update your use count in order to delete a tagname.

To delete an unused tagname

1. Close WindowViewer if it is running.
2. On the **Special** menu, click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears.
3. Click **Select**. The **Select Tagname** (Tag Browser) appears.
4. Select the tagname that you want to delete then click **OK**. The **Tagname Dictionary** dialog box appears displaying the selected tagname's definition.
5. Click **Delete**.

Note The **Delete** button will not be available if WindowViewer is running or InTouch considers the tagname as being used in the application.

You can determine where a tagname is being used through the InTouch cross-reference utility, (On the **Special** menu, click **Cross Reference**.) or you can print a report of all tagname links used in a window. (On the **File** menu, click **Print**.)

If you determined that the tagname to delete is unused and the **Delete** button is not available, open WindowMaker. On the **Special** menu, click Update Use Counts and click **Yes** when prompted, then click **OK**.

For more information on printing reports, see "Printing Tagname Dictionary Details."

Updating Use Counts

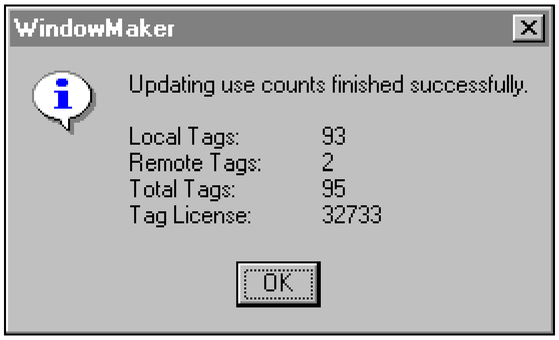
Since InTouch maintains a use count for each item in the database you may need to update the use counts to set all unused tagnames to zero before InTouch will allow you to delete any of them.

To update tagname use counts

1. Close all your windows.
2. On the **Special** menu, click **Update Use Counts**.

Tip A message box appears telling you that updating use counts can take quite a while. You may at that time cancel the command or continue.

3. Click **Yes** to continue updating the use counts. Once the system has completed updating the use counts, the following dialog box appears.



Updating use counts finished successfully.

Local Tags: 93
Remote Tags: 2
Total Tags: 95
Tag License: 32733

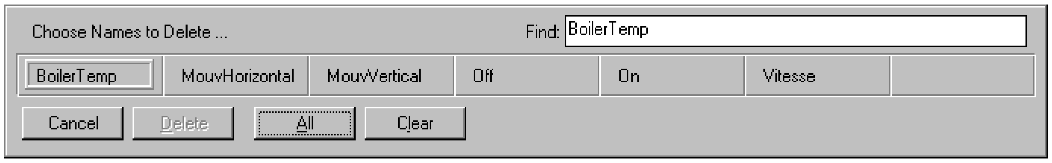
4. Click **OK**.

Deleting Unused Tagnames

After you have updated the use count, InTouch will allow you to delete all unused tagnames. You can either delete them by opening each of them in the **Tagname Dictionary** dialog and clicking **Delete**, or you can delete one or more of them at once by using the **Delete Unused Tags** command.

To delete multiple unused tagnames

1. On the **Special** menu, click **Delete Unused Tags**. The **Choose Names to Delete** dialog box appears.



Find: BoilerTemp

BoilerTemp MouvHorizontal MouvVertical Off On Vitesse

2. Select the tagnames that you want to delete, then click **Delete**.
3. Click **All** to delete all tagnames displayed.

Caution! Tagnames that are only alarmed have no use count, and can be accidentally deleted. To ensure that alarmed only tagnames are included in the use count, you need to use them in a window or QuickScript.

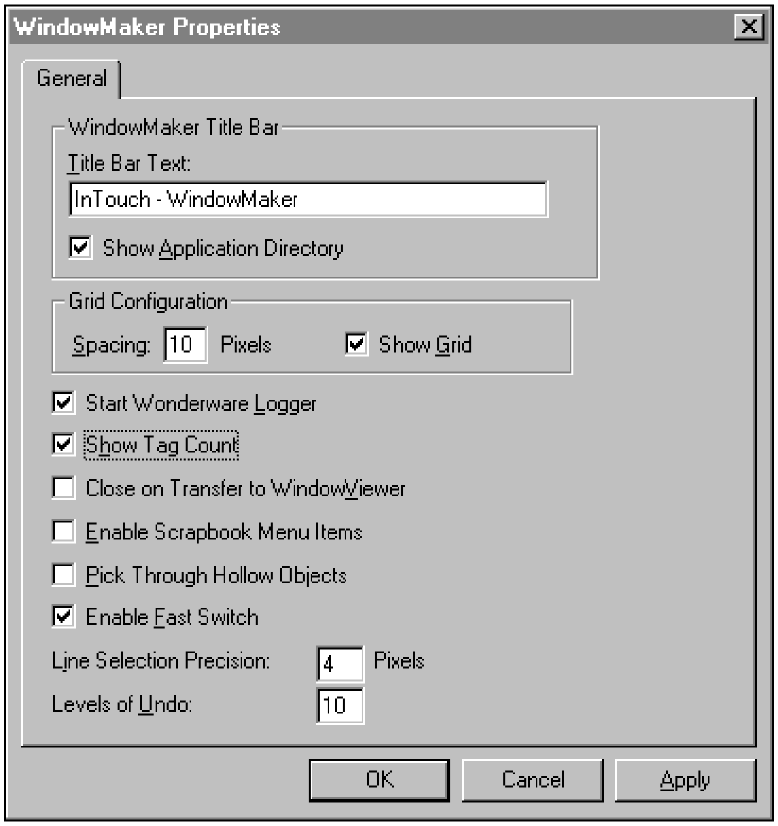
Displaying the Tagname Usage Count

You can display the number of local tagnames that are currently defined in your Tagname Dictionary in the menu bar in WindowMaker. (The tagname count does not include internal system tagnames or remote tagname references.)

To show the tagname count

1. On the **Special** menu, point to **Configure**, and then click **WindowMaker**. The **WindowMaker Properties - General** property sheet appears.

Tip To quickly access the dialog box, in the Application Explorer under **Configure**, double-click **WindowMaker**.



WindowMaker Properties

General

WindowMaker Title Bar

Title Bar Text: InTouch - WindowMaker

☑ Show Application Directory

Grid Configuration

Spacing: 10 Pixels ☑ Show Grid

☑ Start Wonderware Logger

☑ Show Tag Count

☐ Close on Transfer to WindowViewer

☐ Enable Scrapbook Menu Items

☐ Pick Through Hollow Objects

☑ Enable Fast Switch

Line Selection Precision: 4 Pixels

Levels of Undo: 10

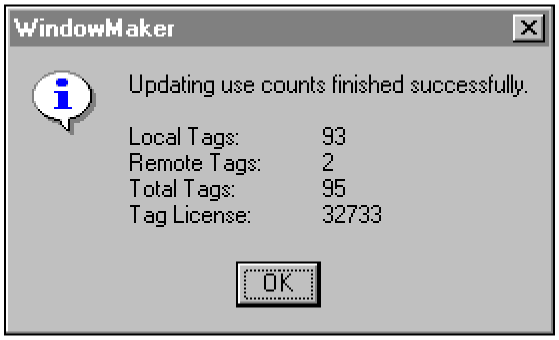
2. Select **Show Tag Count**.
3. Click **OK**.
4. The total number of local tagnames defined in your Tagname Dictionary will now be displayed at the end of your WindowMaker menu bar.

Tip The entire Tagname Dictionary must be read in order to update the displayed tagname count. Therefore, when this option is turned on, performance may be degraded when you are making changes to your Tagname Dictionary. If your Tagname Dictionary is large, you should not select this option.

To determine remote tagname usage

1. On the Special menu, click Update Use Counts.
2. The system will update your tagname usage then display the following dialog box:

Tip Updating use counts can take a while.



WindowMaker

Updating use counts finished successfully.

```
Local Tags:      93
Remote Tags:     2
Total Tags:      95
Tag License:     32733
```

3. The **Remote Tags** line will display the number of remote tagnames used in your application.
4. Click **OK**.

Substituting Tagnames

When you duplicate an object it is an exact replica of the original including links, animation, scripts and so on. However if you need to use a different tagname on an object that you have duplicated you must change the tagname. In WindowMaker, this is called "substituting a tagname." You can select and change the tagnames for any object at any time and you can select multiple objects and change all their tagnames the same time.

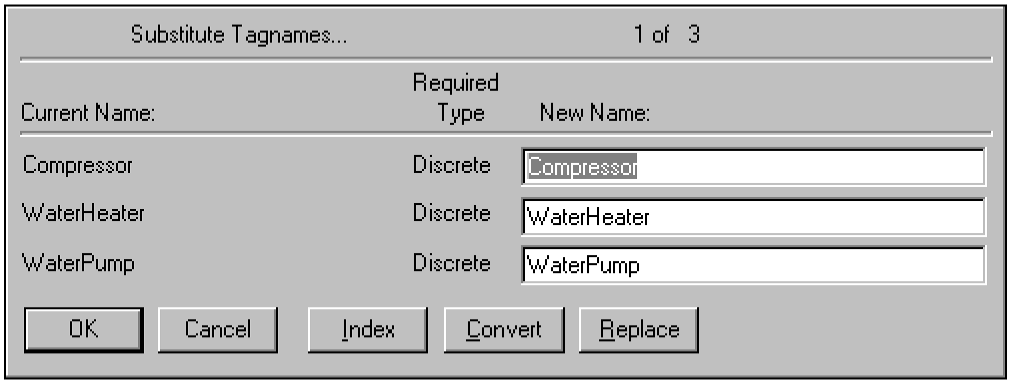
Tip If you change a tagname for an object and WindowViewer is running, you will need to restart WindowViewer for the change to take effect.

If your system's license supports a limited number of tagnames, you can also convert your local tagnames to remote tagname references to reduce the number of tagnames defined in your local Tagname Dictionary.

To change an object's tagname to another local tagname

1. Select the object(s) whose tagname you want to change, and then on the **Special** menu, click **Substitute Tags**. The **Substitute Tagnames** dialog box appears.

Tip To quickly access the dialog box, right-click one of the selected objects, point to **Substitute**, and then click **Substitute Tags**.


Substitute Tagnames... 1 of 3

Current Name:	Required Type	New Name:
Compressor	Discrete	Compressor
WaterHeater	Discrete	WaterHeater
WaterPump	Discrete	WaterPump

2. In the **New Name** box, enter a new tagname, and then click **OK**. The tagname associated with the selected object(s) will automatically be changed.

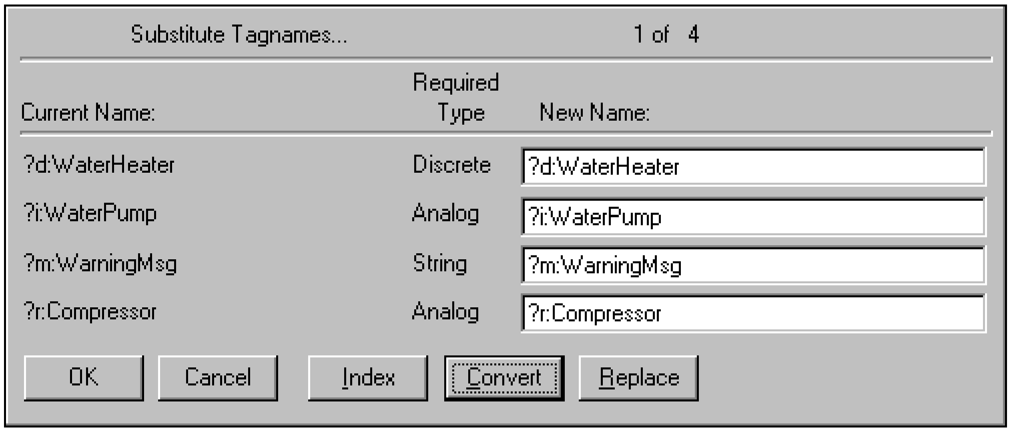
Tip If you right-click the **New Name** box, a menu appears displaying the commands that you can apply to your text.

If you double-click a tagname in the **New Name** box, its definition in the Tagname Dictionary appears.

If you erase the tagname then double-click in the blank **New Name** box, the Tag Browser appears.

Converting Placeholder Tagnames

When you index tagnames (to take them out of service) or you import or export a window or QuickScript to or from your current application, all the tagnames associated with that window or QuickScript are transferred with the window, but they are not added to your new application's database. Instead, they are automatically marked as "placeholder" (index) tagnames. You must convert these placeholder tagnames and, if required, define them in your new application Tagname Dictionary. For example:



Substitute Tagnames... 1 of 4

Current Name:	Required Type	New Name:
?d:WaterHeater	Discrete	?d:WaterHeater
?i:WaterPump	Analog	?i:WaterPump
?m:WarningMsg	String	?m:WarningMsg
?r:Compressor	Analog	?r:Compressor

In this example, to convert the placeholder tagnames to local tagnames, click **Convert**.

Tip When you import a window, if any of the tagnames (except remote tagnames) are not defined in your local Tagname Dictionary, you will be prompted to define each of them before you can convert them. If this is the case, click **OK**. The **Tagname Dictionary** dialog box appears and you can define the tagname(s).

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

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

Remote references will not be shown as placeholders but as a remote tagname references, for example, **PLC2:Temperature**.

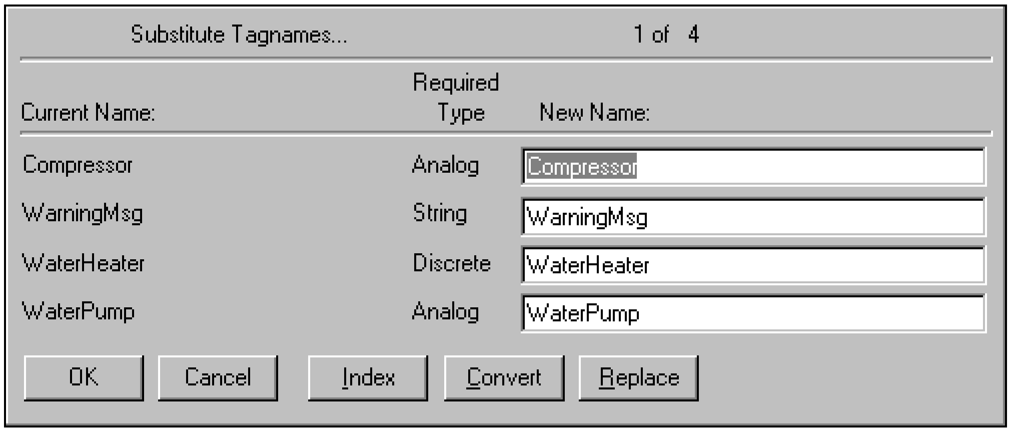
Converting Tagnames to Remote References

There are several methods that you can use in the **Substitute Tagnames** dialog box to convert placeholder (or local) tagnames to remote tagname references. You can directly type in the remote tagname reference, convert the placeholder tagnames associated with an imported window, or launch the Tag Browser and display the tag source's Tagname Dictionary to select the remote tagname reference.

To manually convert tagnames to remote tagname references

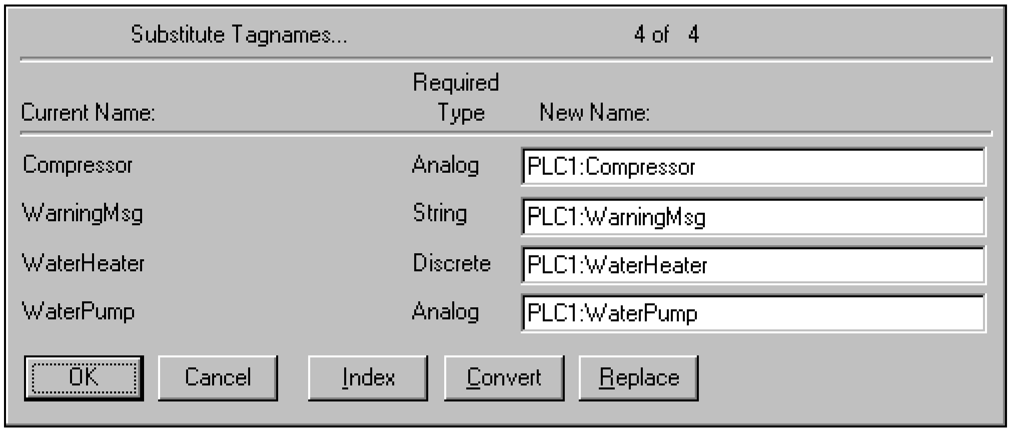
1. Select the object(s) associated with the local tagname that you want to change to a remote tagname reference, and then on the **Special** menu, click **Substitute Tags**. The **Substitute Tagnames** dialog box appears.

Tip To quickly access the dialog box, select all the objects, then right-click one of the selected objects, point to **Substitute**, and then click **Substitute Tags**.


Substitute Tagnames... 1 of 4

Current Name:	Required Type	New Name:
Compressor	Analog	Compressor
WarningMsg	String	WarningMsg
WaterHeater	Discrete	WaterHeater
WaterPump	Analog	WaterPump

2. In **New Name** box, select each tagname that you want to change, and then type in the remote tagname reference:


Substitute Tagnames... 4 of 4

Current Name:	Required Type	New Name:
Compressor	Analog	PLC1:Compressor
WarningMsg	String	PLC1:WarningMsg
WaterHeater	Discrete	PLC1:WaterHeater
WaterPump	Analog	PLC1:WaterPump

3. Click **OK**.

Tip If you use this method and you no longer need the original tagnames to be defined in the local Tagname Dictionary, you can update the tagname use counts and then delete the unused tagnames.

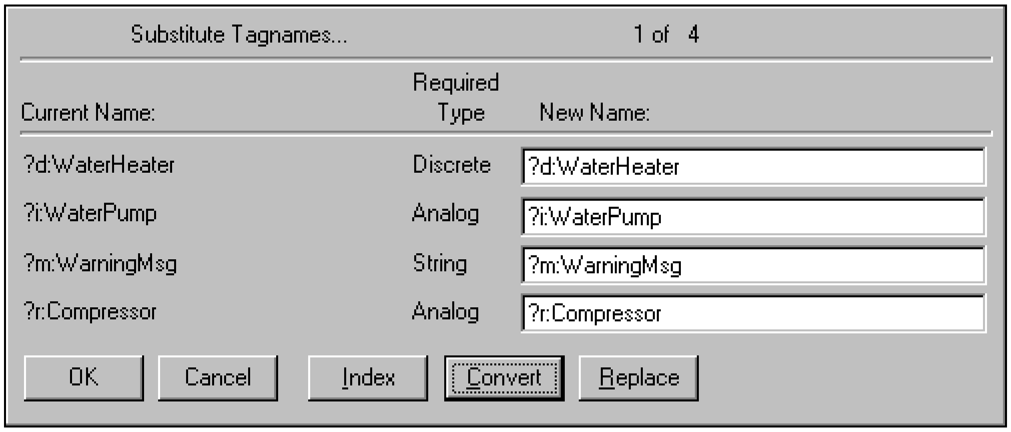
For more information, see "Deleting Tagnames from the Dictionary."

To convert an imported window's tagnames to remote references

1. Open the imported window and select all the object(s), and then on the **Special** menu, click **Substitute Tags**. The **Substitute Tagnames** dialog box appears.

   For more information on importing windows and scripts see, "Chapter 2, "Using WindowMaker.".

   Tip To quickly access the dialog box, press the F2 key (to select all the objects in the window), right-click one of the selected objects, then point to **Substitute**, and then click **Substitute Tags**.



Current Name:	Required Type	New Name:
?d:WaterHeater	Discrete	?d:WaterHeater
?i:WaterPump	Analog	?i:WaterPump
?m:WarningMsg	String	?m:WarningMsg
?r:Compressor	Analog	?r:Compressor

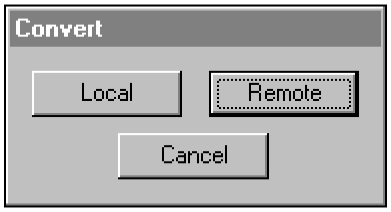
Note The **Index** button turns links into placeholders which disables the animation links to local tagnames, thus freeing them for possible deletion.

You can also use the **Substitute Tags** command to convert local tagnames to remote tagname references. To do so, select the objects associated with the local tagnames, and then select the **Substitute Tags** command to display them in the **Substitute Tagnames** dialog box. Click **Index** to change them to placeholder tagnames, and then click **Convert** and follow the steps below.

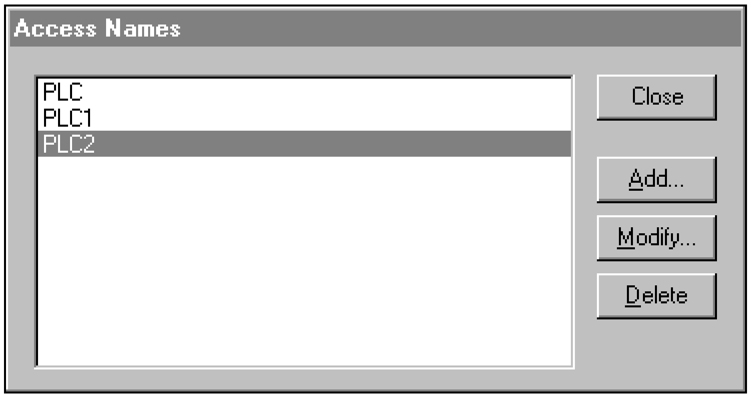
If you use the **Index** button for either of the above options, and you no longer need the original tagnames to be defined in the local Tagname Dictionary, you can update the tagname use counts and then delete the unused tagnames.

For more information, see "Deleting Tagnames from the Dictionary."

2. Click **Convert**. The **Convert** dialog box appears.



3. Click **Remote**. The **Access Names** dialog box appears listing all of the Access Names that you have defined in your local application:



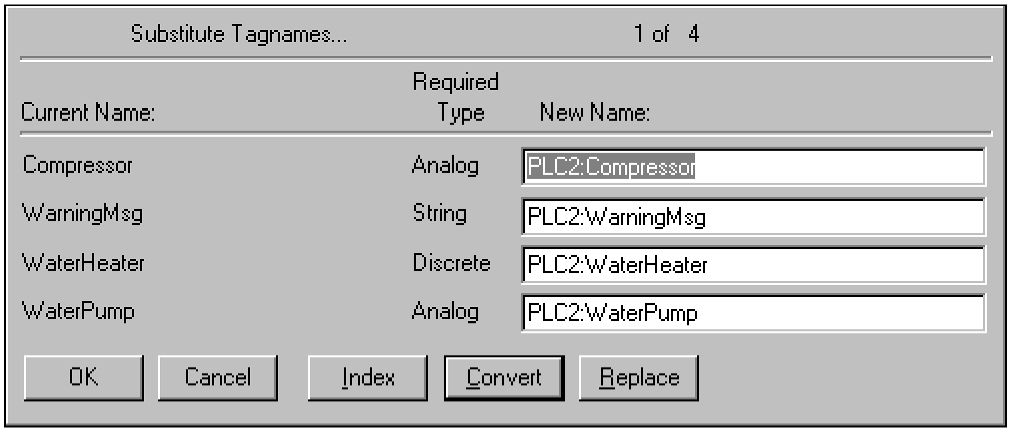
PLC
PLC1
PLC2

4. Double-click the Access Name in the list or select it, and then click **Close**.

Tip To verify that the Access Name correct, you can click **Modify** to view it.

If you do not have an Access Name currently defined that points to the tag source, click **Add** and define it now. The Access Name **<u>must</u>** include the name of the remote node where the application resides.

5. **<u>All</u>** tagnames displayed in the **Substitute Tags** dialog box will automatically be converted to remote tagname references (prefixed with the Access Name that you selected). For example:



Current Name:	Required Type	New Name:
Compressor	Analog	PLC2:Compressor
WarningMsg	String	PLC2:WarningMsg
WaterHeater	Discrete	PLC2:WaterHeater
WaterPump	Analog	PLC2:WaterPump

6. Click **OK**.

Tip This procedure works the same for an imported QuickScript, except you open the QuickScript in the InTouch QuickScript editor and then click **Convert**.

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 instantly be receiving data from hundreds of remote tagnames without defining a single tagname in your local Tagname Dictionary.

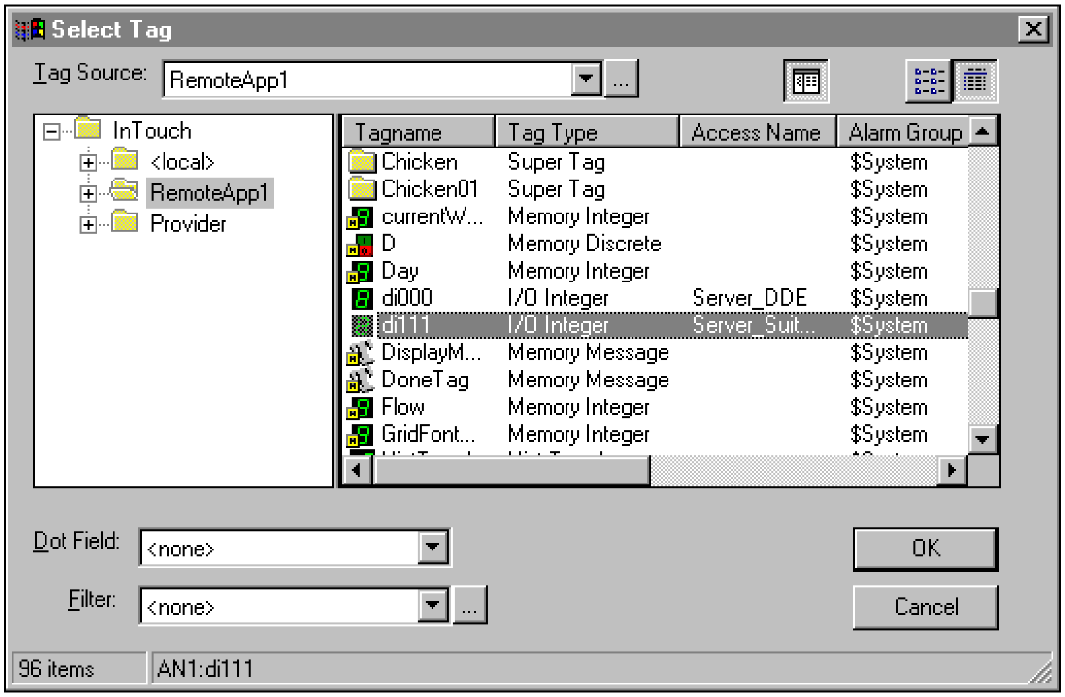
To select a remote tagname reference in the Tag Browser

1. Select the object(s) associated with the local tagname that you want to change to a remote tagname reference, and then on the **Special** menu, click **Substitute Tags**. The **Substitute Tagnames** dialog box appears.

Tip To quickly access the dialog box, select all objects, then right-click one of the selected objects, point to **Substitute**, and then click **Substitute Tags**.

2. Erase the tagname in the **New Name** box that you want to replace with a remote tagname reference, and then double-click the **New Name** box. The **Select Tag** dialog box appears.

3. Click the tool to display the tree view pane:



Tag Source: RemoteApp1

Tagname	Tag Type	Access Name	Alarm Group
Chicken	Super Tag		$System
Chicken01	Super Tag		$System
currentW...	Memory Integer		$System
D	Memory Discrete		$System
Day	Memory Integer		$System
di000	I/O Integer	Server_DDE	$System
di111	I/O Integer	Server_Suit...	$System
DisplayM...	Memory Message		$System
DoneTag	Memory Message		$System
Flow	Memory Integer		$System
GridFont...	Memory Integer		$System

Dot Field: <none>

Filter: <none>

96 items | AN1:di111

4. If the tag source is already defined in the Tag Browser, select it in the tree view to displays its tagnames.

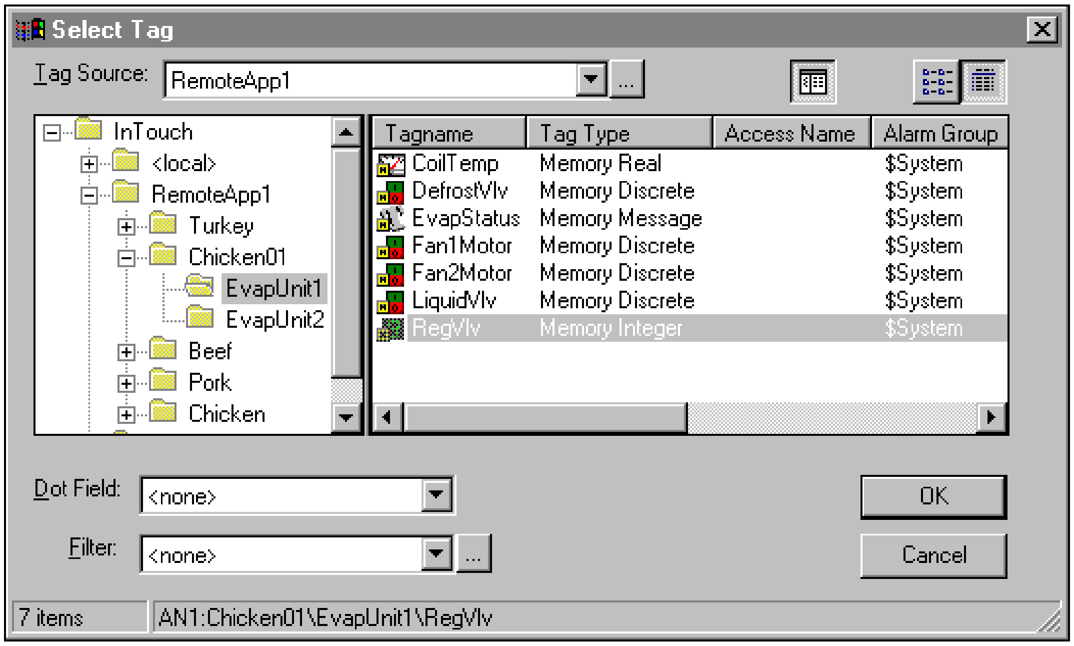
Tip To expand the tree view, double-click the application name or click ⊞.

Note If the tag source is not currently defined, you can define it now.

For more information on defining tag sources, see "Defining Tag Sources."

5. Double-click the remote tagname that you want to use or select it, and then click **OK**.

Tip You can also select SuperTag member tagnames. For example:



Tag Source: RemoteApp1

InTouch
- <local>
- RemoteApp1
  - Turkey
  - Chicken01
    - EvapUnit1
    - EvapUnit2
  - Beef
  - Pork
  - Chicken

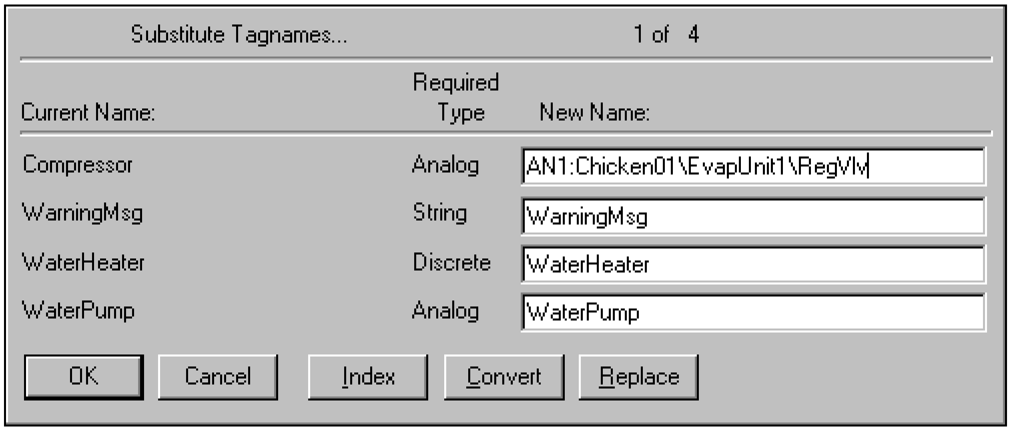
Tagname	Tag Type	Access Name	Alarm Group
CoilTemp	Memory Real		$System
DefrostVlv	Memory Discrete		$System
EvapStatus	Memory Message		$System
Fan1Motor	Memory Discrete		$System
Fan2Motor	Memory Discrete		$System
LiquidVlv	Memory Discrete		$System
RegVlv	Memory Integer		$System

Dot Field: <none>

Filter: <none>

7 items AN1:Chicken01\EvapUnit1\RegVlv

The **Substitute Tags** dialog box appears displaying the remote tagname reference:



Substitute Tagnames... 1 of 4

Current Name:	Required Type	New Name:
Compressor	Analog	AN1:Chicken01\EvapUnit1\RegVlv
WarningMsg	String	WarningMsg
WaterHeater	Discrete	WaterHeater
WaterPump	Analog	WaterPump

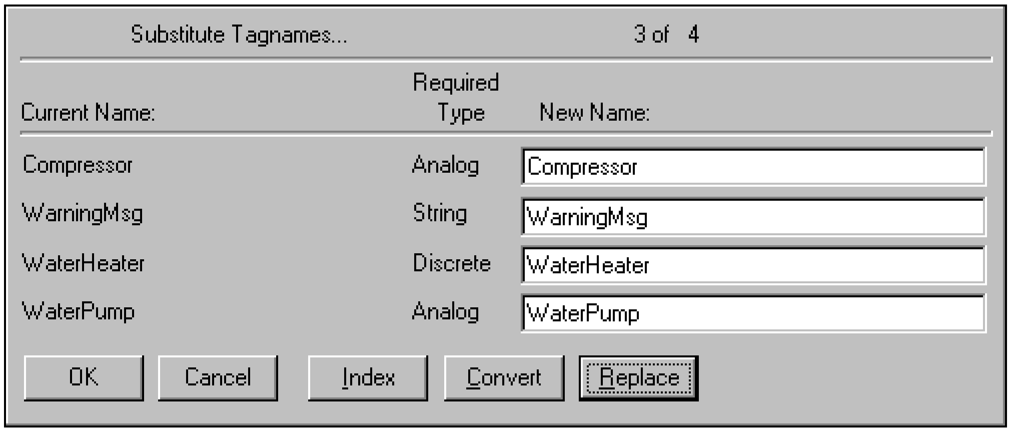
6. Click **OK** to close the dialog box and associate the remote tagname with the selected object(s).

Tip Repeat this procedure for each tagname that you want to change in the **Substitute Tagnames** dialog box.

To replace a tagname

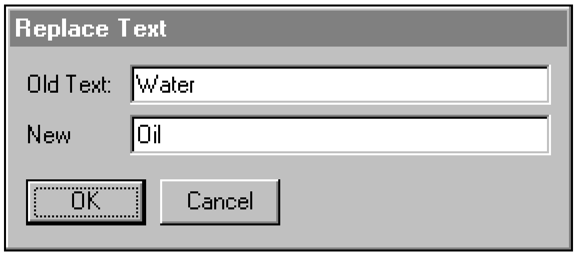
1. Select all objects whose tagname you want to change.
2. On the **Special** menu, click **Substitute Tags**. The **Substitute Tagnames** dialog box appears.

Tip To quickly access the dialog box, right-click the object, point to **Substitute,** and then click **Substitute Tags**.


Substitute Tagnames... 3 of 4

Current Name:	Required Type	New Name:
Compressor	Analog	Compressor
WarningMsg	String	WarningMsg
WaterHeater	Discrete	WaterHeater
WaterPump	Analog	WaterPump

3. Click **Replace**. The **Replace Text** dialog box appears.

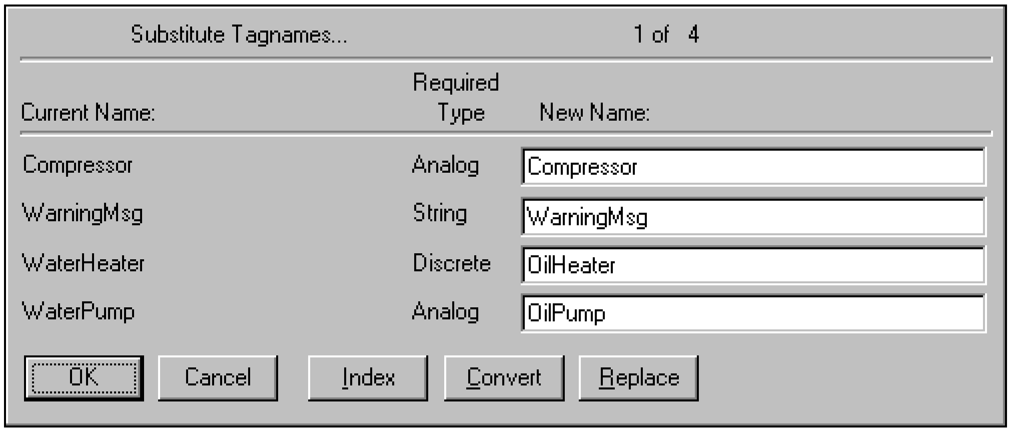

Replace Text

Old Text: Water

New: Oil

4. In the **Old Text** box, type the portion of the tagname that you want to replace.
5. In the **New** box, type the replacement text.

6. Click **OK**. The **Substitute Tagnames** dialog box reappears showing the change made to the tagname:



Substitute Tagnames... 1 of 4

Current Name:	Required Type	New Name:
Compressor	Analog	Compressor
WarningMsg	String	WarningMsg
WaterHeater	Discrete	OilHeater
WaterPump	Analog	OilPump

7. Click **OK**. All tagnames for the selected object(s) containing the text that you replaced will automatically be changed.

Scaling I/O Tagnames

All I/O type tagnames receive their values from other Windows application programs such as Excel and I/O Servers. This value is referred to as the "raw" value. When you define a tagname in the Tagname Dictionary, you must enter values for the "Min Raw" and "Max Raw." These values are used by the database as clamps on the actual raw value received from the I/O device. For example, if you set the "Min Raw" value to 50 and the actual value received from a I/O Server is 0, database will force the Raw value to 50.

InTouch does not display raw values. Instead, it displays engineering units (EU). When you define an I/O type tagname in the Tagname Dictionary, you must specify values for the "Min EU" and "Max EU." These values are used to scale the raw value to the displayed value. If you do not want to do scaling or your I/O device does the scaling for you, set the Min/Max EU values equal to the Min/Max Raw values.

For example, assume that a flow transmitter wired to a PLC register generates a value of zero at no flow and a value of 9999 at 100% flow. The following values would be entered:

```
Min EU = 0Max EU = 100
```

```
Min Raw=0Max Raw = 9999
```

A raw value of 5000 would be displayed as 50.

Let's also assume that a flow transmitter wired to a PLC register generates a value of 6400 at no flow and a value of 32000 at 300 GPM.

```
Min EU = 0Max EU = 300
```

```
Min Raw = 6400Max Raw = 32000
```

In this case, a raw value of 12800 would be displayed as 150. A raw value of 6400 would be displayed as 0 and a Raw value of 0 would be displayed as 0 (all values outside the boundaries set by the Min Raw and Max Raw values are clamped).

The above scaling works in reverse when the I/O tagname data is written from the InTouch Tagname Dictionary to other Windows applications.

For example, an operator could enter a setpoint value of 0-300 GPM in a data entry window and have a value of 6400-32000 transmitted to the PLC register.

This is always valid when the InTouch program is acting as the client (either requesting data from or sending data to another Windows program). However, when InTouch acts as data source (another Windows program is requesting data from InTouch) it will return the EU value to the requesting program.

For example, if a cell in an Excel spreadsheet contained the remote reference formula:

```
=view|tagname!speed
```

The value displayed in the cell would be the current EU value for the tagname speed.

Instrument Failure Monitoring

Beginning with Version 7.0, InTouch supports three tagname **dotfields** (**.RawValue**, **.MinRaw** and **.MaxRaw**) that you can use in InTouch QuickScripts to monitor instrument values to determine out-of-range, out-of-calibration, or failure.

Internal System $Tagnames

InTouch provides you with numerous predefined internal system tagnames that you can use to perform a variety of actions. For example, if you want to display the current time, you could link the system tagname **$TimeString** to a value display string. All internal tagnames are preceded with a dollar sign (**$**). The internal system tagnames are accessed through the Tag Browser.

For more information, see "The Tag Browser."

The following section briefly describes the internal system tagnames:

System Tagname	Description
$AccessLevel	Read only integer security tagname used in expressions or scripts to control the operator's ability to perform specific functions.
$AlarmLogging	No longer supported in InTouch.
$AlarmPrinterError	No longer supported in InTouch.
$AlarmPrinterNoPaper	No longer supported in InTouch.
$AlarmPrinterOffline	No longer supported in InTouch.
$AlarmPrinterOverflow	No longer supported in InTouch.

System Tagname	Description
$ApplicationChanged	Read only discrete tagname that reflects whether or not the remote application has changed in distributed systems. This number is incremented each time the **Notify Clients** command is selected on the Server node's WindowViewer **Special** menu.
$ApplicationVersion	Read only real tagname that reflects the current version number of the application. This number changes each time a tagname or QuickScript is changed, added or deleted.
$ChangePassword	Write only discrete security tagname that allows the operator to set the value of the **$ChangePassword** tagname to 1, causing the generic Change Password dialog box to be displayed for the operator.
$ConfigureUsers	Write only discrete security tagname that can be used on a discrete button to allow the operator to set the value of the **$ConfigureUsers** tagname to 1**,** causing the generic **Configure Users** dialog box to be displayed for editing the security user name list.
$Date	Read only integer tagname that displays the whole number of days which have passed since 1/1/70.
$DateString	Read only memory message tagname that displays the date in the same format set in the Windows registry, for example, 4/18/1992. (This date format is set through the Windows Control Panel.)
$DateTime	Read only real tagname that displays the fractional number of days which have passed since 1/1/70.
$Day	Read only integer tagname that displays the current day (value may be 1-31).
$HistoricalLogging	Read/write discrete tagname that monitors/controls starting and stopping of historical logging. This is a global command for the whole application.
$Hour	Read only integer tagname that displays the current hour of the day (value may be 0-23).
$InactivityTimeout	Read only discrete security tagname that equals 1 when the time configured for automatic log off of the operator has elapsed.
$InactivityWarning	Read only discrete security tagname that equals 1 when the time configured for warning the operator that log off is about to occur has elapsed.

System Tagname	Description
$LogicRunning	Read/write discrete tagname used to monitor and/or control the running of scripts. Asynchronous User Defined Function scripts that are currently executing cannot be stopped. However, you can prevent new scripts from executing.
$Minute	Read only integer tagname that displays the current minute (value may be 0-59).
$Month	Read only integer tagname that displays the current month (value may be 1-12).
$Msec	Read only integer tagname that displays milliseconds (value may be 0-999).
$NewAlarm	Read/write discrete tagname that is equal to 1 each time a new alarm occurs.
$ObjHor	Read only integer tagname used to display the horizontal pixel location of the center of a selected object.
$ObjVer	Read only integer tagname used to display the vertical pixel location of the center of a selected object.
$Operator	Read only security message tagname that can be used in an expression or QuickScript to control the operator's ability to perform specific functions.
$OperatorEntered	Read/write security message tagname that sets the ""User Name"" for the operator.
$PasswordEntered	Write only security message tagname that sets the ""Password"" for the operator.
$Second	Read only integer tagname that displays the current seconds (value may be 0-59).
$StartDdeConversations	Read/write discrete tagname used to start uninitiated conversations during runtime when the **Special** menu has been disabled.
$System	Read only Alarm Group type tagname for the alarm root group. If a tagname is not assigned to a specific Alarm Group name, it is automatically assigned to this root group by default. All defined Alarm Groups are descendants of **$System**.
$Time	Read only integer tagname that displays the time in milliseconds since midnight.

System Tagname	Description
$TimeString	Read only memory message tagname that displays the time in the same format set in the Windows registry, e.g., 12:01:59 PM. (This time format is set through the Windows Control Panel.)
$Year	Read only integer tagname that displays the year in four digits. For example,2008.

For more information on system tagnames, see your online *InTouch Reference Guide.*

Tagname Dotfields

Most discussions concerning InTouch refer to the concept of objects. The concept of objects is very widespread and complex. For our discussion we will limit our definition of an object to a collection of information about a graphical object on the screen or information about a tagname in the Tagname Dictionary.

For example, if a rectangle is drawn on the screen it has certain "attributes" such as the line width, line and fill color, pixel location on the screen, links associated with, etc. Tagnames work in much the same way. For example, if an analog, alarmed tagname is created called "Analog_Tagname," it will have "attributes" associated with it such as the name of the tagname, the tagname's **HiHi** alarm setpoint, and so on. Some of these "attributes" are accessible within InTouch through scripts, expressions and user inputs and are known as **dotfields**. The syntax required to access these data fields associated with a tagname is **Tagname dotfields**.

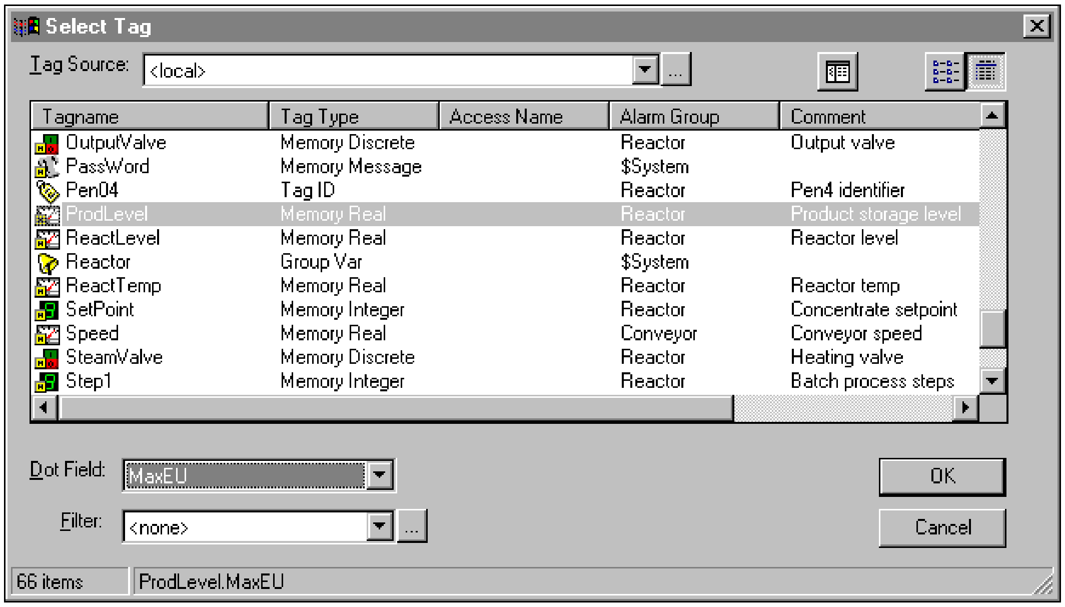
For example, to allow runtime changes to the **HiHi** alarm limit on tagname "Analog_Tagname," you could create an **Analog - User Input** touch link and apply it to a button that is defined with the expression **Analog_Tagname.HiHiLimit**. In runtime, the operator would simply click the button and type in a new value for the **HiHi** alarm limit being used for "Analog_Tagname."

You can use **dotfields** to allow input and output of data associated with a tagname and you can use the historical **dotfields** to allow the user to dynamically modify the historical trend being displayed. For example you can allow the user to control the scrolling, lock or reposition the scooters on the trend, or reassign the pens to new tagnames.

For more information, **dotfields**, see your online *InTouch Reference Guide.*

To access tagname dotfields

Type a tagname plus a period (**tagname.**) in any InTouch QuickScript, animation link tagname, or expression input box, and then double-click to the right of the period (**.**). The Select Tag browser appears displaying the tagnames defined for the current tag source:



Tagname	Tag Type	Access Name	Alarm Group	Comment
OutputValve	Memory Discrete		Reactor	Output valve
PassWord	Memory Message		$System	
Pen04	Tag ID		Reactor	Pen4 identifier
ProdLevel	Memory Real		Reactor	Product storage level
ReactLevel	Memory Real		Reactor	Reactor level
Reactor	Group Var		$System	
ReactTemp	Memory Real		Reactor	Reactor temp
SetPoint	Memory Integer		Reactor	Concentrate setpoint
Speed	Memory Real		Conveyor	Conveyor speed
SteamValve	Memory Discrete		Reactor	Heating valve
Step1	Memory Integer		Reactor	Batch process steps

To select a dotfields

1. Click the **Dot Field** arrow to open the list of **dotfields** that you can associate with the type of tagname currently selected.

   Tip By default, **<none>** will initially be displayed for all types of tagnames.

   Note **Dot Field** is not available when you access the Tag Browser from the Tagname Dictionary or, during runtime, when selecting a tagname for a historical trend pen from the Historical Trend Setup dialog box. (The historical trend must be configured with the Allow runtime changes option selected.)

2. Click the **dotfields** in the list that you want to append to the selected tagname.

   Note Not every tagname type has the same **dotfields**. For example, a **Discrete** type tagname has **.OnMessage**, whereas an analog does not. If you select a **Discrete** type tagname and you assign **.OnMessage** to it, and then you select another **Discrete** type tagname, the displayed **dotfields** list will not change. But, if you select another type of tagname in the control view list, for example an analog, the displayed **dotfields** will revert to **<none>**.

For more information on the Tag Browser, see "The Tag Browser."

The following section briefly describes the tagname **dotfields**:

Dotfields	Description
.Ack	Read/write discrete tagname dotfields that monitors/controls the alarm acknowledgment status of tagnames and Alarm Groups. **.Ack** has an inverse tagname dotfields called **.Unack**. When an unacknowledged alarm occurs, **.Unack** will be set to 1. **.Unack** can then be used in animation links or condition scripts to trigger enunciators for any unacknowledged alarms.
.AckDev	Monitors/controls the alarm acknowledgment status of deviation type alarms active on the tagname.
.AckDsc	Monitors/controls the alarm acknowledgement status of alarms or discrete tags.
.AckROC	Monitors/controls the alarm acknowledgment status of rate-of-change type alarms active on the tagname.
.AckValue	Monitors/controls the alarm acknowledgment status of value type alarms active on the tagname.
.Alarm	Read only discrete tagname dotfields that is equal to 1 when an alarm condition exists for the specified tagname, Alarm Group Name.
.AlarmAccess	Returns the access name of the tagname associated with a selected alarm. The alarm has to be selected by clicking on the summary Distributed Alarm Display.
.AlarmAckModel	Monitors the **Ack Model** associated with the tagname as follows: 0=condition (default) 1=event oriented 2=expanded summary Applies to discrete or analog tagnames with alarms. Read only, but can be configured in WindowMaker.
.AlarmClass	Returns the class of the alarm.
.AlarmComment	Read/Write text string that describes what the alarm is about, not what the tagname is about. By default it is empty in a new application. However, when an InTouch 7.1 application is converted to a 7.11 application, for backward compatibility the tagname comment is copied to the **AlarmComment**.

Dotfields	Description
.AlarmDate	Returns the date of the alarm.
.AlarmDev	Signals that a deviation type alarm exists.
.AlarmDevCount	Tracks the total number of deviation alarms active on a given tagname or alarm group.
.AlarmDevDeadband	Read/write analog (only valid for integer or real tagnames) tagname dotfields that monitors/controls the deviation percentage deadband for both minor and major deviation alarms.
.AlarmDevUnAckCount	Tracks the total number of unacknowledged deviation alarms active on a given tagname or alarm group.
.AlarmDisabled	Disables/enables events and alarms. Applies to discrete and analog tagnames with alarms, or to alarm groups. **Note** The same as the **.AlarmEnabled**, but of opposite polarity.
.AlarmDsc	Indicates that a discrete alarm condition is currently active.
.AlarmDscCount	Tracks the total number of discrete alarms active on a given tagname or alarm group.
.AlarmDscDisabled	Indicates whether or not the tagname can generate discrete alarms. **Note** This dotfield is the same as **.AlarmDisabled** for a discrete tagname.
.AlarmDscEnabled	Indicates whether or not the tagname can generate discrete alarms. **Note** This dotfield is the same as **.**AlarmEnabled for a discrete tagname.
.AlarmDscInhibitor	Returns the name of the inhibitor tagname assigned to the discrete alarm (if any) for this tagname.
.AlarmDscUnAckCount	Tracks the total number of unacknowledged discrete alarms active on a given tagname or alarm group.
.AlarmEnabled	Read/write discrete tagname dotfields that disables/enables events and alarms for a tagname and Alarm Group.
.AlarmGroup	Contains the current query used to populate a distributed alarm display object.
.AlarmGroupSel	Returns the alarm group of the alarm.
AlarmHiDisabled	Disables/enables the **Hi** limit for analog tagnames with alarms.

Dotfields	Description
.AlarmHiEnabled	Disables/enables the **Hi** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dot field, but opposite polarity.
.AlarmHiInhibitor	Returns the **Alarm Inhibitor Tag** reference for the **Hi** limit. Applies to analog tagnames with alarms. Read only but can be configured in Window Maker.
.AlarmHiHiDisabled	Disables/enables the **HiHi** limit for analog tagnames with alarms.
.AlarmHiHiEnabled	Disables/enables the **HiHi** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dot field, but opposite polarity.
.AlarmHiHiInhibitor	Returns the **Alarm Inhibitor Tag** reference for the **HiHi** limit. Applies to analog tagnames with alarms. Read only but can be configured in Window Maker.
.AlarmLimit	Returns the limit of the alarm.
.AlarmLoDisabled	Disables/enables the **Lo** limit for analog tagnames with alarms.
.AlarmLoEnabled	Enables/disables the **Lo** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dot field, but of opposite polarity.
.AlarmLoInhibitor	Returns the **Alarm Inhibitor Tag** reference for the **Lo** limit. Applies to analog tagnames with alarms. Read only but can be configured in Window Maker.
.AlarmLoLoDisabled	Disables/enables the **LoLo** limit for analog tagnames with alarms.
.AlarmLoLoEnabled	Disables/enables the **LoLo** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dot field, but of opposite polarity.

Dotfields	Description
.AlarmLoLoInhibitor	Returns the **Alarm Inhibitor Tag** reference for the **LoLo** limit. Applies to analog tagnames with alarms. Read only but can be configured in Window Maker.
.AlarmMajDevDisabled	Disables/enables the **Major Deviation** limit for analog tagnames with alarms.
.AlarmMajDevEnabled	Disables/enables the **Major Deviation** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dot field, but of opposite polarity.
.AlarmMajDevInhibitor	Returns the **Alarm Inhibitor Tag** reference for the **Major Deviation** limit. Applies to analog tagnames with alarms. Read only but can be configured in Window Maker.
.AlarmMinDevDisabled	Disables/enables the **Minor Deviation** limit for analog tagnames with alarms.
.AlarmMinDevEnabled	Disables/enables the **Minor Deviation** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dot field, but of opposite polarity.
.AlarmMinDevInhibitor	Returns the **Alarm Inhibitor Tag** reference for the **Minor Deviation** limit. Applies to analog tagnames with alarms. Read only but can be configured in Window Maker.
.AlarmName	Returns the name of the alarm.
.AlarmOprName	Returns the operator's name associated with the alarm.
.AlarmOprNode	Returns the operator's node associated with the alarm.
.AlarmPri	Returns the priority value (1-999) for the alarm.
.AlarmProv	Returns the provider for the alarm.
AlarmROC	Signals that a Rate-of-Change type alarm exists.
.AlarmROCCount	Tracks the total number of rate-of-change alarms active on a given tagname or alarm group.
.AlarmROCDisabled	Disables/enables the **Rate-of-Change** limit for analog tagnames with alarms.

Dotfields	Description
.AlarmROCEnabled	Disables/enables the **Rate-of-Change** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dot field, but of opposite polarity.
.AlarmROCInhibitor	Returns the inhibitor tagname reference for the **Rate-of-Change** limit. Applies to analog tagnames with alarms. Read only but can be configured in Window Maker.
.Alarm ROCUnAckCount	Tracks the total number of unacknowledged rate-of-change alarms on a given tagname or alarm group.
.AlarmState	Returns the state of the alarm.
.AlarmTime	Returns the time of the alarm.
.AlarmTotalCount	Tracks the total number of alarms active on a given tagname or alarm group.
.AlarmType	Returns the type of the alarm.
AlarmUnAckCount	Tracks the total number of unacknowledged alarms active on a given tagname or alarm group.
.AlarmUserDefNum1	Read/write real (floating point), default 0 and value not set. Applies to discrete tagnames with alarms, to analog tagnames with alarms, or to alarm groups. **Note** The value of this dot field is attached to the alarm, but ONLY if a value has been set, for example, via a script or a POKE.
.AlarmUserDefNum2	Read/write real (floating point), default 0 and value not set. Applies to discrete tags with alarms, to analog tags with alarms, or to alarm groups. **Note** The value of this dot field is attached to the alarm, but ONLY if a value has been set, for example, via a script or a POKE.
.AlarmUserDefStr	Read/write text string, default "" and value not set. Applies to discrete tags with alarms, to analog tags with alarms, or to alarm groups. **Note** The value of this dot field is attached to the alarm, but ONLY if a value has been set, for example, via a script or a POKE.

Note The **.AlarmUserDefNum1, .AlarmUserDefNum2**, and **AlarmUserDefStr** dotfields allow you to assign one or more values to be attached to the alarm record when it is reported. These values are written to the database by Alarm DB Logger. There are three items you can attach to an alarm: two numbers and a string. By default, they are empty (zero and "").

To simplify setting user values, you can set them on an alarm group as well as on a specific tagname. For example, InBatch could set the batch number in **.AlarmUserDefNum1** all the way up at the **$System** alarm group, causing all alarms to have the batch number attached.

If you set **.AlarmUserDefNum1** on an Alarm Group, it applies to all alarms in that group and any of its sub-groups. You can also specifically set the value of **.AlarmUserDefNum1** on a tagname. In this case, it applies only to that tagname and overwrites any setting of **.AlarmUserDefNum1** in the tagname's Alarm Group.

.

Dotfields	Description
AlarmValDeadband	Read/write analog (only valid for integer or real tagnames) tagname dotfields that monitors and/or controls the value of an alarm's deadband. This field is valid for Alarm Groups as well as ordinary tagnames.
.AlarmValue	Returns the value of the alarm.
.AlarmValueCount	Tracks the total number of value alarms active on a given tagname or alarm group.
.AlarmValueUnAckCount	Tracks the total number of unacknowledged value alarms active on a given tagname or alarm group.
.ChartLength	Read/write integer tagname dotfields used to control the length of time displayed in a Historical Trend graph. **.ChartLength** displays the length of the chart in seconds.
.ChartStart	Read/write integer tagname dotfields used to control the starting time and/or to scroll the corresponding historical trend chart. **.ChartStart** displays the number of elapsed seconds since 12:00 a.m., 1/1/70.
.Comment	Read only message tagname dotfields that is used to display the comment field entered for a tagname in the Tagname Dictionary.
.DevTarget	Read/write analog (only valid for integer or real tagnames) tagname dotfields that monitors and/or controls the target for minor and major deviation alarms.
.DisplayMode	Read/write analog tagname dotfields used to determine the method to be used in displaying values on the trend.

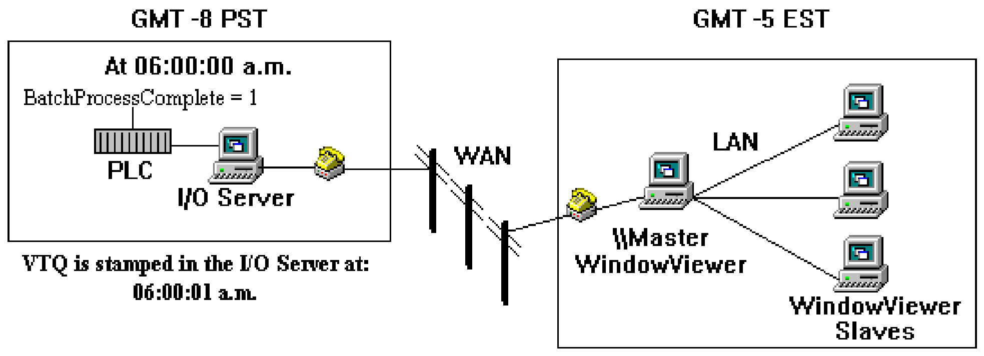
Dotfields	Description
.EngUnits	Read/write analog tagname dotfields used to access the engineering units of an analog tagname as specified in the tagname dictionary. **Note** Writing to these values is not retentive.
.Freeze	Reads/writes the freeze status of the Distributed Alarm Display Object.
.HiLimit, .HiHiLimit, .LoLimit, .LoLoLimit	Read/write analog tagname dotfields that monitors/controls the limits for value alarm checks. These dotfields are only valid for integer and real tagnames.
.HiStatus, .HiHiStatus, .LoStatus, .LoLoStatus	Read only discrete tagname dotfields that determines whether an alarm of a specified type exists.
.ListChanged	Indicates whether there are any new alarms or updates for the Distributed Alarm Object.
.MajorDevPct	Read/write real tagname dotfields that monitors or controls the major percentage of deviation for alarm checking.
.MajorDevStatus	Read only discrete tagname dotfields that determines whether a major deviation alarm exists for the specified tagname.
.MaxEU, .MinEU	Read only integer tagname dotfields that displays the maximum and minimum values for the tagname.
.MaxRange, .MinRange	Read/write real tagname dotfields used to represent the percentage of the tagname's Engineering Unit range that should be displayed for each tagname being trended. The limits for **.MaxRange** and **.MinRange** are from **0** to **100,** and **.MinRange** should always be less than **.MaxRange**. If a value less than **0** or greater than **100** is assigned to either of these fields, the value will be clamped at **0** or **100**. If **.MinRange** is greater than or equal to **.MaxRange**, the trend will not display any data.
.MaxRaw, .MinRaw	The high/low clamp setting for the actual raw value received from an I/O Server by WindowViewer as a client. The value for .**MaxRaw/.MinRaw** tagname dotfields comes from the Max/Min Raw value field in tagname database for the specified I/O tagname. Any raw value which exceeds or falls below this setting is clamped to this value.

Dotfields	Description
.MinorDevPct	Read/write real tagname dotfields used to monitor and/or control the minor percent of deviation for alarm checking.
.MinorDevStatus	Read only discrete tagname dotfields used to determine whether a minor deviation alarm exists for the specified tagname.
.Name	Read/write message tagname dotfields used to display the actual name of the tagname. For example, the name of a **TagID** tagname.
.Normal	Read only discrete tagname dotfields that is equal to 1 when there are no alarms for the specified name. This dotfields is valid for Alarm Groups and ordinary tagnames.
.OffMsg, .OnMsg	Read/write message tagname dotfields used to display the on message and off message specified in the Tagname Dictionary for a discrete tagname. **Note** Writing to these values is not retentive.
Pen1 - .Pen8	Read/write **TagID** type tagname dotfields used to control the tagname being historically trended by each pen. **Note** An easier method of dynamically assigning tagnames to Pens is to use the Historical functions **HTSetPenName** and **HTGetPenName**. For more information, see Chapter 12, "Real-time and Historical Trending." For more information on the TagID tagname type, see "Tagname Types."
.Quality	Read only integer tagname allows the user to access the quality of an I/O tagname as provided by an I/O server.
.QualityLimit	Read only integer tagname dotfields used to display the quality limit of an I/O value provided by data source when the I/O connection is valid.
.QualityLimitString	Read only message tagname dotfields used to display the quality limit string of an I/O value provided by an I/O server when the I/O connection is valid.
.QualityStatus	Read only integer tagname dotfields used to display the quality status of an I/O value provided by an I/O server when the I/O connection is valid.

Dotfields	Description
.QualityStatusString	Read only message tagname dotfields used to display the quality status string of an I/O value provided by an I/O server when the I/O connection is valid.
.QualitySubstatus	Read only integer tagname dotfields used to display the quality sub-status of an I/O value provided by an I/O server when the I/O connection is valid.
.QualitySubstatusString	Read only message tagname dotfields used to display the quality sub-status string of an I/O value provided by an I/O server when the I/O connection is valid. **Note** If the I/O connection becomes invalid, the quality dotfields are automatically reset to the initial value of zero. The **.ReferenceComplete** flag is also set to zero at this time.
.RawValue	Any type (real/discrete) tagname dotfields that is used to display the actual discrete or analog I/O value before InTouch applies scaling.
.Reference	Read/write tagname dotfields used with I/O type tagnames to dynamically change the address of the tagname's source.
.ReferenceComplete	Discrete tagname dotfields that provides a confirmation that the item requested is the same reflected in the **.Value** field. For more information on the .Reference dotfields, see "Dynamic Reference Addressing (DRA)."
.ROCPct	Read/write tagname dotfields used to monitor and/or control the rate-of-change for alarm checking.
.ROCStatus	Read only discrete tagname dotfields used to determine whether a Rate-of-Change alarm exists for the specified tagname.
.ScooterLockLeft	Read/write discrete tagname dotfields. When the value of this field is TRUE, the RIGHT scooter cannot move to the left of the left scooter's position. (0=FALSE, 1=TRUE).
.ScooterLockRight	Read/write discrete tagname. field. When the value of this field is TRUE, the LEFT scooter cannot move to the right of the right scooter's position. (0=FALSE, 1=TRUE).
.ScooterPosLeft	Read/write real tagname dotfields that represents the position of the left scooter (range 0.0 to 1.0).

Dotfields	Description
.ScooterPosRight	Read/write real tagname dotfields that represents the position of the right scooter (range 0.0 to 1.0).
.SuppressRetain	Reads/writes the suppress retain status of the distributed alarm display object.
.TagID	Read only **TagID** tagname dotfields used in conjunction with the Historical Trend **.Pen1-.Pen8 TagID** tagnames to monitor and/or control the tagname being trended by a pen (see previous "Pen" dotfields description).

The following illustrates how the **.Time** fields below receive their data:



VTQ is stamped in the I/O Server at: 06:00:01 a.m.

\\Master Node receives data at 09:00:08 a.m. local time and forwards VTQ to Slaves with local time conversion: 09:00:01 a.m.

Therefore, we know from a global time standpoint that, BatchProcessComplete was set to 1, seven seconds earlier.

Dotfields	Description
.TimeDate	Real integer tagname dotfields used to display the whole number of days which have passed since an I/O value was provided by an I/O server when the I/O connection was valid.
.TimeDateString	Read only message tagname dotfields used to displays the date in the same format set in the Windows registry. For example, 10/31/1997 that an I/O value was provided by an I/O server when the I/O connection was valid.
.TimeDateTime	Read only real tagname dotfields used to display the fractional number of days which have passed since an I/O value was provided by an I/O server when the I/O connection was valid.
.TimeDay	Read only integer tagname dotfields used to display the day an I/O value was provided by an I/O server when the I/O connection is valid. (value may be 1-31).
.TimeHour	Read only integer tagname dotfields used to display the hour of the day that an I/O value was provided by an I/O server when the I/O connection was valid. (value may be 0-23).
.TimeMinute	Read only integer tagname dotfields used to display the minute that an I/O value was provided by an I/O server when the I/O connection was valid. (value may be 0-59).
.TimeMonth	Read only integer tagname dotfields used to display the month that an I/O value was provided by an I/O server when the I/O connection was valid. (value may be 1-12).
.TimeMsec	Read only integer tagname dotfields used to display the time in milliseconds that an I/O value was provided by an I/O server when the I/O connection was valid.
.TimeSecond	Read only integer tagname dotfields used to display the time in seconds that an I/O value was provided by an I/O server when the I/O connection was valid. (value may be 0-59).
.TimeTime	Read only integer tagname dotfields used to display the time in milliseconds (since midnight) that an I/O value was provided by an I/O server when the I/O connection was valid.
.TimeTimeString	Read only message tagname dotfields used to display the time and day of an I/O value provided by an I/O server when the I/O connection is valid.

Dotfields	Description
.TimeYear	Read only integer tagname dotfields used to display the time in the same format set in the Windows registry. (for example, 12:09:45) that an I/O value was provided by an I/O server when the I/O connection was valid. (This time format is set through the Windows Control Panel.)
.Unack	Read/write discrete tagname dotfields used to control the alarm unacknowledgement status of local alarm(s).
.UpdateCount	Read-only integer tagname dotfields that is incremented when a retrieval is complete for the trend.
.UpdateInProgress	Read-only discrete tagname dotfields that shows historical data retrieval status (0=no retrieval in progress, 1=retrieval in progress).
.UpdateTrend	Write only discrete tagname dotfields that can be set to 1 to cause a Historical Trend chart to update using all current values. Historical Trends do not automatically update themselves. A change must be made to either the chart start, chart length, etc. in order for the chart to update and display the current values for the specified tagnames. Using this dotfields in a button QuickScript will allow the operator to update the chart whenever necessary during runtime.
.Value	Read or Read/write analog tagname dotfields that displays the value of the specified tagname.

Several of the alarm **dotfields** are associated with an alarm object, not a tagname.

.

AlarmClass	**.AlarmAccess**	**.AlarmDate**
.AlarmGroupSel	**.AlarmLimit**	**.AlarmName**
.AlarmOprName	**.AlarmOprNode**	**.AlarmPri**
.AlarmProv	**.AlarmState**	**.AlarmTime**
.AlarmType	**.AlarmValue**	**.Freeze**
.ListChanged	**.NumAlarm**	**.PendingUpdates**
.SuppressRetain		

These alarm **dotfields** are accessed by using the function:

GetPropertyM(ControlName.Property, MsgTag).

When executed at runtime, the property is obtained in the **MsgTag** to display for the tagname in the selected row. If multiple rows are selected, the property in the **MsgTag** is the tagname in the first row of the multiple selection.

For more information on alarm **dotfields**, see your online *InTouch Reference Guide*.

Addressing Bit Fields for Analog Tagnames

Single bits within an integer tagname can be addressed by using the bit dotfields. These are all considered to be discrete (0/1) and if written, the analog tagname is promptly updated. You can use the bit dotfields anywhere that a discrete tagname can be used. For example, in I/O, scripts, expressions, and so on.

Bit Dotfield	Description
.00	Least significant bit
.01	next more significant bit
.02	etc.
.	
.	
.	
.31	Most significant bit in the 32-bit integer

The following is an example of how to use the bit fields in an expression:

```
Temperature.08 == 1;
```

The following is an example of how to use the bit fields in a QuickScript:

```
IF Temperature.29 THEN
   Temperature.29 =0;
ENDIF;
```

Tagname Dictionary Utilities

There are two Tagname Dictionary utility programs; DBDump and DBLoad. DBDump is used to export an InTouch application Tagname Dictionary as a text file that can be viewed or edited in another program such as Microsoft Excel. DBLoad allows a properly formatted Tagname Dictionary file (created in another program such as Excel or a DBDump file from another InTouch application) to be loaded into an existing InTouch application. These two programs allow a database (Tagname Dictionary) to be copied, modified or developed in separate portions and then merged into one application.

The DBLoad utility can also be used as an alternative to the InTouch TemplateMaker to create SuperTag instances.

For more information, see "Creating SuperTag Instances."

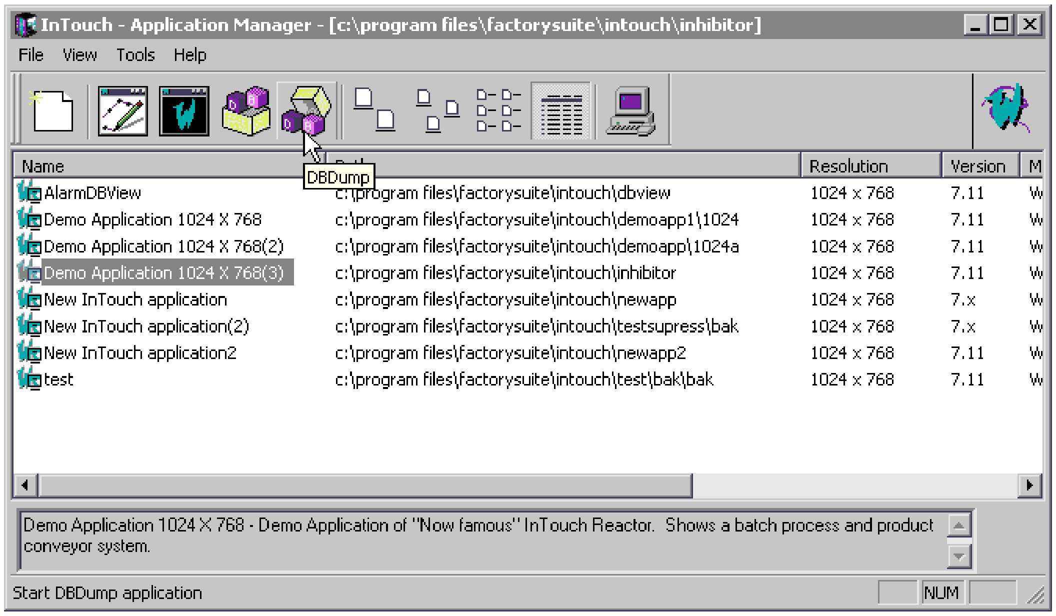
Note Both the DBDump and DBLoad utilities are launched from the InTouch Application Manager (INTOUCH.EXE). Also, you must convert an application created in an earlier version of InTouch before its Tagname Dictionary can be extracted.

For more information on creating database files, see "Creating a Database Input File."

DBDump Utility Program

To extract an existing InTouch application's Tagname Dictionary

1. Close WindowMaker and WindowViewer if they are running.
2. Start InTouch. The **InTouch Application Manager** dialog box appears.

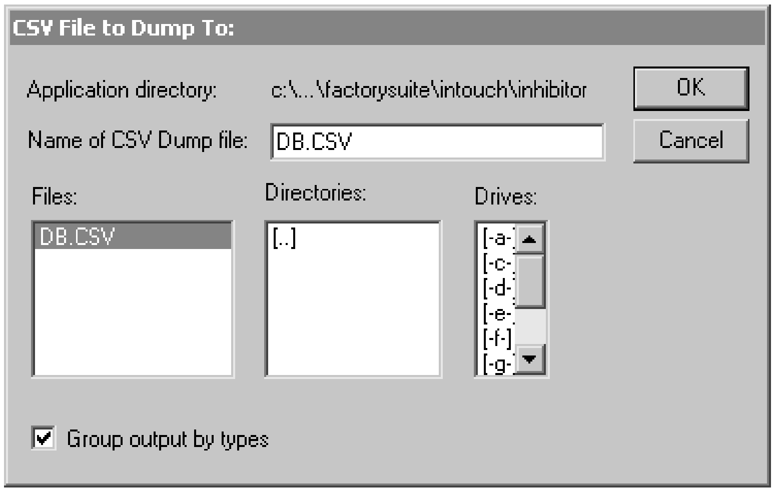

InTouch - Application Manager - [c:\program files\factorysuite\intouch\inhibitor]
File View Tools Help
DBDump

Name	Path	Resolution	Version	M
AlarmDBView	c:\program files\factorysuite\intouch\dbview	1024 x 768	7.11	W
Demo Application 1024 X 768	c:\program files\factorysuite\intouch\demoapp1\1024	1024 x 768	7.11	W
Demo Application 1024 X 768(2)	c:\program files\factorysuite\intouch\demoapp\1024a	1024 x 768	7.11	W
Demo Application 1024 X 768(3)	c:\program files\factorysuite\intouch\inhibitor	1024 x 768	7.11	W
New InTouch application	c:\program files\factorysuite\intouch\newapp	1024 x 768	7.x	W
New InTouch application(2)	c:\program files\factorysuite\intouch\testsupress\bak	1024 x 768	7.x	W
New InTouch application2	c:\program files\factorysuite\intouch\newapp2	1024 x 768	7.11	W
test	c:\program files\factorysuite\intouch\test\bak\bak	1024 x 768	7.11	W

Demo Application 1024 X 768 - Demo Application of "Now famous" InTouch Reactor. Shows a batch process and product conveyor system.

Start DBDump application

3. On the **File** menu, click **DBDump** or click the DBDump tool. The **CSV File to Dump To:** dialog box appears.


CSV File to Dump To:

Application directory: c:\...\factorysuite\intouch\inhibitor

Name of CSV Dump file: DB.CSV

Files: DB.CSV

Directories: [..]

Drives: [-a-] [-c-] [-d-] [-e-] [-f-] [-g-]

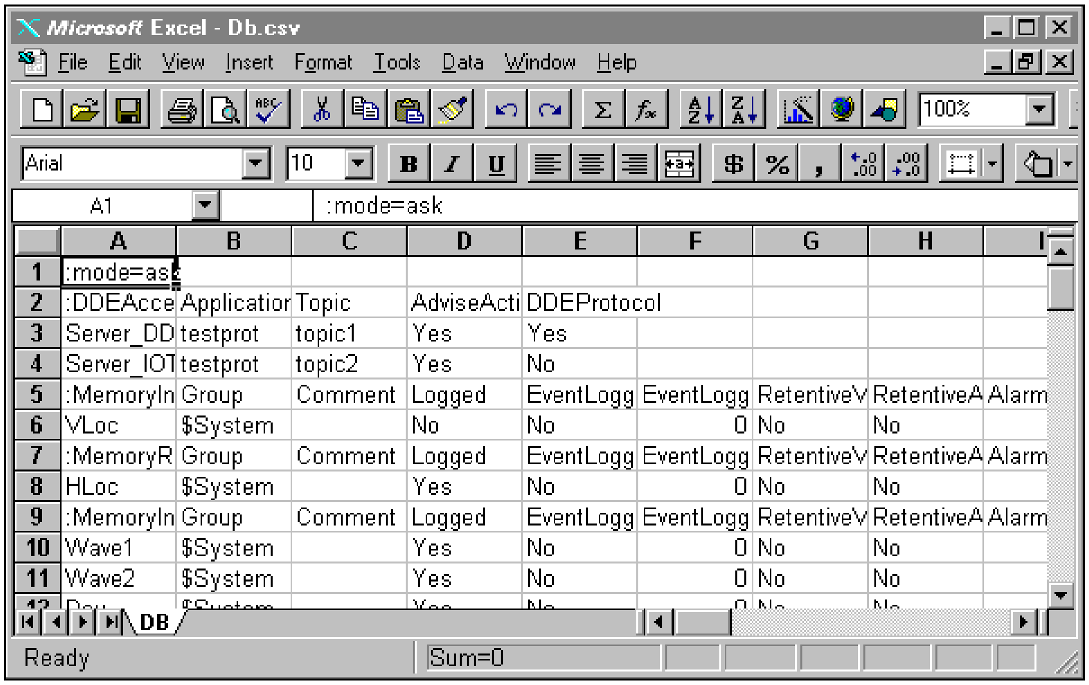
☑ Group output by types

4. In the **Name of CSV Dump file** box, type a name for the file that ends with the .csv (Comma Separated Variable) extension. (If the name already exists, a message box appears.)
5. Select **Group output by types** to group the extracted tagnames by tagname type instead of alphabetically by tagname (default.)

Note In order to extract the database items by groups, the system must read the entire file for each tagname type. Therefore, it takes longer to extract the data. However, when the output file is opened in a .csv supported application such as Microsoft Excel, the grouping makes it much easier to read or edit. A placeholder for each tagname type is included in the dumped file whether tagnames exist for the type or not.

6. Click **OK**. The database information from the selected application directory will be downloaded to the specified filename.

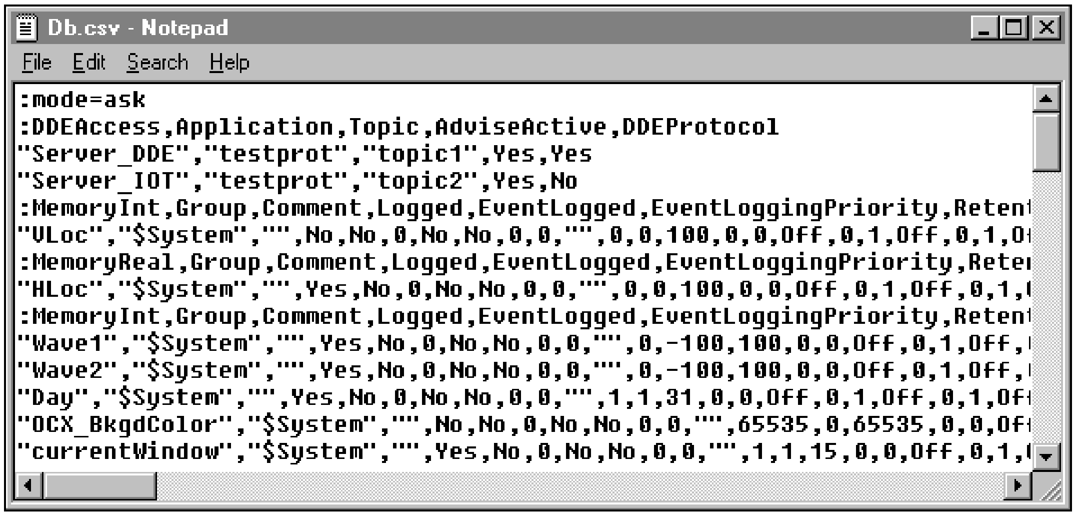
If you open the .csv file in Microsoft Excel, it sees the comma as a delimiter and automatically separates the data records into columns. For example:



:mode=ask

	A	B	C	D	E	F	G	H	I
1	:mode=as								
2	:DDEAcce	Applicatior	Topic	AdviseActi	DDEProtocol				
3	Server_DD	testprot	topic1	Yes	Yes				
4	Server_IOT	testprot	topic2	Yes	No				
5	:MemoryIn	Group	Comment	Logged	EventLogg	EventLogg	RetentiveV	RetentiveA	Alarm
6	VLoc	$System		No	No	0	No	No	
7	:MemoryR	Group	Comment	Logged	EventLogg	EventLogg	RetentiveV	RetentiveA	Alarm
8	HLoc	$System		Yes	No	0	No	No	
9	:MemoryIn	Group	Comment	Logged	EventLogg	EventLogg	RetentiveV	RetentiveA	Alarm
10	Wave1	$System		Yes	No	0	No	No	
11	Wave2	$System		Yes	No	0	No	No	
12	Day	$System		Yes	No	0	No	No	

If you open the .csv file in Notepad, each data record is separate by a comma. For example:



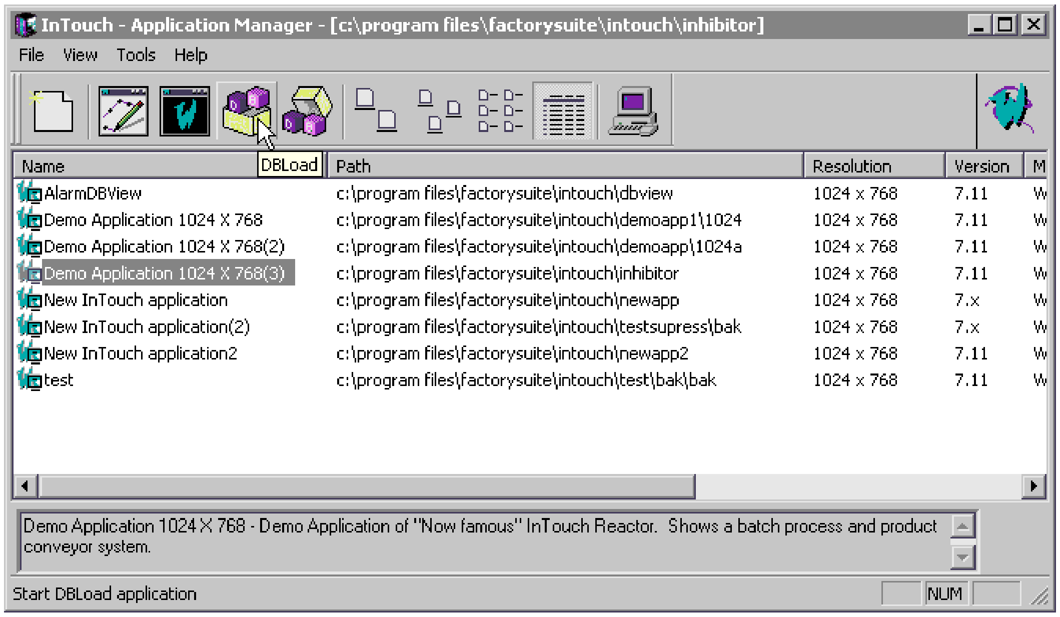
```
:mode=ask
:DDEAccess,Application,Topic,AdviseActive,DDEProtocol
"Server_DDE","testprot","topic1",Yes,Yes
"Server_IOT","testprot","topic2",Yes,No
:MemoryInt,Group,Comment,Logged,EventLogged,EventLoggingPriority,Retent
"VLoc","$System","",No,No,0,No,No,0,0,"",0,0,100,0,0,Off,0,1,Off,0,1,O
:MemoryReal,Group,Comment,Logged,EventLogged,EventLoggingPriority,Rete
"HLoc","$System","",Yes,No,0,No,No,0,0,"",0,0,100,0,0,Off,0,1,Off,0,1,
:MemoryInt,Group,Comment,Logged,EventLogged,EventLoggingPriority,Retent
"Wave1","$System","",Yes,No,0,No,No,0,0,"",0,-100,100,0,0,Off,0,1,Off,
"Wave2","$System","",Yes,No,0,No,No,0,0,"",0,-100,100,0,0,Off,0,1,Off,
"Day","$System","",Yes,No,0,No,No,0,0,"",1,1,31,0,0,Off,0,1,Off,0,1,Of
"OCX_BkgdColor","$System","",No,No,0,No,No,0,0,"",65535,0,65535,0,0,Of
"currentWindow","$System","",Yes,No,0,No,No,0,0,"",1,1,15,0,0,Off,0,1,
```

DBLoad Utility Program

To load/merge a database input file into an existing InTouch

1. Close WindowMaker and WindowViewer if they are running.

2. Start InTouch. The **InTouch Application Manager** dialog box appears.



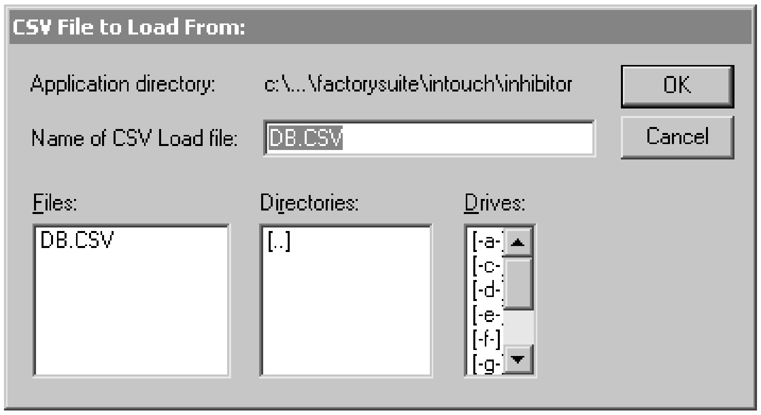
InTouch - Application Manager - [c:\program files\factorysuite\intouch\inhibitor]

Name	Path	Resolution	Version	M
AlarmDBView	c:\program files\factorysuite\intouch\dbview	1024 x 768	7.11	W
Demo Application 1024 X 768	c:\program files\factorysuite\intouch\demoapp1\1024	1024 x 768	7.11	W
Demo Application 1024 X 768(2)	c:\program files\factorysuite\intouch\demoapp\1024a	1024 x 768	7.11	W
Demo Application 1024 X 768(3)	c:\program files\factorysuite\intouch\inhibitor	1024 x 768	7.11	W
New InTouch application	c:\program files\factorysuite\intouch\newapp	1024 x 768	7.x	W
New InTouch application(2)	c:\program files\factorysuite\intouch\testsupress\bak	1024 x 768	7.x	W
New InTouch application2	c:\program files\factorysuite\intouch\newapp2	1024 x 768	7.11	W
test	c:\program files\factorysuite\intouch\test\bak\bak	1024 x 768	7.11	W

Demo Application 1024 X 768 - Demo Application of "Now famous" InTouch Reactor. Shows a batch process and product conveyor system.

Start DBLoad application

3. On the **File** menu, click **DBLoad** or click the DBLoad tool. A message box appears asking if you have backed up your application. Click **Yes** to continue. The **CSV File to Load From:** dialog box appears.



CSV File to Load From:

Application directory: c:\...\factorysuite\intouch\inhibitor

Name of CSV Load file: DB.CSV

4. In the **Name of CSV Load file** box, type the path to the .csv file that you want to load or locate the file using the **Directories** and **Drive** list boxes. (Once the file is properly selected its name appears in the box.)

5. Click **OK**. The database information contained in the selected file will begin uploading to the selected application's Tagname Dictionary.

Creating a Database Input File

The DBDump and DBLoad database utilities are the tools that you use for performing batch type operations on a Tagname Dictionary. Database input files can be created in any program that supports the comma separated variable file format (.csv). (The database input file must be created in the comma separated variable format.) For example, WordPad, Notepad and Microsoft Excel. Once an input file is created, the DBLoad program is used to load/merge the data contained in the file into an existing InTouch application's database.

You can create a database input file "template" by creating a new InTouch application and then running the DBDump program to dump its database to create a correctly formatted .csv file. This makes entering your modifications easier than creating the input file from scratch.

For more information, see "Creating Database Record Templates."

Database Input File Format

The first line of a database input file should specify the operating **:mode** for the file when it is loaded/merged into an application via **DBLoad**.

Tip If you do not specify a mode, **:mode=test**, by default, **Ask** will be used.

For more information on valid mode keywords, see "Database Input File Operating Modes."

All data records must begin with the valid keyword for the tagname's **:type**, followed by the valid keyword for each data record (separated by commas):

```
:mode=test
:IOMsg,Group,Comment,Logged,Event Logged,Event Logging
    Priority, . . .
```

There is a valid **keyword** for each tagname type and data record

For more information on valid tagname types and the data record entries, ":Type and Keyword Entries."

The actual tagname is then entered, followed by the values (separated by commas) for each data record:

```
:mode=test
:IOMsg,Group,Comment,Logged,EventLogged,EventLoggingPriori
    ty, . . .
Ingredient_1,$System,"",No,No,999, . . .
```

In the above example, **IOMsg** = Ingredient_1, **Group** = $System, **Logged** = No, **EventLogged** = No, and **EventLoggingPriority** = 999. The data record **Comment** will be blank since **""** was entered.

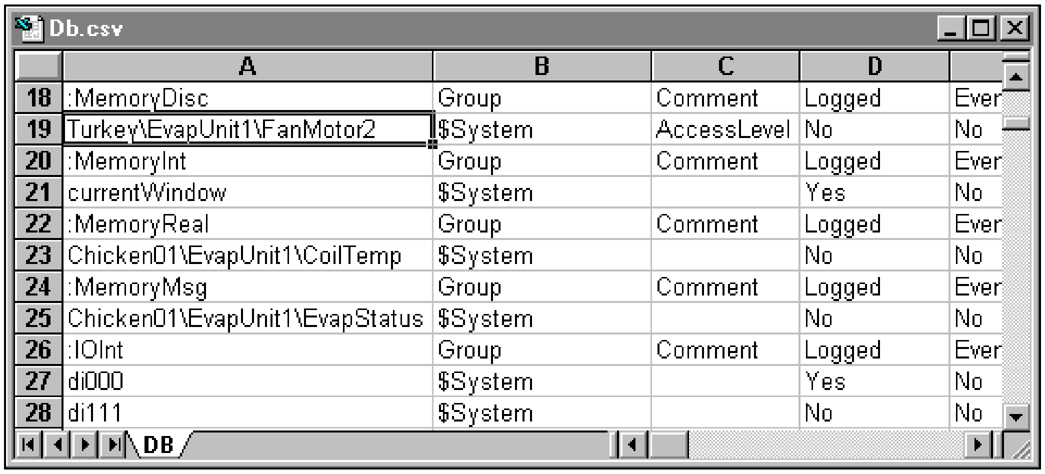
A colon (**:**) must precede the mode and type keywords. To continue a line, a backslash (\) is entered at the end of the line. Comments may be entered by preceding them with a semi-colon (**;**).

Creating SuperTag Instances

In addition to the TemplateMaker, animation links, InTouch QuickScripts and the Tagname Dictionary, InTouch also supports the creation of SuperTags through the DBLoad utility.

Note When you use DBLoad to create SuperTag instances, they are not reflected in the SuperTag template definition in the TemplateMaker.

When you create a SuperTag through DBLoad you must use the valid SuperTag format. and the SuperTag instance data records must begin with the valid keyword for the tagname's **:type**. For example:



Db.csv

	A	B	C	D	
18	:MemoryDisc	Group	Comment	Logged	Ever
19	Turkey\EvapUnit1\FanMotor2	$System	AccessLevel	No	No
20	:MemoryInt	Group	Comment	Logged	Ever
21	currentWindow	$System		Yes	No
22	:MemoryReal	Group	Comment	Logged	Ever
23	Chicken01\EvapUnit1\CoilTemp	$System		No	No
24	:MemoryMsg	Group	Comment	Logged	Ever
25	Chicken01\EvapUnit1\EvapStatus	$System		No	No
26	:IOInt	Group	Comment	Logged	Ever
27	di000	$System		Yes	No
28	di111	$System		No	No

DB

The following syntax examples are valid:

```
ParentInstance\ChildMember
ParentInstance\ChildMember\Submember
```

The following syntax examples are invalid:

```
ParentInstance\
ParentInstance\ChildMember\
```

If an invalid format is used, the an error message box appears informing you that the syntax is invalid.

When you upload the .csv file containing SuperTag instances, they are automatically added to the Tagname Dictionary, and are immediately available for use in animation links and InTouch QuickScripts.

Blank Strings vs. No Entry

There is a difference between string fields which are blank and fields which have no entry. For example:

```
:Comment="HI"
:MemoryDisc,Comment,Group
Tagname1,,$System
Tagname2,"",$System
```

where:

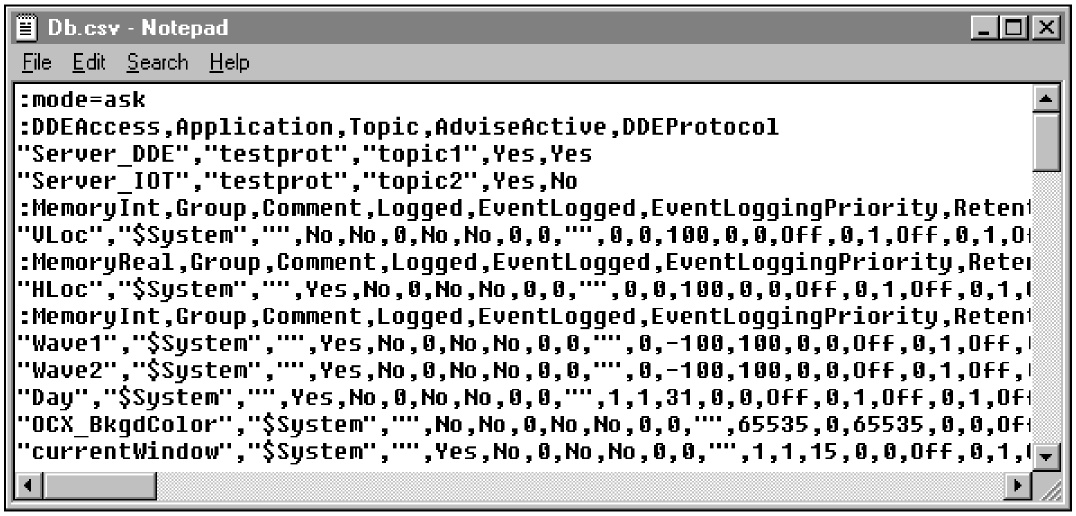
The value of Tagname1's **Comment** field will be **"Hi"** and Tagname2 will have a blank comment. Excel will read in a .csv file such as the one above, but when saving the file, it will save it as follows:

```
:Comment="HI"

:MemoryDisc,Comment,Group

Tagname1,,$System

Tagname2,,$System
```

Thus, to ensure a **blank** string, a space (using the spacebar) must be entered in the cell that is to be **blank**. The following keyword fields are affected in this way:

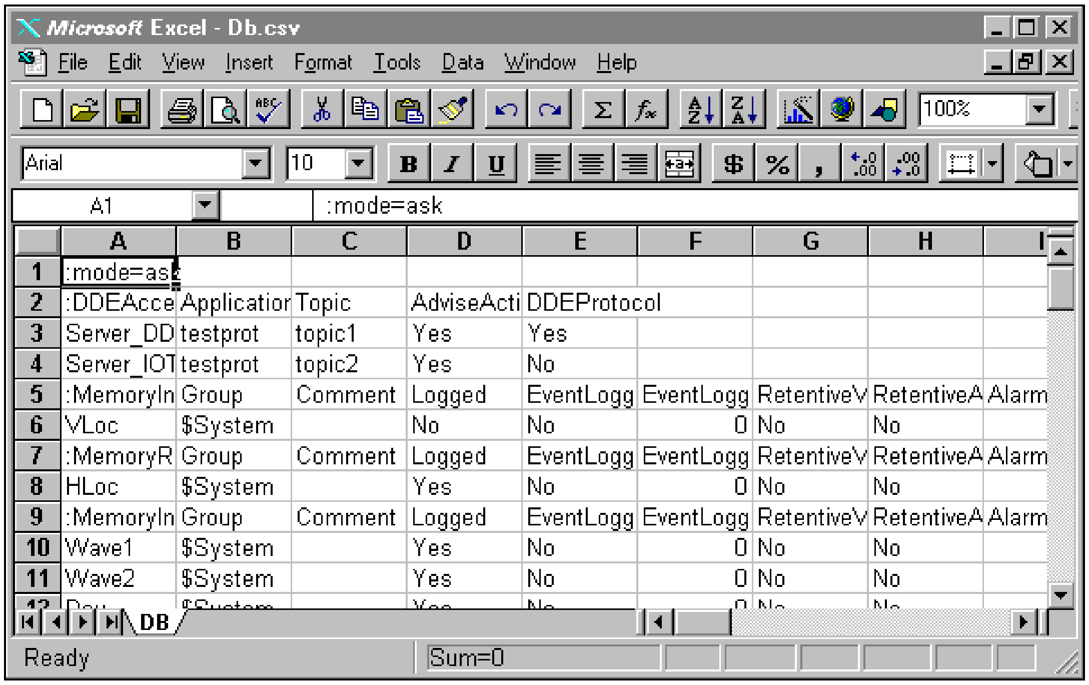
Comment	**Eng Units**
OffMsg	**Initial Message**
OnMsg	**Application**
ItemName	**Topic**

The following illustrates an input file that was created using Windows Notepad:



```
:mode=ask
:DDEAccess,Application,Topic,AdviseActive,DDEProtocol
"Server_DDE","testprot","topic1",Yes,Yes
"Server_IOT","testprot","topic2",Yes,No
:MemoryInt,Group,Comment,Logged,EventLogged,EventLoggingPriority,Reten
"VLoc","$System","",No,No,0,No,No,0,0,"",0,0,100,0,0,Off,0,1,Off,0,1,O
:MemoryReal,Group,Comment,Logged,EventLogged,EventLoggingPriority,Rete
"HLoc","$System","",Yes,No,0,No,No,0,0,"",0,0,100,0,0,Off,0,1,Off,0,1,
:MemoryInt,Group,Comment,Logged,EventLogged,EventLoggingPriority,Reten
"Wave1","$System","",Yes,No,0,No,No,0,0,"",0,-100,100,0,0,Off,0,1,Off,
"Wave2","$System","",Yes,No,0,No,No,0,0,"",0,-100,100,0,0,Off,0,1,Off,
"Day","$System","",Yes,No,0,No,No,0,0,"",1,1,31,0,0,Off,0,1,Off,0,1,Of
"OCX_BkgdColor","$System","",No,No,0,No,No,0,0,"",65535,0,65535,0,0,Of
"currentWindow","$System","",Yes,No,0,No,No,0,0,"",1,1,15,0,0,Off,0,1,
```

When you use a program such as Excel to create an input file, the same formatting requirements apply, except that each entry is made in a separate column. This makes it much easier to read and there is less chance of error. For example:



A1 :mode=ask

	A	B	C	D	E	F	G	H	I
1	:mode=ask								
2	:DDEAcce	Applicatior	Topic	AdviseActi	DDEProtocol				
3	Server_DD	testprot	topic1	Yes	Yes				
4	Server_IOT	testprot	topic2	Yes	No				
5	:MemoryIn	Group	Comment	Logged	EventLogg	EventLogg	RetentiveV	RetentiveA	Alarm
6	VLoc	$System		No	No	0	No	No	
7	:MemoryR	Group	Comment	Logged	EventLogg	EventLogg	RetentiveV	RetentiveA	Alarm
8	HLoc	$System		Yes	No	0	No	No	
9	:MemoryIn	Group	Comment	Logged	EventLogg	EventLogg	RetentiveV	RetentiveA	Alarm
10	Wave1	$System		Yes	No	0	No	No	
11	Wave2	$System		Yes	No	0	No	No	
12	Day	$System		Yes	No	0	No	No	

Sum=0

Database Input File Operating Modes

The following lists the valid operating mode **keywords** and the action which occurs in each mode when a duplicate tagname is encountered during the load:

`:MODE=REPLACE`

`:MODE=UPDATE`

`:MODE=ASK`

`:MODE=IGNORE`

`:MODE=TERMINATE`

`:MODE=TEST`

:MODE=REPLACE

Delete the existing entry and replace it with the new entry.

:MODE=UPDATE

Overwrite the existing definition with only the fields that are explicitly defined in the input file.

Fields are considered explicitly defined if the field is in the record and entered by you or is set by the ":**KEYWORD=value**" mechanism. If a field is not specified in the record and the keyword has been reset via the ":**KEYWORD=**" command, the current field value will not be updated.

The following is an example of what will occur when an input file in the update mode is loaded/merged into an application's database via DBLoad:

```
:Mode=update

:Group=Group1

:IODisc,Group,DConversion

Tagname1,Group2,

; Tagname1's Group updated to Group2 only

Tagname2,,

; Tagname2's Group updated to Group1 and the DConversion
   left as is

Tagname3,,Reverse

; Tagname3's Group updated to Group1 and the DConversion to
   "Reverse"

; the following line "resets" the Group field to its
   default value

:Group=

;  Data field "Group" is reset to its default value

Tagname4,,

; Tagname4 will be left alone
```

Comments are allowed in the .csv file. They must be identified with an opening semi-colon (**;**).

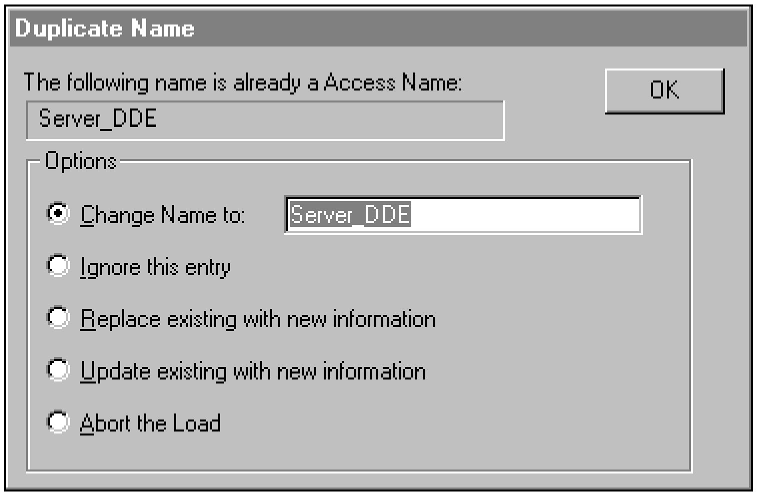
Note The tagname types must be compatible if the type is being changed and the tagname is in use. For example, an existing historical trend tagname cannot be changed to an I/O Integer if the tagname is in use by the application. Also, a tagname cannot be changed to **ReadOnly=yes** if the tagname is being used on an input link in the application. Because of these restrictions, you should update the target application's tagname use counts before running DBLoad.

:MODE=ASK

Change the tagname of the input or the existing entry to your specified tagname and then insert the new definition into the Tagname Dictionary.

Note This is the default input mode if no mode is specified. Each time DBLoad comes across a duplicate tagname, a dialog box appears asking whether the existing tagname is to be replaced.

When the operating mode is set to **ASK**, and a tagname in the input file is a duplicate of a tagname in the target application's Tagname Dictionary, the **Duplicate Name** dialog box appears:


Duplicate Name

The following name is already a Access Name:

Server_DDE

Options

[✓ Change Name to:] Server_DDE

Ignore this entry

Replace existing with new information

Update existing with new information

Abort the Load

Select an option, then click **OK**. A confirmation message box appears when the load is completed.

Option	Description
Change Name to	Replaces the name for the specified tagname with the name typed in the box.
Ignore this entry	Ignores the displayed tagname and processing continues.
Replace existing with new information	Replaces the existing tagname record with the new record.
Update existing with new information	Overwrites the existing tagname record with only the fields that are explicitly defined in the input file.
Abort the Load	Cancels the load function.

Note If a problem occurs that causes the load to fail, a message box appears. The error messages are written to the Logger.

:MODE=IGNORE

Ignore the new record and continue processing.

:MODE=TERMINATE

Terminate processing. Do not update target file.

:MODE=TEST

In this mode, DBLoad will act as if it were in the replace mode, but will not modify the database. It will report errors as found in the Logger and continue with the load. This is useful for verifying the syntax of the input file before actually processing it.

Note Once the input file is set to the test mode, all other modes are ignored. You can enter :**mode=test** as the first line of the file and not be concerned about other mode changes in the file.

For more information on operating modes, see "Creating a Database Input File."

Creating Database Record Templates

KEYWORDs can be used to create template records that supply a "global" entry for the respective data fields in subsequent data records. (There is a keyword for each field value that can be set for the tagname except for the **TAGNAME** and **TYPE** fields.)

Note A template record must follow the formatting requirements previously described for creating a comma separated input file. For example the file must begin with a **:Type** keyword entry.

Setting Field Value Defaults

The keywords can also be used to set the default values for specific fields of a record. For example:

```
:KEYWORD=value
```

This will set the default value of the referenced field for all subsequent data records. This feature can be used to set the default value for fields that will remain unchanged for a number of records. If a field has a default value defined, the default value is used if there is no data in the record for the value. For example, the result of **:GROUP=Reactor_Site** would be that all tagnames having a blank entry for the **GROUP** column will be assigned to the **Reactor_Site** Alarm Group. If the tagname has, for example, **$System** entered for the **GROUP**, it remains assigned to the Alarm Group **$System**.

Resetting Field Value Defaults

The individual keywords can be reset to their original default values by not specifying a value. For example, **:GROUP=**.

Resetting All Field Value Defaults

To reset all keywords, use the **:RESET** command. This command is entered "as is" with no arguments (**:RESET**) and will affect all entries below the command.

Note Default values are the original InTouch values for that tagname type. For example, a memory discrete uses the **Group=$System**, **EventLogging=Off**, **InitialValue=Off**, and so on for its default values.

:Type and Keyword Entries

The table below lists the valid keywords for each tagname **:Type** followed by the keyword for each field value for that type.

:Type & Keywords	Acceptable Values	Default
:MemoryDisc		
ACKModel	Integer [0=condition, 1=event, or 2=expanded]	0
AlarmComment	Any text string, read/write, [Applies to all tag types.]	Blank **Note**: For InTouch 7.1 or earlier, same as tagname description comment
AlarmPri	1 to 999	1
AlarmState	None, On, or Off	None
Comment	Any Text String	""

:Type & Keywords	Acceptable Values	Default
DscAlarmDisable	1, 0	0
DscAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
InitialDisc	1, 0, On, Off, True, False, Yes, or No	0
Logged	On, Off, Yes, or No	No
OffMsg	Any Text String	None
OnMsg	Any Text String	None
RetentiveValue	1, 0, On, Off, True, False, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
:IODisc		
ACKModel	Integer [0=condition, 1=event, or 2=expanded]	0
AlarmComment	Any text string, read/write, [Applies to all tag types.]	Blank **Note**: For InTouch 7.1 or earlier, same as tagname description comment
AlarmPri	1 to 999	1
AlarmState	None, On, or Off	None
Comment	Any Text String	None
DConversion	Direct or Reverse	Direct
AccessName	Valid Access Name	None
DscAlarmDisable	1, 0	0
DscAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
InitialDisc	1, 0, On, Off, True, False, Yes, or No	0
ItemName	Valid Item Name	None
ItemUseTagname	True, False, Yes, or No	Yes
Logged	On, Off, Yes, or No	No
OffMsg	Any Text String	None
OnMsg	Any Text String	None
ReadOnly	Yes or No	No
RetentiveValue	1, 0, On, Off, True, False, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
:MemoryInt & :Memory Real		
ACKModel	Integer [0=condition, 1=event, or 2=expanded]	0
AlarmComment	Any text string, read/write, [Applies to all tag types.]	Blank **Note**: For InTouch 7.1 or earlier, same as tagname description comment
AlarmDevDeadband	Valid Deviation Percentage	0
AlarmValueDeadband	Valid Integer or Real Value	0
Comment	Any Text String	None
Deadband	Valid Integer of Real Value	0
DevTarget	Valid Integer or Real Value	0
EngUnits	Any Text String	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
HiAlarmPri	1 to 999	1
HiAlarmState	Yes, No, On, or Off	No
HiAlarmValue	Valid Integer or Real Value	0
HiHiAlarmPri	1 to 999	1
HiHiAlarmState	Yes, No, On, or Off	No
HiHiAlarmValue	Valid Integer or Real Value	0
InitialValue	Valid Integer or Real Value	0
LoAlarmPri	1 to 999	1
LoAlarmState	Yes, No, On, or Off	No
LoAlarmValue	Valid Integer or Real Value	0
LogDeadband	Valid Integer or Real Value	0
Logged	On, Off, Yes, or No	No
LoLoAlarmPri	1 to 999	1
LoLoAlarmState	Yes, No, On, or Off	No
LoLoAlarmValue	Valid Integer or Real Value	0
MajorDevAlarmPri	1 to 999	1
MajorDevAlarmState	Yes, No, On, or Off	No
MajorDevAlarmValue	Valid Integer or Real Value	0
MaxValue	Valid Integer or Real Value	9999
MinorDevAlarmPri	1 to 999	1

:Type & Keywords	Acceptable Values	Default
MinorDevAlarmState	Yes, No, On, or Off	No
MinorDevAlarmValue	Valid Integer or Real Value	0
MinValue	Valid Integer or Real Value	0
RetentiveAlarmParameters	On, Off, Yes, or No	No
RetentiveValue	On, Off, Yes, or No	No
ROCAlarmPri	1 to 999	1
ROCAlarmState	Yes, No, On, or Off	No
ROCAlarmValue	Any Valid Percentage	0
ROCTimeBase	Sec, Min, or Hr	Min

Alarm Inhibitor Tagnames for Alarm Types	Acceptable Values	Default
HiAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
HiHiAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
LoAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
LoLoAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
MajDevAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
MinDevAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
ROCAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None

Disable Flags	**Acceptable Values**	**Default**
HiAlarmDisable	1, 0	0
HiHiAlarmDisable	1, 0	0
LoAlarmDisable	1, 0	0
LoLoAlarmDisable	1, 0	0
MajDevAlarmDisable	1, 0	0
MinDevAlarmDisable	1, 0	0
ROCAlarmDisable	1, 0	0

:Type & Keywords	**Acceptable Values**	**Default**
:IOInt & :IOReal		
ACKModel	Integer [0=condition, 1=event, or 2=expanded]	0
AlarmComment	Any text string, read/write, [Applies to all tag types.]	Blank **Note**: For InTouch 7.1 or earlier, same as tagname description comment
AlarmDevDeadband	Valid Deviation Percentage	0
AlarmValueDeadband	Valid Integer or Real Value	0
Comment	Any Text String	None
Conversion	Linear or Square Root	Linear
AccessName	Valid Access Name	None
Deadband	Valid Integer or Real Value	0
DevTarget	Valid Integer or Real Value	0
EngUnits	Any Text String	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
HiAlarmPri	1 to 999	1
HiAlarmState	Yes, No, On, or Off	No
HiAlarmValue	Valid Integer or Real Value	0
HiHiAlarmPri	1 to 999	1
HiHiAlarmState	Yes, No, On, or Off	No
HiHiAlarmValue	Valid Integer or Real Value	0
InitialValue	Valid Integer or Real Value	0
ItemName	Valid Item Name	None
ItemUseTagname	True, False, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
LoAlarmPri	1 to 999	1
LoAlarmState	Yes, No, On, or Off	No
LoAlarmValue	Valid Integer or Real Value	0
LogDeadband	Valid Integer or Real Value	0
Logged	On, Off, Yes, or No	No
LoLoAlarmPri	1 to 999	1
LoLoAlarmState	Yes, No, On, or Off	No
LoLoAlarmValue	Yes, No, On, or Off	0
MajorDevAlarmPri	1 to 999	1
MajorDevAlarmState	Yes, No, On, or Off	No
MajorDevAlarmValue	Valid Integer or Real Value	0
MaxEU	Valid Integer or Real Value	9999
MaxRaw	Valid Integer or Real Value	9999
MinEU	Valid Integer or Real Value	0
MinorDevAlarmPri	1 to 999	1
MinorDevAlarmState	Yes, No, On, or Off	No
MinorDevAlarmValue	Valid Integer or Real Value	0
MinRaw	Valid Integer or Real Value	0
ReadOnly	Yes or No	No
RetentiveAlarmParameters	On, Off, Yes, or No	No
RetentiveValue	On, Off, Yes, or No	No
ROCAlarmPri	1 to 999	1
ROCAlarmState	Yes, No, On, or Off	No
ROCAlarmValue	Any Valid Percentage	0
ROCTimeBase	Sec, Min, or Hr	Min

Alarm Inhibitor Tagnames for Alarm Types	Acceptable Values	Default
HiAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
HiHiAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
LoAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
LoLoAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
MajDevAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None

Alarm Inhibitor Tagnames for Alarm Types	Acceptable Values	Default
MinDevAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
ROCAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None

Disable Flags	Acceptable Values	Default
HiAlarmDisable	1, 0	0
HiHiAlarmDisable	1, 0	0
LoAlarmDisable	1, 0	0
LoLoAlarmDisable	1, 0	0
MajDevAlarmDisable	1, 0	0
MinDevAlarmDisable	1, 0	0
ROCAlarmDisable	1, 0	0

:Type & Keywords	Acceptable Values	Default
:MemoryMsg		
Comment	Any Text String	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
InitialMessage	Any Text String	None
Logged	On, Off, Yes, or No	No
MaxLength	1-131 characters	131
RetentiveValue	On, Off, Yes, or No	No
Comment	Any Text String	None
AccessName	Valid Access Name	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
InitialMessage	Any Text String	None
ItemName	Valid Item Name	None
ItemUseTagname	True, False, Yes, or No	No
Logged	On, Off, Yes, or No	No
MaxLength	1-131 characters	131
ReadOnly	Yes or No	No
RetentiveValue	On, Off, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
:GroupVar		
Comment	Any Text String	None
Group	Valid Group Name	$System
EventLogged	On, Off, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
:HistoryTrend		
Comment	Any Text String	None
Group	Valid Group Name	$System

:Type & Keywords	Acceptable Values	Default
:TagID		
Comment	Any Text String	None
Group	Valid Group Name	$System

:Type & Keywords	Acceptable Values	Default
:IndirectDisc		
Comment	Any Text String	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
RetentiveValue	On, Off, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
:IndirectAnalog		
Comment	Any Text String	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
RetentiveValue	On, Off, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
:IndirectMsg		
Comment	Any Text String	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
RetentiveValue	On, Off, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
:Access		
AdviseActive	Yes or No	Yes
NodeName	Valid network node name	None
Application	Valid I/O application name	None
Topic	Valid I/O topic name	None
DDEProtocol	Yes specifies the DDE protocol No specifies the SuiteLink protocol	Yes

:Type & Keywords	Acceptable Values	Default
:AlarmGroup		
Comment	Any Text String	None
EventLogged	On, Off, Yes, or No	No
EventLoggingPriority	1 to 999	999
Group	Valid Group Name	$System
RetentiveValue	On, Off, Yes, or No	No

:Type & Keywords	Acceptable Values	Default
:AlarmDotfields		
AlarmAckModel	0, 1, 2	0
DscAlarmDisable	1, 0	0
DscAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
HiAlarmDisable	1, 0	0
HiHiAlarmDisable	1, 0	0
LoAlarmDisable	1, 0	0
LoLoAlarmDisable	1, 0	0
MajDevAlarmDisable	1, 0	0
MinDevAlarmDisable	1, 0	0
ROCAlarmDisable	1, 0	0
HiAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
HiHiAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
LoAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
LoLoAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
MajDevAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
MinDevAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None
ROCAlarmInhibitor	Tagname reference: any Discrete or Analog tagnames	None

C H A P T E R 7

Creating Animation Links

Once you create a graphic object or symbol you can "bring it to life" by animating it. By attaching animation links, you can make the object or symbol change in appearance to reflect changes in the value of a tagname or an expression. For example, you can create a pump symbol that is red when it is off and green when it is on. You can also make the pump symbol a touch-sensitive push button that the operator can click with the mouse or touch (when using a touch screen) to turn the pump on and off. You can use these and many other special effects by defining animation links for your objects or symbols.

InTouch supports two basic types of links: Touch Links and Display Links. Touch Links allow operator input into the system. Display Links allow output to the operator. Value sliders or push buttons are examples of Touch Links. Color fill, location or blink links are examples of Display Links.

This chapter describes the procedures you use to create each type of animation link.

Contents

Common Animation Link Features

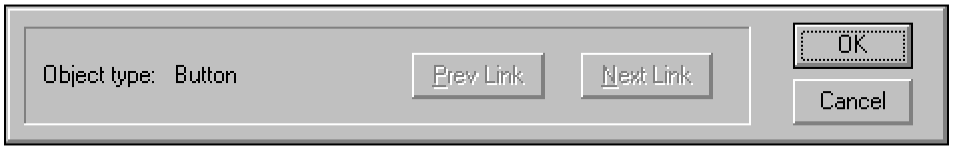
Many of the animation links share the following common controls:

- Object Type Dialog Box
- Common Color Palette
- Quick access to the Tag Browser
- Quick access to the Tagname .Fields
- Support for Key Equivalents

- Right-click mouse support in the **Tagname** or **Expression** input boxes (displays a menu with commands that you can apply to the selected text)

Object Type Dialog Box

The **Object Type** dialog box appears at the top of the screen above the link selection dialog box. It is the header dialog box that is common to all links created. It displays the description of the type of object that you have selected for animation link attachment. For example, **Button**.

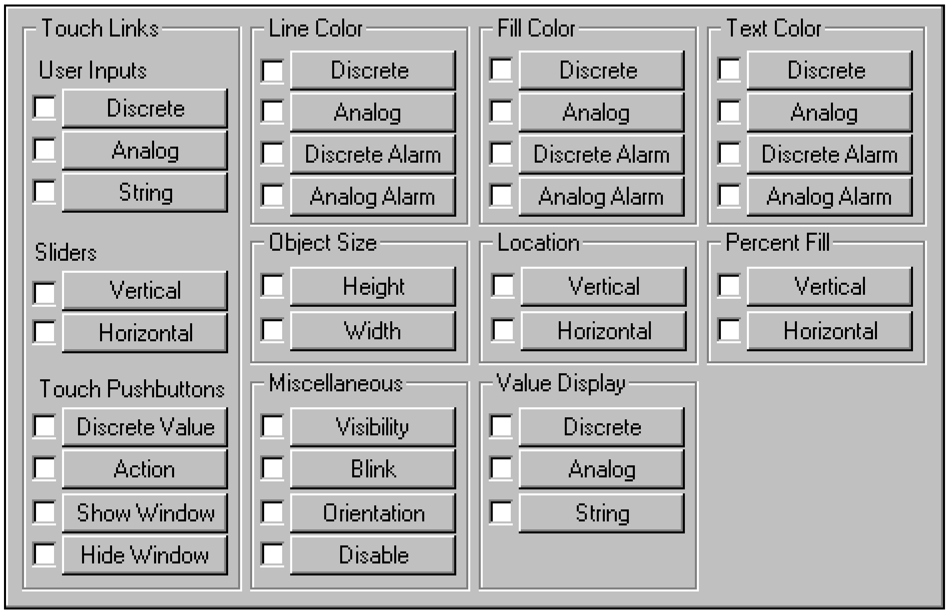

Object type: Button

If multiple links have been attached to an object, you can click **Prev Link** and **Next Link** to quickly page forward or backwards through the link dialog boxes for each link attached to the object.

Tip Links are stored in the order in which they were originally attached to the object.

Animation Link Selection Dialog Box

You can define multiple links for your objects or symbols. By combining various links, you can create almost any screen animation effect imaginable. You can make objects change color, size, location, visibility, fill level, and so on.



Assigning Key Equivalents

You can assign a specific key on the keyboard to activate some animation links. The key equivalent is only operational when the object with the link is visible or selected.

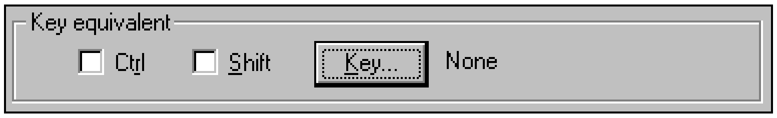
If the object has a visibility or disable link, the key equivalent is not active when the object is invisible or disabled.

You can define the same key in multiple windows. However, the definition in the most recently opened window will be the active one. In the case of overlay windows, the key will be 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 will take precedence over the execution of the **Key Action Script**.

For more information on Key Action Scripts, see Chapter 8, "Creating QuickScripts in InTouch."

The animation links that support key equivalents will display the **Key Equivalent** group in their link dialog boxes. For example:

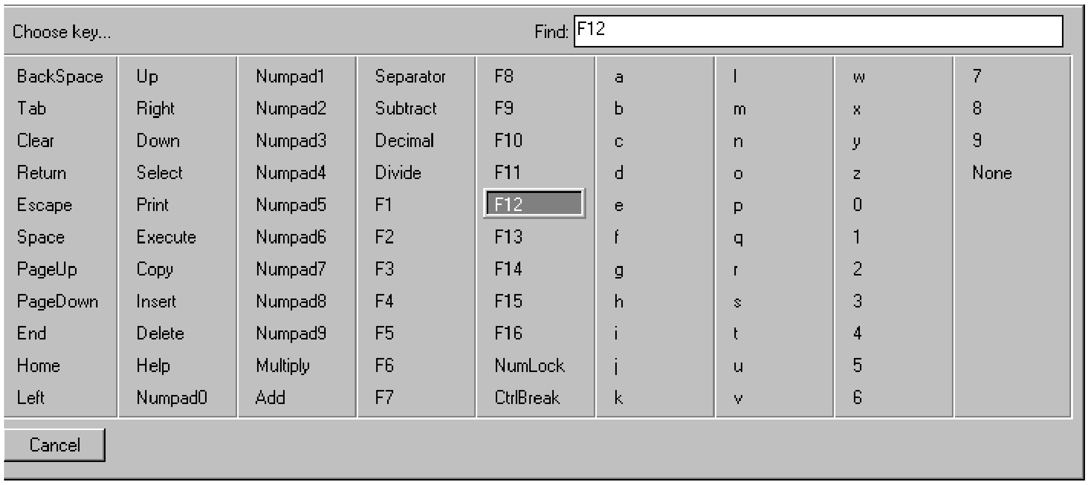


Note Direct choose key links only display function keys 1-16. If you are using a custom keyboard that has more than 16 function keys, you will need to get a device driver from your manufacture that will allow you to access the extended function keys on your system.

To assign a key to a link

1. Select **Ctrl** and/or **Shift** if you want the operator to hold down either or both of these keys when pressing the key equivalent.

2. Click **Key**. The **Choose key** dialog box appears.



Choose key... Find: F12

BackSpace	Up	Numpad1	Separator	F8	a	l	w	7
Tab	Right	Numpad2	Subtract	F9	b	m	x	8
Clear	Down	Numpad3	Decimal	F10	c	n	y	9
Return	Select	Numpad4	Divide	F11	d	o	z	None
Escape	Print	Numpad5	F1	F12	e	p	0	
Space	Execute	Numpad6	F2	F13	f	q	1	
PageUp	Copy	Numpad7	F3	F14	g	r	2	
PageDown	Insert	Numpad8	F4	F15	h	s	3	
End	Delete	Numpad9	F5	F16	i	t	4	
Home	Help	Multiply	F6	NumLock	j	u	5	
Left	Numpad0	Add	F7	CtrlBreak	k	v	6	

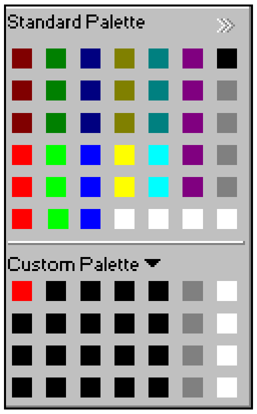
Cancel

3. Click the key that you want to assign to the link. The dialog box will close and the link dialog box reappears displaying the name of the selected key next to the **Key** button.

Applying Color Links

You can apply color to the dynamic properties of lines, rectangles, round rectangles, ellipses, polylines, polygons and text. When you create color links for lines, fill or text objects, you will use the color palette to select the colors that you want linked to the value of the tagname, the tagname's alarm state, and so on.

You must use solid colors for line and text color links. If you select a dithered (mixed) color, WindowMaker, by default, will select the closest solid color. You can create custom color palettes and load them into the standard WindowMaker color palette.



When you attach a color link to an object or symbol and you click on a color box in a link's dialog box, the color palette appears.

Click the color you want to use for the link. The color palette automatically closes and the color you selected appears in the color box in the link detail dialog box.

For more information on customizing the color palette, see Chapter 1, "WindowMaker Program Elements."

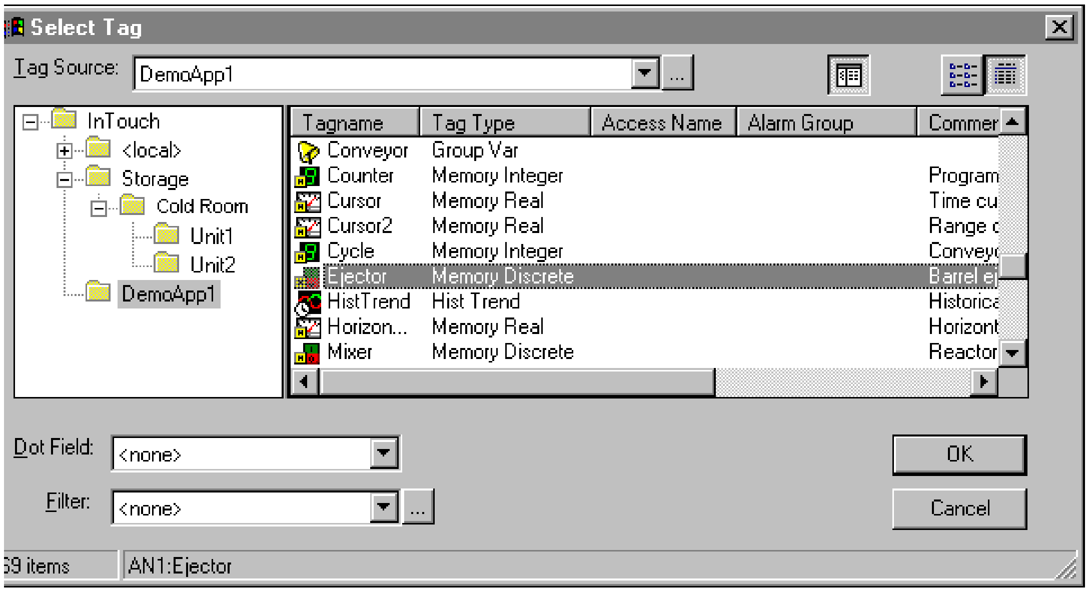
Accessing the Tag Browser

You can quickly view all of the tagnames that are defined in your application when you are creating animation links by accessing the Tag Browser. If you select the tagname that you want assigned to your link from the Tag Browser, it is automatically inserted into the **Tagname** or **Expression** box.

To access the Tag Browser

1. Double-click any blank animation link **Tagname** or **Expression** input box. The Tag Browser appears.

2. Click the tool to show the tree view pane displaying all defined tag sources:



Select Tag

Tag Source: DemoApp1

- InTouch
  - <local>
  - Storage
    - Cold Room
      - Unit1
      - Unit2
  - DemoApp1

Tagname	Tag Type	Access Name	Alarm Group	Commer
Conveyor	Group Var			
Counter	Memory Integer			Program
Cursor	Memory Real			Time cu
Cursor2	Memory Real			Range c
Cycle	Memory Integer			Conveyo
Ejector	Memory Discrete			Barrel e
HistTrend	Hist Trend			Historica
Horizon...	Memory Real			Horizont
Mixer	Memory Discrete			Reactor

Dot Field: <none>

Filter: <none>

OK

Cancel

59 items | AN1:Ejector

Tip If you are not using the tree view mode, click the **Tag Source** arrow and select the name for the tag source that you want to display in the list. The Tag Browser will refresh and the selected tag source's tagnames will be displayed.

3. Select the tagname you want to use for the link then click **OK** or, double-click the tagname to simultaneously select it, close the Tag Browser and insert it into the **Tagname** or **Expression** box.

Tip To use a **.field** with the selected tagname, click the **Dot Field** arrow and select the **.field** that you want to use in the list and then, click **OK**.

To display a tagname's database definition, type the tagname in the **Tagname** or **Expression** box, then double-click it. The **Tagname Dictionary** dialog box appears displaying the tagname's definition.

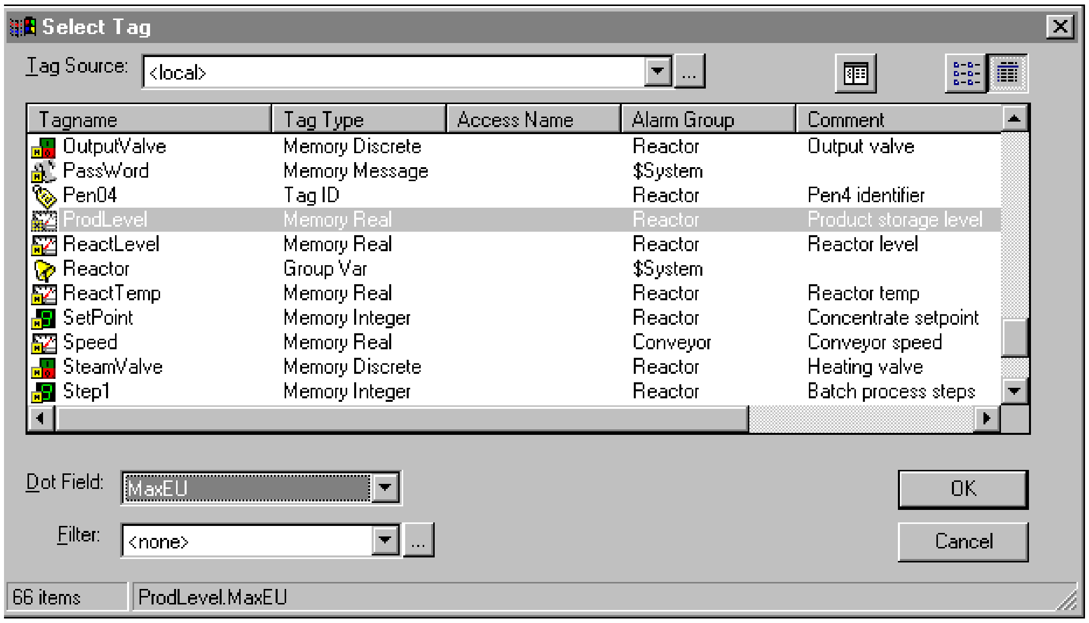
For more information on the Tag Browser, see Chapter 6, "Tagname Dictionary."

Accessing Tagname Dotfields

There are two methods that you can use to access tagname **Dotfields** from an animation link **Tagname** or **Expression** input box. The two methods are described below.

To access tagname Dotfields through the Tag Browser

1. Double-click a blank **Tagname** or **Expression** input box. The Tag Browser appears displaying the tagnames defined for the current tag source:


Select Tag

Tag Source: <local>

Tagname	Tag Type	Access Name	Alarm Group	Comment
OutputValve	Memory Discrete		Reactor	Output valve
PassWord	Memory Message		$System	
Pen04	Tag ID		Reactor	Pen4 identifier
ProdLevel	Memory Real		Reactor	Product storage level
ReactLevel	Memory Real		Reactor	Reactor level
Reactor	Group Var		$System	
ReactTemp	Memory Real		Reactor	Reactor temp
SetPoint	Memory Integer		Reactor	Concentrate setpoint
Speed	Memory Real		Conveyor	Conveyor speed
SteamValve	Memory Discrete		Reactor	Heating valve
Step1	Memory Integer		Reactor	Batch process steps

Dot Field: MaxEU

Filter: <none>

66 items ProdLevel.MaxEU

2. Click the **Dot Field** arrow to open the list of **dotfields** that you can associate with the type of tagname currently selected. By default, **<none>** will initially be displayed for all types of tagnames.

3. Click the **dotfield** in the list that you want to append to the selected tagname.

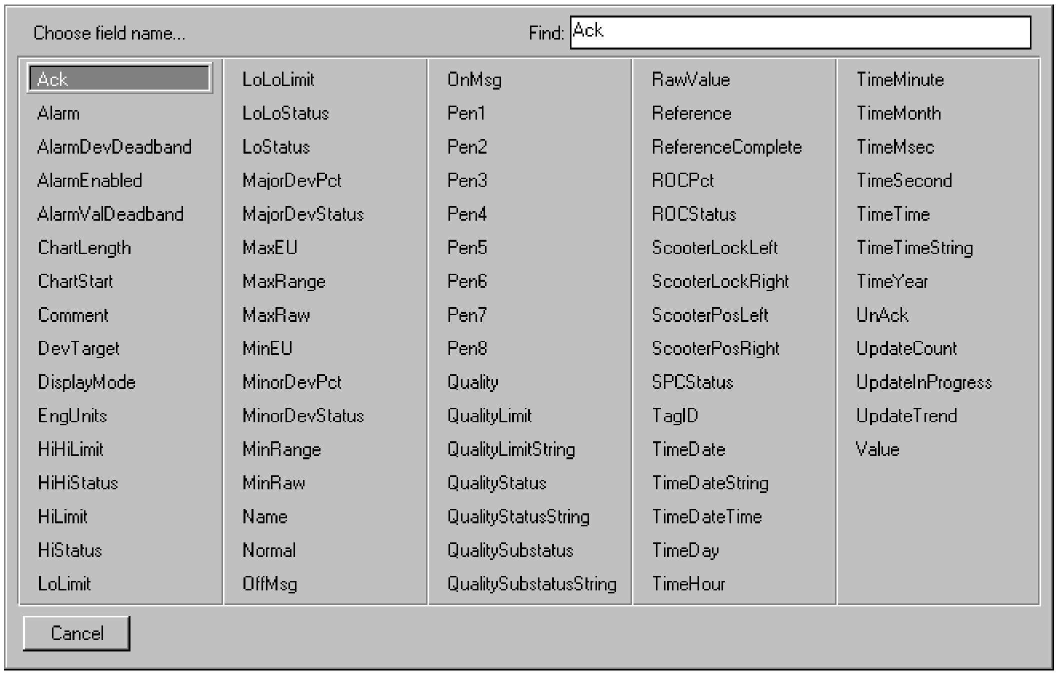
Note Not every tagname type has the same **dotfields**. For example, a **Discrete** type tagname has **.OnMessage**, whereas an analog does not. If you select a **Discrete** type tagname and you assign **.OnMessage** to it and then, you select another **Discrete** type tagname, the displayed **dotfield** list will not change. But, if you select another type of tagname in the control view list, for example an analog, the displayed **dotfield** will revert to **<none>**.

For more information on the Tag Browser, see Chapter 6, "Tagname Dictionary."

For more information on tagname **dotfields**, see your online *InTouch Reference Guide*.

To access the tag .fields through the Choose field name dialog box

1. In any **Tagname** or **Expression** input box, type a tagname plus a period (**tagname.**) and then double-click to the right of it, or type just a period and then, double-click to the right of it. The **Choose field name** dialog box appears displaying all tagname .fields:


Choose field name... Find: Ack

Ack	LoLoLimit	OnMsg	RawValue	TimeMinute
Alarm	LoLoStatus	Pen1	Reference	TimeMonth
AlarmDevDeadband	LoStatus	Pen2	ReferenceComplete	TimeMsec
AlarmEnabled	MajorDevPct	Pen3	ROCPct	TimeSecond
AlarmValDeadband	MajorDevStatus	Pen4	ROCStatus	TimeTime
ChartLength	MaxEU	Pen5	ScooterLockLeft	TimeTimeString
ChartStart	MaxRange	Pen6	ScooterLockRight	TimeYear
Comment	MaxRaw	Pen7	ScooterPosLeft	UnAck
DevTarget	MinEU	Pen8	ScooterPosRight	UpdateCount
DisplayMode	MinorDevPct	Quality	SPCStatus	UpdateInProgress
EngUnits	MinorDevStatus	QualityLimit	TagID	UpdateTrend
HiHiLimit	MinRange	QualityLimitString	TimeDate	Value
HiHiStatus	MinRaw	QualityStatus	TimeDateString	
HiLimit	Name	QualityStatusString	TimeDateTime	
HiStatus	Normal	QualitySubstatus	TimeDay	
LoLimit	OffMsg	QualitySubstatusString	TimeHour	

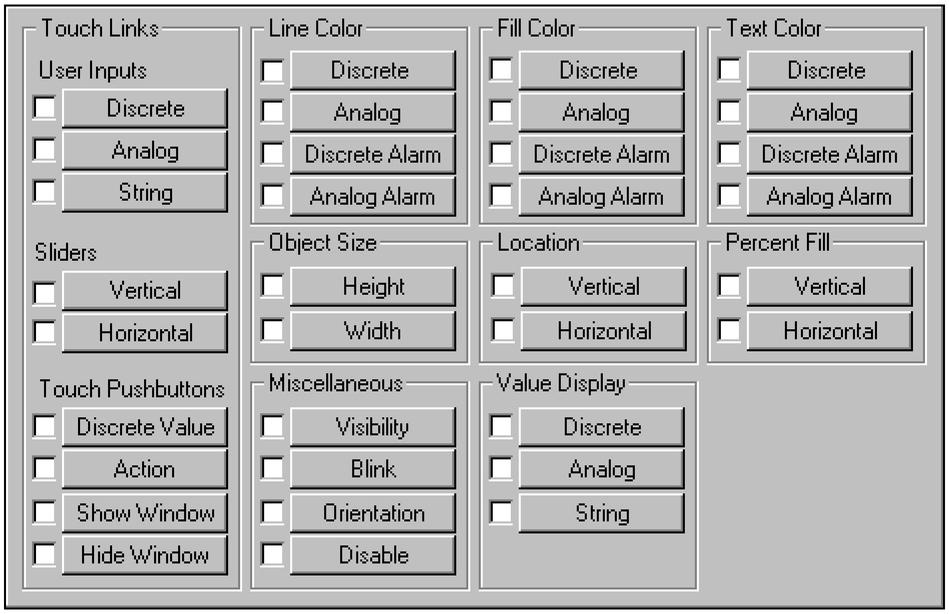
2. Select the **dotfield** that you want to use. The dialog box will close and the selected **dotfield** will automatically be inserted into the **Tagname** or **Expression** input box.

Animating Objects

To animate an object or symbol

1. Create and select the object (line, filled shape, text, button or symbol) that you want to animate.
2. On the **Special** menu, click **Animation Links** or, double-click on the object. The dialog box containing all animation links appears.

Tip You can also right-click the object and then, click **Animation Links**.



3. Click the button for the link that you want to attach to the selected object.

Tip If a link is not applicable for the selected object, its button will not be active.

Tip Clicking the check box only selects the link. Clicking the link name button selects the link and opens its detail definition dialog box. The check box will automatically be selected when you click the link name button and accept the input. However, if you clear a link's check box, the animation link is removed from the selected object.

4. Enter the details for the link definition and then, click **OK**. The **Link Selection** dialog box reappears and if desired, you can create another link for the object.
5. Click **OK** to accept all links for the object and close the dialog box.

Tip When you are creating animation links, the tagname you type in the animation link's tagname or expression box must be defined in the Tagname Dictionary before the link can be assigned to it. If it is not defined, a message box appears asking you if you want to define the tagname now. If you click **Yes**, the Tagname Dictionary appears and you can define the tagname.

Creating Touch Links

You use **Touch Links** on objects or symbols that you want to be "touch-sensitive" in runtime. They allow the operator to input data into the system. For example, the operator may turn a valve on or off, enter a new alarm setpoint, run a complex logic script or log on using text strings, and so on.

Touch links are easily identified in runtime because a "frame" surrounds a touch-sensitive object when you pass the cursor over it or you press the TAB key to move from object to object. If a touch link object or symbol contains text objects that are placed on top of each other, the top text object will be used to display the data value.

The operator activates a touch-sensitive push button by clicking it, touching the object (when using a touch screen), pressing an assigned key equivalent or, pressing the **Enter** key when the object is "framed."

There are nine types of touch links that you can create:

Touch Link	Action
User Inputs	Discrete, Analog, String
Sliders	Vertical, Horizontal
Touch Pushbuttons	Discrete Value, Action, Show Window, Hide Window

Note If the object or symbol used for these links (with the exception of 3-D buttons) contains a text field, all attributes currently set for text, (justification, style, font, and so on) will be applied when the text object is displayed in WindowViewer. When a text field is used for inputting the value, the output value will also be displayed unless the **Input Only** option in the respective tagname's detail dialog box is enabled.

The following sections describe how to create each of the touch links.

Creating User Input Touch Links

You use **User Input Touch Links** to create touch-sensitive objects that allow operator input into the system. For example, push buttons to change discrete states, analog values or security logons.

There are three types of **User Input** touch links:

Link	Description
Discrete	Used to control the value of a discrete tagname. When this link is activated, a dialog box appears prompting the operator to make a selection.
Analog	Used to input the value of an analog (integer or real) tagname. When the link is activated, an input box appears and the value may be entered from the standard keyboard or an optional on-screen keypad.
String	Used to create an object into which a string message may be input.

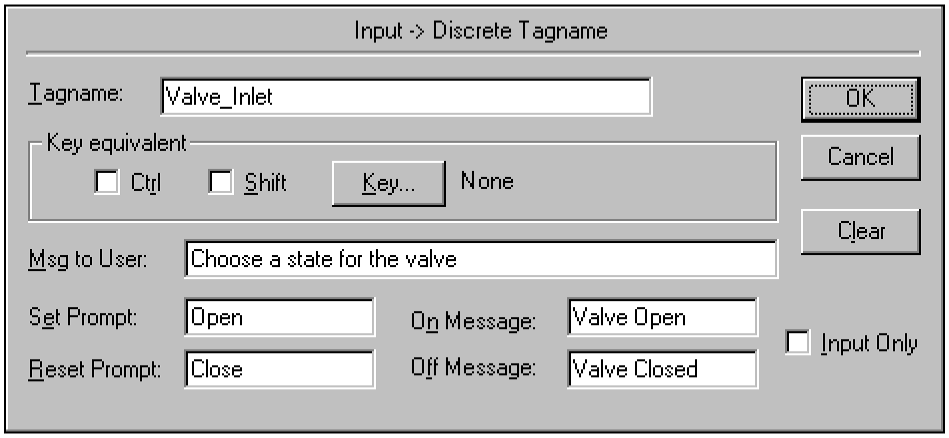
Note In Runtime, when the link is activated, an input box appears for entering the message value or an optional on screen keyboard.

To create a discrete input link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **User Inputs** section, click **Discrete**. The **Input -> Discrete Tagname** dialog box appears.


Input -> Discrete Tagname

Tagname: Valve_Inlet

Key equivalent: [ ] Ctrl [ ] Shift Key... None

Msg to User: Choose a state for the valve

Set Prompt: Open

Reset Prompt: Close

On Message: Valve Open

Off Message: Valve Closed

[ ] Input Only

3. In the **Tagname** box, type a discrete type tagname.

   Tip Right-click the **Tagname** box, to access the commands that you can apply to the selected text.

4. Click **Key** if you want to assign a key equivalent to the link.

   For more information on assigning keys, Assigning Key Equivalents

5. In the **Msg to User** box, type the message that you want to appear in the input dialog box when the input link is activated.
6. In the **Set Prompt** and **Reset Prompt** boxes, type the messages that you want to display on the buttons the operator will click in the input dialog box to turn the discrete value on and off.
7. In the **On Message** and **Off Message** boxes, type the messages that you want to appear in the text field (if any) associated with the object when the object is on or off.
8. Select **Input Only** if you want to prevent the input from being displayed in a text field associated with the object. (This option only applies to an object that has a text field associated with it.)

9. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

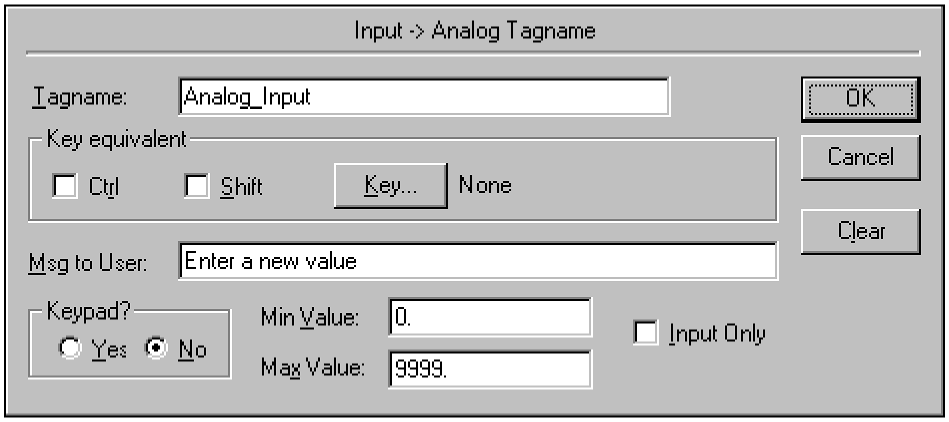
Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

To create an analog input link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **User Inputs** section, click **Analog**. The **Input -> Analog Tagname** dialog box appears.



Tagname: Analog_Input
Key equivalent: Ctrl [ ] Shift [ ] Key... None
Msg to User: Enter a new value
Keypad? Yes [✓ No]
Min Value: 0.
Max Value: 9999.
[ ] Input Only

   Note If a text field is being used for this link, it must be formatted properly in order to display the analog (integer or real) value's output correctly.

   For more information on formatting text fields, see Chapter 2, "Using WindowMaker."

3. In the **Tagname** box, type an analog (integer or real) type tagname.

   Tip Right-click the **Tagname** box, to access the commands that you can apply to the selected text.

4. To assign a key equivalent to the link, click **Key**.

   For more information on assigning keys, see "Assigning Key Equivalents."

5. If you are displaying the optional keypad when this link is activated, in the **Msg to User** box, type the prompt message that you want to appear in keypad.

6. If you want to display an on-screen numeric keypad for inputting the new value of the string, select **Yes**.

7. In the **Min Value** box, type the minimum input value for the tagname.
8. In the **Max Value** box, type the maximum input value for the tagname.
9. Select **Input Only** if you want to prevent the input from being displayed in a text field associated with the object. (This option only applies to an object that has a text field associated with it such as a 3 dimensional button.)
10. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

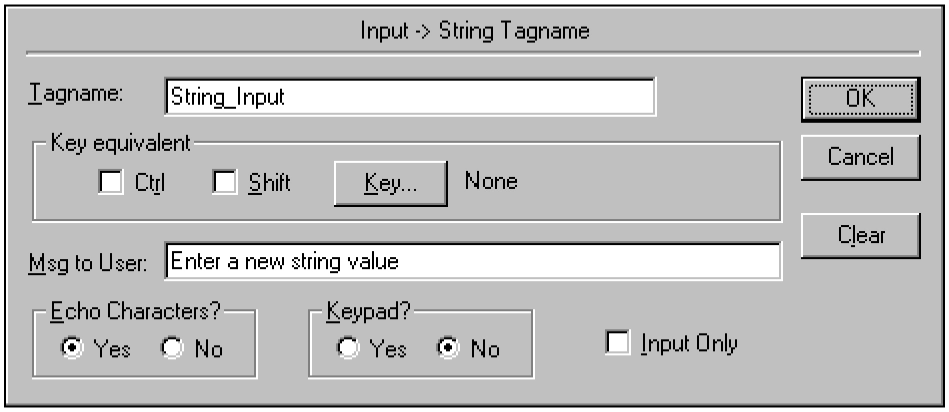
Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

To create a string input link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **User Inputs** section, click **String**. The **Input -> String Tagname** dialog box appears.


Input -> String Tagname

Tagname: String_Input

Key equivalent: Ctrl [ ] Shift [ ] None

Msg to User: Enter a new string value

Echo Characters? [✓ Yes] No

Keypad? Yes [✓ No]

Input Only [ ]

3. In the **Tagname** box, type a message type tagname.

   Tip Right-click the **Tagname** box, to access the commands that you can apply to the selected text.

4. Click **Key** if you want to assign a key equivalent to the link.

   For more information on assigning keys, see "Assigning Key Equivalents."

5. If you are displaying the optional keypad when this link is activated, in the **Msg to User** box, type the prompt message that you want to appear in keyboard.

6. If you want the input string to appear on the screen as it is typed, select **Yes** for the **Echo Characters?** option. If the data is sensitive (for example, a password) and should not be visible on the screen, select **No**.
7. If you want to display an on-screen keyboard for inputting the new value of the string, select **Yes** for the **Keypad?** option.
8. Select **Input Only** if you want to prevent the input from being displayed in a text field associated with the object. (This option only applies to an object that has a text field associated with it such as a 3 dimensional button.)

   Note When WindowViewer is initially started, the string will display the text you typed in the **Initial Value** box when you defined the tagname linked to this object.

9. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

Creating Slider Touch Links

You use **Slider Touch Links** to create objects or symbols that can be moved around the window with the mouse or other pointing devices such as a finger on a touch screen. As the object or symbol is moved, it alters the value of the tagname linked to it. This provides the ability to create devices for setting values in the system.

An object may have a horizontal or a vertical slider touch link, or both. By using both links on a single object, the value of two analog tagnames can be altered simultaneously.

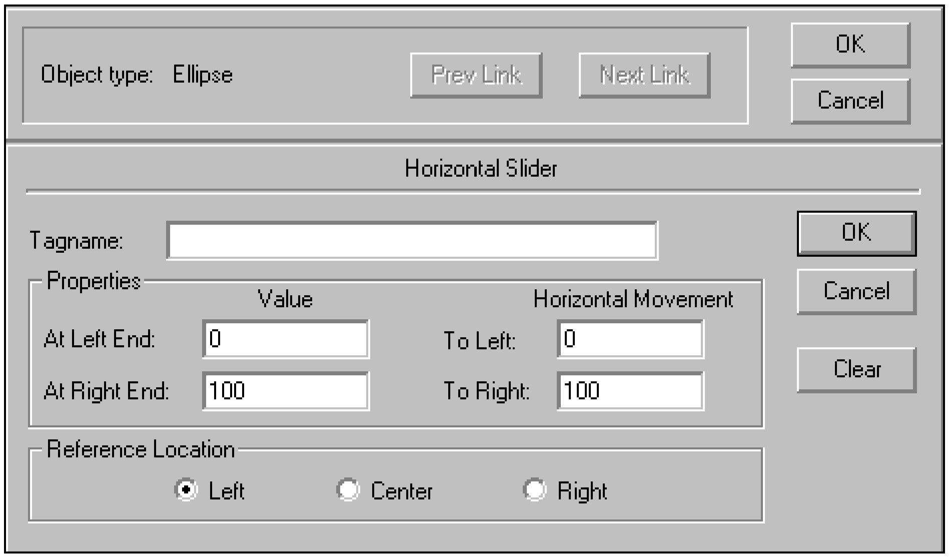
Note The horizontal and vertical slider links are created the same way. This procedure describes the **Horizontal Slider** link.

To create a horizontal (or vertical) slider link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Slider** section, click **Horizontal**. The **Horizontal Slider** dialog box appears.


Object type: Ellipse

Horizontal Slider

Tagname:

Properties

	Value		Horizontal Movement
At Left End:	0	To Left:	0
At Right End:	100	To Right:	100

Reference Location

[✓ Left] Center Right

3. In the Tagname box, type an analog (integer or real) type tagname.

Tip Right-click the **Tagname** box, to access the commands that you can apply to the selected text.

4. In the **At Left End** box, type the value for the tagname when the slider is in its farthest left position.
5. In the **At Right End** box, type the value for the tagname when the slider is in its farthest right position.
6. In the **To Left** box, type the number of pixels the slider can move to the left.

Tip At the far left position, the tagname's value will be equal to the value entered in the **At Left End** field.

7. In the **To Right** box, type the number of pixels the slider can move to the right.

Tip At the far right position, the tagname's value will be equal to the value entered in the **At Right End** field.

8. Select the **Reference Location** on the object that the cursor will lock on for moving the object.
9. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

Creating Touch Pushbuttons Touch Links

You use **Touch Pushbutton Touch Links** to create object links that immediately perform an operation when clicked with the mouse or touched (when touch screen is being used). These operations can be **Discrete Value Changes**, **Action Script** executions, **Show** or **Hide Window** commands. There are four types of **Touch Pushbutton** links:

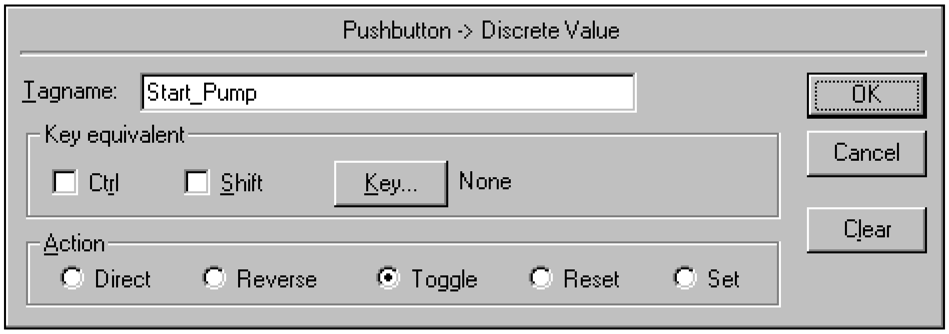
Link	Description
Discrete Value	Used to make any object or symbol into a pushbutton that controls the state of a discrete tagname. Pushbutton actions can be set, reset, toggle, momentary on (direct) and momentary off (reverse) types.
Action	Allows any object, symbol or button to have up to three different action scripts linked to it; On Down, While Down and On Up. Action scripts can be used to set tagnames to specific values, show and/or hide windows, start and control other applications, execute functions, and so on.
Show Window	Used to make an object or symbol into a button that opens one or more windows when it is clicked or touched.
Hide Window	Used to make an object or symbol into a button that closes one or more windows when it is clicked or touched.

To create a discrete value touch pushbutton link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Touch Pushbutton** section, click **Discrete Value**. The **Pushbutton -> Discrete Value** dialog box appears.


Pushbutton -> Discrete Value

Tagname: Start_Pump

Key equivalent: Ctrl [ ] Shift [ ] Key... None

Action: Direct Reverse [✓ Toggle] Reset Set

3. In the **Tagname** box, type a discrete type tagname.

Tip Right-click the **Tagname** box, to access the commands that you can apply to the selected text.

4. Click **Key** if you want to assign a key equivalent to the link.

5. For more information on assigning keys, see "Assigning a Key to an Animation Link."
6. Select the **Action** option that you want to use for the pushbutton as follows:

Action	Description
Direct	Sets the value equal to 1 (True, On, Yes) as long as the pushbutton is pressed and held down. The value automatically resets to 0 (False, Off, No) when the button is released.
Reverse	Sets the value equal to 0 (False, Off, No) when the pushbutton is pressed and held down. The value automatically resets to 1 (True, On, Yes) when the button is released.
Toggle	Reverses the state of the discrete tagname when it is pressed. For example, if the tagname is equal to 1 and the button is pressed, it is reset to 0 and vice-versa.
Reset	Sets the value equal to 0 (False, Off, No) when the pushbutton is pressed.
Set	Sets the value equal to 1 (True, On, Yes) when the pushbutton is pressed.

7. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

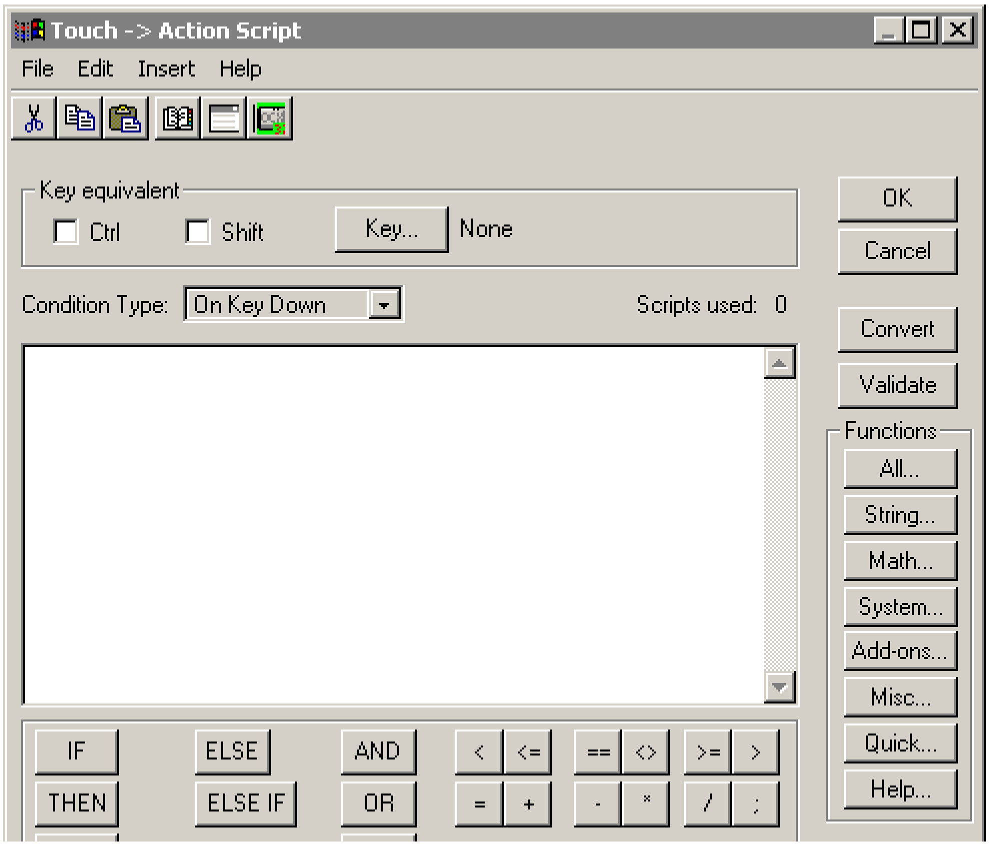
Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

To create an action touch pushbutton link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Touch Pushbutton** section, click **Action**. The **InTouch -> Action Script** editor appears.



For more information on writing QuickScripts, see Chapter 8, "Creating QuickScripts in InTouch."

3. Click the **Condition Type** arrow and select the script type that you want to apply to the object. You can apply all three script types to the same key:

Script	Description
On Key Down	Executes the script one time when the key is initially pressed.
While Down	Executes the script continuously on a time interval as long as the key is held down.
On Key Up	Executes the script one time when the key is released.

Tip A **While Down** script begins executing after the specified number of milliseconds has elapsed. To cause immediate execution, create a duplicate **On Key Down** script.

Note If any object or action pushbutton in the active window is assigned to the same key used for a **Key Script**, the key equivalent link on the key in the active window will take precedence over the execution of the **Key Script**.

For more information on scripts, see Chapter 8, "Creating QuickScripts in InTouch."

4. Click in the script editor's window and type the script that you want executed when the object is activated.
5. Click **OK** to attach the script to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

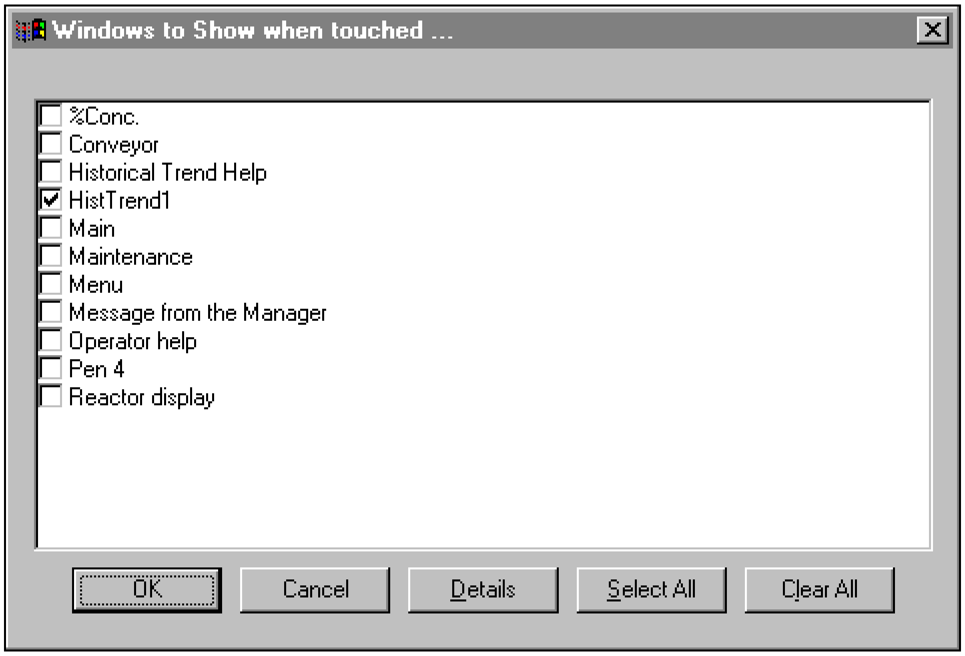
To create a show (or hide) window touch pushbutton link

Note The **Show Window** and **Hide Window** links are created in the same way. This procedure describes the **Show Window** link.

1. Double-click the object or select it, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Touch Pushbutton** section, click **Show Window**. The **Windows to Show when touched** dialog box appears.



- [ ] %Conc.
- [ ] Conveyor
- [ ] Historical Trend Help
- [x] HistTrend1
- [ ] Main
- [ ] Maintenance
- [ ] Menu
- [ ] Message from the Manager
- [ ] Operator help
- [ ] Pen 4
- [ ] Reactor display

3. Select the windows that you want to open when the object is clicked or touched.
4. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Tip When showing more than one window where one of the windows is a **Replace** type, if it intersects any other windows, they will close before they are displayed (giving the illusion that your **Show Window** animation link is not working).

To change a window's type, right-click a blank area of the open window and then, click **Window Properties**. The **Window Properties** dialog box appears and you can change the type. (You cannot change the window's type if WindowViewer is running.)

For more information on window properties, see Chapter 2, "Using WindowMaker."

Creating Display Links

You use the various **Display Links** to provide output to the operator. There are eight types of display links that you can create:

Display Link	Types
Line, Fill & Text Color	Discrete, Analog, Discrete Alarm, Analog Alarm
Object Size	Height, Width
Location	Horizontal, Vertical
Percent Fill	Horizontal, Vertical
Miscellaneous	Visibility, Orientation, Blink, Disable
Value Display	Discrete, Analog, String

The following sections describe how to create each of the display links.

Creating Color Links

You use color links to animate the **Line Color**, **Fill Color**, and **Text Color** attributes of an object.

Note You must use solid colors for line and text color links. If you select a dithered (mixed) color, WindowMaker, by default, will select the closest solid color. To avoid dithered colors, your video card must have at least 2MB and your color depth settings must be higher than 256 colors, such as 32K or 65K (sometimes called "high color".)

For more information on the color palette, see "Applying Color Links."

Each of these color attributes may be made dynamic by defining a color link for the attribute. The color attribute may be linked to the value of a discrete expression, analog expression, discrete alarm status or analog alarm status.

There are four types of line, fill and text color:

Color Link	Description
Discrete	Used to control the fill, line and text colors attributes of an object or symbol that is linked to the value of a discrete expression.
Analog	The line, fill, and text color of an object or symbol can be linked to the value of an analog tagname (integer or real) or an analog expression. Five value ranges are defined by specifying four breakpoints. Five different colors can be selected which will be displayed as the value range changes.
Discrete Alarm	The text, line, and fill color of an object can all be linked to the alarm state of a tagname, Alarm Group, or Group Variable. This color link allows a choice of two colors; one for the normal state and one for the alarm state of the tagname. This link can be used for both analog and discrete tagnames. If it is used with an analog tagname, it responds to any alarm condition of the tagname.
Analog Alarm	The text, line, and fill color of an object can all be linked to the alarm state of an analog tagname, Alarm Group, or Group Variable. Allows a specific color to be set for the normal state as well as a separate color for each alarm condition defined for the tagname. **Note** Objects will not go into an alarm state when using an Analog Alarm animation link while the animation link is using a remote tagname that is accessing tagname information from an unconverted application that was created prior to InTouch Version 7.0.

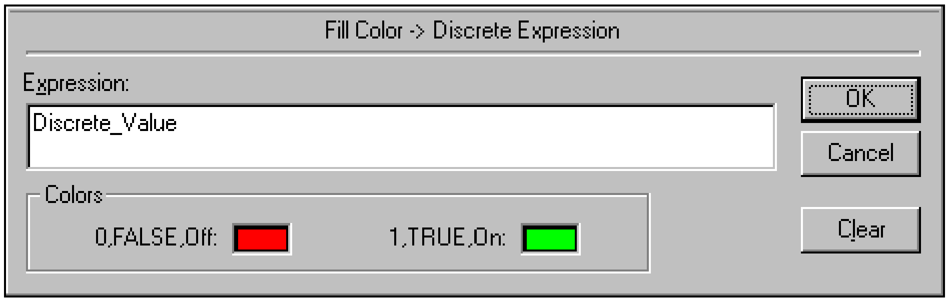
To create a discrete fill color link

Note All of the **Line Color** and **Text Color** links are created in the same way as the **Fill Color** links. The following procedure describes creating a **Fill Color** link.

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Fill Color** (**Line Color** or **Text Color**) section, click **Discrete**. The **Fill Color -> Discrete Expression** dialog box appears.



Expression: Discrete_Value

0,FALSE,Off: 🟥 1,TRUE,On: 🟩

3. In the **Expression** box, type a discrete tagname or an expression that equates to true or false.

Tip Discrete expressions can also 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 fill color of the object will change.

Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. In the **Colors** group, click each color box to open the color palette. Click the color in the color palette that you want to use for each tagname state.

Note You must use solid colors for line and text color links. If you select a dithered (mixed) color, WindowMaker, by default, will select the closest solid color. To avoid dithered colors, your video card must have at least 2MB and your color depth settings must be higher than 256 colors, such as 32K or 65K (sometimes called "high color".)

For more information on the color palette, see "Applying Color Links."

5. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

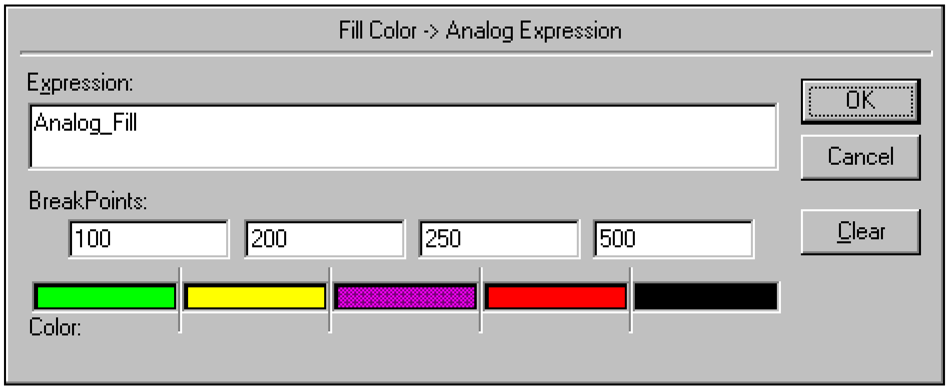
Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

To create an analog expression color link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Fill Color** (**Line Color** or **Text Color**) section, click **Analog**. The **Fill Color -> Analog Expression** dialog box appears.



Analog_Fill

100 200 250 500

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

Tip Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. In each **Break Points** box, you can specify the breakpoint values (decimals are valid for real type tagnames) where the object will change color.

Tip You do not have to use four different values. For example, if you only want the object to change color three times, type three values then use the same color for the third and fourth values.

5. In the **Colors** group, click each color box to open the color palette. Click the color in the color palette that you want to use for each breakpoint.

Note You must use solid colors for line and text color links. If you select a dithered (mixed) color, WindowMaker, by default, will select the closest solid color. To avoid dithered colors, your video card must have at least 2MB and your color depth settings must be higher than 256 colors, such as 32K or 65K (sometimes called "high color".)

.For more information on the color palette, see "Applying Color Links."

6. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

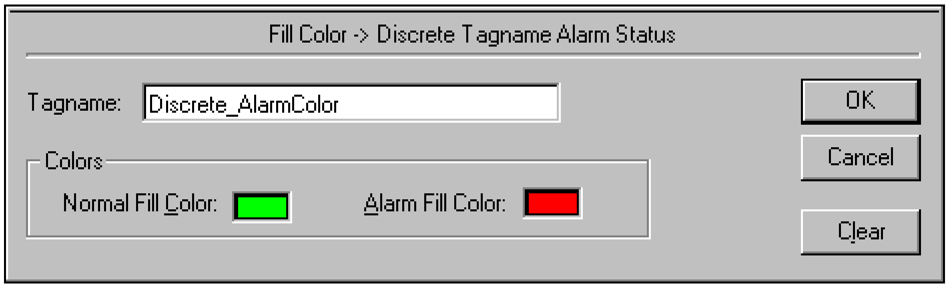
Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

To create a discrete alarm status color link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Fill Color** (**Line Color** or **Text Color**) section, click **Discrete Alarm**. The **Fill Color -> Discrete Tagname Alarm Status** dialog box appears.


Fill Color -> Discrete Tagname Alarm Status

Tagname: Discrete_AlarmColor

Colors

Normal Fill Color: 🟩 Alarm Fill Color: 🟥

3. In the **Tagname** box, type the discrete tagname whose alarm status you want associated with the object.

Tip Right-click the **Tagname** box, to access the commands that you can apply to the selected text.

4. In the **Colors** group, click each color box to open the color palette. Click the color in the color palette that you want to use for each color state.

Note You must use solid colors for line and text color links. If you select a dithered (mixed) color, WindowMaker, by default, will select the closest solid color. To avoid dithered colors, your video card must have at least 2MB and your color depth settings must be higher than 256 colors, such as 32K or 65K (sometimes called "high color".)

For more information on the color palette, see "Applying Color Links."

5. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

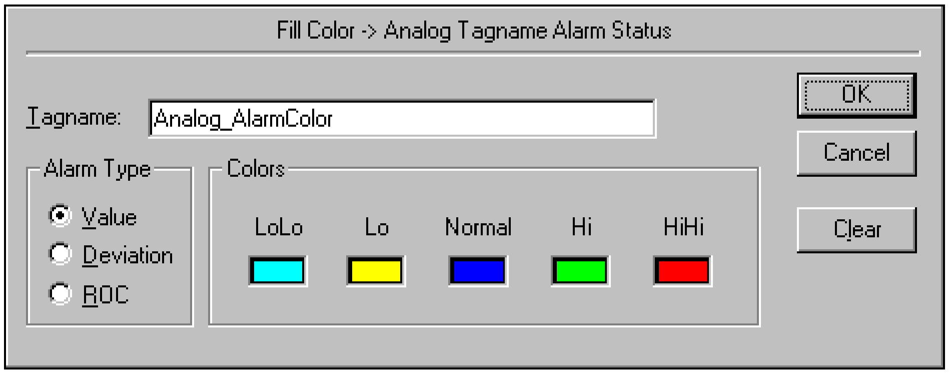
Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

To create an analog alarm status color link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Fill Color** (**Line Color** or **Text Color**) section, click **Analog Alarm**. The **Fill Color -> Analog Tagname Alarm Status** dialog box appears.



Fill Color -> Analog Tagname Alarm Status

Tagname: Analog_AlarmColor

Alarm Type: [✓ Value] Deviation ROC

Colors				
LoLo	Lo	Normal	Hi	HiHi
🟦	🟨	🟦	🟩	🟥

3. In the **Tagname** box, type the analog (integer or real) tagname whose alarm status you want associated with the object.

   Tip Right-click the **Tagname** box, to access the commands that you can apply to the selected text.

4. In the **Alarm Type** group, select type of alarm that you want to associate with the object. There are three mutually exclusive types of analog color links that you can use:

Alarm Type	Description
Value Alarm	You can select up to five different colors depending on the status of the value alarms defined for the tagname (example above).
Deviation	You can select up to three different colors depending on the status of the deviation alarms defined for the tagname (example above).
ROC (Rate of Change)	**ROC** (Rate-of-Change) -You can select two different colors depending on the status of the rate-of- of change alarm defined for the tagname.

5. In the **Colors** group, click each color box to open the color palette. Click the color in the color palette that you want to use for each color state.

   Note You must use solid colors for line and text color links. If you select a dithered (mixed) color, WindowMaker, by default, will select the closest solid color. To avoid dithered colors, your video card must have at least 2MB and your color depth settings must be higher than 256 colors, such as 32K or 65K (sometimes called "high color".)

   For more information on the color palette, see "Applying Color Links."

6. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Creating Object Size Links

You use **Object Size** links to vary the height and/or width of an object according to the value of an analog (integer or real) tagname or analog expression. Size links provide the ability to control the direction in which the object enlarges in height and/or width by setting the "anchor" for the link. Both height and width links can be attached to the same object.

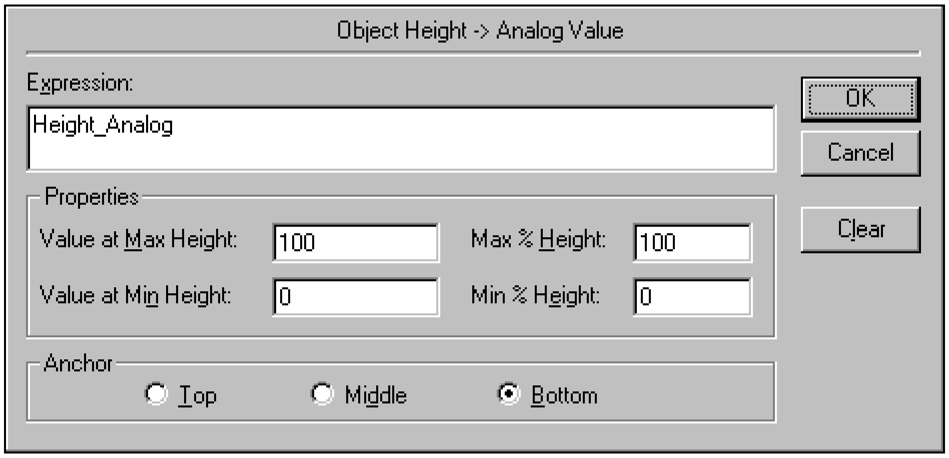
Note The height and width links are created the same way. This procedure describes the **Height** link.

To create a height (or width) link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Object Size** section, click **Height**. The **Object Height -> Analog Value** dialog box appears.



Expression: Height_Analog

Value at Max Height: 100
Max % Height: 100
Value at Min Height: 0
Min % Height: 0

Anchor: Top ( ) Middle ( ) [✓ Bottom]

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

Tip Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. In the **Value at Max Height** box, type the value of the tagname or expression that will result in the object reaching its maximum height.
5. In the **Value at Min Height** box, type the value of the tagname or expression that will result in the object reaching its minimum height.
6. In the **Max % Height** box, type the percentage (0-100) of its height that the object will be when the tagname or expression reaches the value set in the **Value at Max Height** field.
7. In the **Min % Height** box, type the percentage (0-100) of its height that the object will be when the tagname or expression reaches the value set in the **Value at Min Height** field.

Tip The percent height figures are expressed as a percentage of the actual "drawn size" of the object, which is 100%.

8. Select the **Anchor** point from which the object will enlarge in height.

Tip Selecting **Top** will cause the object to be enlarged from its top downward. Selecting **Middle** will cause the object to be enlarged from its centerpoint outwards in both directions. Selecting **Bottom** will cause the object to be enlarged from its bottom upwards.

9. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

Creating Location Links

You use **Location Links** to make an object automatically move horizontally, vertically, or in both directions in response to changes in the value of an analog tagname or expression.

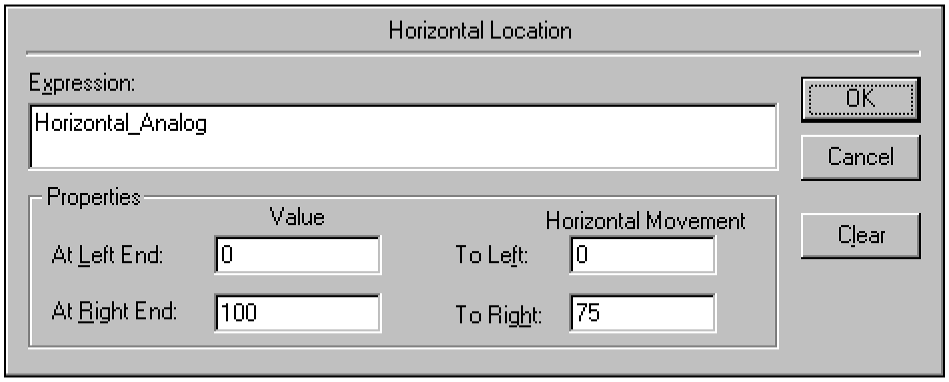
Note The **Horizontal** and **Vertical Location** links are created the same way. This procedure describes the **Horizontal Location** link.

To create a horizontal location link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Location** section, click **Horizontal**. The **Horizontal Location** dialog box appears.



Horizontal Location

Expression: Horizontal_Analog

Properties

	Value		Horizontal Movement
At Left End:	0	To Left:	0
At Right End:	100	To Right:	75

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

   Tip Right-click the **Expression** box, to access the commands that you can apply to the selected text.

   Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

   A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

   For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. In the **At Left End** box, type the value for the tagname when the object is located at its farthest left position.

5. In the **At Right End** box, type the value for the tagname when the object is located at its farthest right position.
6. In the **To Left** box, type the number of pixels the object can move to the left of its drawn position.

   Tip At the far left position, the tagname's value will be equal to the value entered in the **At Left End** field.

7. In the **To Right** box, type the number of pixels the object can move to the right of its drawn position.

   Tip At the far right position, the tagname's value will be equal to the value entered in the **At Right End** field.

8. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

Creating Percent Fill Links

You use **Percent Fill Links** to provide the ability to vary the fill level of a filled shape (or a symbol containing filled shapes) according to the value of an analog tagname or an expression that computes to an analog value. For example, this link may be used to show the level of liquids in a vessel. An object or symbol may have a horizontal fill link, a vertical fill link, or both.

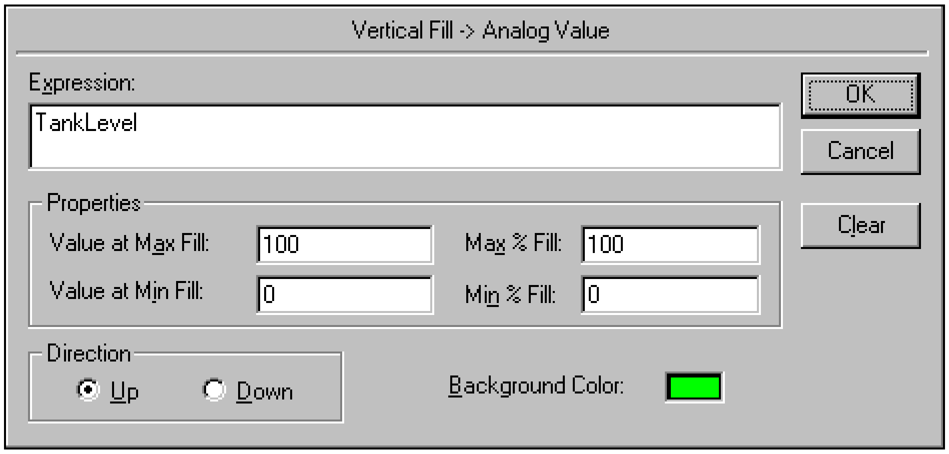
Note The **Horizontal** and **Vertical Percent Fill** links are created the same way. This procedure describes the **Vertical Percent Fill** link.

To create a vertical percent fill link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Percent Fill** section, click **Vertical**. The **Vertical Fill -> Analog Value** dialog box appears.



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

Tip Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. In the **Value at Max Fill** type the value the expression that will result in the object being filled to its maximum level.
5. In the **Value at Min Fill** type the value the expression that will result in the object being filled to its minimum level.
6. 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.

Tip If the value of the expression is greater than this number, it will be ignored.

7. 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.

Tip If the value of the expression is greater than this number, it will be ignored.

8. Select the **Direction** that you want the object to fill from.

Tip If **Up** is selected, it will be filled from the bottom to the top. If **Down** is selected, it will be filled from the top to the bottom.

9. In the **Background Color** box to open the color palette. Click on the desired color. The color palette will be removed from the screen.

Tip This **Background Color** selection is for 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 in either of their link dialog boxes will be used as the background color.

10. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

Creating Miscellaneous Links

There are four type of miscellaneous links.

Misc Link	Description
Visibility	Use to control the visibility of an object based on the value of a discrete tagname or expression.
Blink	Used to make an object blink based on the value of a discrete tagname or expression.
Orientation	Used to make an object rotate based on the value of a tagname or expression.
Disable	Used to disable the touch functionality of objects based on the value of a tagname or expression.

Tip Often used as part of a security strategy.

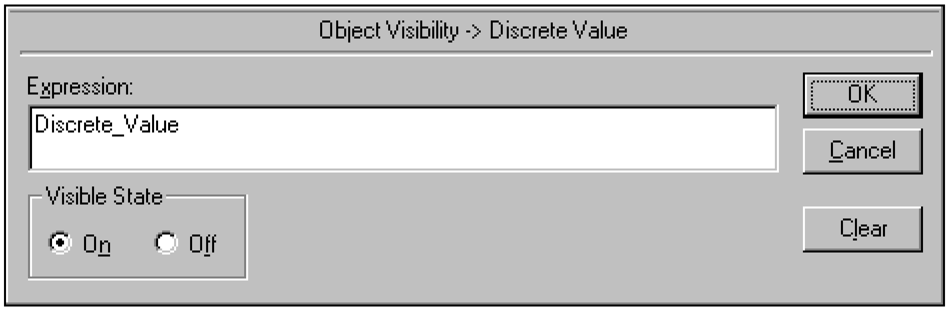
For more information on applying security to your application, see Chapter 5, "Building a Distributed Application."

To create a visibility link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

Tip To quickly access the animation link selection dialog box, right-click the object and then, click Animation Links.

2. In the **Miscellaneous** section, click **Visibility**. The **Object Visibility -> Discrete Value** dialog box appears.



Discrete_Value

Visible State: [✓ On] Off

3. In the **Expression** box, type a discrete tagname or an expression that equates to a discrete value.

Tip Discrete expressions can also contain analog tagnames, for example, TankLevel >= 75. In this example, when the value of the tagname, TankLevel is greater than or equal to 75, the object will become visible in the window.

Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. Select the **Visible State** for the object. If you select **On**, the object will be invisible when the value of the expression is true. If you select **Off**, the object will be visible when the value of the expression is true.
5. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

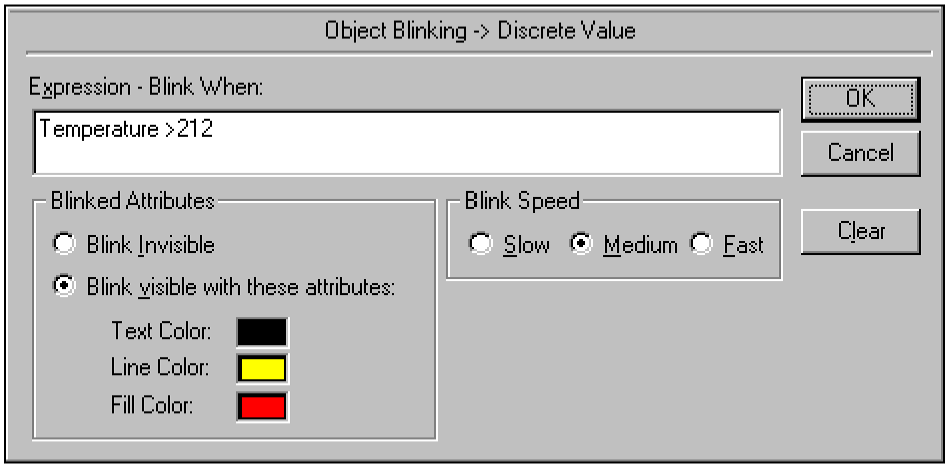
Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

To create a blink link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Miscellaneous** section, click **Blink**. The **Object Blinking -> Discrete Value** dialog box appears.


Object Blinking -> Discrete Value

Expression - Blink When: Temperature >212

Blinked Attributes: Blink Invisible [✓ Blink visible with these attributes:]

Text Color: ⬛
Line Color: 🟨
Fill Color: 🟥

Blink Speed: Slow [✓ Medium] Fast

3. In the **Expression - Blink When** box, type a discrete tagname or an expression that equates to a discrete value.

Tip Discrete expressions can also 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 will blink.

Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. Select the **Blinked Attributes** that you want for the object.

   If you select **Blink Invisible**, the object/symbol blinks by disappearing and reappearing in the window. If you select **Blink visible with these attributes**, the object/symbol remains visible in the window and the changing of the color attributes selected creates the blinking effect.

   Click the **Text Color**, **Line Color** and **Fill Color** boxes to open the color palette. Click on the desired color. The color palette will be removed from the screen.

   Tip Choosing a "fill" blink color that is the same as the object's "fill" color will not allow the object to "blink."

5. Select the **Blink Speed** that you want to use for the blinking speed of the object.

   Tip To configure the blink speed for **Slow**, **Medium**, and **Fast**, on the **Special** menu, point to **Configure** and then click **WindowViewer**. The **WindowViewer General** property sheet appears. In the **Blink Frequency** group, type the number of milliseconds you want to use for the speeds.

   Any changes you make to these settings are global and will effect the blink speeds of all blink links throughout your application.

6. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

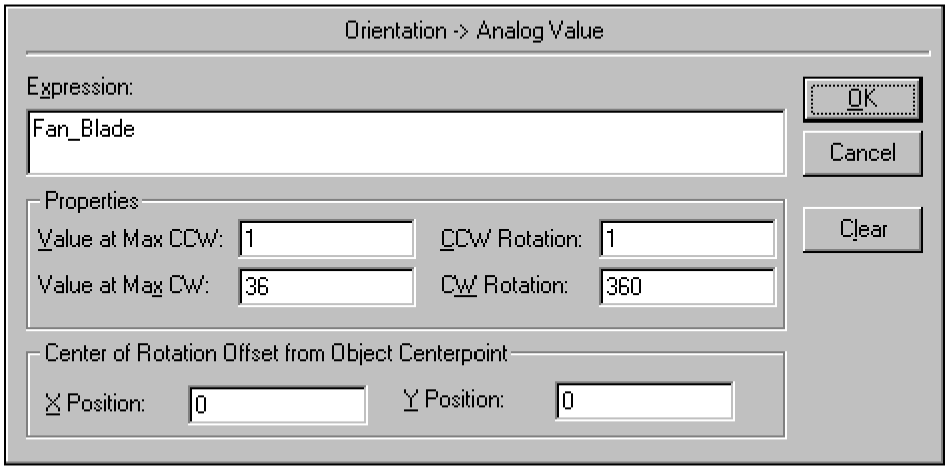
Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

To create an orientation link"

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Miscellaneous** section, click **Orientation**. The **Orientation -> Analog Value** dialog box appears.


Orientation -> Analog Value
OK
Cancel
Clear
Properties
Center of Rotation Offset from Object Centerpoint

Expression: Fan_Blade

Value at Max CCW: 1
CCW Rotation: 1
Value at Max CW: 36
CW Rotation: 360

X Position: 0
Y Position: 0

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

Tip Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. In the **Value at Max CCW** box, type the value the expression must be for the object to be rotated to its maximum counter-clockwise position.

Tip If the value of the expression is greater than this number, it will be ignored.

5. In the **Value at Max CW** box, type the value the expression must be for the object to be rotated to its maximum clockwise position.

Tip If the value of the expression is greater than this number, it will be ignored.

6. In the **CCW Rotation** box, type the degrees the object will rotate counter-clockwise when the **Value at Max CCW** is reached.

7. In the **CW Rotation** box, type the degrees the object will rotate clockwise when the **Value at Max CW** is reached.

Tip The object is rotated clockwise or counter-clockwise based on its original drawn position when drawn in WindowMaker.

To force an object like text to a specific angle, simply set **Value at Max CCW** to 360 and **Value at Max CW** to 0, **CCW Rotation** to 360 and **CW Rotation** to 0 and then, in the **Expression** box, type the angle value such as 90 (for 90 degrees). Remember, without a tagname, this expression will never change and the object will always hold its 90 degree position.

Text can be set in WindowMaker, but not rotated in WindowViewer on a tagname value.

8. In the **X Position** box, type the number of pixels the rotation centerpoint is to be moved horizontally from the centerpoint of the object. (Positive values are to the right of centerpoint.)

   Tip The orientation link uses the center of the object or symbol as the center of rotation.

9. In the **Y Position** box, type the number of pixels the rotation centerpoint is to be moved vertically from the centerpoint of the object. (Positive values are to the left of centerpoint.)
10. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

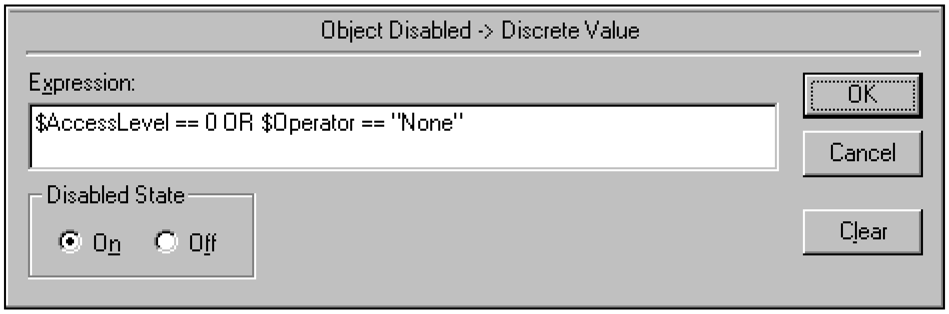
To create a disable link

Tip The disable link is very useful when you are applying security to your application. For example, you can disable objects based upon the logged on operator's access level or name.

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Miscellaneous** section, click **Disable**. The **Object Disabled -> Discrete Value** dialog box appears.



Expression: $AccessLevel == 0 OR $Operator == "None"

Disabled State: [✓ On] Off

3. In the **Expression** box, type a discrete tagname or an expression that equates to a discrete value.

Tip By using the above expression if no one is logged on, the object or button is secured from tampering.

Discrete expressions can also 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 will be disabled.

Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. Select the **Disabled State** that will turn off or on functionality of the object when the discrete tagname or expression is true.

Tip A disabled state of "on" means the touch functionality of the object or button are turned off and cannot be clicked as long as the expression is true.

5. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

Creating Value Display Links

Value Display Links provide the ability to use a text object to display the value of a discrete, analog, or string tagname. There are three types:

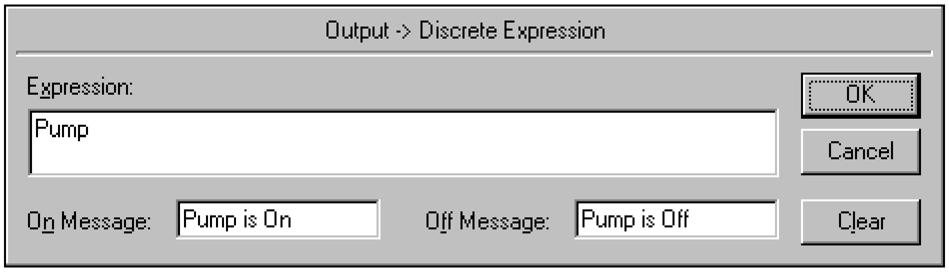
Value Display Type	Description
Discrete	Uses the value of a discrete expression to display an On or Off user defined message in a text object.
Analog	Displays the value of an analog expression in a text object.
String	Displays the value of a string expression in a text object.

To create a discrete value display link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

   Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Value Display** section, click **Discrete**. The **Output -> Discrete Expression** dialog box appears.



Expression: Pump

On Message: Pump is On

Off Message: Pump is Off

3. In the **Expression** box, type a discrete tagname or an expression that equates to a discrete value.

   Tip Discrete expressions can also 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 appropriate message will be displayed.

   Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. In the **On Message** box, type the message that you want displayed when the value of the discrete expression equals 1 (True, On, Yes).
5. In the **Off Message** box, type the message that you want displayed when the value of the discrete expression equals 0 (False, Off, No).

Tip The messages will be displayed 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 displayed message at runtime.

6. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

Tip You can also use a **Value Display Output -> String Expression** link to display the on and off messages for a discrete tagname. For the link, you would type the following expression:

```
DText (Pump, Pump.OnMsg, Pump.OffMsg);
```

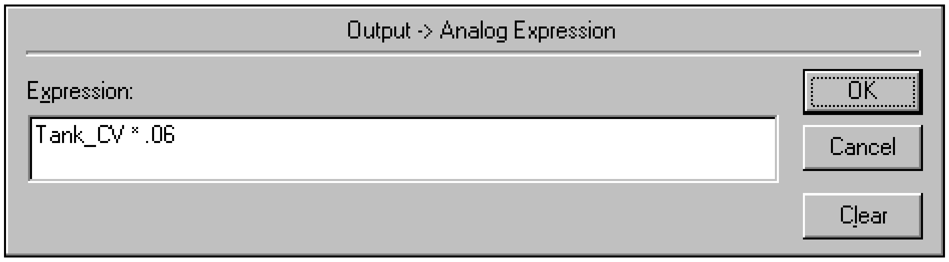
In this expression, the **.OnMsg** and **.OffMsg** strings will be extracted from the InTouch Tagname Dictionary definition for the discrete tagname, Pump.

To create an analog value display link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

Tip To quickly access the animatio n link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Value Display** section, click **Analog**. The **Output -> Analog Expression** dialog box appears.



Expression:
Tank_CV * .06

3. In the **Expression** box, type an analog (integer or real) tagname or an expression that equates to an analog value. (You can also use a discrete type tagname in this expression. It will simply display a 1 or 0.)

Tip Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

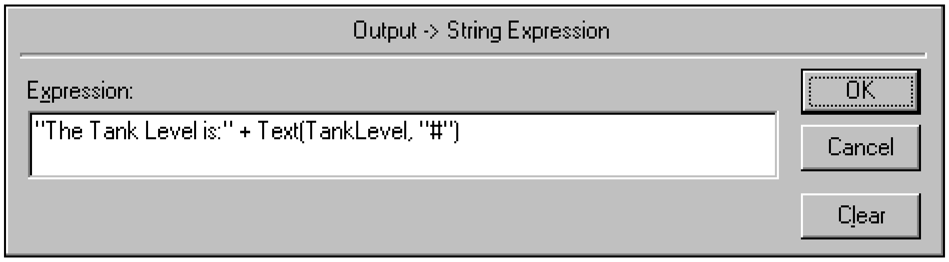
For more information on formatting analog display objects, see Chapter 2, "Using WindowMaker."

To create a string value display link

1. Double-click the object or select it and then, on the **Special** menu, click **Animation Links**. The link selection dialog box appears.

Tip To quickly access the animation link selection dialog box, right-click the object and then, click **Animation Links**.

2. In the **Value Display** section, click **String**. The **Output -> String Expression** dialog box appears.



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

3. In the **Expression** box, type a message tagname or an expression that equates to a message tagname.

Tip In the above expression, the function **Text()** is used to convert the value of the integer tagname, TankLevel, to a string.

Right-click the **Expression** box, to access the commands that you can apply to the selected text.

Note Up to 256 characters may be typed for your expression. If you need to use a larger expression, create a QuickFunction then call it in your expression.

A "trigger" tagname must be used as a parameter for the QuickFunction to force the Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

For more information on QuickFunctions, see Chapter 8, "Creating QuickScripts in InTouch."

4. Click **OK** to attach the link to the object and return to the animation links dialog box. You can now attach another link to the object if desired.

Note If the tagname you entered is not defined in your Tagname Dictionary (except for remote tagnames) you will be prompted to define it now.

C H A P T E R 8

Creating QuickScripts in InTouch

InTouch scripting is one of the most powerful features of an InTouch application. The InTouch QuickScript capabilities allow you to execute commands and logical operations based on specified criteria being met. For example, a key being pressed, a window being opened, a value changing, and so on.

QuickFunctions are scripts that you create that can be called from other scripts and animation link expressions. The reused code is stored in one script and in one location, thereby supporting update of all script instances with one edit session.

By using scripts, a wide variety of customized and automated system functions can be created.

Contents

- Script Editor Error Messages

InTouch QuickScripts

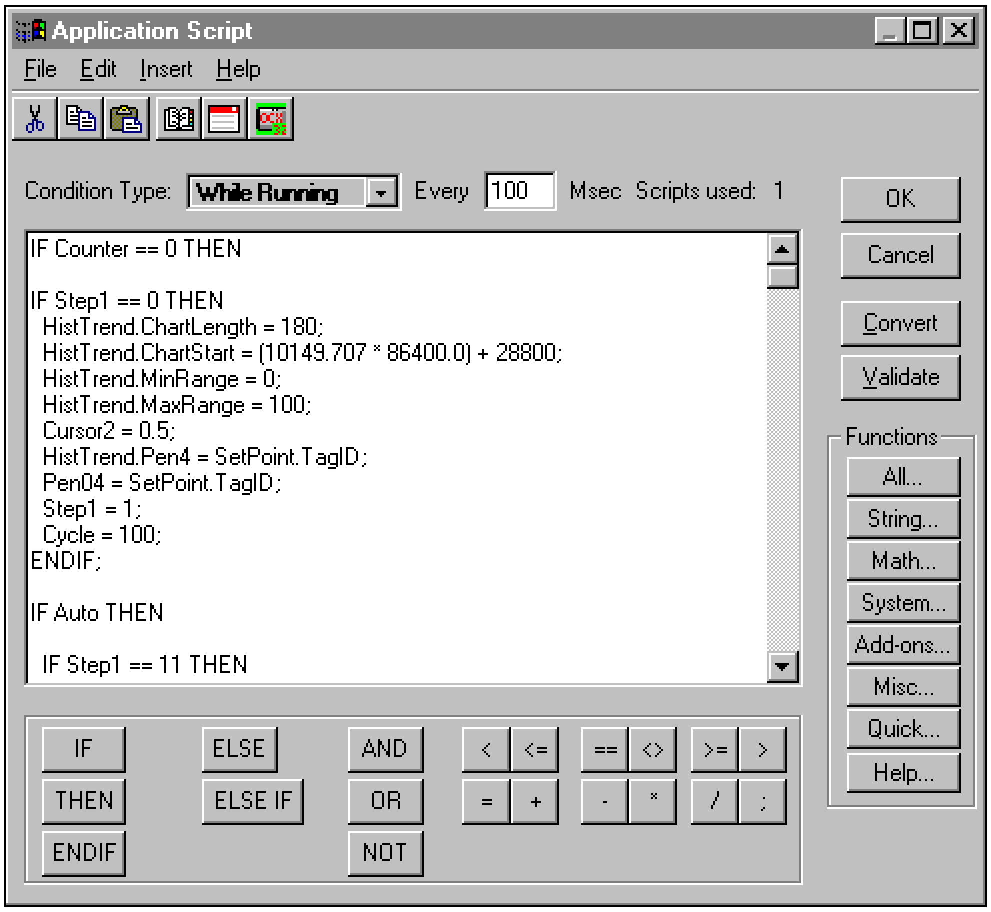
All InTouch QuickScripts are event driven. The event may be a data change, condition, mouse click, timer, and so on. The order of processing is application specific. While it may appear that there is some inherent order in the way multiple scripts initiated by the same event are scheduled, there is no guarantee of any specific order. Therefore, you should not build any dependency on the order of processing.

The following briefly describes the types of scripts that you can create:

Script Type	Description
Application	Linked to the entire application.
Window	Linked to a specific window.
Key	Linked to a specific key or key combination on the keyboard.
Condition	Linked to a discrete tagname or expression.
Data Change	Linked to a tagname and/or **tagname dotfield** only.
QuickFunctions	Scripts you create that can be called from other InTouch QuickScripts or animation link expressions. QuickFunctions can be both synchronous and asynchronous while all other script types are synchronous only.
Action Pushbutton	Associated with an object that you link to an **Touch Link - Action Pushbutton**.
ActiveX Event	Execute ActiveX control events in runtime.
Wizard	Access wizard properties for enhanced functionality in runtime.

Using the InTouch QuickScript Editor

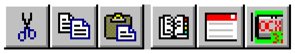
The InTouch QuickScript editor is basically the same for all QuickScript types. Therefore, to avoid redundancy, its common functions and features are described in this section. The items that are unique to a QuickScript type are described in that QuickScript type's respective section later in this chapter.



Condition Type: While Running Every 100 Msec Scripts used: 1

```
IF Counter == 0 THEN

IF Step1 == 0 THEN
  HistTrend.ChartLength = 180;
  HistTrend.ChartStart = (10149.707 * 86400.0) + 28800;
  HistTrend.MinRange = 0;
  HistTrend.MaxRange = 100;
  Cursor2 = 0.5;
  HistTrend.Pen4 = SetPoint.TagID;
  Pen04 = SetPoint.TagID;
  Step1 = 1;
  Cycle = 100;
ENDIF;

IF Auto THEN

 IF Step1 == 11 THEN
```

The QuickScript Editor Toolbar



The QuickScript Editor toolbar gives you quick access to editor functions:

Tool	Description
	Click to cut selected text from the QuickScript.
	Click to copy selected text from the QuickScript.
	Click to paste text into the QuickScript from the clipboard.

Tool	Description
📖	Click to open the Tagname Dictionary.
🗔	Click to insert a window name from the current application.
▦	Click to open the ActiveX Control Browser.

QuickScript Editor Common Procedures

This section describes the generic procedures that you will use when writing scripts in the various InTouch QuickScript editor dialog boxes. The procedures that are unique to a script type are described in that script type's respective section later in this chapter.

There are text, equivalency and mathematical operator buttons at the bottom of the QuickScript editor that you can click to quickly insert the displayed keyword, function or symbol into your script at the cursor location.

To indent/unindent text in a script

Position the cursor at the beginning of the line that you want to indent, and then press the TAB key. To remove the indent, press hold down the SHIFT key while you press the TAB key.

To create a new script

On the **Script** menu, click **New**.

Note The **Script** menu does not exist for **Application, Window Scripts** or **Touch Pushbutton Action** scripts.

Additionally, if you open a script for editing by double-clicking it in the Application Explorer, new script functionality is not allowed. To create a new script, close the currently open script then, on the **Special** menu, point to **Scripts** and then, select the type of script you want to create.

Quick Functions, Condition scripts and Data Change scripts are created with default names. Therefore, you must save (or cancel with <Esc>) the currently displayed script, before you can load another script into the editor.

When creating a QuickScript, it is not possible to declare a local variable with the same name as a tagname that was defined earlier.

To delete a script from your application

Select the text you want to delete, and then on the **Script** menu, click **Erase**. The script is deleted from your application entirely.

Note Deleted text is not written to the Windows Clipboard.

To undo your last action

On the **Edit** menu, click **Undo**. Your last editing operation, a paste, for example, is reversed.

Tip To quickly execute this command, right-click the script window, and then click **Undo**. The **Undo** command will not be active unless you have performed an action that can be reversed.

To select the entire script

On the **Edit** menu, click **Select All**. The entire script is selected.

Tip To quickly execute the command, right-click the script window, and then click **Select All**. You can now copy, cut or delete the entire script.

To cut selected text from a script

Select the text you want to remove, and then on the **Edit** menu, click **Cut**. The cut text is deleted from the script and copied to the Windows Clipboard. You can now paste the cut text at another location in this script or you can paste it in another script.

Tip To quickly perform this command, right-click the script window, and then click **Cut**. The **Cut** command will not be active unless you have selected text to cut.

To copy selected text from a script

Select the text to be removed, and then on the **Edit** menu, click **Copy**. The copied text is written to the Windows Clipboard. You can now paste the copied text at another location in this script, or you can paste it in another script.

Tip To quickly perform this command, right-click the script window, and then click **Copy**. The **Copy** command will not be active unless you have selected text to copy.

Note When you cut or copy text, it is automatically written to the Windows Clipboard. This information remains on the Clipboard until you perform a subsequent cut or copy command.

To paste text into a script

On the **Edit** menu, click **Paste**. The contents of the Windows Clipboard is pasted into your script at the cursor location.

Tip To quickly execute the **Paste** command, right-click the script window, and then click **Paste**. (The **Paste** command will not be active if there is nothing in the Windows Clipboard to paste.)

To clear the text in a script

On the **Edit** menu, click **Clear**. All the text in the script is erased. However, the script is not deleted from your application. If you select this command then close the script editor and reopen it, the script reappears. If you want to delete only part of the script, you can highlight the text to be deleted, right-click a blank area of the script window and click **Delete**.

Tip To completely delete the script, you must use the **Erase** command on the **Script** menu or select the entire script, then right-click a blank area of the script window, and then click **Delete**.

To insert a function into a script

1. On the **Insert** menu, point to **Functions**, and then click the name of the function category. The respective **Choose function** dialog box appears.
2. Click the function that you want to use. The dialog box will close and the function will automatically be inserted into your script at the cursor location.

   The types of functions available are:

Function	Description
All	The Choose function dialog box appears displaying all available functions including the functions for each installed add-on program (Recipe Manager, SPC Pro and SQL Access Manager or the functions installed from the KBCD). **Tip** You can also click the All button in the Functions group to access these functions.
String	The **Choose function** dialog box appears displaying all available string functions. **Tip** You can also click the **String** button in the **Functions** group to access these functions.
Math	The **Choose function** dialog box appears displaying all available mathematical functions. **Tip** You can also click the **Math** button in the **Functions** group to access these functions.
System	The **Choose function** dialog box appears displaying all available system functions. For example, the functions to start and/or activate another application, read and/or write file and disk information, and so on. **Tip** You can also click the **System** button in the **Functions** group to access these functions.

Function	Description
Add-ons	The **Choose function** dialog box appears displaying all available functions for each installed add-on program (Recipe Manager, SPC Pro and SQL Access Manager). **Tip** You can also click the **Add-ons** button in the **Functions** group to access these functions.
Misc	The **Choose function** dialog box appears displaying all available miscellaneous functions. For example, the functions for alarms, historical trending, windows controls, ActiveX controls, and so on. **Tip** You can also click the **Misc** button in the **Functions** group to access these functions.
Help	The **Choose function to Obtain Help for** dialog box appears listing all available functions. Click a function to open its respective Help topic. **Tip** You can also click the **Help** button in the **Functions** group to access these functions.
Quick Functions	The **Choose function** dialog box appears listing the names of all the QuickFunctions available for calling from the current script. **Tip** You can also click the **Quick** button in the **Functions** group to access all QuickFunctions.

For more information on the individual script functions, see "Script Functions."

To insert a tagname into a script

1. On the **Insert** menu, click **Tagname**. The Tag Browser appears in the unlimited selection mode.

   Tip The tagnames defined in the last tag source accessed through the Tag Browser will be displayed. To change the tag source, click the **Tag Source** arrow and select a different tag source in the list.

   Click **Define Tag Sources** to add or remove a tag source from the **Tag Source** list.

2. Double-click the tagname you want to use or select it, and then click **OK**. The Tag Browser will close and the tagname will automatically be inserted into your Quick Script at the cursor location.

Tip To quickly access the Tag Browser, double-click a blank area in the QuickScript window.

To access a specific tagname's definition in the Tagname Dictionary, type the tagname in the QuickScript window, and then double-click it.

For more information on the Tag Browser, see Chapter 6, "Tagname Dictionary."

To insert a tagname .field into a script

1. On the **Insert** menu, click **Tagname**. The Tag Browser appears in the unlimited selection mode.

Tip The tagnames defined in the last tag source accessed through the Tag Browser will be displayed. To change the tag source, click the **Tag Source** arrow and select a different tag source in the list.

Click **Define Tag Sources** to add or remove a tag source from the **Tag Source** list.

2. Select the tagname that you want to use, and then click the **Dot Field** arrow. Select the **.field** that you want to use with the tagname in the list.
3. Click **OK**. The selected tagname**.field** will be inserted into your QuickScript at the cursor location.

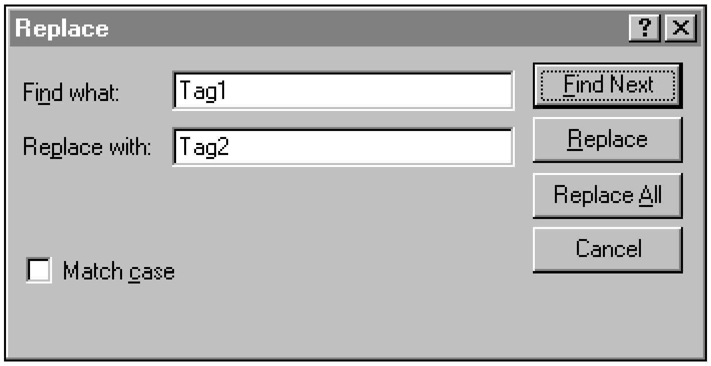
Tip To quickly insert a tagname **.field**, type the tagname followed by a period (**.**), and then double-click to the right of the period. The **Choose field name** dialog box appears. Click the **.field** that you want to use. The dialog box will close and the selected **.field** will automatically be inserted into your QuickScript at the cursor location.

For more information on the Tag Browser, see Chapter 6, "Tagname Dictionary."

For more information on the Tagname **.fields**, see your online *InTouch Reference Guide*.

To find or replace a tagname in a script

1. On the **Edit** menu, click **Find**. The **Replace** dialog box appears.


Find what: Tag1

Replace with: Tag2

2. In the **Find what** box, type the tagname that you want to find (or replace), and then click **Find Next**.
3. In the **Replace with** box, type the new tagname that you want to use to replace the old tagname then click **Replace** or **Replace All**.

Tip If you only want to replace certain instances of an old tagname, click **Find Next**. **InTouch** will begin searching your script for the old tagname. When the old tagname is found, it will be highlighted. Click **Replace** to replace it with the new tagname or click **Find Next** to skip it and continue searching. If you want to replace all occurrences of a specific tagname, click **Replace All** at any time during the search.

4. Select the **Match case** option if you find specific upper or lowercase instances of the tagname.
5. Click **Cancel** to close the dialog box.

To insert a window name into a script

1. On the **Insert** menu, click **Window**. The **Window Name to Insert** dialog box appears displaying the names of all the windows in your application.
2. Click the window name that you want to use. The dialog box will close and the window name will automatically be inserted into your script at the cursor location.

To validate a script

Click **Validate** to verify that your script syntax is accurate at any time while you are writing the script.

Tip Validation is automatically performed when you click **OK** or **Save**. If the system encounters errors when validating your script, a corresponding error message box appears.

For more information on script errors, see "Script Editor Error Messages."

To save a script

If you are writing multiple scripts, after you have finished writing one, you can click **Save** to save it, and then on the **Script** menu, click **New** to write another new script.

Note **Application** and **Window** scripts don't support this function. Otherwise, the save function is automatically performed when you click the **OK** button.

Tip If the system encounters errors when saving your script, a corresponding error message box appears.

To restore a script

If you change a script and decide that you want to clear your changes and restore the original script, click **Restore**.

Note You cannot restore a script once you have saved it. **Application** and **Window** scripts don't support this function.

To exit the script editor

On the **Script** menu, click **Exit**. The script editor will close and the script will be saved unless an error is encountered. In the **Application Script Editor**, point to **File**, and click **Exit**.

Tip You can also close the script editor by clicking **OK** once you have completed writing your script.

Note When you select **Exit**, **OK** or you click the **X** button in the upper right hand corner of the dialog box, the system automatically verifies your script for accuracy.

For more information on script errors, see "Script Editor Error Messages."

To specify a script's execution frequency

In the **While Running/Showing/Down Every 0 Milliseconds** boxes, type the number of milliseconds that you want to elapse before the script executes.

Tip When you create an Application **While Running** script, Window **While Showing** scripts, Condition **While On True/On False** scripts or Key and Touch Pushbutton Action **While Down** scripts you must specify the frequency (in milliseconds) that they will be executed.

Note WindowViewer will make every attempt possible to run these types of scripts as fast as the time you specify. However, the performance cannot be guaranteed. Also scripts can never run any faster than the **Tick Interval** setting that you specify when you configure WindowViewer's properties.

Scripts cannot execute faster than every 10 milliseconds on the Windows NT operating system or every 50 milliseconds on Windows 2000 (or later).

For more information on the Tick Interval setting, see Chapter 2, "Using WindowMaker."

Application Scripts

The Application Scripts are linked to the entire application. You can use application scripts to start other applications, create process simulations, calculate variables, and so on. There are three types of Application Scripts that you can apply to an application:

Application Script	Description
On Startup	Executes one time when the application is initially started up.

While Running	Executes continuously at the specified frequency while the application is running.
On Shutdown	Executes one time when the application is exited.

Note The Application **On Startup** QuickScript executes before any window opens or any runtime initialization occurs. Therefore, you cannot refer to ActiveX methods, properties or events in an Application **On Startup** QuickScript.

Similarly, I/O communications are initialized after the Application **On Startup** QuickScript executes. Therefore, you cannot refer to I/O type tagnames or remote tagname references in an Application **On Startup** QuickScript. Additionally, I/O type tagnames and remote tagname references will not update in an Application **On Shutdown** QuickScript.

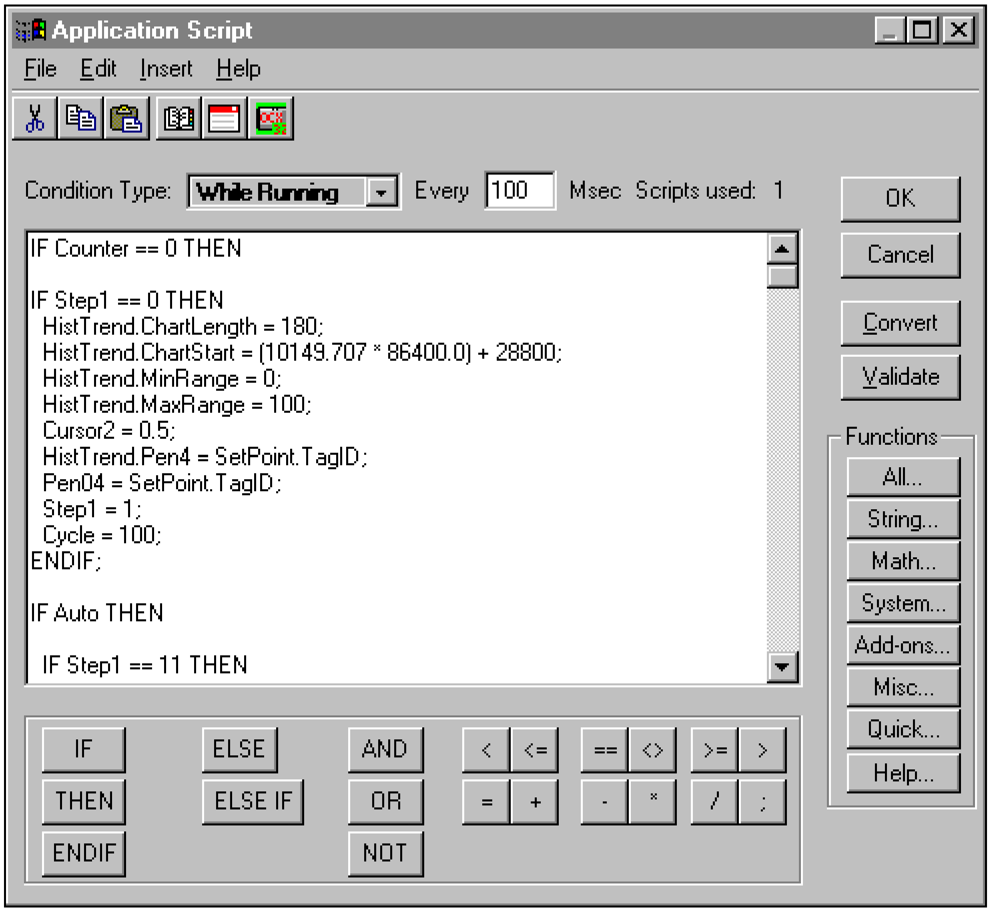
Data Change or Condition QuickScripts may not execute from an Application **On Startup** QuickScript.

Also, you cannot use an Application **On Shutdown** QuickScript to startup other applications.

To access the Application Script editor

On the **Special** menu, point to **Scripts**, and then click **Application Scripts,** or in the Application Explorer under **Scripts**, double-click **Application**. The **Application Script** editor appears.

In the Application Explorer under **Scripts**, you can also right-click **Application**, and then click **Open**.



Condition Type: While Running Every 100 Msec Scripts used: 1

```
IF Counter == 0 THEN

IF Step1 == 0 THEN
  HistTrend.ChartLength = 180;
  HistTrend.ChartStart = (10149.707 * 86400.0) + 28800;
  HistTrend.MinRange = 0;
  HistTrend.MaxRange = 100;
  Cursor2 = 0.5;
  HistTrend.Pen4 = SetPoint.TagID;
  Pen04 = SetPoint.TagID;
  Step1 = 1;
  Cycle = 100;
ENDIF;

IF Auto THEN

  IF Step1 == 11 THEN
```

When you select a **While Running** script, the **Every 0 Milliseconds** box becomes active. In the box, type the number of milliseconds that you want to elapse before the script executes. If you want the script to execute immediately, create an identical **On Startup** script. However, as long as the condition or event for the **While Running** script is met, the script will repeatedly execute at the specified frequency.

Window Scripts

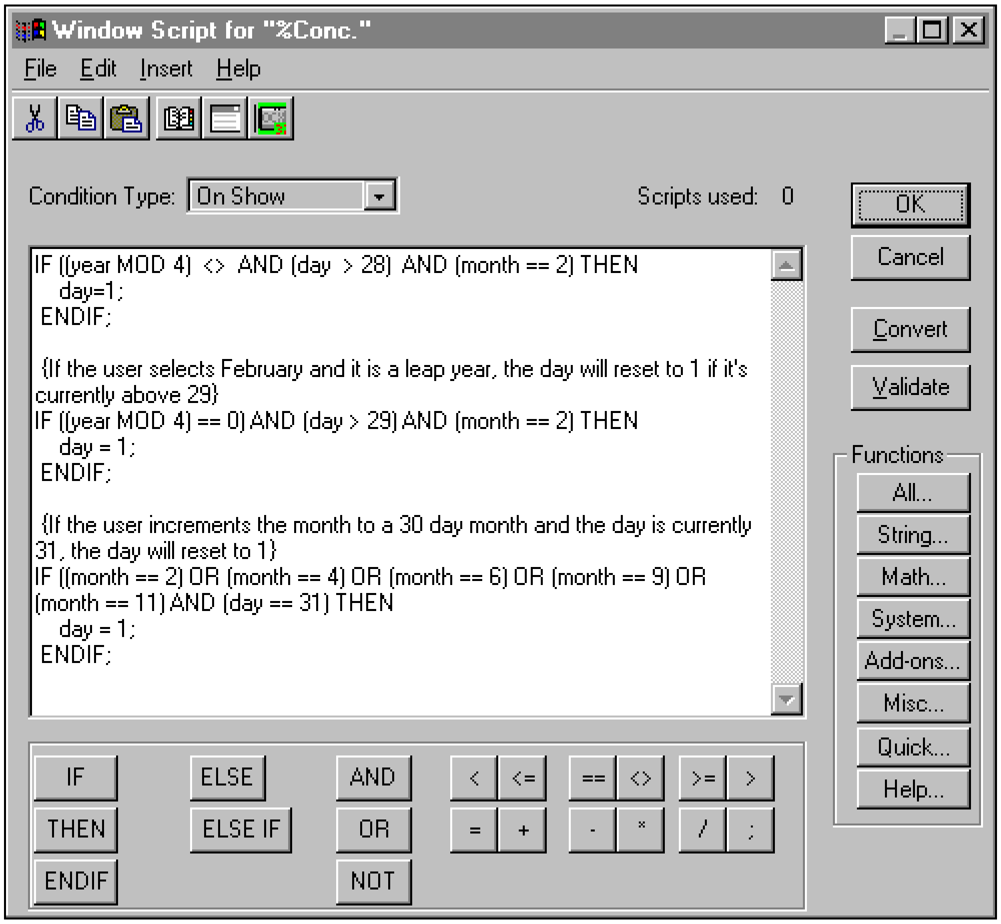
Window Scripts are linked to a specific window. There are three types of scripts that you can apply to a window:

Window Script	Description
On Show	Executes one time when the window is initially shown.
While Showing	Executes continuously at the specified frequency while the window is showing.
On Hide	Executes one time when the window is hidden.

To create Window Scripts

On the **Special** menu, point to **Scripts**, and then click **Window Scripts**. The **Window Script** editor appears.

Tip To quickly access the Window Script editor for a specific window, in the Application Explorer, under **Windows**, right-click the window name and then, click **Window Scripts**. You can also right-click a blank area of an open window, and then click **Window Scripts**. If a script exists for the selected window, it will be displayed.



Condition Type: On Show

Scripts used: 0

```
IF ((year MOD 4) <> AND (day > 28) AND (month == 2) THEN
   day=1;
 ENDIF;

 {If the user selects February and it is a leap year, the day will reset to 1 if it's
currently above 29}
IF ((year MOD 4) == 0) AND (day > 29) AND (month == 2) THEN
   day = 1;
 ENDIF;

 {If the user increments the month to a 30 day month and the day is currently
31, the day will reset to 1}
IF ((month == 2) OR (month == 4) OR (month == 6) OR (month == 9) OR
(month == 11) AND (day == 31) THEN
   day = 1;
 ENDIF;
```

When you select **While Showing**, the **Every 0 Milliseconds** box becomes active. In the box, type the number of milliseconds that you want to elapse before the script executes. If you want the script to execute immediately, create an identical **On Show** script. However, as long as the condition or event for the **While Showing** script is met, the script will repeatedly execute at the specified frequency.

Note If you attach a Window Script to the active window and then you create a new window, the scripts from the active window can be copied to the new window. A message dialog box appears asking if you want to copy the script(s).

Key Scripts

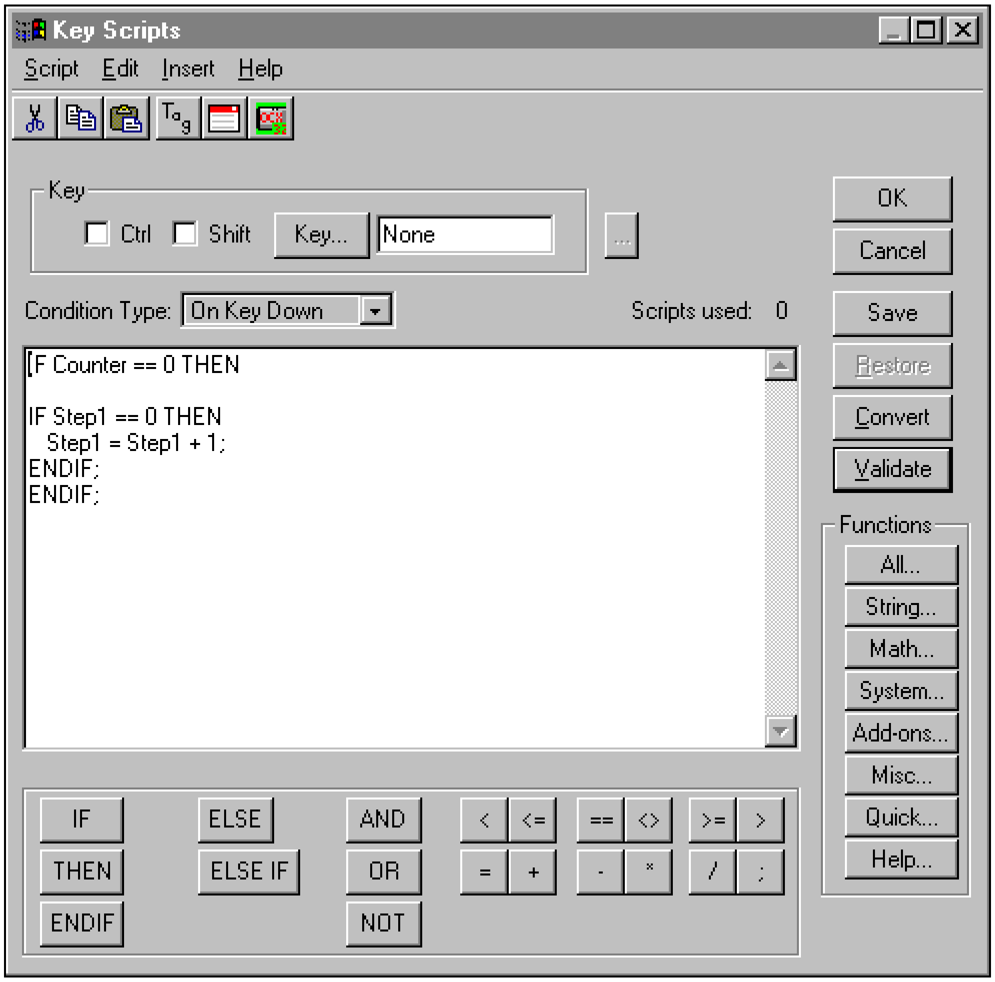
Key Scripts are linked to a specific key or key combination on the keyboard. You can use them to create global keys for the application. For example, returning to a main menu window, logging off the operator, and so on. There are three types of Key Scripts that you can apply to a key:

Key Script	Description
On Key Down	Executes one time when the key is initially pushed down.
While Down	Executes continuously at the specified frequency while the key is held down.
On Key Up	Executes one time when the key is released.

To access the Key Script editor

On the **Special** menu, point to **Scripts**, and then click **Key Scripts**, or in the Application Explorer under **Scripts**, double-click **Key**. The **Key Script** editor appears.

Tip In the Application Explorer, under **Scripts**, you can also right-click **Key**, and then click **Open**.



Key: Ctrl ☐ Shift ☐ Key... None

Condition Type: On Key Down   Scripts used: 0

```
IF Counter == 0 THEN

IF Step1 == 0 THEN
  Step1 = Step1 + 1;
ENDIF;
ENDIF;
```

When you select **While Down**, the **Every 0 Milliseconds** box becomes active. In the box, type the number of milliseconds that you want to elapse before the script executes. If you want the script to execute immediately, create an identical **On Key Down** script. However, as long as the condition or event for the **While Down** script is met, the script will repeatedly execute at the specified frequency.

For more information on assigning a key to the script, see "Assigning a Key Equivalent to a Script."

Note The key equivalents used in the local active window's for Touch Pushbutton Action scripts will override any global Key Scripts with the same key equivalents.

In Windows 2000, all SendKey QuickFunctions that include the ALT key must go into an On Key Up script. The ALT key is not processed in the On Key Down script.

Touch Pushbutton Action Scripts

Touch Pushbutton Action Scripts are similar to Key Scripts, except they are associated with an object that you link to a **Touch Link- Action Pushbutton**. (The script editor is accessed through the animation link selection dialog box.) They are executed when the operator clicks or presses the object or button assigned to the link. There are three types of Touch Action Scripts that you can apply to an object:

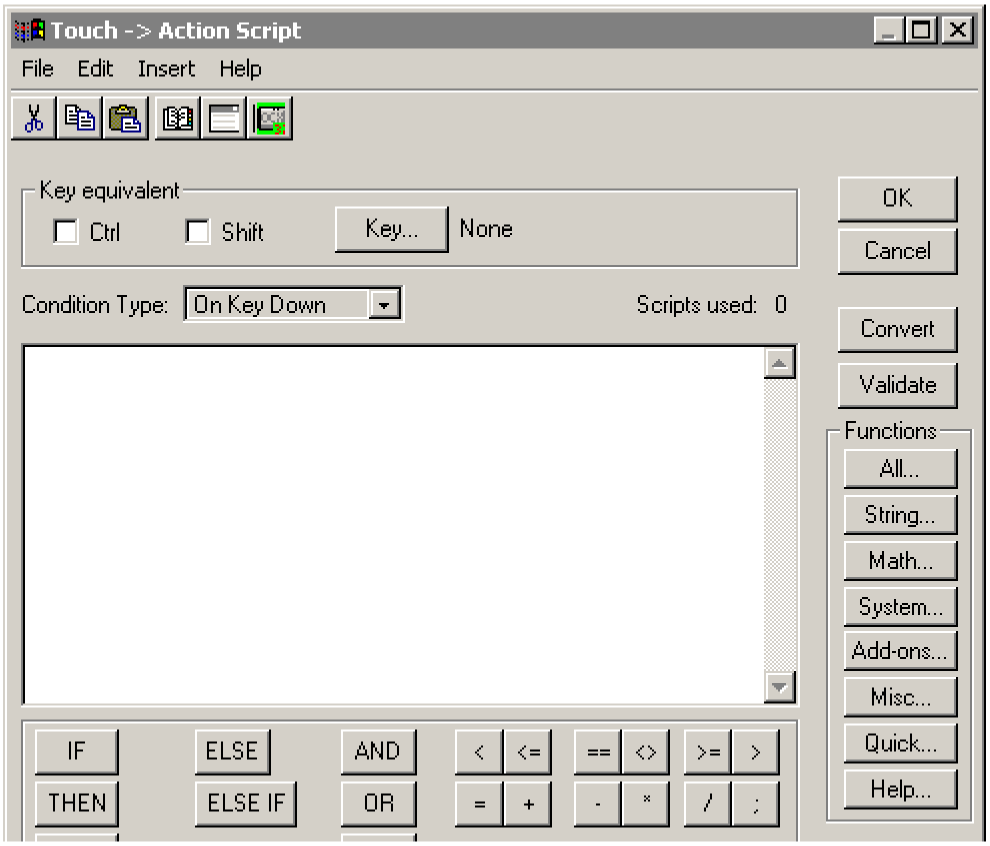
Touch Pushbutton Action Script	Description
On Key Down	Executes one time when the key is initially pushed down.
While Down	Executes continuously at the specified frequency while the key is held down.
On Key Up	Executes one time when the key is released.

To create an action pushbutton script

1. Draw the object or button that you want to be linked to the script.
2. Double-click the object or select it, and then on the **Special** menu, click **Animation Links**. The animation link selection dialog box appears.

Tip To quickly access the dialog box, right-click the object, and then click **Animation Links**.

3. In the **Touch Pushbutton** section, click **Action**. The **InTouch -> Action Script** editor appears.



When you select **While Down**, the **Every 0 Milliseconds** box becomes active. In the box, type the number of milliseconds that you want to elapse before the script executes. If you want the script to execute immediately, create an identical **On Key Down** script. However, as long as the condition or event for the **While Down** script is met, the script will repeatedly execute at the specified frequency.

For more information on assigning a key to the script, see "Assigning a Key Equivalent to a Script."

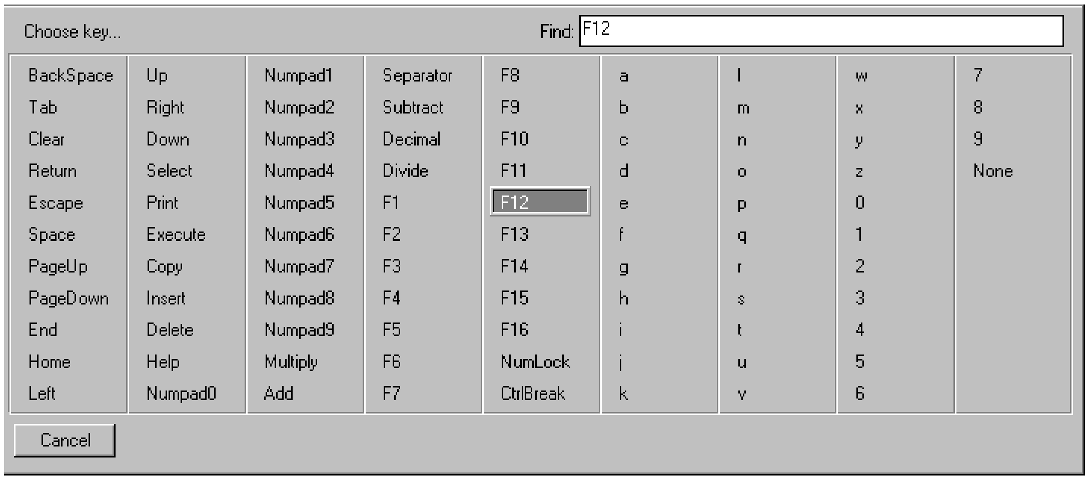
Note The key equivalents used in the local active window's for Touch Pushbutton Action scripts will override any global Key Scripts with the same key equivalents. Also, key equivalents are only active when the window with the object is active.

Assigning a Key Equivalent to a Script

The Key Script and the Touch Action Script editors are a little different from the other QuickScript editors. Since you are creating scripts that apply to a key, you must specify the key(s) that you want the operator to press to execute the script.

To assign a key to a Key Script

1. Select **Ctrl** and/or **Shift**, if you want the operator to have to hold down the **Ctrl** and/or **Shift** keys while pressing the key to execute the script.
2. Click **Key** to select the key you want assigned to the script. The **Choose key** dialog box appears.



Choose key... Find: F12

BackSpace	Up	Numpad1	Separator	F8	a	l	w	7
Tab	Right	Numpad2	Subtract	F9	b	m	x	8
Clear	Down	Numpad3	Decimal	F10	c	n	y	9
Return	Select	Numpad4	Divide	F11	d	o	z	None
Escape	Print	Numpad5	F1	F12	e	p	0	
Space	Execute	Numpad6	F2	F13	f	q	1	
PageUp	Copy	Numpad7	F3	F14	g	r	2	
PageDown	Insert	Numpad8	F4	F15	h	s	3	
End	Delete	Numpad9	F5	F16	i	t	4	
Home	Help	Multiply	F6	NumLock	j	u	5	
Left	Numpad0	Add	F7	CtrlBreak	k	v	6	

3. Click the desired key. The dialog box automatically closes and your selection is automatically entered in the **Key** box.

Condition Scripts

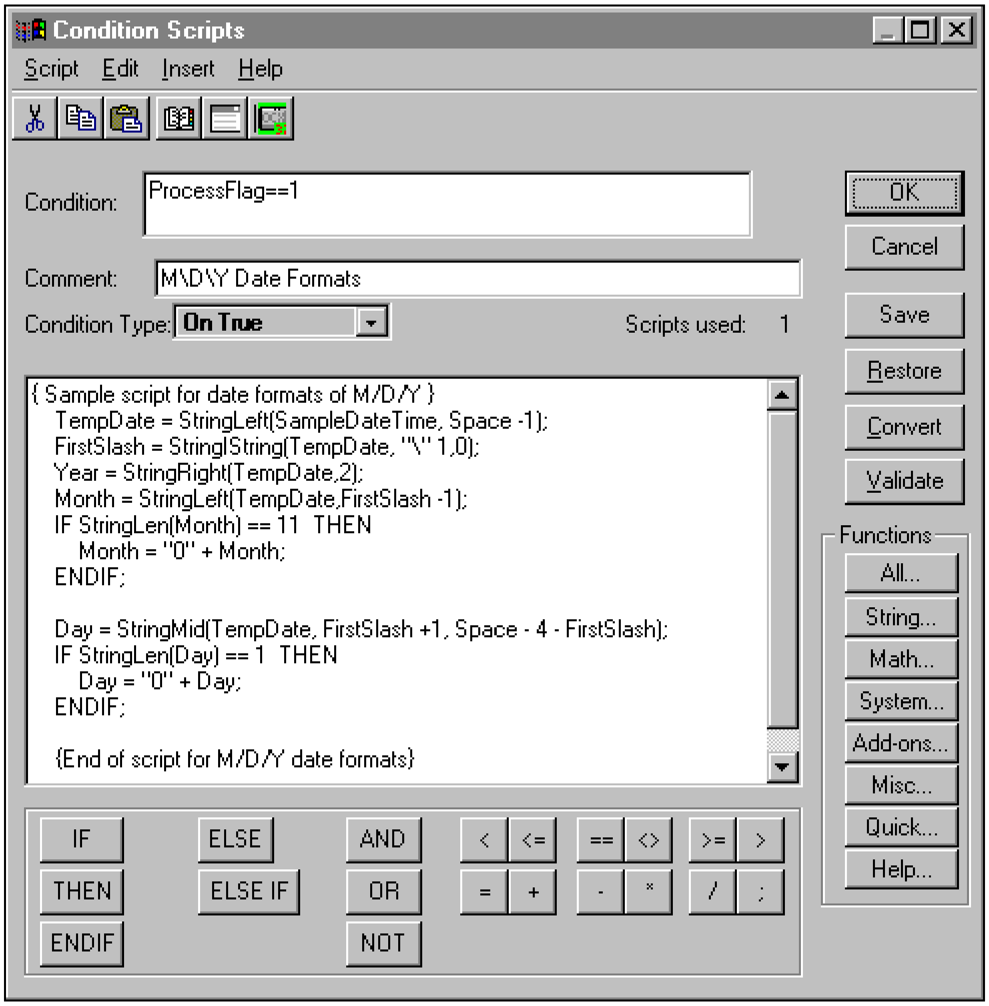
Condition Scripts are linked to a discrete tagname or expression that equates to true or false. You can also use discrete expressions that contain analog tagnames (see example below). There are four types of scripts that you can apply to a condition:

Condition Script	Description
On True	Executes one time when the condition transitions to true.
On False	Executes one time when the condition transitions to false.
While True	Executes continuously while the condition is true.
While False	Executes continuously while the condition is false.

To access the Condition Script editor

1. On the **Special** menu, point to **Scripts**, and then click **Condition Scripts**, or in the Application Explorer under **Scripts**, double-click **Condition**. The **Condition Script** editor appears.

Tip In the Application Explorer under **Scripts**, you can also right-click **Condition**, and then click **Open**.



Condition: ProcessFlag==1

Comment: M\D\Y Date Formats

Condition Type: On True

Scripts used: 1

```
{ Sample script for date formats of M/D/Y }
   TempDate = StringLeft(SampleDateTime, Space -1);
   FirstSlash = StringIString(TempDate, "\" 1,0);
   Year = StringRight(TempDate,2);
   Month = StringLeft(TempDate,FirstSlash -1);
   IF StringLen(Month) == 11  THEN
      Month = "0" + Month;
   ENDIF;

   Day = StringMid(TempDate, FirstSlash +1, Space - 4 - FirstSlash);
   IF StringLen(Day) == 1  THEN
      Day = "0" + Day;
   ENDIF;

   {End of script for M/D/Y date formats}
```

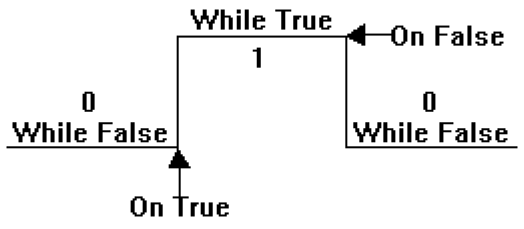
2. Since Condition Scripts are executed based upon a condition being met, you must specify the condition (a discrete tagname or expression) in the **Condition** box.

Note You can only apply one script per condition.

Tip You can also use a discrete expression that equates an analog tagname to true or false. For example, TankLevel >= 75. In this example, when the value of the tagname TankLevel is greater than or equal to 75, the script will execute.

The value for the condition **must** **transition** to become true or false before the script will execute. For example, if the initial value when WindowViewer starts is true, the value must become false and then true again for an **On True** script to execute.

You can apply all four script types to the same condition. Both **While True** and **While False** scripts will begin executing after the specified number of milliseconds have elapsed. To cause immediate execution, create duplicate **On True** and/or **On False** scripts. For example:



3. In the **Comment** box type any miscellaneous comments that you want on file regarding the script.

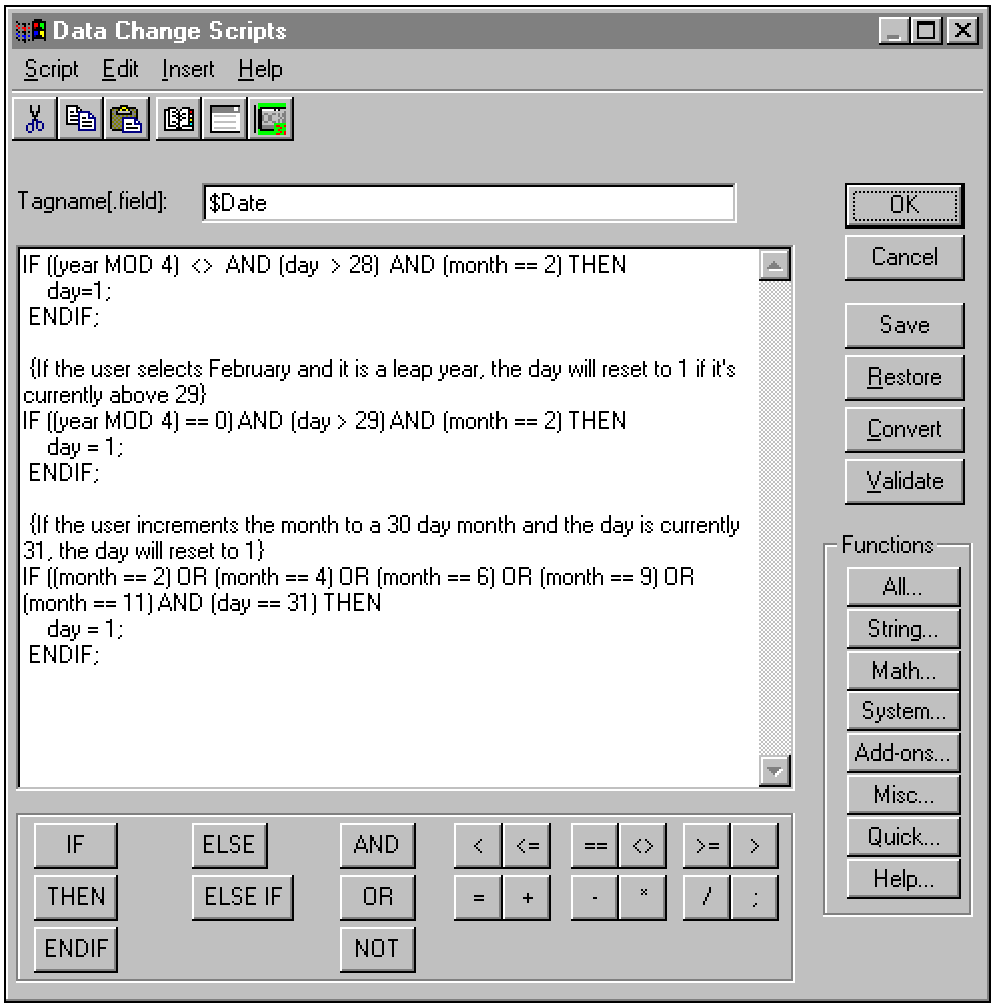
Data Change Scripts

Data Change Scripts are linked to a tagname and/or **tagname.field** only. They are executed one time when the value of the tagname or **tagname.field** changes by a value greater than the deadband that you defined for the tagname in the Tagname Dictionary.

To access the Data Change Script editor

1. On the **Special** menu, point to **Scripts**, and then click **Data Change Scripts**, or in the Application Explorer under **Scripts**, double-click **Data Change**. The **Data Change Script** editor appears.

Tip In the Application Explorer under **Scripts**, you can also right-click **Data Change**, and then click **Open**.



Tagname[.field]: $Date

```
IF ((year MOD 4)  <>  AND (day  > 28)  AND (month == 2) THEN
    day=1;
 ENDIF;

 {If the user selects February and it is a leap year, the day will reset to 1 if it's
currently above 29}
IF ((year MOD 4) == 0) AND (day > 29) AND (month == 2) THEN
    day = 1;
 ENDIF;

 {If the user increments the month to a 30 day month and the day is currently
31, the day will reset to 1}
IF ((month == 2) OR (month == 4) OR (month == 6) OR (month == 9) OR
(month == 11) AND (day == 31) THEN
    day = 1;
 ENDIF;
```

2. Since Data Change Scripts are executed based upon a change in a data value, you must specify a tagname or **tagname.field** in the **Tagname[.field]** box.

3. On the **Insert** menu, click **Tagname** or double-click the script window. The **Select Tag** dialog box appears.

   To select a tagname without a **.field**, double-click the tagname or select it, and then click **OK**. The selected tagname will be inserted into your script at the cursor location.

Tip To quickly access the **Select Tag** dialog box, double-click a blank area in the script editing window. To access a specific tagname's definition in the Tagname Dictionary, type the tagname, and then double-click it.

To select a **.field**, first select the tagname that you want to use, then click the **Dot Field** arrow and select the **.field** that you want to associate with the selected tagname. Click **OK**. The selected tagname**.field** will be inserted into your script at the cursor location.

Tip To quickly insert a tagname **.field**, type the tagname followed by a period (.), and then double-click to the right of the period. The **Choose field name** dialog box appears. Click the **.field** that you want to use. The dialog box will close and the selected **.field** will automatically be inserted into your script at the cursor location.

For more information on tagname**.fields**, see your online *InTouch Reference Guide*.

For more information on the Tag Browser, see Chapter 6, "Tagname Dictionary."

Caution! Tagnames that are modified (written to) in a Condition Script or a Data Change Script should **not** be used as the tagname for a Data Change Script or in the expression of a condition script.

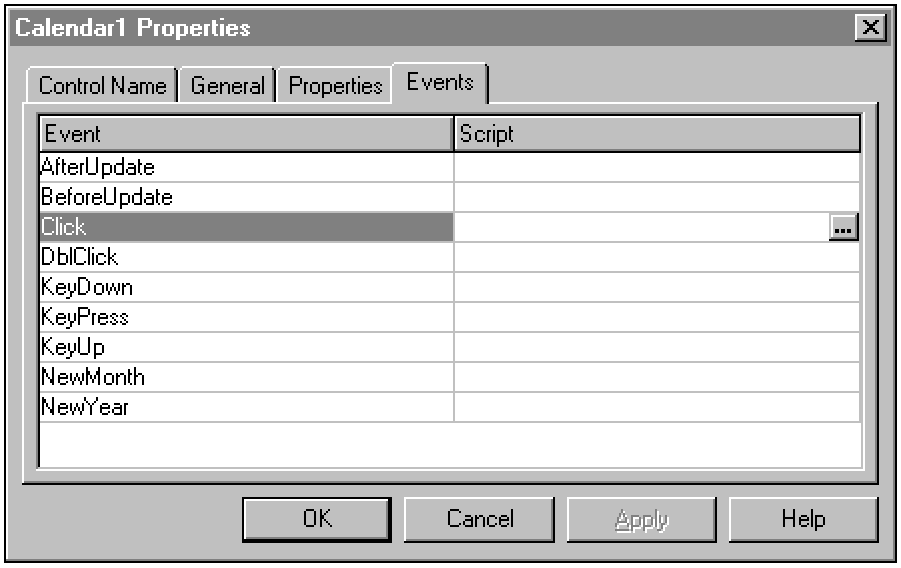
For example: Script1 calls Script2 which calls Script3 which attempts to call Script1 (which is still active). The final call will not be executed. In order to avoid this instance do not use tagnames that are modified in a Condition script or a Data Change script as the tagname for a Data Change script or in the expression of a Condition script.

ActiveX Event Scripts

Most ActiveX controls have events associated with them. For example, click, double-click, mouse down and key press are typical events used in many ActiveX controls. InTouch ActiveX Event scripts are provided to support event actions. You can associate one ActiveX Event script to each event. You execute ActiveX control events in runtime (WindowViewer).

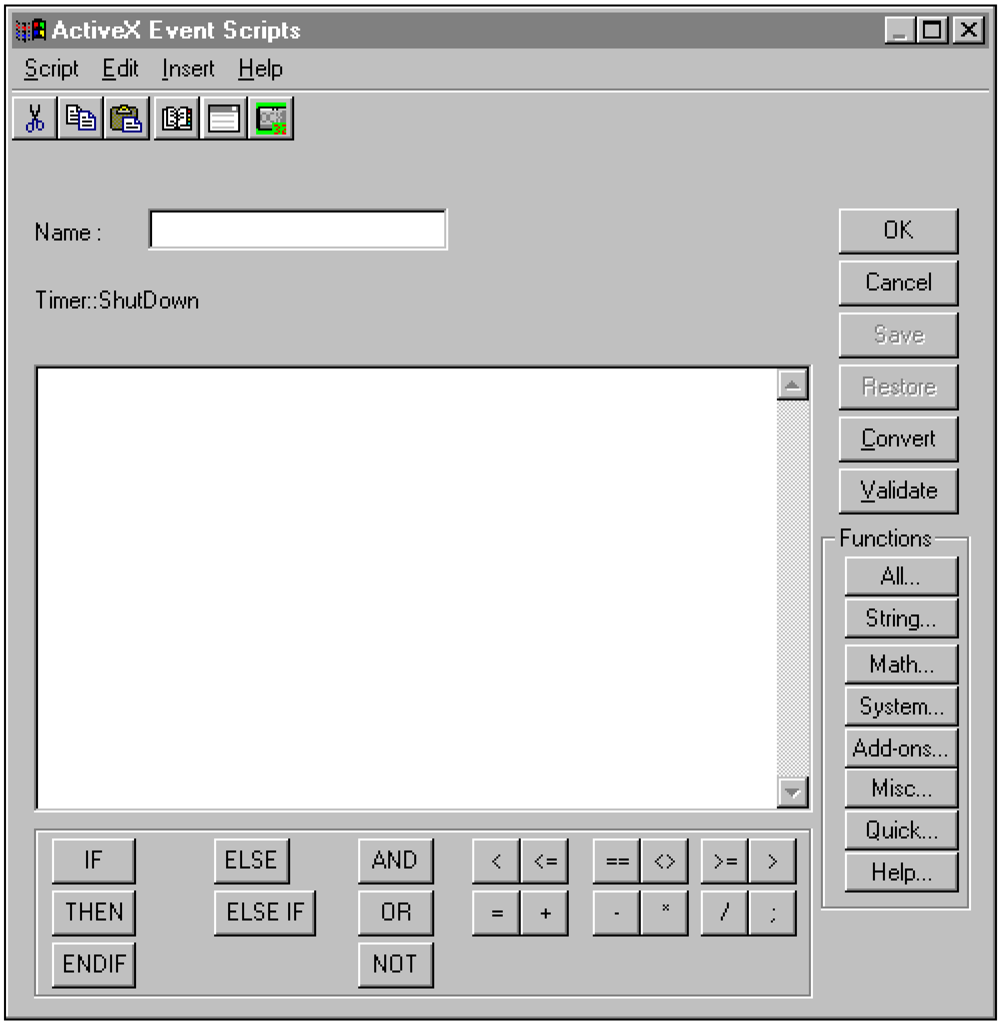
To access the ActiveX Event Script editor

1. Click the **Events** tab in the ActiveX control's **Properties** dialog box to activate the **Events** property sheet. For example:



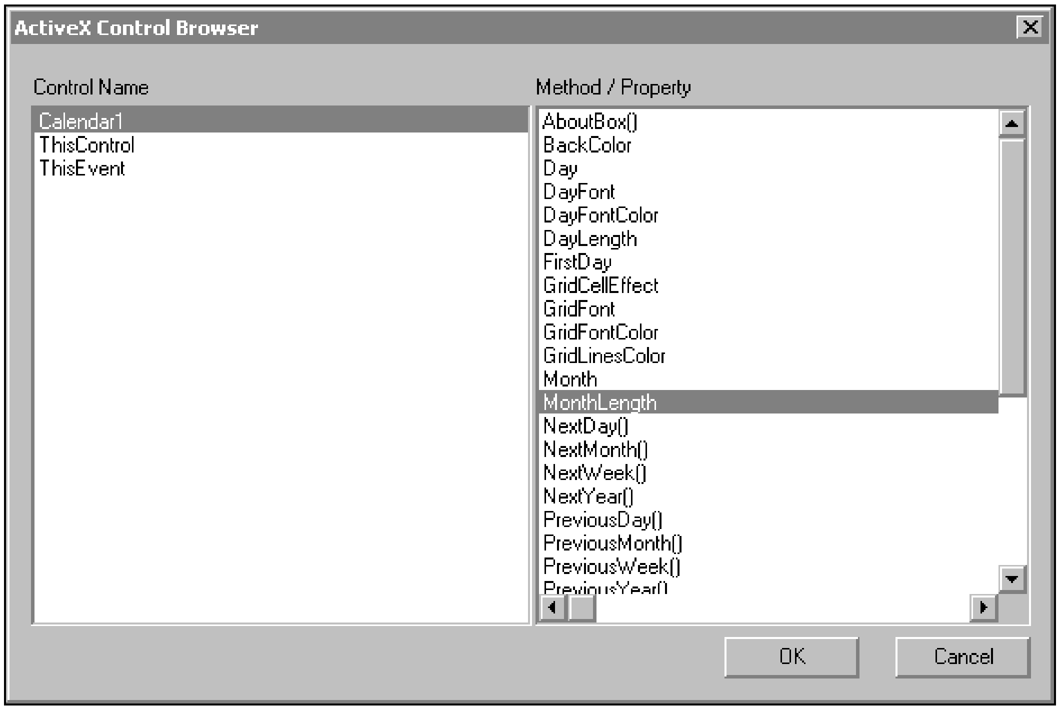
Event	Script
AfterUpdate	
BeforeUpdate	
Click	
DblClick	
KeyDown	
KeyPress	
KeyUp	
NewMonth	
NewYear	

2. Double-click a blank cell in the **Script** column, or type a name for the ActiveX Event script and click **OK**.
3. If an ActiveX Event script for the name you type does not currently exist, a message box appears asking you if you want to create it now. Click **OK**. The **ActiveX Event Scripts** editor appears.



4. In the **Name** box, type the name that you want to use to identify the ActiveX Event script.

5. ActiveX control methods are similar to ActiveX control properties. Properties are data members associated with the object while methods are data functions that control the object. To access the ActiveX control methods, on the **Insert** menu, click **ActiveX**. The **ActiveX Control Browser** appears.



The **ActiveX Control Browser** will display the names of all ActiveX controls being used in your application. When you select a control's name, its respective methods will be displayed. Select the method that you want to insert into your script, and then click **OK**.

QuickFunctions

QuickFunctions are scripts that you can write and call from other scripts or expressions. They are stored in the application in which they are created. Calling QuickFunctions from other scripts or expressions allows you to create a script one time and then reuse it. Reusing these scripts decreases your application maintenance by reducing the amount of duplicate code that is copied and pasted over and over into scripts. The reused code is stored in one script and in one location, thereby supporting update of all script instances with one edit session.

QuickFunctions can be defined as asynchronous, so that, when executed, they will run in the background of the main WindowViewer (runtime) process. This functionality allows WindowViewer to separate time-consuming operations such as SQL database calls, from the main program flow. When these time consuming operations need to be performed, by creating an asynchronous QuickFunction, you provide a way for all animation links and other InTouch functionality to remain active while the operation executes.

For example, let's assume that you are in need of doing a calculation in a for next loop. You can use an asynchronous QuickScript so that you will not hold up execution of your InTouch system while the calculation is taking place. You can then use the **IsAnyAsyncFunctionBusy()** function to determine when the calculation is complete and display the result to the operator.

```
If IsAnyAsyncFunctionBusy(120) ==1 then  Show "Calculation
    running Window";  Endif;
```

Note We do not recommend running SQL commands in an asynchronous QuickScript as failures in any SQL command may cause system instability.

For more information on asynchronous scripts, see Asynchronous QuickFunction Scripts."

Tip The animation link expression boxes are limited to 255 characters. However, you can create a more complex QuickFunction and then, call it from an animation link expression box. This allows you to use the **CALL** statement to call a complex script that contains a **RETURN** statement that returns the result back to the expression.

For example, when several tagnames of 30 characters each are added (using "**&**" and "**:**") together, you can only use 8 tagnames (plus spaces) in the expression. However, by using the statement, **CALL MYSCRIPT($SECOND)**, in the expression, you can execute a QuickFunction that might contain hundreds of 30 character tagnames. This QuickFunction will use the RETURN statement to provide a value back to the expression.

Note A "trigger" tagname must be used as a parameter for the QuickFunction to force an Animation Link to update. For example, using the tagname $Second as the parameter for the QuickFunction will cause the expression for the Animation Link to be evaluated every time $Second changes value thus, resulting in the QuickFunction being called every second.

The QuickFunction RETURN statement cannot be used to return message or string type values.

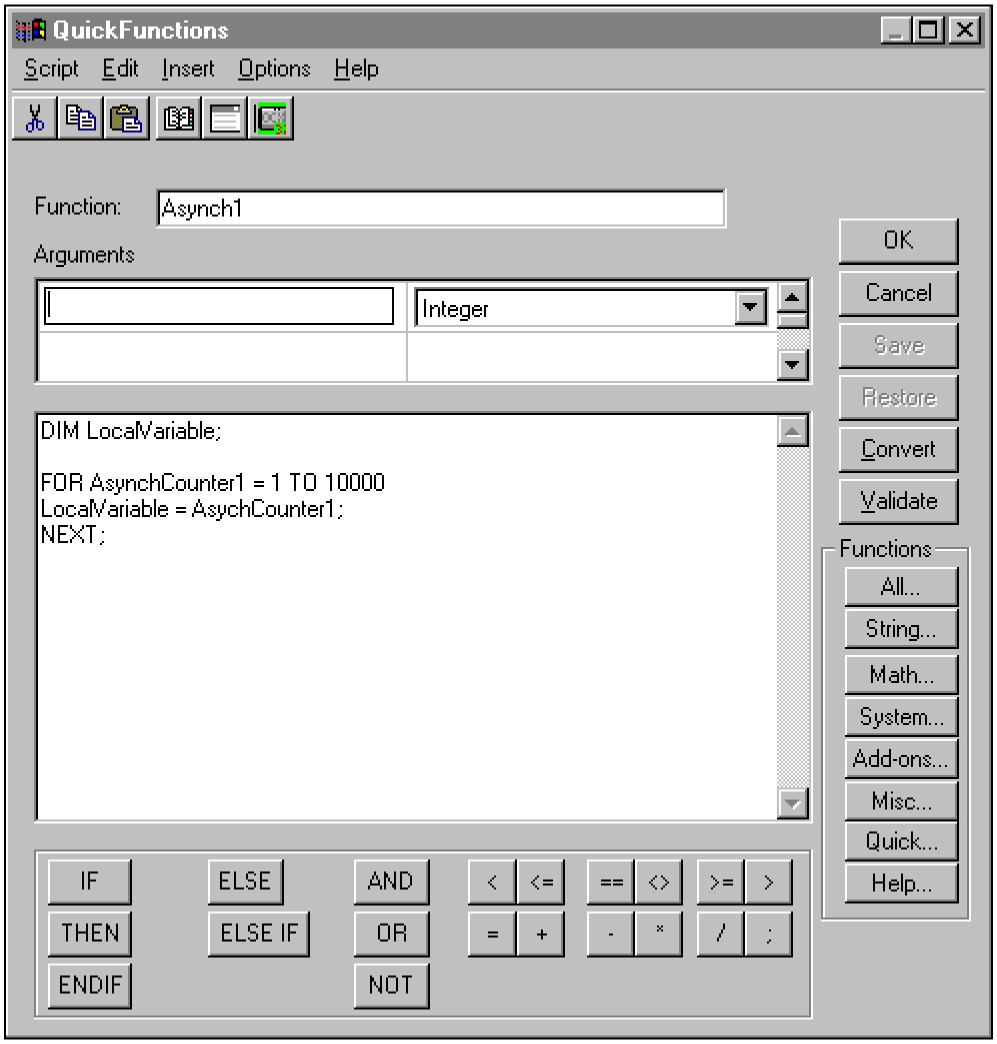
Once you create a QuickFunction and save it, you can immediately call it from any other script or expression by its name.

For more information on the syntax used for QuickFunctions, see your online *InTouch Reference Guide*.

To create a QuickFunction

1. On the **Special** menu, point to **Scripts**, and then click **QuickFunctions**, or in the Application Explorer under **Scripts**, double-click **QuickFunctions**. The **QuickFunctions** script editor appears.

Tip In the Application Explorer, under **Scripts**, you can also right-click **QuickFunctions**, and then click **Open**.



Function: Asynch1

```
DIM LocalVariable;

FOR AsynchCounter1 = 1 TO 10000
LocalVariable = AsychCounter1;
NEXT;
```

2. In the **Function** box, type the name for the QuickFunction.

Tip The name can be up to 31 characters in length. (Blank spaces and duplicate names cannot be used.) This is the name that other QuickScripts or expressions will use to call the QuickFunction. This name will also appear in the **Choose function** dialog box when you click either the **All** or the **Quick** buttons in the QuickScript editor, or on the **Insert** menu, under **Functions**, when you click either the **All** or **Quick Functions** command.

3. In the **Arguments** boxes, type each argument name for your QuickScript and then, click the argument's arrow and select its data type in the list.

   The valid data types are:

Data Type	Description
Integer	Used to pass integer variable, tagname or constant values.
Real	Used to pass real variable, tagname or constant values.
Discrete	Used to pass discrete variable, tagname or constant values.
Message	Used to pass string variable, tagname or constant values.

   The following are reserved keywords that should not be used as argument names:

Return	**Call**	**Dim**	**As**	**RetVal**
Integer	**Real**	**Discrete**	**Message**	

   Argument names are local variables that exist only within the QuickFunction in which they are defined. You can use up to 16 arguments per QuickFunction. The argument names can be up to 31 characters in length and spaces cannot be used. The argument names must also begin with an alpha character (A-Z). Duplicate names cannot be used.

   Do not use tagnames for argument names. Tagnames take precedence over argument names that are the same name and your QuickScript will not execute properly. An argument name does not consume a tagname count because they are treated as local variables.

4. Once you have typed in the argument names and selected their type, you are ready to write your QuickFunction.

QuickFunction Argument Expressions

Script parameters are passed by value. Argument expressions can be any script expression that returns an integer, real, discrete, message data type value. All argument expression values are resolved by the calling script before executing the QuickFunction. Examples:

```
CALL Stuff (5.6, 237, "PI");
```

In this expression, The Real constant 5.6 as argument1, Integer constant 237 as argument 2, Message constant "PI" as argument 3.

```
CALL Temp (IntegerTag);
```

IntegerTag passed as argument expression value.

```
CALL ValveOpen (Tag.MaxEU -5);
```

Calculated value (Tag.MaxEU -5) passed as argument expression value.

Argument Data Type Matching

There must be a strict left to right correspondence between the data types of the calling statement's argument list and the data types of the saved QuickFunction which is called. There must also be an exact number of arguments to match the number of arguments in the saved QuickFunction argument list as well. Coercion is used to type cast Real values to Integer and Integer values to Real. This ability to modify the type of a value, allows any analog argument to be passed to any other analog types.

For example, If you pass a Real value of 1.23 to an Integer argument it will use only the 1 and the .23 will be lost. Similarly, if you pass an Integer value of 1 to a Real argument it will promote the 1 to 1.0. However, despite this coercion capability we recommend that you make a strict match of calling argument types to the corresponding QuickFunction argument list.

Valid QuickFunction Syntax

QuickFunctions return a value. The QuickFunction statement syntax and form are as follows:

```
CALLQuickFunctionName ( [arg1, ... arg16] );
```

Where:

CALL	Is the required keyword in all QuickScripts and expressions to call a QuickFunction.
QuickFunctionName	Is a 1 to 31 character string that corresponds to the name assigned to the saved QuickFunction.
([arg1, ...arg16])	Is 0 to 16 comma separated argument expressions enclosed in parentheses.

For more information on the syntax used for QuickFunctions, see your online *InTouch Reference Guide*.

Using the Return Statement

A QuickFunction is a script that can be called from another script (calling script). The following is an example of a script that is calling a QuickFunction:

```
RETURNRESULT = CALL MyFunction( StartHour, EndHour);
```

The **RETURN** statement is used by a QuickFunction to force a value to be 'returned' to the calling script. When **RETURN** is encountered, an immediate end to the execution of the QuickFunction occurs. At this point, the QuickFunction returns a value back to the calling script. The data type returned can be a discrete, integer, or real.

In the example above, the tagname, **RETURNRESULT**, must be a discrete, integer, or real type tagname to receive the transferred value. When a **RETURN** statement is encountered, no further execution of the QuickFunction is processed.

In the example below, if the tagname ConvertType is equal to 0, the expression abs((Max / Maxprogress) * 100) is calculated and this data type real result is returned to the calling script.

```
IF ConvertType == 0 THEN
   RETURN abs( (Max / Maxprogress) * 100 );
ELSE
   RETURN 0;
ENDIF;
```

The data type of the return value is determined by the context. For example:

```
RETURN AnalogTag;
```

If the Tagname, **AnalogTag**, is defined as a Memory Integer, the **RETURN** statement will send a analog value back to the calling location. Only one value can be returned.

Asynchronous QuickFunction Scripts

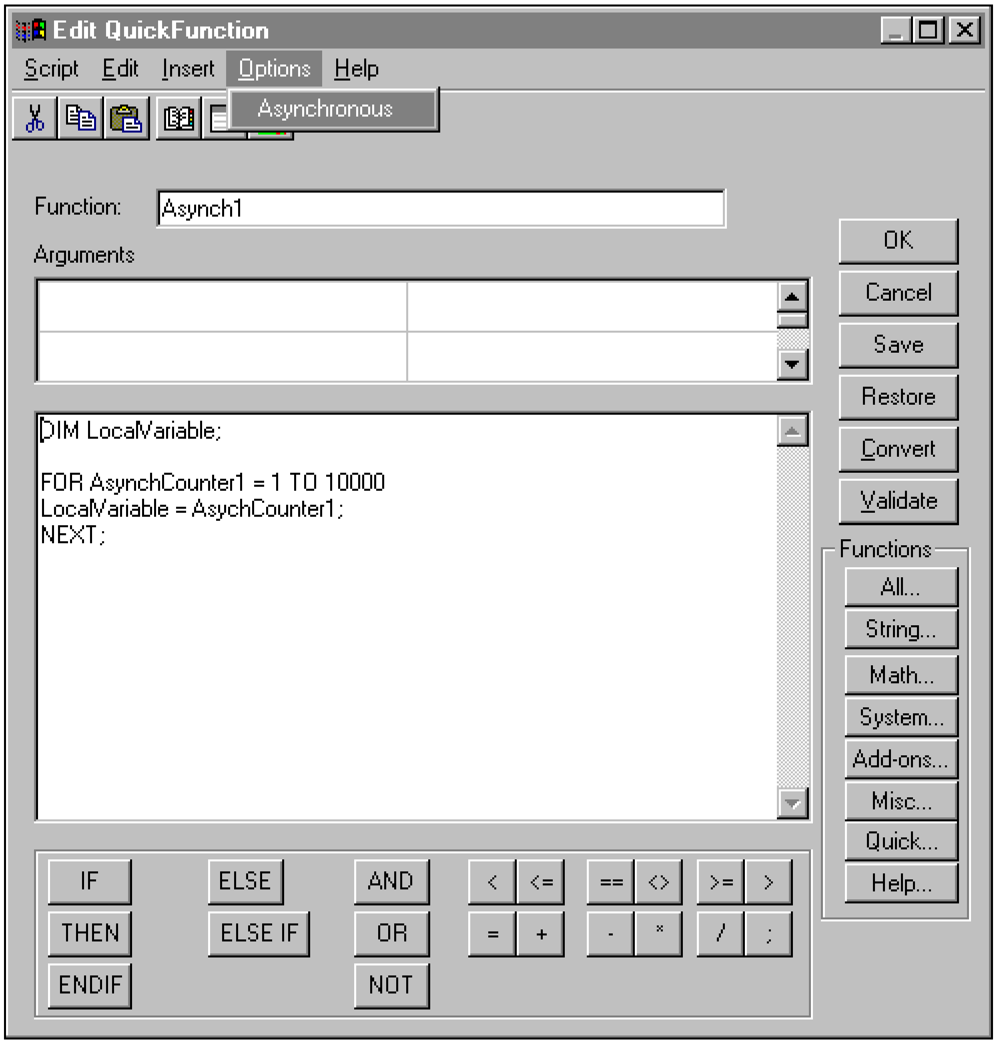
You can define your QuickFunctions as asynchronous. (QuickFunctions are the only InTouch script type that support asynchronous functionality.) When WindowViewer encounters a call to an asynchronous QuickFunction, a separate thread gets spawned. Once the new thread is spawned, WindowViewer is free to continue to call more scripts (including more asynchronous QuickFunctions), wait for the asynchronous scripts to end, or do graphic window updates. It is from the newly spawned thread that the asynchronous QuickFunctions will actually be launched. Once the asynchronous script completes execution, the newly spawned thread ends. The threading mechanism is transparent to the operator.

Note Asynchronous QuickFunctions cannot return a value. Therefore, asynchronous QuickFunctions cannot be used in animation link expressions. Additionally, there is no limit to the number of asynchronous QuickFunctions that you can execute simultaneously. However, to ensure adequate system performance, it is highly recommended that no more than three be executed simultaneously. Additionally, it is not possible to run more than one instance of any asynchronous QuickFunction concurrently.

To create an asynchronous QuickFunction

1. Create a QuickFunction.

2. On the **Options** menu, click **Asynchronous**. For example:



Function: Asynch1

Arguments

```
DIM LocalVariable;

FOR AsynchCounter1 = 1 TO 10000
LocalVariable = AsychCounter1;
NEXT;
```

Note In runtime, asynchronous QuickFunctions cannot be stopped once they begin executing. However, if the operator stops all scripts from running (through the **Logic** menu **Halt Logic** command or by pressing a button linked to the **$LogicRunning** system tagname) no new asynchronous QuickFunctions will begin executing.

Controlling Asynchronous Scripts

You can use the **IsAnyAsyncFunctionBusy()** function to find out if any asynchronous QuickFunctions are running. This function can be used to make the QuickScript that calls an asynchronous QuickFunction wait for all other asynchronous QuickFunctions to complete processing. This allows the QuickScript to re-synchronize itself.

The valid syntax for this function is:

DiscreteTag **= IsAnyAsyncFunctionBusy(***timeout***);**

Where:

DiscreteTag	Is a discrete type tagname to which a value is returned as follows: If the function times out, waiting all executing QuickFunctions to complete, a 1 (true) is returned to *DiscreteTag.* If there are no asynchronous QuickFunctions running, a value of 0 (false) will be returned immediately, or the QuickFunction will wait for the timeout to elapse. It will also return a 0 if after timing out there are no asynchronous QuickFunctions running.
timeout	Is an integer value representing the number of seconds to wait to determine if any asynchronous QuickFunctions are running.

For example, let's assume you want to connect to several SQL databases using asynchronous QuickFunctions and, you know that it will take 2 minutes to make those connections. First, you would execute the asynchronous QuickFunctions to connect to the SQL databases. Next, you would launch the function, **IsAnyAsyncFunctionBusy(120)** to allow enough time for SQL to make the connections before completing the QuickFunction.

However, if after 2 minutes the connections have not been made and the asynchronous QuickFunctions are still busy trying to make the connections, a value of 1 (true) will be returned by the **IsAnyAsyncFunctionBusy()** function. You can now display an error message telling the operator that the SQL connections were unsuccessful.

For example, you could use the following window **On Show** QuickScript:

```
IF IsAnyAsyncFunctionBusy(120) == 1 THEN
   SHOW "SQL Connection Error Dialog";

ENDIF;
```

Using Local Variables

You can declare local variables within a script to store temporary results and create complex calculations with intermediate scripting values without impacting or decreasing your licensed tagname count and increase performance.

Local variables or tagnames can be used interchangeably within the same script. However, local variables loose their value and meaning once the script ends where, tagnames are global and retain their values. Unlike tagnames, local variables are declared within the body of the script. The number of local script variables that you can declare within a given script body is limited only by your available memory. Once you have declared a local variable, you can include it in one or more expressions within the same script body. The expression and syntax rules for the placement of local variable names within a script body are the same as those for tagnames, with one exception, local variables do not support **.field** references.

Like tagnames, local variables can be used on both the left and right hand side of statements and expressions that include other local variables and tagnames of different data-types.

Note It is not possible to define a local variable with the same name as an existing tagname. It is possible to create a tagname with the same name as an already defined local variable. If this situation occurs the local variable takes precedence over the tagname and the tagname value will reflect any changes to the local variable.

Valid Local Variable Syntax

Each local variable must be declared within the script as a separate **DIM** statement. (One per line cascading is not permitted.) The **DIM** statement syntax and format are as follows:

```
DIM LocalVarName [ AS data-type ];
```

Where:

DIM	Is a required keyword.
LocalVarName	Is a variable name that conforms to tagname format and restrictions. Variable names can be up to 32 characters long and must begin with A-Z or a-z. The remaining characters can be: A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &. If there is a conflict in your script between a declared variable name and a tagname, (both are the same name) the variable name takes precedence over the tagname. For example, let's assume that you have a tagname defined in your database as "Temp," and you declare "**DIM** *Temp* **AS** *Integer*;". Within the declaring script, expressions using "Temp" in a statement will refer to the value associated with the local variable "Temp" rather than the tagname "Temp."

AS	Is an optional keyword.
	
	If you omit the **AS** clause from the **DIM** statement, by default, the variable will be declared as an integer data-type. For example:
	
	`DIM LocVar1;`
	
	is equivalent to:
	
	`DIM LocVar1 AS Integer;`

data-type	Can be any one of the following four keywords:
	Integer
	`DIM LocVar1 AS Integer;`
	
	Real
	`DIM LocVar2 AS Real;`
	
	Discrete
	`DIM LocVar3 AS Discrete;`
	
	Message
	DIM *LocVar4* **AS** *Message*;

The InTouch **DIM** statement cannot be cascaded. For example, the following examples are invalid and cannot be used:

```
DIM LocVar1 AS Integer, LocVar2 AS Real;
DIM LocVar3, LocVar4, LocVar5, AS Message;
```

To declare the multiple variables in InTouch, you must enter a separate DIM statement for each variable. For example, the following examples are valid:

```
DIM LocVar1 AS Integer;
DIM LocVar2 AS Real;
```

Note Data-type keywords are case insensitive.

The **DIM** statement line must be terminated with a semi-colon (;).

Cascaded **DIM** statements are not supported.

The **DIM** statement can be located anywhere within the script body. But must precede the first referencing script statement or expression.

If the local variable is referenced before the **DIM** statement, the script editor will read it as a tagname when the script is validated and you will be asked to define it.

For more information on the syntax used for local variables, see your online *InTouch Reference Guide*.

Creating FOR-NEXT Loop Scripts

A FOR-NEXT loop is used to perform a function (or set of functions) within a script several times during a single execution of a script. The general format of the FOR-NEXT loop is as follows:

```
FOR AnalogTag = start_expression TO end_expression [STEP
   change_expression]
   ...statements...
IF (condition) THEN
   [EXIT FOR;]
   ENDIF;
   ...statements...
```

```
NEXT;
```

Where:

[ ]	Items enclosed in brackets are optional parameters.
BOLDCASE	Bold items in **UPPERCASE** denote script language reserved keywords.
italics	Items in lowercase *italics* denote variable data.
AnalogTag	An InTouch Analog type tagname.
start_expression	A valid InTouch expression, to initialize *AnalogTag* to a value for execution of the loop.
end_expression	A valid InTouch expression, if *AnalogTag* is greater than *end_expression*, execution of the script jumps to the statement immediately following the **NEXT** statement. (This holds true if loop is incrementing up, otherwise, if loop is decrementing, loop termination will occur if IntegerTag is less than *end_expression.*)
change_expression	A valid InTouch expression, to define the increment or decrement value of *AnalogTag* after execution of the **NEXT** statement. **Note** The *change_expression* can be either positive or negative. If *change_expression* is positive, *start_expression* must be less than or equal to *end_expression* or the statements in the loop will not execute. If *change_expression* is negative, *start_expression* must be greater than or equal to *end_expression* for the body of the loop to be executed. If **STEP** is not set, then *change_expression* defaults to 1.
...statements...	One or more valid InTouch script language statements. These could be nested **FOR** loops. Nested loops require different *change_expressions* from outer loops.
FOR	Signals the beginning of the "For" loop.
TO	Signals the beginning of the *end_expression*.
STEP	Signals the beginning of the *change_expression*.
EXIT FOR	Immediately terminates the loop with script execution jumping to the statement immediately following the **NEXT** statement.
NEXT	Signals the end of the loop statement.

When you execute a FOR...LOOP function, InTouch:

1. Sets AnalogTag equal to start_expression.
2. Tests to see if AnalogTag is greater than end_expression. If so, InTouch exits the loop. (If change_expression is negative, InTouch tests to see if AnalogTag is less than end_expression.)
3. Executes the statements.
4. Increments AnalogTag by 1 - or by change_expression if it is specified.

5. Repeats steps 2 through 4.

Nesting FOR-NEXT Loops

FOR-NEXT loops may be nested. Number of levels of nesting possible depend on your system's memory and resource availability.

Screen Updates and Performance Penalties

During execution of the FOR-NEXT loop, the screen update sub-system within InTouch will pause until the loop is complete.

- All animation on the screen will pause during execution of the FOR-NEXT loop. Therefore, you cannot use the FOR-NEXT loop to animate an object to move across the screen, since all movement will occur only after the loop has completed.
- Real-time trends will pause.
- Historical trends will pause, if "updating."
- Displayed values will not update on the screen during loop execution. Although there may be new values present within those variables, those new values will only be displayed after the loop has completed.
- The value of any I/O type tagname modified within the body of a FOR-NEXT loop will be transmitted to the I/O server only after loop execution is finished. Therefore, if you modify the value of a I/O type tagname during each interaction of a FOR-NEXT loop, only the final value of that I/O type tagname will be transmitted to the PLC.

Note FOR-NEXT loops will pause all operations in InTouch. While executing, no data moves in or out of WindowViewer, no animation links are updated and, no other scripts are executed including asynchronous QuickFunctions. However, FOR-NEXT loops used inside asynchronous QuickFunctions do not pause other operations.

Loop Execution Time Limit

By default, FOR-NEXT loops must complete execution within 5 seconds. This is a safety limit built into the FOR-NEXT sub-system. The time limit will be enforced for all FOR-NEXT loops. For example:

```
FOR X = 1 TO 1000000
   FileWriteMessage("C:\LOG.TXT","Hello");
NEXT;
```

Note You can extend this limit by adding the following switch to your INTOUCH.INI file in your application directory:

```
LoopTimeout=20
```

Where; 20 is the number of seconds before terminating the loop prematurely.

This loop will most likely exceed the time limit of 5 seconds. In the Logger a message appears that indicates the following:

```
95/03/07 07:34:40.550/VIEW    /
   Exceeded loop time limit of 5 seconds.
```

```
95/03/07 07:34:40.550/VIEW    /
   FOR-NEXT Timeout at X = 65464
```

This error message indicates that the FOR-NEXT loop has terminated before meeting the *end_condition*, it also provides the value of the loop variable at the time of the loops termination. This information will allow you to track down which FOR-NEXT is having the problem.

Note The 5 second time limit is only evaluated each time the NEXT; statement is reached within the FOR-NEXT loop. For example, if you were executing the following script:

```
FOR Index = 1 to 10

SQLInsert(ConnectionID,"ORG","list1");
SQLInsert(ConnectionID,"ORG","list2");
SQLInsert(ConnectionID,"ORG","list3");
SQLInsert(ConnectionID,"ORG","list4");

NEXT;
```

If each **SQLInsert()** took 12 seconds to complete, all four inserts would be executed to completion before the loop exited because the five second time limit is evaluated only when the NEXT; statement is reached..

Loop Variable Value After Loop Execution

As in Visual Basic (and most other popular Basic programming languages) the value of the loop variable at the end of loop execution is defined as follows:

The Index continues to increase in value, starting at the *Start_Condition*, incremented each iteration by the value of *Step_Expression*, until it reaches the last iteration at which the Index value exceeds that of the *End_Expression*.

Therefore, if you had a loop as follows:

```
FOR Index = 2 TO 25 STEP 7
   { Some script statements }
NEXT;
```

The value of Index would progress as follows:

Iteration	Value	Calculation
1	2	
2	9	2 + 7
3	16	2 + 7 + 7
4	23	2 + 7 + 7 + 7
5	30	2 + 7 + 7 + 7 + 7

At that point, when the value reached 30, the loop would stop executing because it exceeded the *End_Expression*. The ending value of Index would be 30.

Nested Control Structures

Control structures can be placed inside other control structures (such as an IF...THEN block within a FOR...NEXT loop). A control structure inside another control structure is known as *nested*.

Example:

```
FOR TagX = 1 TO 5

FOR TagY = 1 TO 10
    ...statements...
    IF (condition) THEN
    [EXIT FOR;]
    ENDIF;
    ...statements...
    NEXT;

NEXT;
```

Where:

The first NEXT closes the inner FOR loop and the last NEXT closes the outer FOR loop. Likewise, in nested IF statements, the ENDIF statements automatically apply to the nearest prior IF statement.

Exiting a Control Structure

The EXIT FOR statement allows you to exit directly from a FOR loop. Syntactically, the EXIT FOR statement is simple. EXIT FOR can appear as many times as needed inside a FOR loop. For example:

```
FOR TagX = 1 TO 10;

...statements...

IF (condition) THEN
    EXIT FOR;

ENDIF;

...statements...

NEXT;
```

Below are some examples of various FOR-NEXT loop scripts:

Example 1"Simple Math 2"

This loop allows a person to configure the number with which to raise to a power, as well as the power to which they would like to raise it by setting up the value input links for the tagnames NumberToRaise and Power:

```
Product = 1;

NumberToRaise = 4;

Power = 12;

FOR Index = 1 TO Power
    Product = Product * NumberToRaise;

NEXT;
```

Once the above script has completed processing the value of the Product will be 16,777,216.

Example 2 "Complex FOR-NEXT using indirect tagnames"

This loop utilizes the "EXIT FOR" and "STEP" portions of the FOR-NEXT construct to perform a search on a set of 100 tagnames, searching for the tagname to which NumberEntered is equivalent.

Note For this example, it is assumed that there are 100 **Memory Integer** tagnames (TAG1 - TAG100) already existing. An operator enters a number into the tagname NumberEntered, and this loop searches TAG1 - TAG100 for a matching value. In addition, there is an indirect analog tagname created: *IndirectTag*

```
Found = 0;

FOR Index = 1 TO 100
   IndirectTag.NAME = "TAGNAME" + TEXT( Index, "#" );
   IF (IndirectTag.NAME == ("TAGNAME"+
      Text(NumberEntered,"#"))) THEN
      Found = 1;
      EXIT FOR;
   ENDIF;

NEXT;

IF (Found==1) THEN
   Show "NumberFound"; {window notifying search was
      successful}

ELSE
   Show "NumberNotFound";

ENDIF;
```

Once the script has completed processing, a window will be displayed either indicating that the number was found, or not.

Note Notice the use of two addition functions within this script: **Show()** and **TEXT()**.

Example 3

This loop performs an odd calculation, but illustrates the use of nested FOR-NEXT loops, as well as the use of the "STEP" portion of the FOR-NEXT construct:

```
MyTag = -1;

FOR Index = 1000 TO -1000 STEP -5
   IF (MyTag > Index) THEN
      FOR Index2 = 1 TO 10 STEP 2
         MyTag = MyTag * (Index + 11);
      NEXT;
   ENDIF;

NEXT;
```

Once the script has completed processing the values will be:

MyTag = -7776, Index = -1005 and Index2 = 11.

Script Editing Styles and Syntax

The InTouch script editor supports two "styles" of scripts: "Simple" and "Complex." Simple scripts allow assignments, comparisons, simple math functions, and so on. Complex scripts provide the ability to perform logical operations in the form of IF-THEN-ELSE type statements. In addition, InTouch also supports the use of built-in complex functions, as well as native QuickFunctions.

An example of a function would be the **StartApp(*ApplicationName*)** function actually started the Windows application identified in the argument, "(*ApplicationName*)". Functions may be used in both Simple and Complex scripts. The following section includes complete descriptions of each style.

Required Syntax for Expressions and Scripts

The syntax used in QuickScripts and animation link expression dialog boxes is similar to the algebraic syntax of a calculator. Most expressions are assignment statements written in the following form:

```
a = (b - c) / (2 + x) * xyz;
```

This statement will cause the value of the expression to the right of the equal sign (=) to be placed in the variable location named "a." Every expression must end with a semicolon (;). The operands in an expression may be constants or variables. A single tagname must appear to the left of the assignment operator =.

Memory or **I/O Message** type tagnames can be concatenated by using the plus (+) operator. For example, tagnames can be concatenated for use in **Indirect** type tagnames. If a Data Change Script such as the one below is created, each time the value of "Number" changes, the indirect tagname "Setpoint" will change accordingly:

```
Number=1;

Setpoint.Name = "Setpoint" + Text(Number, "#" );
```

Where: The result is "Setpoint1."

Simple Scripts

Simple scripts provide the ability to implement logic such as assignments, math and functions. An example of this type of scripting is:

```
React_temp = 150;

ResultTag = (Sample1 + Sample2)/2;

{this is a comment}

Show "Main Menu";
```

In this example, the script will assign the value of "150" to the tagname "React_temp." "Sample1" will be added to "Sample2" and the result divided by "2"and the "Main Menu" window appears on the screen when the script is run.

Note Notice that each logical statement must end with a semicolon (;) and that several logical statements may be included in one script. Also notice that comments are allowed within the script editor. Comments are identified by a pair of braces {}. The function **Show** was also used with the argument "Main Menu" (WindowName) to cause the specified window to open.

In addition to simple assignments, mathematical operators and functions, InTouch supports several other "Operations" for use on "Operands," that is, tagnames, number constants, and so on.). **Discrete**, **Integer** and **Real** tagname types are supported for all operations listed below. **Message** tagname types may be used with comparison and assignment operations only. The following is a list of InTouch supported operations:

Operations that Require 1 Operand

~	Complement
-	Negation
NOT	Logical NOT

Operations that Require 2 Operands

*	Multiplication
/	Division
+	Addition and Concatenation
-	Subtraction
=	Assignment
MOD	Modulo
SHL	Left Shift
SHR	Right Shift
&	Bitwise AND
^	Exclusive OR
|	Inclusive OR
**	Power
<	Less than
>	Greater than
<=	Less than or Equal to
>=	Greater than or Equal to
==	Equivalency ("is equivalent to")
<>	Not Equal to
AND	Logical AND
OR	Logical OR

Precedence of Operators

The following list shows the order of precedence for evaluation of operators. The first operator in the list is evaluated first, the second operator is evaluated second, and so on. Operators in the same line in the list have equivalent precedence. Operators are listed from highest precedence to lowest precedence.

( )Highest Precedence

- , NOT, ~

**

* , /, MOD

+, -

SHL, SHR

<, >, <=, > =

==, <>

&

^

|

=

AND

ORLowest Precedence

Precedence Examples

Since * has higher precedence than +,

B + C * D; is equivalent to B + (C * D);

Since * and / have equivalent precedence,

B / C * D; is equivalent to (B / C) * D;

Some other examples to note:

B * - D; is equivalent to B * (-D);

B or C and D; is equivalent to B or (C and D);

Descriptions of Operators

Arguments for the previously listed operators can be numbers or tagnames. Putting parentheses around the arguments is optional, and the operator names are not case-sensitive.

Parentheses ( )

Parentheses are used to ensure the correct order of evaluation for the operations. They can also make a complex expression easier to read. Operations in parentheses are evaluated first (preempting the other rules of precedence that would apply in the absence of parentheses). If the precedence is in question or needs to be overridden, use parentheses. In the example below parentheses are used to force B and C to be added together before multiplying by D:

```
( B + C ) * D;
```

Negation (-)

Negation is an unary operator that converts a positive integer or real number into a negative number.

Complement (~)

This operator yields the one's complement of a 32-bit integer. In other words, it converts each zero-bit to a one-bit and each one-bit to a zero-bit. The one's complement operator is an unary operator that accepts an integer operand.

Power (**)

This binary operator returns the result of a number (the base) raised to the power of a second number (the power). The base and the power can be any real or integer numbers, subject to the following restrictions:

A zero base and a negative power are invalid.

Example: **"0 ** - 2"** and **"0 ** -2.5"**

A negative base and a fractional power are invalid.

Example: **"-2 ** 2.5"** and **"-2 ** -2.5"**

Invalid operands yield a zero result. Moreover, the result of the operation should not be so large or so small that it cannot be represented as a real number. Example:

```
1 ** 1 = 1.0

3 ** 2 = 9.0

10 ** 5 = 100,000.0
```

Multiplication (*), Division (/), Addition (+), Subtraction (-)

These binary operators perform basic mathematical operations. The plus (+) is also used to concatenate **Memory** or **I/O Message** type tagnames. For example, tagnames can be concatenated for use in **Indirect** tagnames. If a Data Change Script such as the one below were created, each time the value of "Number" changed, the indirect tagname "Setpoint" would change accordingly:

```
Number=1;

Setpoint.Name = "Setpoint" + Text(Number, "#" );
```

Where: The result would be "Setpoint1."

Modulo (MOD)

MOD is a binary operator that divides an integer quantity to its left by an integer quantity to its right. The remainder of this division is the result of the MOD operation. Example:

```
97 MOD 8 yields 1

63 MOD 5 yields 3
```

Shift Left (SHL), Shift Right (SHR)

SHL and **SHR** are binary operators that use only integer operands. The binary content of the 32-bit word referenced by the quantity to the left of the operator is shifted (right or left) by the number of bit positions specified in the quantity to the right of the operator. Bits shifted out of the word are lost. Bit positions vacated by the shift are zero-filled. (The shift is an unsigned shift.)

Bitwise AND (&)

This is a bitwise binary operator which compares 32-bit integer words with each other, bit for bit. A common use of this operator is to mask a set of bits. The operation in this example would "mask out" (set to zero) the upper 24 bits of the 32-bit word. For example:

```
result = name & 0xff;
```

Exclusive OR (^) and Inclusive OR (|)

The **OR**s are bitwise logical operators which compare 32-bit integer words to each other, bit for bit. The Exclusive **OR** looks for opposite bit status's in corresponding locations. If the corresponding bits are the same, a zero is the result. If the corresponding bits differ, a one is the result. Example:

0 ^ 0 yields 0

0 ^ 1 yields 1

1 ^ 0 yields 1

1 ^ 1 yields 0

The Inclusive **OR** examines the corresponding bits for a one condition. If either bit is a one, the result is a one. Only when both corresponding bits are zeros is the result a zero. For example:

0 | 0 yields 0

0 | 1 yields 1

1 | 0 yields 1

1 | 1 yields 1

Assignment (=)

Assignment is a binary operator which accepts integer or real operands. Each statement may contain only one assignment operator. Only one name can be on the left side of the assignment operator. Read the equal sign (=) of the assignment operator as "is assigned to" or "is set to."

Note Do not confuse the equal sign with the equivalency sign (==) used in IF-THEN-ELSE clauses and relational contacts.

Comparisons (<, >, <=, >=, ==, <>)

These operators are used in IF-THEN-ELSE statements to execute various instructions based on the state of an expression.

AND, OR, and NOT

These operators only work on discrete tagnames. Although, if these operators are used on integers or reals, they are *converted* as follows:

Real to Discrete: If real is 0.0, discrete is 0, otherwise discrete is 1.

Integer to Discrete: If integer is 0, discrete is 0, otherwise discrete is 1.

Thus, if the statement were: "Disc1 = Real1 AND Real2;" and Real1 was 23.7 and Real2 was 0.0, Disc1 would have 0 assigned to it, since Real1 would be converted to 1 and Real2 would be converted to 0.

Complex Scripts

Complex scripts provide the ability to perform logical operations in the form of

IF-THEN-ELSE type scripts and the ability to process loops using FOR-NEXT script structures. The following is an example of an IF-THEN-ELSE script:

```
IF React_temp > 200 THEN
   React_temp_sp = 150;
   PRValve = 1;
   PlaySound("c:\alert.wav",1);

ELSE
   PRValve = 0;
   PlaySound("c:\All_Ok.wav",1);

ENDIF;
```

In this example, the script checks if the reactor temperature is higher than "200." If so, then the "React_temp_sp" tagname is assigned the value of "150," the "PRValve" is turned on and the "alert.wav" file is played by calling the **Playsound()** function. Else, the reactor temperature is lower than "200," the "PRValve" is turned off and the "All_Ok.wav" file is played.

Note Notice that each IF statement requires an ENDIF statement. Also be aware that an ELSE statement is not required if unnecessary for the script's functionality. Notice the use of the Function **PlaySound(*path_text,number*)** in this complex script.

Simple Math

This loop performs a simple iterative mathematical calculation. When executed, Product equal the value of NumberToRaise raised to the power of 10, that is, **Product=NumberToRaise^{10}**.

```
Product = 1;

NumberToRaise = 4;

FOR Index = 1 TO 10
   Product = Product * NumberToRaise;

NEXT;
```

Once the script executes, the value of the "Product" will be "1048576."

Note FOR-NEXT loops pause all operations in InTouch. While executing, no data moves in or out of WindowViewer, no animation links are updated and, no other scripts are executed including asynchronous QuickFunctions. However, FOR-NEXT loops used inside asynchronous QuickFunctions do not pause other operations.

IF-THEN-ELSE and Comparisons in Scripts

The IF-THEN-ELSE statement is used to conditionally execute various instructions based on the state of an expression. The following comparison operators are used to set up the conditions in an IF-THEN-ELSE statement:

<Less than

>Greater than

<=Less than or Equal to

>=Greater than or Equal to

==Equivalency ("is equivalent to")

<>Not Equal to

Below are some examples of various complex scripts:

IF-THEN statement with no ELSE clause:

```
IF a <> 0 THEN
   a = a + 100;
ENDIF;
```

IF-THEN-ELSE statement with one ELSE clause:

```
IF temp > 500 THEN
   Disc = 1;
   Real = 43.7;
ELSE
   Disc = 0;
   Real = 93.4;
ENDIF;
```

IF-THEN-ELSE statement with one ELSE IF clause and no ELSE clause:

```
IF temp > 500 THEN
   Disc = Disc * 10;
ELSE
   IF temp > 250 THEN
      x = y / z;
      a = abc + def;
   ENDIF;
ENDIF;
```

IF-THEN-ELSE statement with one ELSE IF clause and one ELSE clause:

```
IF temp > 500 THEN
   Disc = Disc - 10;
ELSE
   IF temp < 250 THEN
      Disc = Disc + 10;
```

```
    ELSE
        Disc = Disc + 50;
        Real = 100;
    ENDIF;
ENDIF;
```

Note Each IF must have a matching ENDIF and a semicolon must be entered at the end of each statement line.

IF-THEN-ELSE statement with multiple ELSE IF clauses and one ELSE clause:

```
IF temp > 100 THEN
    temphihi = 1
    Disc = 50;
ELSE
    IF temp > 80 THEN
        temphi = 1;
ELSE
    IF temp < 10 THEN
        templo = 1;
ELSE
    IF temp < 30 THEN
        templolo = 1;
ELSE
        tempok = 1;
    ENDIF;
ENDIF;
ENDIF;
ENDIF;
```

IF-THEN-ELSE statement that tests for Condition 1 *or* Condition 2:

```
IF (pump1 < 50.0) OR (pump2 < 50.0) THEN
    alarm-1 = 1;
ELSE
    alarm-1 = 0;
ENDIF;
```

IF-THEN-ELSE statement that tests for Condition 1 *and* Condition 2:

```
IF (pump1 < 50.0) AND (pump2 < 50.0) THEN
    alarm-2 = 1;
ELSE
    alarm-2 = 0;
ENDIF;
```

IF-THEN-ELSE statement that tests for equivalency:

```
IF a > 50 THEN
    IF b == 100 THEN
        c = 0;
    ENDIF;
ENDIF;
```

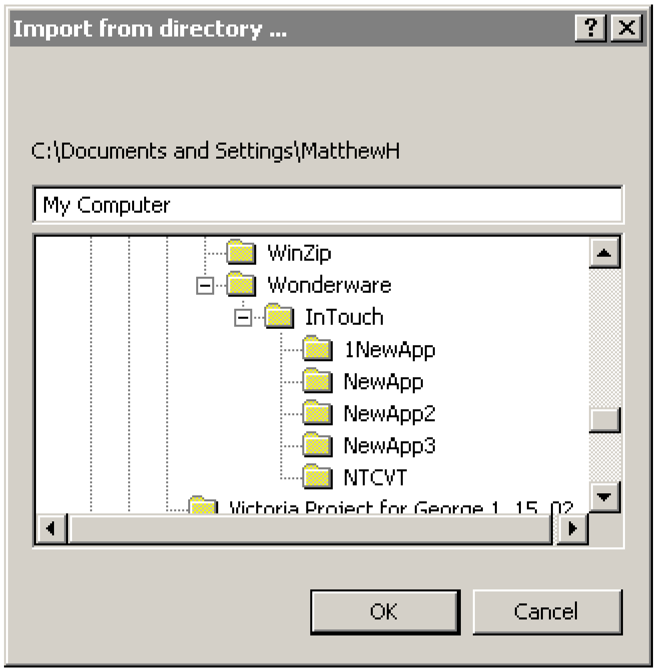
Importing QuickScripts

Importing QuickScripts from one InTouch application to your current application can save you a considerable amount of development time. It also provides you with a quick and easy method for creating remote tagname references. It allows you to reuse your previously created QuickScripts. When you move QuickScripts from one InTouch application to another, you must use the **Import** command on the **File** menu.

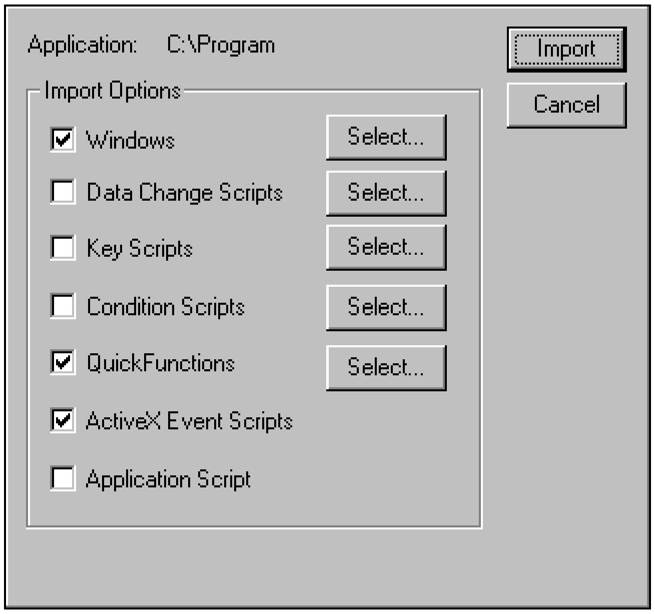
For more information on remote tagname references, see Chapter 6, "Tagname Dictionary."

To import a QuickScript

1. Close all windows in your current application
2. On the **File** menu, click **Import**. The **Browse for Folder** dialog box appears.


Import from directory ...
C:\Documents and Settings\MatthewH
My Computer
WinZip
Wonderware
InTouch
1NewApp
NewApp
NewApp2
NewApp3
NTCVT
Victoria Project for George 1_15_02
OK
Cancel

3. Locate and select the application directory (folder) containing the QuickScript(s) the you want to import.
4. Click **OK**. The following dialog box appears.



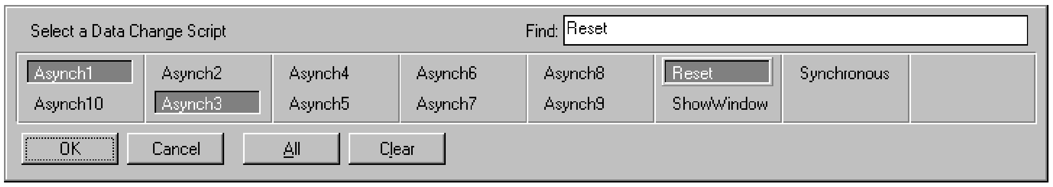
Application: C:\Program

Import Options

- [✓] Windows — Select...
- [ ] Data Change Scripts — Select...
- [ ] Key Scripts — Select...
- [ ] Condition Scripts — Select...
- [✓] QuickFunctions — Select...
- [✓] ActiveX Event Scripts
- [ ] Application Script

Import | Cancel

5. Select the QuickScript type(s) that you want to import.
6. Click **Select**. The **Select a *ScriptType* Script** dialog box appears.



Select a Data Change Script

Find: Reset

Asynch1	Asynch2	Asynch4	Asynch6	Asynch8	Reset	Synchronous	
Asynch10	Asynch3	Asynch5	Asynch7	Asynch9	ShowWindow		

OK | Cancel | All | Clear

7. Select the QuickScript(s) that you want to import, and then click **OK** to close the dialog box.

Note When you import ActiveX Event scripts, from one application to another, all ActiveX Events scripts are imported. Additionally, in order for an imported ActiveX Event script to function properly in the new application, the same ActiveX control and the same event for which the script was originally created, must also be used in the new application, and it must be loaded into memory. If the window containing an ActiveX control is closed, its ActiveX Event scripts, or any other InTouch QuickScripts containing script functions associated with that ActiveX control, will not execute properly.

8. Click **Import**. The system will automatically begin to import the selected QuickScript(s) into your current application.

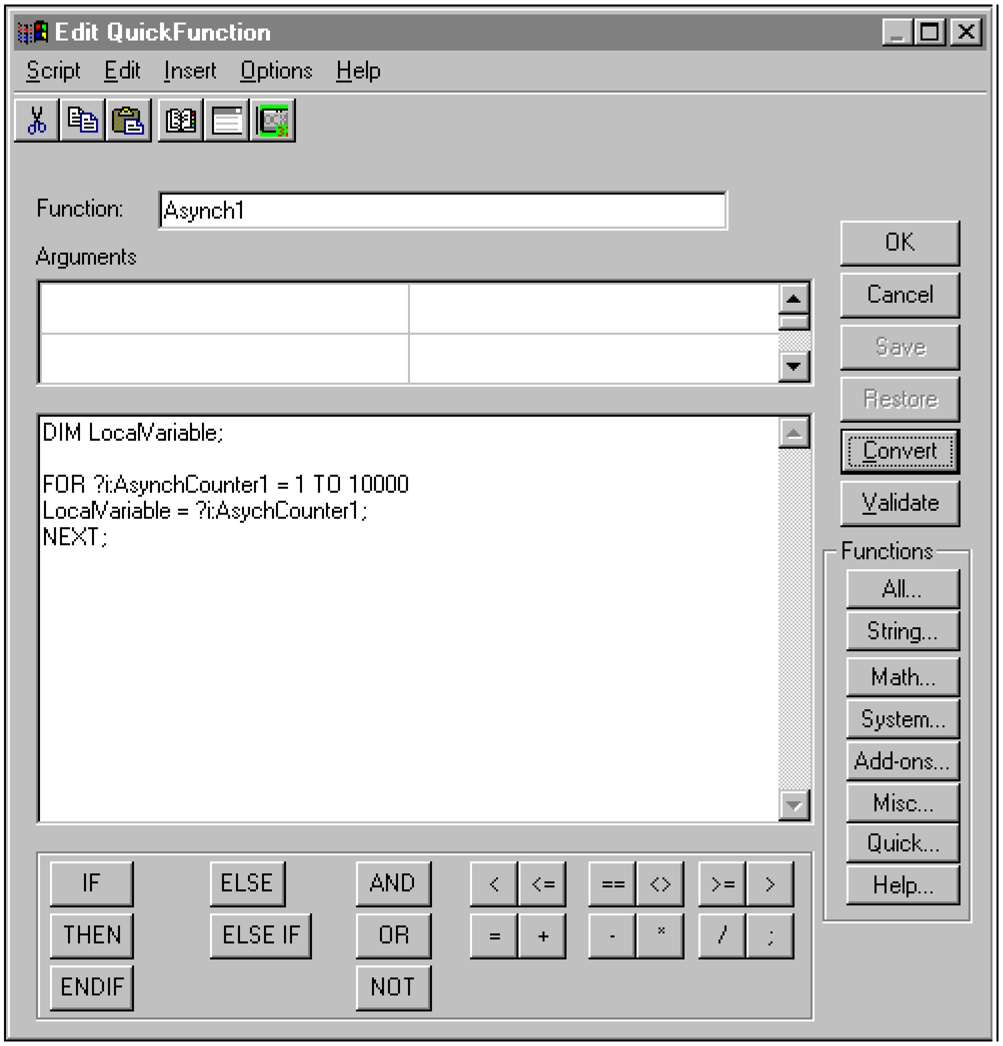
Note When you import a QuickScript into a new application, all the tagnames in the QuickScript are imported with the QuickScript, but they are not added to your Tagname Dictionary. Instead, they are automatically converted to "placeholder" tagnames. You must convert the placeholder tagnames in order to use them and, if they are not currently defined in the new application's Tagname Dictionary, you will be asked to define each of them.

When the tagnames in an imported QuickScript are converted to placeholder tagnames three index characters are added to the beginning of each tagname. For example, when a discrete tagname is imported, the tagname is prefixed with the three characters **?d:**. When a tagname of 30, 31 or 32 characters in length is imported, the three indexing characters will still be added to the beginning of each tagname. However, the addition of these three characters will not truncate the length of your existing tagname. For example, for placeholder tagnames only, a 32 character tagname is increased to 35 characters. These three additional spaces are allotted only for placeholder tagnames. This increase in the tagname length is not supported for standard tagnames.

To convert placeholder tagnames in an imported script

1. On the **Special** menu, point to **Scripts**, and then click the type of QuickScript you imported or in the Application Explorer under **Scripts**, double-click the QuickScript type that you imported. The QuickScript editor appears displaying the first QuickScript on file for the type of script you selected. For example, if you imported a **QuickFunction** script, the **QuickFunctions** script editor appears.

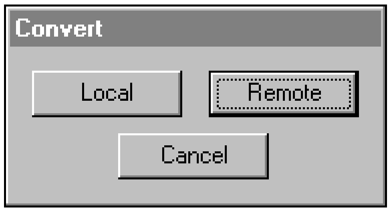
Tip To quickly open the imported QuickScript, double-click **Scripts** in the Application Explorer, and then double-click the QuickScript type.



Function: Asynch1

```
DIM LocalVariable;

FOR ?i:AsynchCounter1 = 1 TO 10000
LocalVariable = ?i:AsychCounter1;
NEXT;
```

2. Click **Convert**. The **Convert** dialog box appears.



3. Click **Local** to convert the tagnames in the QuickScript to local tagnames.
4. After the tagnames are converted, click **OK** in the QuickScript editor.

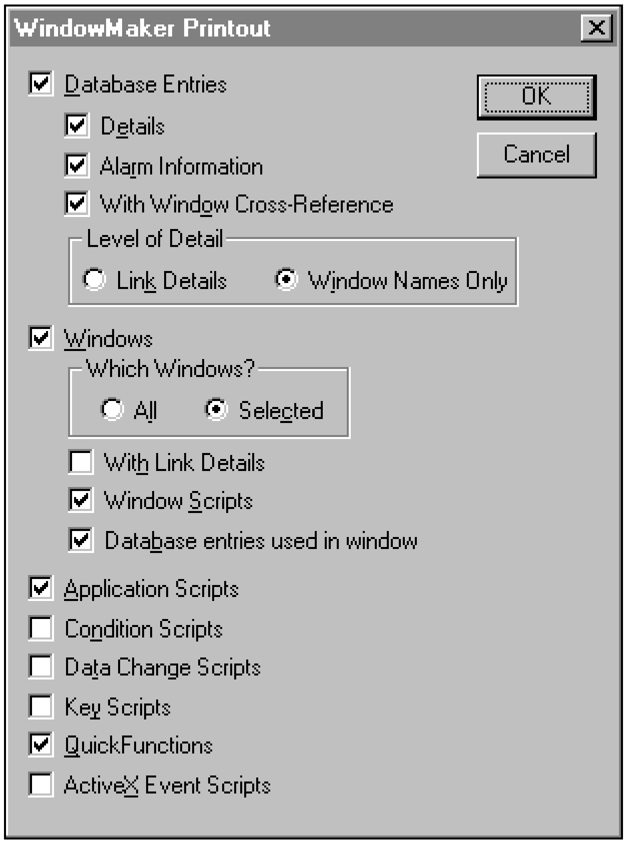
For more information on converting to remote tagname references, see Chapter 6, "Tagname Dictionary."

Printing Scripts

You can print all scripts in each InTouch QuickScript category.

To print a script

1. On the **File** menu, click **Print**. The **WindowMaker Printout** dialog box appears.



WindowMaker Printout

- ✓ Database Entries
  - ✓ Details
  - ✓ Alarm Information
  - ✓ With Window Cross-Reference
  - Level of Detail: Link Details [✓ Window Names Only]
- ✓ Windows
  - Which Windows?: All [✓ Selected]
  - ☐ With Link Details
  - ✓ Window Scripts
  - ✓ Database entries used in window
- ✓ Application Scripts
- ☐ Condition Scripts
- ☐ Data Change Scripts
- ☐ Key Scripts
- ✓ QuickFunctions
- ☐ ActiveX Event Scripts

2. To print window scripts, select **Windows**, and then select **Window Scripts**. In the **Which Windows?** group, select **All** to print the scripts for all windows in the application. To print a specific window's script, select **Selected**. The **Windows to Print** dialog box appears. Select the window(s), whose script you want to print, and then click **OK**.

Tip If you select a window that does not have a script linked to it, the following will be printed on the report: "Window Scripts for *Window Name*: none."

3. To print all scripts for a QuickScript type, select the QuickScript type, and then click **OK**.

Script Functions

InTouch provides you with numerous built-in functions that you can be link to objects or push buttons or use in scripts to perform a multitude of tasks. For example, acknowledging alarms, hiding windows, changing the tagname being trended by a pen, and so on.

These functions are accessible through the **Insert** menu or by clicking the various buttons in the **Functions** section of the Script Editor. Once you select a function in its respective dialog box, the function and its required arguments are automatically pasted into your script at the cursor location. After the function is pasted into your script, you highlight the argument you want to modify and type in the new value.

String Functions

String functions are used on string variables. The following briefly describes each string script function.

Function	Description
DText	Changes a message tagname based on the value of a discrete tagname.
StringASCII	Returns the ASCII value of the first character in a specified message tagname.
StringChar	Returns the character corresponding to a specified ASCII code.
StringFromIntg	Converts an integer value into its string representation in another base.
StringFromReal	Converts a real value into its string representation, either as a floating-point number or in exponential notation.
StringFromTime	Converts a time value (in seconds since Jan-01-1970) into a particular string representation.
StringInString	Returns the position in *Text* where *Search For* first occurs.
StringLeft	Returns the number of characters specified by Chars starting with the leftmost character of text.
StringLen	Returns the length of text to integer result.
StringLower	Converts all the uppercase characters in text to lower case, and places the resulting sting in MessageResult.
StringMid	Extract from text the specific numbers of characters specified by Chars, starting at the position *StartChar.* This function is slightly different from its counterparts **StringLeft()** function and **StringRight()** function in that it allows the user to specify both the start and end of the string which is to be extracted from the message tag.
StringReplace	Replaces or changes specific parts of a provided string.

Function	Description
StringRight	Returns the number of character specified by Chars starting at the rightmost character of text.
StringSpace	Generates a string of spaces either within a message tagname or an expression.
StringTest	Tests the first character of text to determine whether it is of a certain type.
StringToIntg	Converts the numeric value of a message tagname to an integer value to which mathematical calculations can be applied.
StringToReal	Converts the numeric value of a message tagname to a real (floating point) value to which mathematical calculations can be applied..
StringTrim	Removes unwanted spaces from text.
StringUpper	Converts all the lowercase character in text to uppercase.
Text	Causes a message type tagname to display the value of an analog (integer or real) tagname based upon the specified ***Format_Text***..

For more information on the valid syntax for each function and examples of how you use each function, see your online *InTouch Reference Guide.*

Math Functions

Math functions are used on integer or real tagnames. In the following math functions, the **ResultNumericTags** and **InputNumericTags** can be either **Real** or **Integer** and freely intermixed. Keep in mind, however, that returning a non-integer result of a function to an **Integer** tagname will result in the truncation of the result. (The portion to the right of the decimal point will be lost.) The following examples assume that ***ResultNumericTag*** has been defined as either a **Memory Real** or **I/O Real** type tagname.

The following briefly describes each math script function.

Function	Description
Abs	Returns the absolute value (unsigned equivalent) of a specified number.
ArcCos	Given a number between -1 and 1 (inclusive), returns an angle between 0 and 180 degrees whose *cosine* is equal to that number.
ArcSin	Given a number between -1 and 1 (inclusive), returns an angle between -90 and 90 degrees whose *sine* is equal to that number.
ArcTan	Given a number, returns an angle between -90 and 90 degrees whose *tangent* is equal to that number.
Cos	Returns the *cosine* of an angle given in degrees.
Exp	Returns the result of ***e*** raised to a power**.**

Function	Description
Int	Returns the next integer less than or equal to a specified number.
Log	Returns the log of a number.
LogN	Returns the values of the *logarithm* of ***x*** to base ***n***.
Pi	Returns the value of *Pi*.
Round	Rounds a real number to a specified precision.
Sgn	Determines the sign of a value (whether it is positive, zero, or negative).
Sin	Returns the *sine* of an angle given in degrees.
Sqrt	Returns the *square root* of a number.
Tan	Returns the *tangent* of an angle given in degrees.
Trunc	Truncates a real (floating point) number by simply eliminating the portion to the right of the decimal point.

For complete details on the valid syntax for each function and examples of how you each function, see your online *InTouch Reference Guide.*

System Functions

System functions are used to perform actions on your system such as activating another Windows application, copying, deleting or moving files and retrieving information regarding your application. There are two types of system functions; File and Info. The System File functions are used to read and write data from files. They each have two common parameters, ***Filename*** and ***FileOffset***.

The ***Filename*** refers to the name of the file which will be read from or written to. This filename must include the full path. The ***FileOffset*** refers to the position in the file where the read or write operation will begin (measured in bytes from the beginning of the file). The first byte of the file is ***FileOffset*** 0. Upon completion, each function returns the byte position directly following the data that was just read from or written to the file. For example, if the function reads 5 bytes of data starting at byte position 10, the function will return 15.

The ***FileOffset*** tagname can be used as both a parameter to the functions and as the return tagname. This can facilitate continuous operations.

Example:

FileOffset=FileReadMessage(*Filename,FileOffset,Message_Tagname*,*0*);

In the previous example, a line of text is read from ***Filename***. The starting location is specified by the original value of ***FileOffset*** (0 for instance, being the beginning of the file). The position where the next read will begin is then returned to ***FileOffset*** in preparation for the next call to **FileReadMessage()**. Every time the function is called, ***FileOffset*** gets larger and larger as the **FileReadMessage()** reads through the file.

The following briefly describes each system script function.

Function	Description
ActivateApp	Activates another currently running Windows application.
FileCopy	Copies a *SourceFile* to a *DestFile*, similar to the DOS Copy command or the Copy function in Windows File.
FileDelete	Deletes unnecessary or unwanted files.
FileMove	This is similar to FileCopy() except that it moves the file from one location to another instead of making a copy.
FileReadFields	Reads a Comma Separated Variable (CSV) record from a specified file.
FileReadMessage	Reads a specified number of bytes (or a whole line) from a specified file.
FileWriteFields	Writes a Comma Separated Variable (CSV) record to a specified file.
FileWriteMessage	Writes a specified number of bytes (or a whole line) to a specified file.
InfoAppActive	Tests whether an application is active.
InfoAppTitle	Returns the Application Title or Windows Task list name of a specified program which is currently running.
InfoDisk	Returns information about a specific local (or network) disk drive.
InfoFile	Returns information about a specific file or subdirectory.
InfoInTouchAppDir	Returns the current InTouch application directory.
InfoResources	Returns various system resource values as follows: **Case 1 and Case 2:** GDI and USER are hard-coded to return 50% on Windows NT. **Case 3:** On Windows NT returns "free bytes of paging file." **Case 4:** On Windows NT returns the result of search of all the top level windows. It only counts the windows that are visible and do not have any owners. This is not the actual "number of tasks currently running" in the system. Its closest approximation would be the count of items on the Applications tab when you run the Task Manager in Windows NT.

Function	Description
IsAnyAsyncFunctionBusy	Used to find out if any asynchronous QuickFunctions are running. This function can be used to make the QuickScript that calls an asynchronous QuickFunction wait for all other asynchronous QuickFunctions to complete processing. This allows the QuickScript to re-synchronize itself.
StartApp	Automatically starts another Windows application.

For more information on the valid syntax for each function and examples of how you use each function, see your online *InTouch Reference Guide.*

Misc Functions

Miscellaneous functions are used to perform miscellaneous actions such as, hiding a window, monitoring and controlling historical trends, printing windows, sending keys to other applications, and so on.

The following briefly describes each miscellaneous script function.

For the list of the new alarm functions, see Chapter 9, "Alarms/Events."

For the details of the new alarm functions, see *InTouch Reference Guide.*

Tip The function names that begin with "alm" are used for distributed alarm systems only. Function names beginning with "wc" are used for Windows Controls objects (list boxes, text boxes, combo boxes, and so on.). Function names beginning with "HT" are used for historical trend objects only.

Function	Description
Ack	Acknowledges any unacknowledged alarm. This function can be applied to a tagname, Alarm Group or Group Variable.
almAckAll	Acknowledge all alarms in current query including those not currently displayed in the alarm display object.
almAckDisplay	Acknowledge only those alarms currently visible in the alarm display object.
almAckRecent	Acknowledge the most recent alarm that has occurred.
almAckSelect	Acknowledge only those alarms selected in the alarm display object.
almDefQuery	Performs a query to update an alarm display object using the default properties.
almMoveWindow	Scrolls the alarm display object window.
almQuery	Performs a query to update an alarm display object.

Function	Description
almSelectAll	Toggles the selection of all the alarms in an alarm display object.
almSelectItem	Toggles the selection of the item that is highlighted in an alarm display object.
almShowStats	Displays the alarm display object statistics screen.
ChangePassword	Displays the Change Password dialog box allowing the logged on operator to change his/her password.
DialogStringEntry	Displays an alphanumeric keyboard on the screen, allowing the operator to change the current string value of a message tagname in the Tagname Dictionary.
DialogValueEntry	Displays the numeric keypad on the screen, allowing the operator to change the current value of a discrete, integer or real tagname in the Tagname Dictionary.
GetNodeName	Returns the NetDDE node name to a string variable.
GetPropertyD	Retrieves the specified property's discrete value during runtime.
GetPropertyI	Retrieves the specified property's integer value during runtime.
GetPropertyM	Retrieves the specified property's message value during runtime.
Hide	Hides various windows from within a script. A Hide function must precede the name of each window to be hidden.
HideSelf	Hides the currently active window.
HTGetLastError	Determines if there was an error during the last retrieval of a specified pen.
HTGetPenName	Returns the tagname of the tagname currently used for the specified pen # of the specified trend.
HTGetTimeAtScooter	Returns the time in seconds since 00:00:00 hours GMT, January 1, 1970 for the sample at the scooter location specified by *ScootNum* and *ScootLoc*.
HTGetTimeStringAtScooter	Returns the string containing the time/date for the sample at the scooter location specified by *ScootNum* and *ScootLoc*.
HTGetValue	Returns a value of the requested type for the entire trend's specified pen.
HTGetValueAtScooter	Returns a value of the requested type for the sample at the specified scooter position, trend and pen #.

Function	Description
HTGetValueAtZone	Returns a value of the requested type for the data contained between the right and left scooter positions for a trend's specified pen.
HTScrollLeft	Sets the start time of the trend to a value older than the current start time by a percentage of the trend's width. The effect is to scroll the date/time of the chart to the left by a given percent.
HTScrollRight	Sets the start time of the trend to a value newer than the current start time by a percentage of the trend's width. The effect is to scroll the date/time of chart to the right by a given percent.
HTSelectTag	Displays the Select Tag dialog box and the operator can select a different tagname for specified pen. (The dialog box only lists the tagnames that have been defined for historical logging (Log Data option selected) in the Tagname Dictionary.)
HTSetPenName	Assigns a different tagname to a trend's pen.
HTUpdateToCurrentTime	Causes the data to be retrieved and displayed with an end time equal to the current time. The start time will be equal to EndTime minus the Width of the chart.
HTZoomIn	Calculates a new chart width and start time. If the trend's .**ScooterPosLeft** is 0.0 and the .**ScooterPosRight** is 1.0, then the new chart width equals the old chart width divided by two.
HTZoomOut	Calculates a new chart width and start time. The new chart width is the old chart width multiplied by two.
IOReinitialize	Closes all existing I/O conversations and restarts the entire process of setting up I/O conversations. All I/O points are affected when this function executes.
IOGetApplication	Returns the application name defined for a specific Access Name to the specified tagname.
IOGetNode	Returns the node information (address) defined for a specific Access Name to the specified tagname.
IOGetTopic	Returns the topic name defined for a specific Access Name to the specified tagname.

Function	Description
IOReinitialize	Closes all existing I/O conversations and restarts the entire process of setting up I/O conversations. All I/O points are affected wMisc
IOSetAccessName	Modifies the *node*, *application* or *topic* name portions of an Access Name during runtime which allows implementing of hot-backup strategies for InTouch.
IOSetItem	Changes the access name and/or item in a I/O type tagname ***.Reference*** field.
LogMessage	Writes a user-defined message to the Logger.
PlaySound	Plays a wave form sound specified by a .wav filename or an entry in the [sounds] section of the WIN.INI file through the Windows sound device (if installed).
PrintHT	Can be used in a button for printing the Historical Trend chart associated with the specified Hist Trend type tagname. The Historical Trend must be visible on the screen when using this function.
PrintScreen	Prints the specified screen.
PrintWindow	Prints the specified window.
ReloadWindowViewer	Allows the user control over reloading WindowViewer.
RestartWindowViewer	Allows the user control over shutting down and restarting WindowViewer.
SendKeys	Sends keys to an application.
SetPropertyD	Specifies the property's discrete value that is to be written during runtime.
SetPropertyI	Specifies the property's integer value that is to be written during runtime.
SetPropertyM	Specifies the property's message value that is to be written during runtime.
Show	Displays a specified window. (Window name must be enclosed in quotation marks.)
ShowAt	Specifies the horizontal and vertical pixel location of a window when it is shown. When the window opens, it will be centered on the horizontal and vertical coordinates.

Function	Description
ShowHome	Displays the "home" window(s). Home windows are those you selected to open automatically when WindowViewer is started. (The home windows are selected in the WindowViewer Properties - Home Windows property sheet.) For more information on home windows, see Chapter 2 - Using WindowMaker.
ShowTopLeftAt	Specifies the horizontal and vertical pixel location of the top left corner of a window when it is shown. When the window opens, its top left corner will be positioned where the horizontal and vertical coordinates meet.
wcAddItem	Adds the supplied string to the list box or combo box.
WcClear	Removes all items from the list box or combo box.
wcDeleteItem	Deletes the item associated with the item index argument in both list or combo boxes.
wcDeleteSelection	Deletes the currently selected item from the list. Applies to list boxes and combo boxes
wcErrorMessage	Given an error number, **wcErrorMessage**(), returns a string message describing the error. Applies to list boxes, text boxes, combo boxes, radio buttons and check boxes.
wcFindItem	Determines the corresponding index of the first item in the list box or combo box that matches the supplied string.
wcGetItem	Returns the value property of an item string associated with a corresponding index in a list box or combo box.
wcGetItemData	Retrieves the integer value associated with a list item in a list box or combo boxes.
wcInsertItem	Inserts a string into a list box or combo box.
wcLoadlist	Replaces the contents of the list box or combo box with new items.
wcLoadText	Replaces the contents of the text box with a new string.
wcSavelist	Replaces the contents of a filename with the items in a list object.
wcSaveText	Saves the text contained in a text box to a filename.
wcSetItemData	Assigns an integer value to an item in a list box.

For complete details on the valid syntax for each function and examples of how you use each function, see your online *InTouch Reference Guide.*

WW DDE Functions

You should not use the WW DDE functions as a replacement for normal InTouch DDE communications. Whenever possible, you should create an DDE type tagname to get data from or send data to an external application. The WW DDE functions are intended to support applications that cannot communicate using the DDE Advises supported by InTouch. For example, some applications support only DDE Executes or Pokes.

The **WWExecute()**, **WWPoke()** and **WWRequest()** functions use the same Windows functions as MS Visual Basic (DDEML). A single function actually does several things. For example, a **WWPoke()** will perform a DDE Initiate, a DDE Poke and a DDE Terminate all in one function. This makes WW DDE functions less error prone, but also less efficient in processing many DDE messages. As general guidelines for the use of these functions, you should never:

Loop these functions (call them over and over).

Call several of the DDE functions in a row and in the same script.

Use them to call a lengthy task in another DDE application.

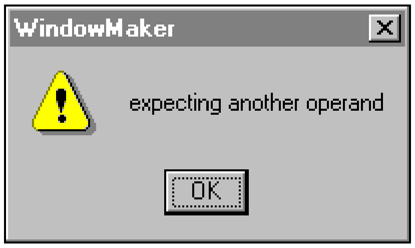
If the DDE command executes a lengthy task in another application it can use up all of the available processor time. However, even if communication difficulties occur, no loss of data will occur. Even if the I/O Server cannot send messages to InTouch, it will continue to try.

The following briefly describes each script function. For details on the valid syntax for each function and examples of how to use each function, see your online *InTouch Reference Guide.*

Function	Description
WWControl	Allows you to Restore, Minimize, Maximize*,* or Close an application from InTouch.
WWExecute	Sends a command (using a DDE Execute) to a specified *Application* and *Topic*.
WWPoke	Pokes a value (using an DDE Poke) to a specified *Application*, *Topic*, and *Item*.
WWRequest	Makes a one-time request for a value (using an DDE Request) from a particular *Application*, *Topic*, and *Item*.

Script Editor Error Messages

If the script editor detects any errors in the script when it is validated a corresponding message box will be displayed. For example:



expecting another operand

Tip In most cases, when an error is encountered, InTouch will place the cursor at the position in the script where the error occurred. However, in some cases, such as a missing ENDIF, the cursor will be at the end of the script. All errors must be corrected before the system will accept the script.

Error Message	Definition
Can only compare alarm groups for equality	Cannot compare alarm groups for <, >, <=, >=.
Cannot add, subtract, multiply or divide with strings	These operations are not supported for strings.
Cannot mix another type with alarm group	Trying to compare an Alarm Group with another type (e.g. integer) or trying to use something other than an Alarm Group somewhere where an Alarm Group is expected.
Cannot mix another type with string	Trying to compare a string with another type (for example, integer) or trying to use something other than a string somewhere where a string is expected.
Cannot negate alarm groups	Minus sign (-) has been used.
Cannot negate Access Name	A "-" or "~" is not allowed prior to a DDE Access Name.
Cannot negate strings	Minus sign (-) has been used.
Cannot negate TagID	Minus sign (-) has been used.
Cannot negate window	A "-" or "~" is not allowed prior to a window name.
Cannot use Access Name in this manner	A DDE Access Name cannot be used in this context.
Cannot use HistTrendTag in this manner	Trying to compare a string with another type (for example, integer) or trying to use something other than a string somewhere where a string is expected.

Error Message	Definition
Cannot use TagID in this manner	A TagID type variable cannot be used in this context.
Cannot use window name in this manner	A window name cannot be used in this context.
E Format must be between -38 and +38	The maximum "e" format number is between e-38 and e+38.
E format must have digit following E	Valid "e" format is n.nnen, 1.e is not legal.
Expecting ")" after function arguments	A matching right parenthesis is required to match the left parenthesis following the function name.
Expecting "(" after function name	A left parenthesis is required after this function name.
Expecting a number after 0x	Hexadecimal (base 16) numbers can be entered in InTouch. To start a hexadecimal number, start with 0x and follow with digits.
Expecting an expression after IF	Missing discrete expression.
Expecting analog argument for function	This argument for this function requires an analog value.
Expecting another argument for	The function requires more arguments than are present.
Expecting another operand	If "a + " is entered, InTouch will display this error message.
Expecting assignment	In an action script, a tagname was entered and the next logical operation would be an assignment.
Expecting comma and other argument(s) to function	More arguments are required for this function.
Expecting DLL Name	A DLL name must be used in this context.
Expecting ENDIF	Must have an ENDIF for each IF.
Expecting ENDIF or ELSE	Every IF/THEN must be matched with an ENDIF or ELSE.
Expecting expression after assignment (=)	In an action script, a tagname and assignment were entered and no value was given for the assignment. This can also happen if => is entered instead of >=.
Expecting Function Name	A Function Name must be used in this context.
Expecting right parentheses	A matching ")" was not found.
Expecting name	Expecting a tagname for this argument.
Expecting name in statement	Missing name in statement.

Error Message	Definition
Expecting semicolon	Semicolon is missing at end of line.
Expecting string	The given argument must be a string expression (the name of a string tagname or a constant string (text surrounded by double quotes ("))).
Expecting THEN	THEN missing after IF statement.
Expecting window name - must be string expression	The given argument must be a string expression (the name of a string tagname or a constant string (text surrounded by double quotes ("))).
Extra expressions	For example, the expression "a b" is not legal, "a + b" is OK.
Function only legal in action scripts or logic	Some functions are only legal in scripts, not in expressions.
IF expression must be discrete (use == instead of =)	This error is received because of using the assignment (=) instead of comparison (==). For example, "IF a = b THEN ..." should be "IF a == b THEN ...". May also be received if "IF x THEN..." is used and x is not a discrete tagname.
Invalid or missing operand	The operand required for an operator is either invalid or missing.
Invalid placeholder name - must have chars follow ?x:	Describing characters must follow ?x: in placeholder name.
Invalid placeholder name - second char must be d, i, a, r, m, v, g, h, t	Describing 2nd digit character is invalid for placeholder name.
Invalid placeholder name - third char must be ":"	Describing 3rd digit character is invalid for placeholder name.
Logical AND/OR must use discrete	Using AND/OR operators must be done with discrete expressions. Thus "x AND y" is OK, if x and y are discrete tagnames; if they are of any other type, this error message will be received.
Logical NOT must use discrete	Using the NOT operator must be done with discrete expressions. Thus, "NOT x" is OK if x is a discrete tagname, but if x is of any type other than discrete, this error message appears.
Maximum string 131 characters	The string is longer than the max allowed.
Must assign the return value of function	Certain functions require function evaluation of the return value.
Must have a digit after decimal point	The syntax "1." is not legal.

Error Message	Definition
Must have hist trend variable for this argument	A HistTrend type variable must be used in this context.
Must have writeable analog variable or name.field for this argument .field	The argument must be an integer or real variable or integer or real of a variable.
Must have writeable discrete variable for this argument	The argument for this function must be a tagname whose type is discrete and for which read-only is NOT checked.
Must have writeable integer variable for this argument	The argument for this function must be a tagname whose type is integer and for which read-only is NOT checked.
Must have writeable message variable for this argument	The argument for this function must be a tagname whose type is message and for which read-only is NOT checked.
Must have writeable real variable for this argument	The argument for this function must be a tagname whose type is real and for which read-only is NOT checked.
Must have writeable variable for this argument	The argument for this function must be a tagname for which read-only is NOT checked.
Must not assign the return value of function	Certain functions do not return a value and a return value cannot be evaluated for them.
Name too long	Tagnames must be <= 32.
No closing comment	Opening comment delimiters ({) must be matched with closing comment delimiters (}).
No closing quote	Missing closing quotation mark (").
Not enough room in display buffer	Not enough memory for this operation. Clear up memory and this operation will complete.
Not enough room in expression buffer	Not enough memory for this operation at this time. Clear up memory and this operation could complete.
Only 8 digits allowed after 0x	When entering a hexadecimal number, only 8 digits are allowed.
Number too large	Absolute value of numbers must be < 2e38.
Number too small	Absolute value of numbers must be > 2e-38.
Too many arguments	Supplied more arguments to this function than are necessary.
Trying to assign to a read-only name	It is not legal to assign a value to a tagname for which read-only is checked.

Error Message	Definition
Undefined field name	The .field name is not defined, most likely due to a spelling error.
Unrecognized character	The highlighted character is not a legal character for expressions or action scripts.
Using a reserved field name (e.g., SP) for normal tagname	Can't use a field name for tagname.

Error Messages for Windows Controls and Distributed Alarms

The Window Controls and distributed alarm QuickScript functions return a value based on the result of processing the QuickScript function. The return value, used for error diagnostics, can be assigned to an Integer tagname. For example:

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

Where:

ErrorNumber is an **Integer** tagname type that will hold the returned error value. The return value of the function can be passed to the **wcErrorMessage()**. The **wcErrorMessage()** will return a string description of the error. For example:

```
ErrorMsg = wcErrorMessge(ErrorNumber);
```

Where:

ErrorMsg is a **Message** type tagname that holds the text description of the returned error. The following table identifies the numeric value and its description:

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

* *-x* represents any other number.

C H A P T E R 9

Alarms/Events

InTouch provides a notification system to inform operators of process and system conditions. This system supports the displaying, logging, and printing of process alarms and system events. The alarm displays are described in Chapter 10, "Alarm/Event Clients." The alarm logger and alarm printer are described in Chapter 11, "Alarm Utilities." Alarms represent warnings of process conditions, while events represent normal system status messages.

InTouch supports a Distributed Alarm System that allows the display of alarms and events generated by the local InTouch application and by alarm systems of other networked InTouch applications. Alarms can be acknowledged on the local InTouch or from a remote node on the network.

This chapter describes the alarm system, the various types of alarm conditions, and the grouping hierarchies. Specific sections cover how you add, modify and delete Alarm Groups, assign tagnames to Groups, define the alarm conditions for a tagname, and how you configure the alarm system.

Contents

- Alarm Visibility Controls
- Configuring the Alarm System
- Attaching Comments to an Alarm Ack Function
- Acknowledging Local Alarms
- Migrating from an Older InTouch Standard Alarm System to the Distributed Alarm System
- Migrating from Older Master/Slave Alarms to the Distributed Alarm System
- Hot Backup and Synchronization
- Hot Backup Usage Example
- View and Stored Procedure Column Definitions
- Alarm History Database Views
- Event History Database Views
- Alarm-Event History Database View
- AlarmCounter Database Stored Procedure
- Viewing the Definition of a Stored Procedure in Enterprise Manager
- EventCounter Database Stored Procedure
- AlarmSuite Alarm Log Database View

Introduction

The InTouch Distributed Alarm System is a set of software components that are separate from WindowViewer. These components allow InTouch and other FactorySuite programs to communicate alarm information in a multiple node system. A FactorySuite program acting as an *Alarm Provider* performs the actual detection of an alarm condition. Alarm Providers pass the notification to the Distributed Alarm System via API calls. Conversely, a FactorySuite program acting as an *Alarm Consumer* retrieves notifications and statuses from the Distributed Alarm System via API calls, and then performs the display of alarm information.

In previous versions of InTouch, the focus of the Distributed Alarm System was on communication and a set of Alarm Consumers that supported the basic but flexible functions of display and storage: alarm display in an InTouch View window, alarm history storage, alarm logging and alarm printing. The enhanced Distributed Alarm System improves performance and also supports the following features:

- Alarm Disablement and Inhibition
- Alarm Display Suppression
- Alarm SQL Database Storage

The Distributed Alarm System continues to provide you with the services to display, log, print and acknowledge process alarms and system events. It handles alarms and events generated by the local InTouch application and by InTouch applications running on other nodes. The applications on the different nodes do not have to be identified. In fact, alarms can be handled for Alarm Providers other than InTouch if they are configured to use the Distributed Alarm System.

The Distributed Alarm System features also include:

- The ability to display and acknowledge alarms from any InTouch node on a network.
- ActiveX Alarm displays that have built-in scroll bars, sizable display columns, multiple alarm selections, an update status bar, dynamic display types, and display colors based on alarm priority. For more information on the ActiveX Alarm displays, see Chapter 10, "Alarm/Event Clients."
- QuickScript functions that provide dynamic control over the alarm display and alarm acknowledgment. For more information on QuickScript functions, see Chapter 9, "Alarms/Events."
- A grouping mechanism that allows multiple Alarm Groups across different applications to be called via a single name.
- The capability of adding comments to alarms when acknowledged.

Support for Non-InTouch Alarm Providers

In the past, InTouch has been the main Alarm Provider (generator of alarms) in the FactorySuite environment. Beginning with InTouch Version 7.11, non-InTouch Alarm Providers are supported by the Distributed Alarm System, for example, some of the other FactorySuite components such as InBatch and InControl, I/O Servers and alarm print servers as well as third-party components.

Note The Wonderware *Alarm Toolkit* must be used to create third-party software that generates and tracks alarms. For more information on the *Alarm Toolkit*, contact your local Wonderware distributor.

Distributing Your Application

Your application can be distributed either manually or by using the NAD system. When the application is distributed, the alarm group list file is automatically distributed, as it is part of the application.

For more information on NAD, see Chapter 5, "Building a Distributed Application."

General Background on Alarms

There are a number of terms and concepts that apply to alarms generally, regardless of the particular system in which they are used, or how they are implemented.

- **Alarms**: In general an **Alarm** is a specific type of the more general concept of a **condition** - in particular, an alarm is an ***abnormal condition***. Usually, the intention of an alarm is to signal that something has gone wrong, or that a particular stage of processing has been reached. For example, an alarm might indicate that a boiler has exceeded a safe temperature limit - or it might simply signal the end of a work shift.
- **Priority**: A level of **severity** or **priority** is associated with an alarm to indicate how "bad" the situation is or, how "important" the condition is. In the case of a boiler temperature limit, the severity might be "very bad" or "very important," requiring immediate attention to save life and property. By contrast, in the case of the end of a shift, the severity would be "not so bad" or even minimal. The severity of an alarm usually depends upon the circumstances - the factory application, the nature of the equipment, safety, availability of backup systems, potential costs of damage or downtime and so on.

Note InTouch uses a **priority scale** where 1 is the most important and 999 is the least important.

For more information on priority, see "Alarm Priorities."

- **Sub-states**: An alarm condition may also have **sub-conditions**, in which case it is called a **multi-state alarm**. For example, an analog alarm typically has several limits, with "High" and "Low" bounding the normal operating range, and "HiHi" and "LoLo" marking the extreme departures from normal. The boiler temperature level mentioned above could be in the *alarmed* condition for any one of these *sub-states*. It could also transition between any two sub-states while continuing to remain in the overall *alarmed* condition.
- **Events**: An **event** is a ***detectable occurrence***, which may or may not be associated with an alarm. A *transition* into or out of an alarmed state certainly constitutes one kind of event. An event might also mark an operator action, a change to the system configuration or some kind of system error. It is important to make a distinction between a **condition** and an **event**. A **condition** may persist for minutes, hours, days or weeks. An **event** is momentary; it takes place and is immediately over. An *alarm* is a condition; an *alarm notification* is an event.
- **Acknowledgment**: One major purpose of an alarm is to alert an observer - a human being or another part of the system - about the condition. The person or system should then ***acknowledge*** the alarm, indicating that is has been seen. This is separate from the issue of taking corrective action, which might not happen right away. It is also separate from the issue of whether the alarm condition returns to normal - which it might do on its own, even without any external intervention. Acknowledgment merely indicates that someone has noticed the occurrence of the alarm. A high or medium priority alarm usually requires acknowledgment, while a very low priority alarm might not. Although the condition that generated the alarm may go away (for example, a temperature rises too high and then becomes lower again), the alarm itself is not considered "handled" until it is acknowledged.

  For more information on acknowledgment, see "Alarm Acknowledgment Models."

- **Groups**: Alarms may be organized into **groups** to facilitate tracking and management. These groups might represent different areas of a factory, pieces of equipment, operator responsibility or a specific functionality. Groups may also be organized into a **hierarchy** of parent groups and sub-groups.

  For more information on groups, see "Alarm Groups."
- **Areas**: Generally, factories are organized into **areas**, representing physical location, operator responsibility, phases of a process, types of equipment and so on. Like groups, areas can be hierarchical - divided into main areas and sub-areas. However, **areas** should be mentioned separately from **groups**, because some factories make a distinction between the two. The Distributed Alarm System provides no explicit support for Areas. However, alarm Groups may be used to divide alarms into collections that correspond to Areas.

Alarms and Events

InTouch has two types of notifications to inform operators of process activity: Alarms and Events. Alarms represent warnings of process conditions that could cause problems, and require an operator response. A typical alarm is triggered when a process value exceeds a user-defined limit, such as an analog value exceeding a hi-limit threshold. This triggers an *unacknowledged* alarm state that can be used to notify the operator of a problem. Once the operator acknowledges the alarm, the system returns to an *acknowledged* state. InTouch can be configured to require an alarm to be acknowledged even if the condition causing the alarm has passed. This ensures that an operator is aware of events that caused a temporary alarm state but have returned to normal.

Events represent normal system status messages, and do not require an operator response. A typical event is triggered when a certain system condition takes place, such as an operator logging into InTouch. If configured to do so, InTouch can log an event to the alarm database and/or print it out to a printer.

You can configure any tagname to do event monitoring while you are defining it in the Tagname Dictionary. When you define a tagname to do event monitoring, an event message is logged to the alarm system each time the tagname value changes. The event message logs how the value changed, whether the operator, I/O, scripts or the system initiated the change.

For more information on configuring a tagname to do event monitoring, see Chapter 6, "Tagname Dictionary."

Summary Alarms versus Historical Alarms

InTouch uses the terms ***summary alarms*** and ***historical alarms*** to refer to alarms that are "currently active" and alarms that are "finished," respectively. The idea is that an operator might want to see a "summary" on screen of all current alarms that are awaiting acknowledgment, whereas all other alarm information is of historical interest and of less urgency. Inside the Distributed Alarm System, these two kinds of **alarm records** are kept in different storage caches.

Terminal Services Alarm Support

By using the Distributed Alarm System with Terminal Services for InTouch, alarm clients running on different terminal sessions can select what alarm data to display and how to present it.

Alarm Providers identify themselves by a name that uniquely identifies their application, and the instance of their application. This information is made available to the Distributed Alarm System when the Alarm Provider or the Alarm Consumer registers with the Distributed Alarm System.

The node on which an Alarm Provider is running is identified by a name that uniquely identifies the computer node in the system. This information is made available to the Distributed Alarm System when an instance of it starts up on the computer node.

When an alarm event is logged, the node and complete Alarm Provider name identify the source of the alarm.

Note Alarm Providers are not supported on Terminal sessions. They are only supported on the Terminal Console.

For more information on Terminal Services, refer to your online *Terminal Services for InTouch Deployment Guide.*

Support for Other Alarm Sources

The Distributed Alarm System supports alarms coming from a variety of sources (also called Alarm Providers), not just from InTouch. In particular, it supports alarms from RTUs when coupled with Wonderware I/O Servers. The Distributed Alarm System is able to insert "batch" RTU alarm records in sequence when they are received in a batch mode.

The date/time stamp for these alarm records is provided by the Alarm Provider, and not automatically generated by the Distributed Alarm System.

Client requests for these alarms, such as acknowledgments, are accomplished by the Distributed Alarm System sending a message to the Alarm Provider. The Alarm Provider then processes the message internally, and then responds with an updated alarm record indicating the new state or configuration, if any.

The Alarm Provider places all information specific to the Alarm Provider in the alarm record. The Alarm Provider (not the Distributed Alarm System) places alarm names, alarm classes, limits, enablements, status, and any other configurations in the alarm record.

For more information on Terminal Services, refer to your online *Terminal Services for InTouch Deployment Guide.*

Alarm Types

InTouch classifies alarms into several general categories based on their characteristics. These categories are known as *Class* and *Type*. The Distributed Alarm System categorizes all alarms into five general *Conditions*: Discrete, Deviation, Rate-of-Change, Value, and SPC. The table below summarizes the alarm condition for both systems:

Alarm Condition	Distributed Class	Distributed Type
Discrete	DSC	DSC
Deviation - Major	DEV	MAJDEV
Deviation - Minor	DEV	MINDEV
Rate-of-Change	ROC	ROC
SPC	SPC	SPC
Value - LoLo	VALUE	LOLO
Value - Low	VALUE	LO
Value - High	VALUE	HI
Value - HiHi	VALUE	HIHI

You can associate each alarm with an InTouch tagname. Depending upon a tagname's type, you can define one or more of the alarm classes or types for it. You define your alarm conditions in the Tagname Dictionary.

For information on defining alarm conditions, see Chapter 6, "Tagname Dictionary."

You can also configure alarm printing and/or the distributed alarm display object or Alarm Viewer ActiveX Control to show the alarm *Class* field and/or the alarm *Type* field.

Event Types

InTouch also classifies events into general categories based on their characteristics. These categories are known as *Event Types*. The table below summarizes the classification of events:

Event	Condition
SYS	A system event occurred
USER	$Operator changed
DDE	The tagname value was poked from a DDE client
LGC	A QuickScript modified the tagname value
OPR	The operator modified the tagname value using the Value Input

The first two events listed are configured automatically when event logging is enabled. The remaining three must be defined by users for each tagname in the Tagname Dictionary.

For more information, on events, see "Alarms and Events."

Alarm Priorities

Each alarm configured in InTouch has a priority value associated with it. This value represents the severity of the alarm and can range from 1 to 999 with 1 being the most severe. By creating alarm ranges using these priorities and assigning alarms to each, you can easily filter out critical alarms from non-critical ones. You can also create animation links, acknowledgment scripts, and filtered viewing and printing all based on the priority range.

For example, if a process plant has determined that they need four levels of severity, they could establish ranges as shown below:

Alarm Severity	Priority Range
Critical	1 - 249
Major	250 - 499
Minor	500 - 749
Informational	750 - 999

As the plant engineers create InTouch tagnames and alarm conditions, each alarm will be assigned to one of these severity levels by choosing a priority number within that range. With these ranges configured, the plant operators may now easily display and print only certain severity levels.

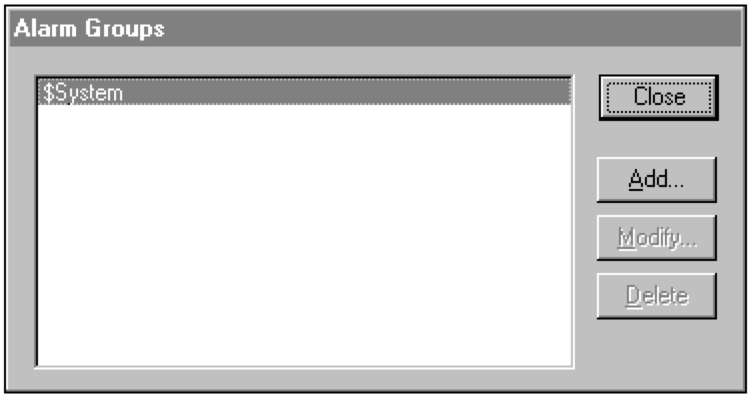
Note It is possible and even desirable to use only four priorities: One for critical, two for major, three for minor, and four for informational (ie. 1, 2, 3, 4).

Alarm Groups

Each InTouch alarm is assigned to a logical Alarm Group. These groups are user-definable and can be arranged into a hierarchy up to 32 levels deep. The groups provide a way of categorizing alarms based on an organization, plant layout, or any other metric. Alarm Groups are useful for filtering alarm displays, Alarm Printers, and acknowledgment scripts.

Every tagname is associated with an Alarm Group. If you do not associate an Alarm Group name to a tagname, then by default InTouch automatically associates it with the root group, **$System**. Any Alarm Group may have both tagnames and other Alarm Group names associated with it. Alarm Groups are organized into a hierarchical tree structure with the root group, **$System**, at the top of the tree. All defined Alarm Groups automatically become descendants of the root group.

This tree may have up to 32 levels. Each Alarm Group may have a maximum of 32 subgroups. Each subgroup may have a maximum of 32 subgroups, and so on, until the maximum of 32 levels is reached.



This illustration displays only Alarm Groups, not the tagnames within each group. This tree concept is analogous to the Windows folder directory structure, where a directory may contain other sub-directories (analogous to groups) and file names (analogous to tagnames).

The Distributed Alarm System also uses these groups as the basis for its Alarm Group Lists.

For more information, see "Distributed Alarm Group Lists."

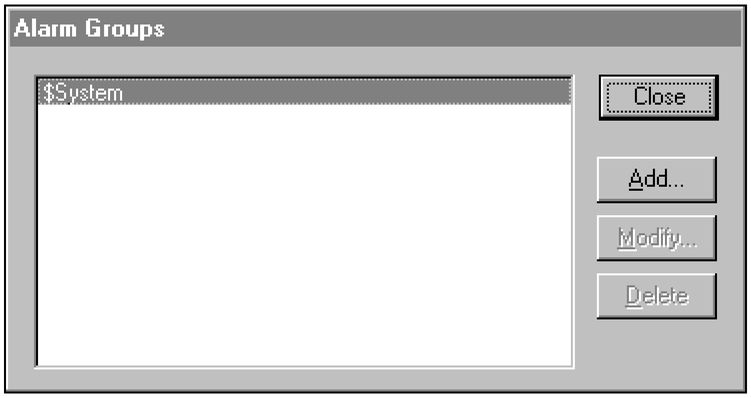
Note While Alarm Groups do not count as tagnames with InTouch licensing, they do count as tagnames in the database. Therefore, the total number of Alarm Groups plus actual tagnames cannot exceed 61,402.

WARNING! Large numbers of alarm groups (100's/1000's) will cause performance problems.

To create an Alarm Group

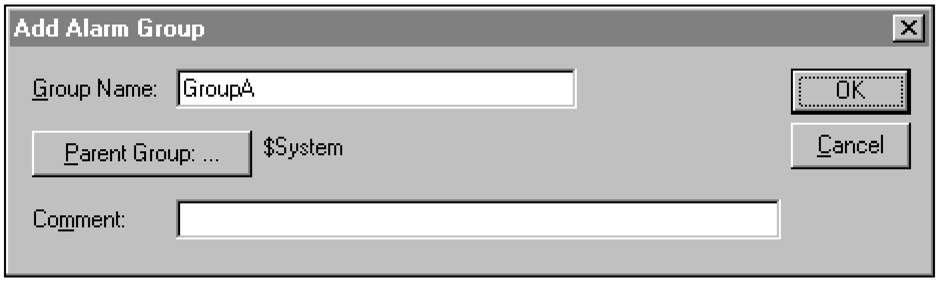
1. On the **Special** menu, select **Alarm Groups**. The **Alarm Group** dialog box appears.

Tip You can also create Alarm Groups and associate tagnames with them while you are defining your tagnames in the Tagname Dictionary.



2. Click **Add**. The **Add Alarm Group** dialog box appears.

Tip The **Modify** and **Delete** buttons are not available until an Alarm Group is defined. The **$System** Alarm Group cannot be modified or deleted.



Group Name: GroupA

Parent Group: $System

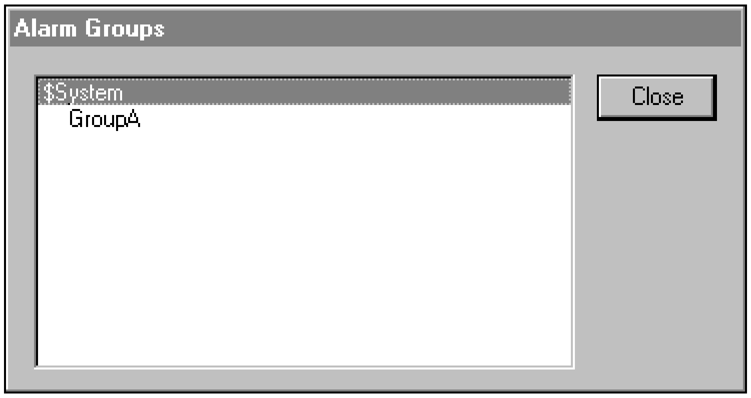
Comment:

Tip If you right-click a text box in any alarm configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

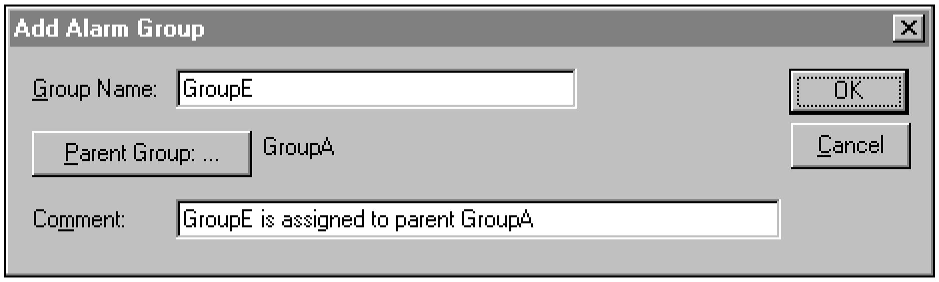
3. In the **Group Name** box, type the name for the new Alarm Group. Since this is the first Alarm Group you have created, it is automatically assigned to the **$System** Parent Group.

Tip After you have created an Alarm Group, it can be used as a Parent Group.

4. Click **Parent Group** to assign your Alarm Groups, to a different Parent Group. The **Alarm Group Selection** dialog box appears.



5. In the **Select Alarm Group** list, double-click the name of the Alarm Group that you want to use as the Parent Group (or select it, then click **OK**) for the new Alarm Group. The **Add Alarm Group** dialog box reappears displaying the selected Parent Group. For example:



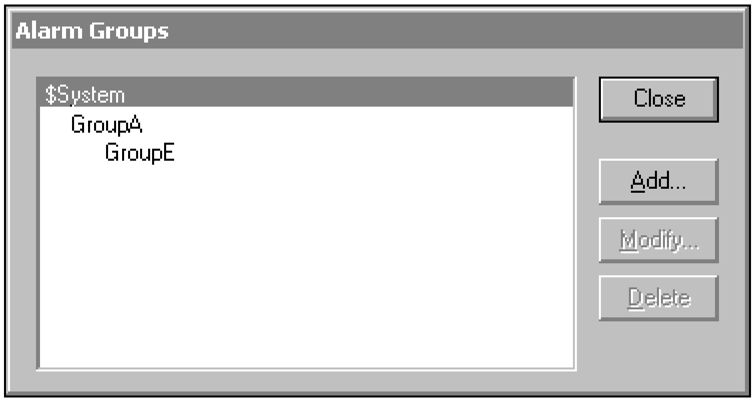
Group Name: GroupE

Parent Group: ... GroupA

Comment: GroupE is assigned to parent GroupA

6. In the **Comment** box, type any comment for the new Alarm Group.

7. Click **OK**. The **Alarm Group** dialog box appears displaying your Alarm Group hierarchy:

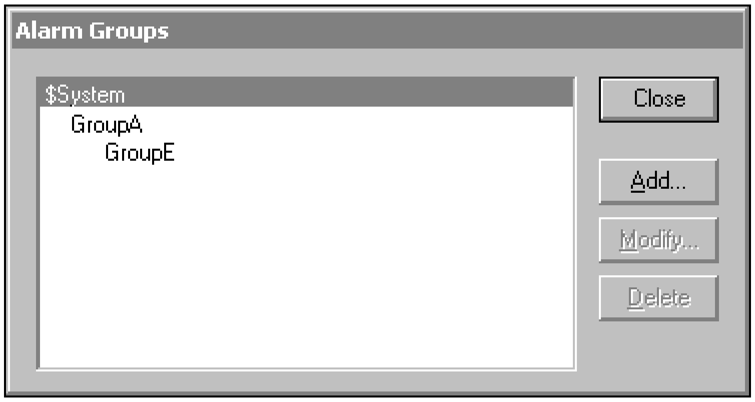


8. Click **Close**.

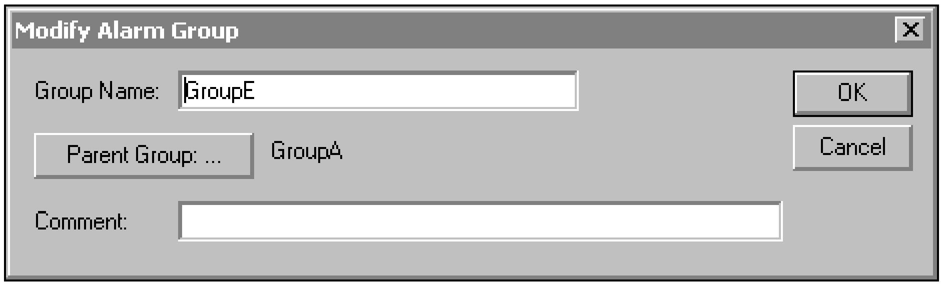
To modify an Alarm Group

1. On the **Special** menu, select **Alarm Groups**. The **Alarm Groups** dialog box appears.

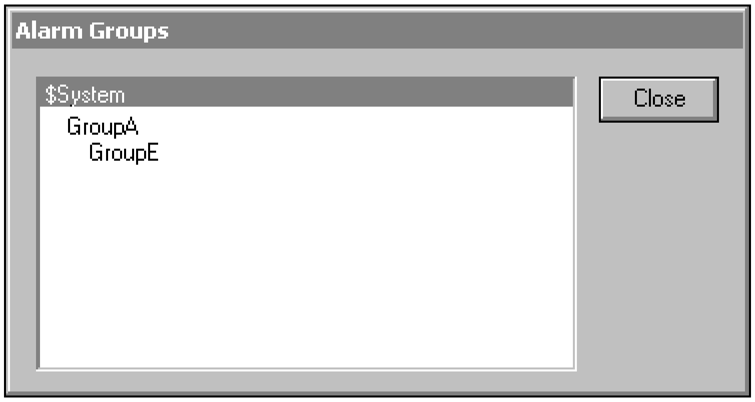
Tip You can also modify Alarm Groups while you are defining your tagnames in the Tagname Dictionary.



2. Select the Alarm Group that you want to modify in the list, and then click **Modify**. The **Modify Alarm Group** dialog box appears.

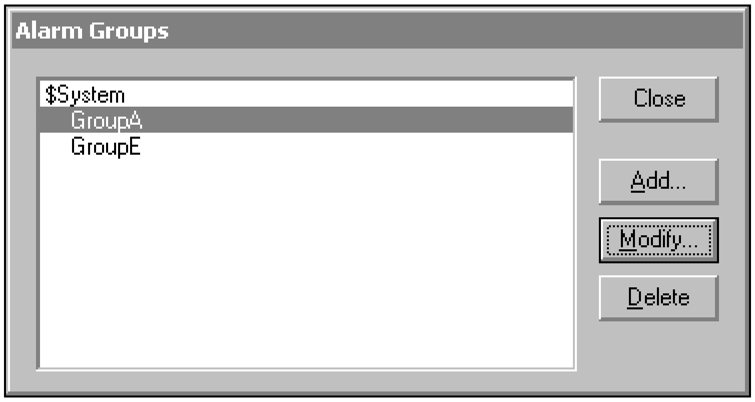


Make the required changes to the Alarm Group. If you want to change the parent group for the Alarm Group, click **Parent Group**. The **Alarm Groups** dialog box appears.



Select the new parent group, and then click Close. The Modify Alarm Group dialog box reappears displaying the new parent group.

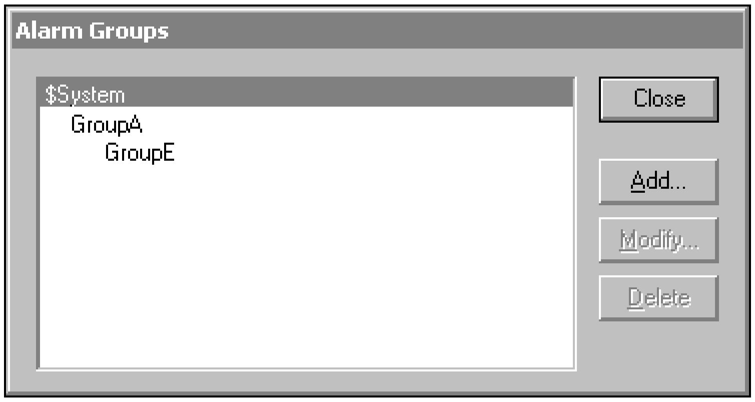
3. Click **OK**.



To delete an Alarm Group

1. On the **Special** menu, select **Alarm Groups**. The **Alarm Group** dialog box appears.

Tip You can also delete Alarm Groups while you are defining your tagnames in the Tagname Dictionary.



2. Select the Alarm Group that you want to delete in the list, and then click **Delete**. A message box will appear asking you to confirm the deletion. Click **Yes** to delete the Alarm Group, or click **No** to cancel the deletion.
3. Click **Close**.

Note Any alarms configured for the deleted group will automatically be assigned to the parent group.

Distributed Alarm Group Lists

The Distributed Alarm System uses the Alarm Group mechanism to group alarms into a local hierarchical tree structure that the distributed alarm display can use to filter alarms for display. The Distributed Alarm System also allows you to view these alarm groups from multiple nodes on a network. In addition, the Distributed Alarm System allows you to organize your viewing of alarms by creating **Alarm Group Lists.** For more information on viewing alarms, see Chapter 10, "Alarm/Event Clients."

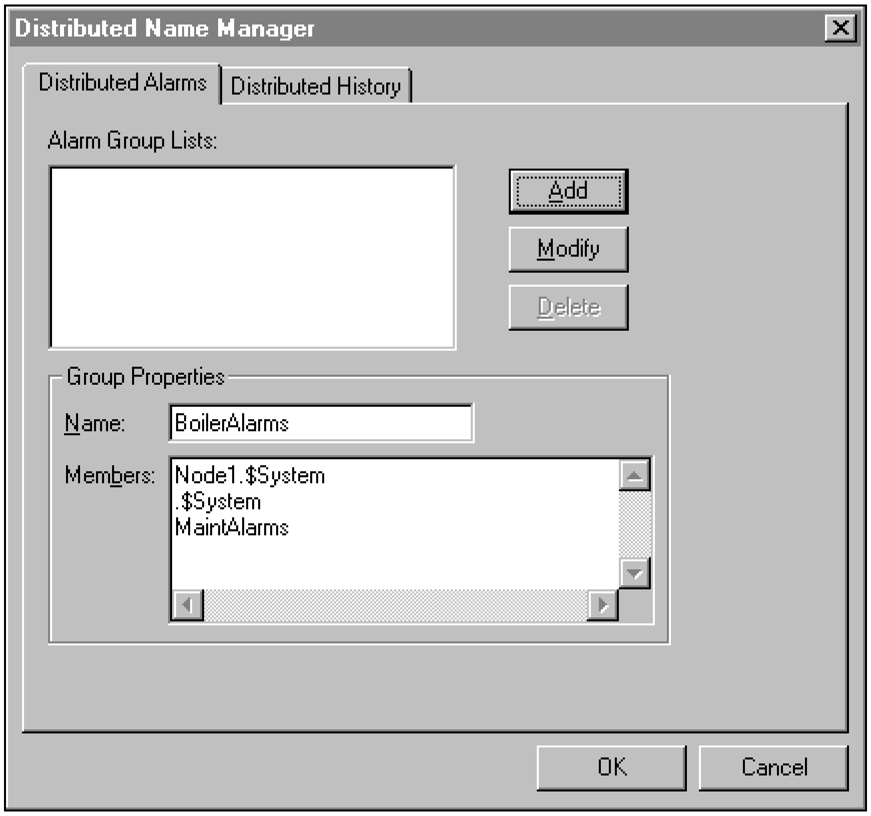
An **Alarm Group List** is a named list consisting of InTouch nodes and the Alarm Groups defined on each of those nodes. It can also contain other Alarm Group List Names and local Alarm Groups. An Alarm Consumer, such as the Distributed Alarm Display, uses this list to query for alarms. For more information on the Distributed Alarm Display, see Chapter 9, "Alarms/Events."

For example, if you were interested in all of the boiler alarms across several InTouch nodes, you could build a query called "BoilerAlarms." The list attached to that query would contain all the Alarm Groups on all the nodes that correspond to the boiler alarms.

To create an Alarm Group list

1. On the **Special** menu, point to **Configure**, and then select **Distributed Name Manager**. The **Distributed Name Manager** dialog box appears with the **Distributed Alarms** property sheet active:

Tip To quickly access the dialog box, in the Application Explorer under **Configure**, double-click **Distributed Name Manager**.



Name: BoilerAlarms

Members:
Node1.$System
.$System
MaintAlarms

2. In the **Group Properties** section, type the name of the query in the **Name** box.

3. In the **Members** box, type the list of InTouch nodes and Alarm Groups that you want to include in your query. The valid syntax for these lists include:

   Standard Group Entries

\\Node\InTouch!Group	Fully qualified path to an Alarm Group on a remote node
\InTouch!Group	Same as above, but Node assumed to be local
GroupList	Another Group List

   Shortcut Group Entries:

Node.Group	Shortcut that equates to **\\Node\InTouch!Group**
.Group	Shortcut that equates to **\InTouch!Group**

   Where **Node** identifies the name of the InTouch remote node and .Group identifies the Alarm Group on that node. If the Alarm Group is local, you can enter just the Alarm Group name with a period. For example, **.AlarmGroup**.

Tip The **Shortcut Group Entries** provide you with an easy way to enter node and Alarm Group information into the dialog box. It's important to remember that this information is translated into the **Standard Group Entry** format when you save the Alarm Group List.

Note The **Node.Group** and **.Group** syntax can only be used in this configuration dialog box. It is not valid in the alarm display configuration or any alarm QuickScript function.

4. Click **Add** to add this list to your Alarm Group file. The syntax of the **Members** will automatically be converted.
5. Click **OK**.

Alarm Acknowledgment Models

This section describes how alarms are reported, tracked and acknowledged.

Whenever an item goes from a **Normal** state to an **Alarmed** state, a new *instance* of its alarm is generated. If the alarm is a multi-state alarm, any change to a new sub-state while still remaining alarmed is treated as a transition of the same alarm instance. The *lifetime* of an alarm instance stops when the alarm returns to **Normal** - a subsequent transition to the alarmed state generates a new alarm instance.

The InTouch Distributed Alarm system tracks each *instance* of an alarm - when it first enters the alarmed state, when it makes a sub-state transition, when it returns to **Normal**, whether it is waiting for an **Acknowledgment**, and when it is **Acknowledged**.

- **Alarms as conditions:** Acknowledgment is for the *instance* of the alarm. An alarm instance begins waiting for an **ACK** when it first enters the **alarmed** state. If it is **ACKed** and subsequently transitions to a new alarmed sub-state (for example from "Hi" to HiHi"), it begins waiting for another **ACK**. Whenever the **ACK** is received, it is accepted and applies to all transitions of the alarm that have occurred so far. The alarm is considered **ACKed** when the most recent instance has been **ACKed**.
- **Alarms as events:** Acknowledgment is for the *instance* of the alarm, and must be for the most recent transition to an alarmed state or sub-state. An alarm *instance* begins waiting for an **ACK** when it first enters the alarmed state. If it is **ACKed** and subsequently transitions to a new alarmed sub-state, it begins waiting for another **ACK**. Each subsequent transition is assigned a sequence number, and the **ACK** must have attached to it the sequence number of the transition to which it is responding. The **ACK** is accepted only if it is for the most recent transition. A rejected **ACK** may be logged for diagnostic purposes, but is not otherwise tracked in the system. If it is accepted, it applies to all transactions of the alarm that have occurred so far. The alarm is considered **ACKed** when the most recent instance has been **ACKed**. The event-oriented model ensures that if an alarm is changing between different states, the **ACK** corresponds to up-to-date information. In systems with small latency times, this may look just like condition-oriented alarms - but in other environments, such as the Internet, the features of this model may become important.

- **Expanded Summary Alarms:** Acknowledgment is for each transition of the alarm. The initial entry to the alarmed state must be **ACKed**, and the return to normal must also be separately **ACKed**. Any transition to a new alarm sub-state is treated as a new occurrence that must be **ACKed**, and whose **RTN** must also be **ACKed**. Sub-state transitions are treated as belonging to a "**RTN group**," starting with the first entry into an ***alarmed*** state when the item was previously ***normal***. If the item ***returns to normal*** and subsequently enters the ***alarmed*** state again, a new **RTN group** is created. Each transition must be acknowledged individually and explicitly; and the alarm is considered acknowledged only when the item has ***returned to normal*** and all transitions in all pending **RTN groups** have been ***acknowledged***.

Note Here the term "summary" is clearly employed as meaning "awaiting acknowledgment." This alarm model is also known in the industry as "ring-back alarms."

To summarize, for **condition-oriented** alarms, an acknowledgment counts against all entries into the ***alarmed*** state up to the time of the acknowledgment. For **event-oriented** alarms (as in OPC) an acknowledgment is accepted only if it refers to the most recent "activation" event. For **Expanded Summary** alarms, all events - to "active," "inactive," and different sub-states - must be acknowledged before the overall alarm is considered acknowledged.

Expanded Summary Alarms

When an alarm occurs it generates a record in the alarm display object showing that an alarm condition has occurred with the date and time stamp of the alarm. This record does not leave the display until an operator has acknowledged the alarm and a return-to-normal state (RTN) has occurred. If the return-to-normal state occurs before the alarm is acknowledged, then two records are displayed in the alarm object.

For example, if a boiler’s temperature exceeds the hi-limit state, thus triggering an alarm, but returns to the normal temperature range before an operator acknowledges the alarm state, a record will be generated showing the alarm state and an additional record will be generated showing that the alarm state was not acknowledged.

An acknowledgment is only for a particular transition, whether to an alarmed state, to a sub-state, or a return to normal. Each transition from the normal state marks the beginning of a new RTN group. All transitions in an RTN group must be acknowledged individually before the overall RTN group is considered acknowledged.

Using Expanded Summary Alarms

When you define a tagname and select **Expanded Summary** as its ACK Model it requires an operator to acknowledge that an alarm occurred even if the state triggering the alarm has returned-to-normal. Acknowledging an alarm state will change the color of the alarm entry but does not change the displayed time stamp. Alarms will only clear from the display when they are acknowledged and the alarm has returned to a normal state.

Note When you define a tagname with the **Expanded Summary** ACK mode, the **RTN Implies Ack** option in the **Alarm Properties** dialog box does not apply to the tagname.

For more information, see Chapter 6, "Tagname Dictionary."

Publish/Subscribe Mechanism

On any given node there can be a collection of **Alarm Providers** (Publishers) and **Alarm Consumers** (Subscribers). The InTouch Distributed Alarm system provides the **Publish/Subscribe mechanism** by which alarm information is passed between nodes and between software components.

Alarm Provider

An **Alarm Provider** keeps track of ***alarmable*** items - that is, items that may go into an ***alarmed*** condition - and provides the Distributed Alarm System with the list of these items, including information on any hierarchical grouping of the items. It also notifies the Distributed Alarm System when the status of such an item changes. Status changes include whether the item is in or out of the ***alarmed*** state and whether the most recent alarm has been ***acknowledged***.

In addition the Alarm Provider keeps track of whether an item is disabled.

Note In Windows 2000, if the Alarm Provider is in a different domain from the Alarm Consumer, the Alarm Consumer will not be able to see the alarms unless the query has the fully qualified name of the provider machine or the IP address of the provider name. For example, an Alarm provider in a different domain, can be specified in the query as:
\\provider1.b3.wonderware.com\intouch!$system
where "provider1" is the machine name and the "b3.wonderware.com" is the Primary DNS Suffix for the domain.

Alarm Consumer

An **Alarm Consumer** provides the Distributed Alarm System with a set of **queries** identifying alarmable items about which it wishes to receive notifications. A **query** remains active until changed or removed by the Alarm Consumer, and specifies an Alarm Provider or group of alarms - much like a SQL query with "wildcards." Whenever an Alarm Provider issues notification of a change, the Distributed Alarm System checks the alarm for matches with all registered queries and passes updates to the corresponding Alarm Consumers. Upon receiving updates, an Alarm Consumer displays or logs information relating to the status of the items or their transitions. An alarm Consumer may also ***acknowledge*** an alarm. This amounts to sending an acknowledgement notification to the Distributed Alarm System, identifying the alarm and the Alarm Provider. The notification is passed to the Alarm Provider, which then updates the status of the item to ***acknowledged*** (if appropriate) and in turn notifies the Distributed Alarm System, thereby ensuring that the update gets distributed to all interested Alarm Consumers.

Note The majority of communication in the Distributed Alarm System consists of sending **alarm queries** and **alarm records** from one node to another. Within a node, alarm queries and alarm records are tracked and cached by the **Alarm Buffer** to minimize network traffic.

Alarm Data Storage

There are several forms of data storage used in the Distributed Alarm System:

- **Alarm Cache:** Most information about current and recent alarms is cached in memory on the various computer nodes. InTouch's original alarm system used two caches: one for "summary" alarms (current) and one for "historical" alarms. This model is also used in the Distributed Alarm System.

  The cache for summary (current) alarms is allowed to grow as large as need to be, up to the limit of available memory, to accommodate all current alarms. The cache for historical alarms is allowed to grow only up to a pre-determined limit that is configured in an .INI file. Once the historical cache reaches this limit, the oldest alarm records are discarded as new ones are added. In a multi-node environment, the alarm caches on the various nodes constitute a distributed in-memory database.
- **Alarm Logging:** The Alarm DB Logger creates a database, keeping track of when an alarm occurs, makes a sub-state transition, is acknowledged, and when it returns to normal. Essentially, these constitute a permanent or quasi-permanent history of alarms in the system.

The Distributed Alarm System supports logging to a database, such as Microsoft SQL Server or MSDE. These databases support access via an open, non-proprietary interface, which facilitates the examination or analysis of the database contents.

Note The original InTouch standard alarm system logged to a **flat text file** using a format that could be configured by the user. This form of logging has been retired with the rest of the Standard Alarm Subsystem.

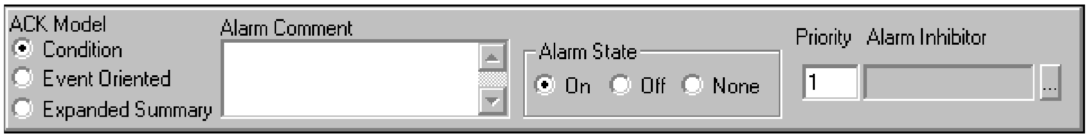
Because it is based on the use of **Queries**, the Distributed Alarm System supports using one computer node to log alarms for several other nodes.

Tagname Alarm Configuration

InTouch allows you to define an alarm configuration for each tagname that you define in the Tagname Dictionary. (By default, all tagnames have their alarms disabled.) The basic concept is that any tagname can be configured as ***alarmable***, specifying the type of alarm and user-selected limits. For such a tagname, whenever the value of the tagname changes, the **alarm logic** is invoked. This alarm logic is a subroutine internal to InTouch that checks the type of alarm, compares the new value to the indicated limits, and determines whether the tagname is in an ***alarmed*** condition. Any transitions of state are then reported to the Distributed Alarm System.

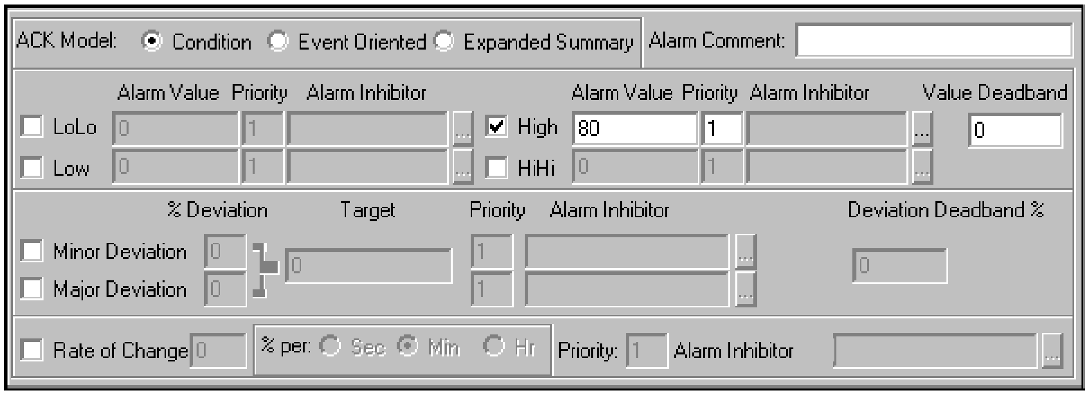
There are three basic alarm types (or classes) defined in InTouch. These are subdivided into several additional alarm sub-types:

- **Discrete:** A discrete alarm corresponds to a discrete tagname. You can configure whether the *alarmed* state corresponds to the TRUE state or the FALSE state of the discrete tagname, and the associated *priority* of the alarm. The configuration dialog for a discrete tagname is as follows:


ACK Model: [✓ Condition] Event Oriented / Expanded Summary
Alarm Comment:
Alarm State: [✓ On] Off / None
Priority: 1
Alarm Inhibitor:

Analog: An analog alarm corresponds to either an integer or real (floating point) tagname. Within the analog type there are several alarm classes:

- **Value:** The current value is compared to one or more limits. If the value exceeds the limit, the alarmed state is declared. You can individually configure values and priorities for the "LoLo" limit, "Lo" limit, "Hi" limit and "HiHi'" limit and indicate whether or not each limit is to be used.
- **Deviation:** The current value is compared to a target value, and then the absolute value of the difference is compared to one or more limits, expressed as a percent of the ***range*** of the tagname value - that is, the total difference between its configured maximum and minimum allowable values. You can individually configure values and priorities for the "Minor Deviation" limit and the "Major Deviation" limit, and indicate whether or not each limit is to be used. You can also configure a value for a "Deviation Deadband," also expressed as a percent of the tagname's range. This controls the percentage the tagname value that must drop before it is taken out of alarm.
- **Rate-of-Change**: The current value <u>and</u> the previous value are used in this computation, along with the current time and the time of the previous update. If the absolute value of the rate of change exceeds the limit, the alarmed state is declared. You can configure the value and priority for the ROC limit, and whether or not the limit is to be used. The limit is expressed as a percent of the range of the tagname value - which can be per second, per minute, or per hour. The configuration dialog for an analog tagname is as follows:


ACK Model: [✓ Condition] Event Oriented / Expanded Summary
Alarm Comment:

	Alarm Value	Priority	Alarm Inhibitor
☐ LoLo	0	1	
☐ Low	0	1	
✓ High	80	1	
☐ HiHi	0	1	

Value Deadband: 0

	% Deviation	Target	Priority	Alarm Inhibitor
☐ Minor Deviation	0	0	1	
☐ Major Deviation	0		1	

Deviation Deadband %: 0

☐ Rate of Change: 0 — % per: Sec [✓ Min] Hr — Priority: 1 — Alarm Inhibitor:

SPC: The SPCPro program can generate a Statistical Process Control alarm. This is actually defined outside InTouch itself, and operates through a separate software execution path from the rest of the alarms.

For more information on SPC alarms, see your online *SPCPro User's Guide.*

For more information on defining alarm conditions, see "Defining Tagname Alarm Conditions."

Alarm Dotfields

InTouch provides various alarm "**dotfields**" that allow you to dynamically control and/or monitor various alarm conditions. Many of these **dotfields** are accessible using I/O, expressions and/or scripts. I/O access provides the ability to monitor and/or control a specific tagname's alarm information using other Windows applications, such as Excel, or a remote View application (described later in this chapter).

For example, if you create an analog alarmed tagname called **Analog_Tagname**, it will have "attributes" associated with it such as its name, its **HiHi** setpoint, and so on. Some of these "attributes" are accessible through logic scripts, expressions and user inputs and are known as **dotfields**.

The syntax required to access these dot fields associated with a tagname is **Tagname.dotfield**. For example, if you want to allow runtime changes to the **HiHi** alarm limit on **Analog_Tagname**, you could create an **Analog - User Input** touch link to a button and **Analog_Tagname**.**HiHiLimit** would be entered as the expression in the link's dialog box. During runtime, the operator would simply click on the button and type in a new value for the **HiHi** alarm limit being used for **Analog_Tagname**.

The following briefly describes each example of how to use the alarm dotfields.

Dotfield	Description
.Ack	Read/write discrete tagname dotfield that monitors/controls the alarm acknowledgment status of tagnames and Alarm Groups. **.Ack** has an inverse tagname dotfield called **.UnAck.** When an unacknowledged alarm occurs, **.UnAck** will be set to 1. **.UnAck** can then be used in animation links or condition scripts to trigger annunciators for any unacknowledged alarms.
.AckDev	Monitors/controls the alarm acknowledgment status of deviation type alarms active on the tagname.
.AckDsc	Monitors/controls the current acknowledge state of a discrete tag.
.AckROC	Monitors/controls the alarm acknowledgment status of rate-of-change type alarms active on the tagname.
.AckValue	Monitors the alarm acknowledgment status of value type alarms active on the tagname.

Dotffield	Description
.Alarm	Signals that an alarm condition exists.
.AlarmAccess	Returns the name of the tagname associated with a selected alarm. The alarm has to be selected by clicking on the summary Distributed Alarm Display.
.AlarmAckModel	Monitors the **ACK model** associated with the tagname as follows: 0=condition (default) 1=event 2=expanded Applies to discrete or analog tagnames with alarms. Read only, but can be configured in WindowMaker.
.AlarmClass	Returns the class of the current alarm.
.AlarmComment	Read/Write text string that describes what the alarm is about, not what the tagname is about. By default it is empty in a new application. However, when an InTouch 7.1 application is converted to a 7.11 application, for backward compatibility the tagname comment is copied to the **AlarmComment**.
.AlarmDate	Returns the date of the current alarm.
.AlarmDev	Signals that a deviation alarm exists.
.AlarmDevCount	Tracks the total number of deviation alarms active on a given tagname or alarm group.
.AlarmDevDeadband	Monitors/controls the deviation percentage deadband for both minor and major deviation alarms.
.AlarmDevUnAckCount	Tracks the total number of unacknowledged deviation alarms active on a given tagname or alarm group.
.AlarmDisabled	Disables/enables events and alarms. Applies to discrete and analog tagnames with alarms, or to alarm groups. **Note** The same as the **.AlarmEnabled**, but of opposite polarity.
.AlarmDsc	Indicates that a discrete alarm condition is currently active.
.AlarmDSCCount	Tracks the total number of discrete alarms active on a given tagname or alarm group.
.AlarmDscDisabled	Indicates whether or not the tagname can generate discrete alarms. **Note** This dotfield is the same as **.AlarmDisabled** for a discrete tagname.

Dotffield	Description
.AlarmDscEnabled	Indicates whether or not the tagname can generate discrete alarms. **Note** This dotfield is the same as **.AlarmEnabled** for a discrete tagname.
.AlarmDscInhibitor	Returns the name of the inhibitor tagname assigned to the discrete alarm (if any) for this tagname.
.AlarmDscUnAckCount	Tracks the total number of unacknowledged discrete alarms active on a given tagname or alarm group.
.AlarmEnabled	Disables/enables events and alarms.
.AlarmGroup	Contains the current query used to populate a distributed alarm display object.
.AlarmGroupSel	Returns the alarm group of the alarm.
.AlarmHiDisabled	Disables/enables the **Hi** limit for analog tagnames with alarms.
.AlarmHiEnabled	Disables/enables the **Hi** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dotfield, but opposite polarity.
.AlarmHiInhibitor	Returns the inhibitor tagname reference for the **Hi** limit. Applies to analog tagnames with alarms. Read only but can be configured in WindowMaker.
.AlarmHiHiDisabled	Disables/enables the **HiHi** limit for analog tagnames with alarms.
.AlarmHiHiEnabled	Disables/enables the **HiHi** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dotfield, but of opposite polarity.
.AlarmHiHiInhibitor	Returns the inhibitor tagname reference for the **HiHi** limit. Applies to analog tagnames with alarms. Read only but can be configured in WindowMaker.
.AlarmLimit	Returns the limit of the current alarm.
.AlarmLoDisabled	Disables/enables the **Lo** limit for analog tagnames with alarms.
.AlarmLoEnabled	Disables/enables the **Lo** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dotfield, but of opposite polarity.

Dotffield	Description
.AlarmLoInhibitor	Returns the inhibitor tagname reference for the **Lo** limit. Applies to analog tagnames with alarms. Read only but can be configured in WindowMaker.
.AlarmLoLoDisabled	Disables/enables the **LoLo** limit for analog tagnames with alarms.
.AlarmLoLoEnabled	Disables/enables the **LoLo** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dotfield, but of opposite polarity.
.AlarmLoLoInhibitor	Returns the inhibitor tagname reference for the **LoLo** limit. Applies to analog tagnames with alarms. Read only but can be configured in WindowMaker.
.AlarmMajDevDisabled	Disables/enables the **Major Deviation** limit for analog tagnames with alarms.
.AlarmMajDevEnabled	Disables/enables the **Major Deviation** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dotfield, but of opposite polarity.
.AlarmMajDevInhibitor	Returns the inhibitor tagname reference for the **Major Deviation** limit. Applies to analog tagnames with alarms. Read only but can be configured in WindowMaker.
.AlarmMinDevDisabled	Disables/enables the **Minor Deviation** limit for analog tagnames with alarms.
.AlarmMinDevEnabled	Disables/enables the **Minor Deviation** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dotfield, but of opposite polarity.
.AlarmMinDevInhibitor	Returns the inhibitor tagname reference for the **Minor Deviation** limit. Applies to analog tagnames with alarms. Read only but can be configured in WindowMaker.
.AlarmName	Returns the name of the current alarm.
.AlarmOprName	Returns the operator for the current alarm.
.AlarmOprNode	Returns the operator node for the current alarm.
.AlarmPri	Returns the priority value (1-999) for the current alarm.
.AlarmProv	Returns the provider for the current alarm.
.AlarmROC	Signals that a rate-of-change type alarm exists.

Dotffield	Description
.AlarmROCCount	Tracks the total number of rate of change alarms active on a given tagname or alarm group.
.AlarmROCDisabled	Disables/enables the **Rate of Change** limit for analog tagnames with alarms.
.AlarmROCEnabled	Disables/enables the **Rate of Change** limit for analog tagnames with alarms. **Note** The same as the corresponding disabled dotfield, but of opposite polarity.
.AlarmROCInhibitor	Returns the inhibitor tagname reference for the **Rate of Change** limit. Applies to analog tagnames with alarms. Read only but can be configured in WindowMaker.
.AlarmROCUnAckCount	Tracks the total number of unacknowledged rate of change alarms on a given tagname or alarm group.
.AlarmState	Returns the state of the current alarm.
.AlarmTime	Returns the time of the current alarm.
.AlarmTotalCount	Tracks the total number of alarms active on a given tagname or alarm group.
.AlarmType	Returns the type of the current alarm.
.AlarmUnAckCount	Tracks the total number of unacknowledged alarms active on a given tagname or alarm group.
.AlarmUserDefNum1	Read/write real (floating point), default 0 and value not set. Applies to discrete tagnames with alarms, to analog tagnames with alarms, or to alarm groups. **Note** The value of this dot field is attached to the alarm, but ONLY if a value has been set, for example, via a script or a POKE.
.AlarmUserDefNum2	Read/write real (floating point), default 0 and value not set. Applies to discrete tagnames with alarms, to analog tagnames with alarms, or to alarm groups. **Note** The value of this dot field is attached to the alarm, but ONLY if a value has been set, for example, via a script or a POKE.
.AlarmUserDefStr	Read/write text string, default "" and value not set. Applies to discrete tagnames with alarms, to analog tagnames with alarms, or to alarm groups. **Note** The value of this dotfield is attached to the alarm, but ONLY if a value has been set, for example, via a script or a POKE.

Dotffield	Description
.AlarmUserDefNumSet1	Read/write discrete. TRUE of a script has defined the .AlarmUserDefNum1 for the corresponding tag. To undefine the value of .AlarmUserDefNum1 for the tag, set this dotfield to FALSE. The default is FALSE.
.AlarmUserDefNumSet2	Read/write discrete. TRUE of a script has defined the .AlarmUserDefNum2 for the corresponding tag. To undefine the value of .AlarmUserDefNum2 for the tag, set this dotfield to FALSE. The default is FALSE.
.AlarmUserDefStrSet	Read/write discrete. TRUE if a script has defined the .AlarmUserDefStr for the corresponding tag. To undefine the value of .AlarmUserDefStr for the tag, set this dotfield to FALSE. The default is FALSE.

Note The .**AlarmUserDefNum1**, .**AlarmUserDefNum2**, and **AlarmUserDefStr** dotfields allow you to assign one or more values to be attached to the alarm record when it is reported. These values are written to the database by Alarm DB Logger. There are three items you can attach to an alarm: two numbers and a string. By default, they are empty (zero and "").

To simplify setting user values, you can set them on an alarm group as well as on a specific tagname. For example, InBatch could set the batch number in .**AlarmUserDefNum1** all the way up at the **$System** alarm group, causing all alarms to have the batch number attached.

If you set .**AlarmUserDefNum1** on an Alarm Group, it applies to all alarms in that group and any of its sub-groups. You can also specifically set the value of .**AlarmUserDefNum1** on a tagname. In this case, it applies only to that tagname and overwrites any setting of .**AlarmUserDefNum1** in the tagname's Alarm Group. You can undefine a user-defined dotfield by setting the corresponding "set" dotfield to FALSE. For example, you can set Tag 1 .AlarmUserDefNum1Set = FALSE to un-define the value of Tag 1 .AlarmUserDefNum1.

Dotffield	Description
.AlarmValDeadband	Monitors/controls the value of an alarm's deadband.
.AlarmValue	Returns the value of the current alarm.
.AlarmValueCount	Tracks the total number of value alarms active on a given tagname or alarm group.
.AlarmValueUnAckCount	Tracks the total number of unacknowledged value alarms active on a given tagname or alarm group.
.DevTarget	Monitors/controls the target for minor and major deviation alarms.
.Freeze	Reads/writes the freeze status of the Distributed Alarm Display Object.

Dotffield	Description
.HiLimit, .HiHiLimit, .LoLimit, .LoLoLimit	Read/write analog tagname dotfields that monitors /controls the limits for value alarm checks. These dotfields are only valid for integer and real tagnames.
.HiStatus, .HiHiStatus, .LoStatus, .LoLoStatus	Read only discrete dotfields that determines whether an alarm of a specified type exists. These fields are only valid for integer and real tagnames.
.MajorDevPct	Read/write integer dotfield that monitors or controls the major percentage of deviation for alarm checking.
.MajorDevStatus	Read only discrete dotfield that determines whether a major deviation alarm exists for the specified tagname.
.MinorDevPct	Read/write integer dotfield monitors and/or controls the minor percentage of deviation for alarm checking.
.MinorDevStatus	Read only discrete dotfield used to determine whether a minor deviation alarm exists for the specified tagname.
.Name	Read/write message dotfield used to display the actual name of the tagname. For example, it can be used to determine the name of an Alarm Group that a Group Variable is pointing to, or the name of a TagID tagname. It can also be written to change the Alarm Group that a Group Variable is pointing to.
.ListChanged	Indicates whether there are any new alarms or updates for the Distributed Alarm Object.
.Normal	Read only discrete dotfield that is equal to 1 when there are no alarms for the specified tagname. This dotfield is valid for Alarm Groups and Group Variables as well as ordinary tagnames.
.NumAlarm	Shows the total number of alarms in the Distributed Alarm Object.
.PendingUpdates	Indicates if there are any updates pending for an Alarm Display Object.
.ROCPct	Read/write dotfield used to monitor and/or control the rate of change for alarm checking.
.ROCStatus	Read only discrete dotfield used to determine whether a Rate-of-Change alarm exists for the specified tagname.
.SuppressRetain	Reads/writes the suppress retain status of the distributed alarm display object.

The following alarm dotfields are associated with an alarm object (Distributed Alarm Display), not a tagname.

.AlarmClass	**.AlarmAccess**	**.AlarmDate**	**.AlarmGroupSel**
.AlarmLimit	**.AlarmName**	**.AlarmOprName**	**.AlarmOprNode**
.AlarmPri	**.AlarmProv**	**.AlarmState**	**.AlarmTime**
.AlarmType	**.AlarmValue**	**.Freeze**	**.ListChanged**
.NumAlarm	**.PendingUpdates**	**.SuppressRetain**	

These alarm dotfields are accessed by using the function

```
GetPropertyM(ControlName.Property, MsgTag)
```

When executed at runtime, the property is obtained in the **MsgTag** to display for the tagname in the selected row. If multiple rows are selected, the property in the **MsgTag** is the tagname in the first row of the multiple selection.

For more information on alarm dotfields, see your online *InTouch Reference Guide.*

Alarm Visibility Controls

For various reasons, at times it may be necessary or desirable for you to turn some alarms "off" without actually removing the alarm configurations for an item. InTouch supports three basic types of alarm visibility control: **disablement**, **inhibition**, and **suppression**. These are described in terms of a three-stage alarm model.

- **Alarm stimulus:** The value or status of an ***alarmable*** item is tracked.
- **Alarm state:** The alarmable item is compared against limits and conditions, to determine if it is in an ***alarmed*** state. State transitions are tracked and reported.
- **Alarm annunciation:** Reports and updates of the alarm state are ***displayed*** and/or ***logged*** via one or more alarm clients.

With this model in mind, the following visibility controls are handled at the Alarm Provider:

- **Alarm Disablement:** An alarm can be **disabled** at the Alarm Provider by setting a flag (for example a status bit) that marks it as **disabled**. No other change to the alarm configuration is involved. While an alarm is disabled, any checking may or may not continue regarding conditions that would put the item into an ***alarmed*** state. But while the disablement is in effect those conditions cannot put the item into the ***alarmed*** state. Essentially, the item is in a "forced normal" state.

  Disablement breaks the connection between the **alarm stimulus** and the **alarm state.**

  - Note that disablement takes place entirely within the Alarm Provider.
  - All Alarm Consumers will see the same results.
  - Since the alarm cannot go into the ***alarmed*** state, while an alarm is disabled, no entries are made to the alarm history for that alarm.

- **Alarm Inhibition:** An alarm can be **inhibited** by identifying a tagname that marks it as inhibited. This tagname is called an ***inhibitor tag.*** No other change to the alarm configuration is involved. While the inhibitor tagname is FALSE (zero or NULL), the alarm is handled normally; but while the inhibitor tagname is TRUE (non-zero or non-NULL), the item cannot alarm. In essence, it is treated the same as if the alarm were **disabled**. We then say that the alarm is **actively inhibited**. Clearly, alarm inhibition is a two-stage process:

  1. Assignment of the inhibitor tagname.
  2. A change of the state of the inhibitor tagname from FALSE to TRUE or vice-versa.

As with disablement, **inhibition** breaks the connection between the **alarm stimulus** and the **alarm state**.

Note Assigning a tagname as an inhibitor tagname for an alarm will increase its cross-reference use count.

With the three-stage alarm model in mind, the following visibility control is handled at the Alarm Consumer:

- **Alarm Suppression:** One or more alarms can be **suppressed** at an Alarm Consumer by identifying a set of exclusion criteria. If an alarm matches the exclusion criteria, it will not be visible at the Alarm Consumer. That is, it will not appear on a display, be printed, or be logged at that particular Alarm Consumer (according to its function). The actual generation of alarms is completely unaffected by suppression. Essentially, suppression causes an Alarm Consumer to ignore certain alarms.

  Suppressing an alarm breaks the connection between the **alarm state** and the **alarm annunciation**.

  - Note that suppression takes place entirely within the Alarm Consumer.
  - Each Alarm Consumer will see different results. That is, each Alarm Consumer can have its own set of exclusion criteria, thereby making different sets of alarms invisible. Essentially, suppression is a refinement on the Alarm Query.
  - Since the actual generation of alarms is unaffected, even while one or more Alarm Consumers suppress an alarm, entries can still be made to the alarm history for that alarm (assuming the Alarm DB Logger itself is not also suppressing the alarm).

  The Logger will log an entry whenever an Alarm Consumer changes its suppression criteria.

Note If the line "SuppressionLog=1" is added in alarmbuf.ini, "SuppressionLog=0" turns off the logging of entry into Logger for changes in suppression criteria. By default, this is off.

Disablement and Inhibition State Transitions During Runtime

InTouch lets you disable or enable all alarms of a tagname at once. In addition, for an alarm that has sub-states, each sub-state can be disabled individually. Thus, all sub-states can be disabled, or some sub-states can be disabled while others remain enabled. For example, an analog VALUE alarm can have "Hi" enabled and "HiHi" disabled.

As with disablement, for an alarm that has sub-states, you may also assign inhibitor tagnames to individual sub-states. Each sub-state can be inhibited by a different tagname, and you can leave some sub-states with no inhibitor tagname assigned. For example, an analog VALUE alarm can have "Hi" inhibited by one tagname, "HiHi" inhibited by another tagname, and the "Lo" and "LoLo" not inhibited at all.

InTouch permits changing the inhibitor tagname only in WindowMaker.

In WindowViewer, whenever the transition causes an alarm to change from being **actively inhibited**, the checking logic is executed to determine whether the Alarm Provider should put the item in the ***alarmed*** state.

During runtime, the Alarm Provider will not generate alarms for an alarm or sub-state that is **disabled**, or for an alarm or sub-state that is **actively inhibited**. Changes to whether an alarm is disabled or enabled can be made at runtime. Changes to the value of an inhibitor tagname can also be made at runtime. However, the assignment of inhibitor tagnames to alarms can be set or changed only in WindowMaker.

An alarm or sub-state can be independently disabled, inhibited, or both. If the alarm is either currently disabled or actively inhibited, the alarm is turned "off." Only if the alarm is both enabled and not actively inhibited is the alarm capable of becoming active. Therefore, even when the alarm is enabled, the alarm might not get to the consumer from the provider because of active inhibition by another tag.

If an alarm or sub-state has no inhibitor tagname assigned to it, the effect is the same as if it had an inhibitor tagname that is always FALSE - that is, the item is never actively inhibited.

Whenever an alarm transitions from **disabled** to **enabled**, the checking logic is executed to determine whether the item should be put in the ***alarmed*** state by the Alarm Provider.

If an alarm becomes **disabled** or **actively inhibited** while the item is in an ***alarmed*** state, the item will be forced to a different (valid) state. What that state should be depends upon which states are available and whether they have also been disabled. This activity is handled by the Alarm Provider according to the type of alarm, limit values, and so on.

An alarm that is disabled or actively inhibited is not waiting for an ***acknowledgment***. If the alarm has sub-states, it can only be waiting for an ***acknowledgment*** on sub-states that are still available.

Examples:

As an example of simple **disablement**, an InTouch application engineer could set up a control or a script to set the appropriate dotfield of a tagname: .AlarmDisabled to disable the alarm, .AlarmEnabled to enable it. This dot field can also be controlled in an external application using InTouch as an I/O server via NetDDE or SuiteLink.

For more information on alarm dotfields, see your online *InTouch Reference Guide.*

As an example of **inhibition**, an InTouch application engineer could set up a tagname named "Phase3" that becomes True during a particular stage of execution. For specific alarms in the application, the engineer would assign Phase3 as the inhibitor tagname. This means that when Phase3 is True, those alarms are disabled; when Phase3 is False, the alarms function normally.

Configuring the Alarm System

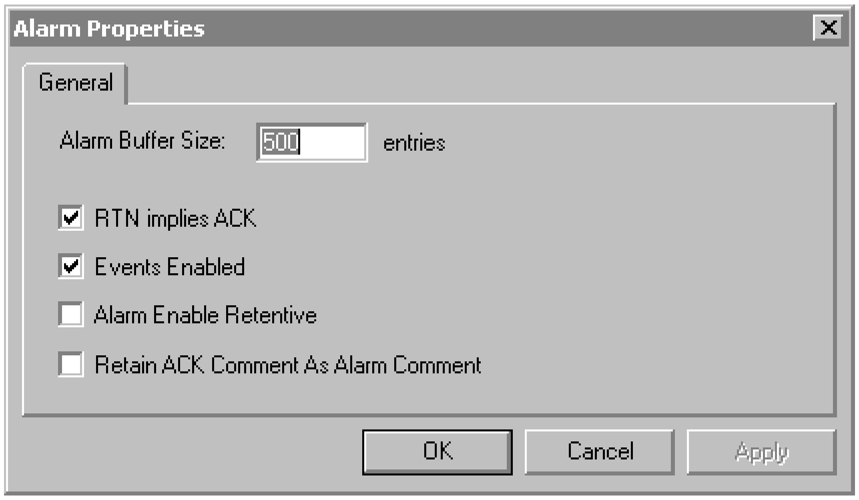
You can configure various parameters for the alarm system such as whether to enable events, whether alarms require an acknowledgment when they return to normal, and so on.

Note The configuration dialog box behaves like any standard Windows property sheet in that no settings are recorded until you click **OK**. If you click **Cancel**, all input is ignored and the dialog box closes.

Alarm/Event General Properties

To configure alarms/events general properties

1. On the **Special** menu, point to **Configure**, and then select **Alarms**, or in the Application Explorer under **Configure**, double-click **Alarms**. The **Alarm Properties** dialog box appears:



Alarm Buffer Size: 500 entries

[✓] RTN implies ACK
[✓] Events Enabled
[ ] Alarm Enable Retentive
[ ] Retain ACK Comment As Alarm Comment

Tip If you right-click a text box in any alarm configuration dialog box, a popup menu will appear displaying the commands that you can apply to the selected text.

2. In the **Alarm Buffer Size** box, the number of "in-memory" alarm events you want WindowViewer to maintain. (The maximum number of alarms that the node can store for summary or historical queries.)

Note Only "in-memory" alarm events can be displayed in alarm display objects. If alarms are not being used, this value may be set to 1 to conserve memory.

If you set this value too high, it can slow down the performance of your system. For the Distributed Alarm System, a value of 500 is recommended.

3. Select **RTN implies ACK** if you want to automatically acknowledge alarmed tagnames that return to the "normal" state (RTN). Do not select this option if you want the operator to acknowledge an alarm after it returns to normal.

Note When you define a tagname with the **Expanded Summary** ACK model, the **RTN Implies ACK** option does not apply to that tagname..

4. Select **Events Enabled** if you want to turn on event logging of all data changes that are initiated by the operator, I/O, QuickScripts, or the system. (Only tagnames with **Log Events** selected will be affected.)

   For more information on events, see "Alarms and Events."

5. Select **Alarm Enable Retentive** if you want the state of the **.AlarmEnabled** variable to be retained when WindowViewer is closed.

6. Select **Retain ACK Comment as Alarm Comment** if you want comments entered with alarm acknowledgements to be kept as updates to the tagname’s Alarm Comment dot field. When an operator ACKs an alarm, there is an option to provide a comment regarding the alarm. An ACK comment can also be provided via a script function. If the box for Retain ACK Comment for AlarmComment is checked, the ACK comment will be retained as the alarm comment for the corresponding tagname. InTouch will also update the Alarm Comment in the tagname dictionary.

   For more information, see "Attaching Comments to an Alarm Ack Function."

7. Click **OK** to save your settings and close the dialog box.

Attaching Comments to an Alarm Ack Function

A new dot field, the AlarmComment, has been added to provide a comment specifically for alarms. The AlarmComment can be up to 131 characters in length.

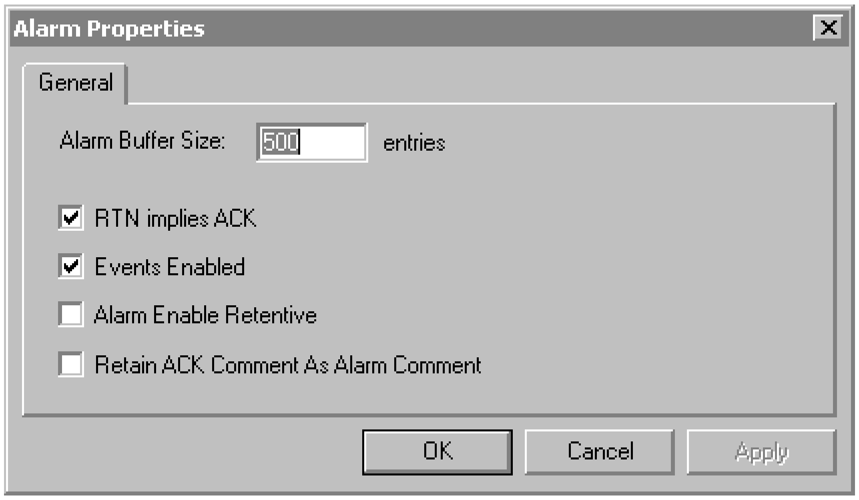
Each alarm acknowledgment function can have a comment attached to it, whether the ACK is done via the Distributed Alarm Object, a script function or any other means. The operator acknowledging the alarm can use this comment to add information about the alarm.

When an alarm becomes active, the Distributed Alarm System creates an alarm record to track that instance of the alarm. For the comment relating to the onset of the alarm, InTouch uses the Alarm Comment box of the tagname’s alarm definition in the database. If the operator provides a comment when ACKing an alarm, InTouch adds this to the Alarm Record as the ACK comment for that instance of the alarm and the comments for the Distributed Alarm System. Both are logged in the alarm database. The Distributed Alarm Object and the Alarm Printer show the Alarm Comment or the ACK Comment according to whether the instance of the alarm has been ACKed. The next time an alarm occurs on the same tag, the Alarm Comment is used again at the onset of the new alarm instance. The operator can enter a different ACK Comment when acknowledging the new instance.

You can also configure InTouch to "Retain ACK Comment as Alarm Comment," which enables you to use the ACK Comment to update the Alarm Comment in the tagname database (this feature used to update the tagname comment, but now updates the AlarmComment). If you enable this feature, the AlarmComment dot field will be overwritten during runtime -- including the AlarmComment entry in the Tagname Data Dictionary.

To use alarm acknowledge comments to update the AlarmComment field

1. On the **Special** menu, point to **Configure** and select **Alarms**. The **Alarm Properties** dialog box appears with the **General** properties sheet active:



Alarm Buffer Size: 500 entries

[✓] RTN implies ACK

[✓] Events Enabled

[ ] Alarm Enable Retentive

[ ] Retain ACK Comment As Alarm Comment

Tip If you right-click a text box in any alarm configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

2. Select **Retain ACK Comment as Alarm Comment** if you want comments entered with alarm acknowledgments to be kept as updates to the corresponding tagname's AlarmComment dot field and be copied to the Tagname Dictionary. If this box is not checked, the ACK Comment will display with the ACKed alarm (in the database, printouts and displays), but the Alarm Comment will not change.

   For more information on alarm comments, see "Chapter 6, "Tagname Dictionary."

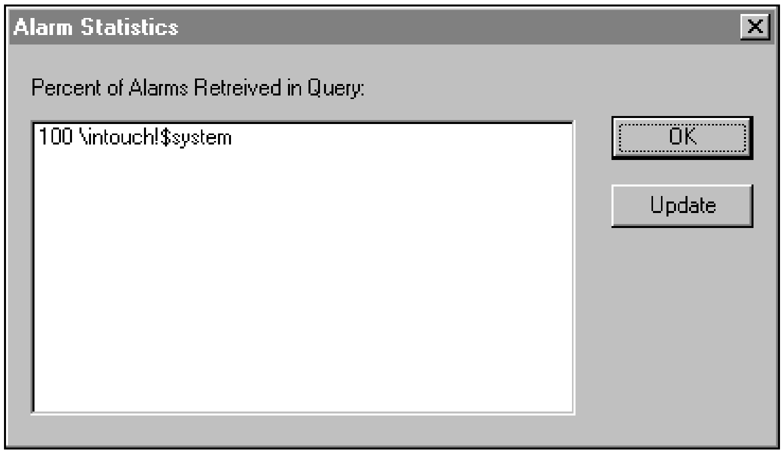
3. Click **OK**.

Displaying Alarm Statistics

The Distributed Alarm System provides a built-in alarm statistics dialog box. The application developer can design the application to call up the **Alarm Statistics** dialog box to list the status of the current query for a particular alarm display. This dialog box can also be invoked by selecting **Stats** from the right-click menu.

For more information, see "Selecting and Configuring Alarm Query Favorites."

The **Alarm Statistics** dialog box provides you with an overview of the current alarm query for a particular alarm display. It lists the actual alarm provider requests and the results of each. It's important to keep in mind that even though you may have requested a single Alarm Group List name, that name may equate to several individual Alarm Provider queries. For example:



```
100 \intouch!$system
```

Tip Each row in the dialog box lists a number and a query. The number represents the percentage of that query that has been returned. The dialog box provides a static display of the query results.

To update the percent of alarms retrieved in query list

1. Right-click the distributed alarm object.
2. Select **Stats** from the sub-menu.
3. Select the alarm query you want to update, then click **Update**.
4. Click **OK** to close the dialog box.

Acknowledging Local Alarms

You can acknowledge local alarms by using the **.Ack (dotfield)** in an action or key QuickScript, or by using the context sensitive right-click menu at the Distributed Alarm Object.

Note Not recommended for Expanded Summary alarms.

To create a local alarm acknowledge button

1. Create a 3-D button or any other object to which an action or Key QuickScript be linked.
2. Double-click the object, or select it and then on the **Special** menu, select **Animation Links**.

3. In the **Touch Pushbuttons** section of the Animation link selection dialog box, click **Action**. The QuickScript editor appears.

   Type any of the following statements for the QuickScript:

Statement	Description
Ack $System;	Acknowledges all local alarms in the system.
Ack Group Name;	Acknowledges all local alarms in a specific Alarm Group.
Ack Tagname;	Acknowledges a specific tagname's alarms.
$System.Ack=1;	Acknowledges all local alarms in the system.
Group Name.Ack=1;	Acknowledges all local alarms in a specific Alarm Group.
Tagname.Ack=1;	Acknowledges a specific tagname's alarms.
Tagname.AckDev=1;	Acknowledges a specific tagname's deviation alarm.
Tagname.AckROC=1;	Acknowledges a specific tagname's rate-of-change alarm.
Tagname.AckValue=1;	Acknowledges a specific tagname's value alarm.

4. Click **OK**.

For more information on the QuickScript editor and its features, see Chapter 8, "Creating QuickScripts in InTouch."

Timestamps in the Distributed Alarm System

The present Alarm Record for the Distributed Alarm system has five timestamps:

Timestamp	Description
ar_OrigTime	Time of origination, i.e. onset of alarm, OAT.
ar_SubTime	Time of last sub-state change, e.g. VALUE/HI to VALUE/HIHI
ar_AckTime	Time alarm was last acknowledged
ar_Time	Last changed time, LCT
ar_RtnTime	Time of return to normal, RTN.

Migrating from an Older InTouch Standard Alarm System to the Distributed Alarm System

InTouch automatically migrates InTouch applications that use Standard Alarm Object. When you try to open any application with Standard Alarm Objects in WindowMaker, InTouch automatically converts all the Standard Alarm Objects to Distributed Alarm Objects with the default values. The colors, fonts, the expressions, and the alarm query settings will not be migrated.

When you try to open an application built prior to 7.11 in Window Viewer, a dialog pops up prompting you to run WindowMaker to convert the application. If the Standard Alarm Object is in the application, the conversion will migrate the Standard Alarm Object to the Distributed Alarm Object.

If you click **Yes**, the application opens in WindowMaker and the application is converted. If you click **No**, the Viewer closes.

Migrating from Older Master/Slave Alarms to the Distributed Alarm System

As mentioned in the "Migrating from an Older InTouch Standard Alarm System to the Distributed Alarm System" section all the Standard Alarm Displays in the Master/Slave application will be replaced with Distributed Alarm Displays. The new Distributed Alarm Displays will have the default query as "\\nodename\intouch!$system" where node name is the name of the Master node.

You will now be able to view and acknowledge the remote alarms using the distributed alarm display.

Acknowledgment of remote alarms via POKEs to the .Ack fields will continue to work as before. No change to the application is needed. Depending on whether the I/O tagname was configured for NetDDE or SuiteLink, it may be necessary to enable NetDDE. However, you may decide you no longer need separate controls for issuing acknowledgements, as the alarms can now be ACKed via the Distributed Alarm Display.

Monitoring of alarm statuses via ADVISEs of the .Alarm fields will continue to work as before. No change to the application is needed. Depending on whether the I/O tagname was configured for NetDDE or SuiteLink, it may be necessary to enable NetDDE.

Hot Backup and Synchronization

The InTouch Distributed Alarm System provides the capability for Alarm Provider applications, such as InTouch, to issue notifications and to receive acknowledgments about alarms across a network. It also provides the capability for Alarm Consumer applications (clients) on remote nodes to query, display, and acknowledge these alarms. In a distributed alarm environment there may be a number of Alarm Providers and Alarm Consumers connected across the network.

In addition, it is possible to configure some of the Alarm Providers as ***backups*** to other Alarm Providers in the system. The purpose of having a backup Alarm Provider is to establish a fail-safe system - to ensure that alarms will be generated for certain critical circumstances by having a ***secondary*** Alarm Provider take over issuing alarm notifications if the ***primary*** Alarm Provider fails. For example, you can set up two separate computer nodes running identical InTouch applications interfacing to the same equipment, perhaps even with a redundant set of I/O connections. Alternatively, the primary Alarm Provider may be a "full-blown" application, while the secondary Provider is a "minimal" system, handling only the most critical parts to keep the equipment functioning and undamaged. Generally, only certain key Alarm Providers will be backed-up with secondary Alarm Providers.

To set up a backup configuration, you define a ***Hot Backup pair***, giving it a ***name*** and identifying a ***primary*** Alarm Provider and a ***secondary*** Alarm Provider.

Note The alarms from the secondary provider should be displayed only when the primary provider is inactive.

The idea of a backup configuration is that the Alarm Consumer (client) uses an alarm query that references the name of the ***Hot Backup pair***. For that ***pair*** the client sees only one set of alarms: the alarms from the primary, or if the primary has failed, the alarms from the secondary. In addition, the acknowledgements for the alarms on the primary and secondary are completely ***synchronized***. That is, if an alarm has been acknowledged on the primary Provider it is also acknowledged on the secondary Provider.

The InTouch Distributed Alarm System Hot Backup and synchronization feature accomplishes the following:

1. Provides you with a configuration utility that simplifies the process of establishing alarm backup pairs and designating which is primary and which is secondary.
2. Provides you with a configuration utility to perform ***alarm mapping***.
3. Built-in synchronization of alarm acknowledgments.
4. Handshaking during the startup/shutdown of an Alarm Provider.

Notes Regarding Hot Backup Pairs

1. Hot Backup only supports InTouch 7.11and later and no other clients.
2. Hot Backup is not supported for Expanded Summary Alarms or for event-oriented alarms.
3. If you have an alarm client (Distributed Alarm display object) query a Hot Backup pair and then have it also query the Primary Provider of the backup pair once again separately, the Distributed Alarm display object will display duplicate records.
4. A Provider should not be configured as a primary or secondary of more than one Hot Backup pair.

5. If a record at the primary Provider is acknowledged and the secondary Provider (which was down when the acknowledgment took place) comes up at a later time, the time stamp of the acknowledged record at the secondary Provider will be that of the secondary Provider and will not be identical to the primary Provider.
6. The alarm client (Distributed Alarm display object) querying a Hot Backup pair, will show the pair name as the Provider in the respective column, and will not show the Provider node name.
7. You can choose any combination of **Design-Time** and **Run-time** alarm record fields for mapping. However, it is important that you ensure that the mappings do not result in multiple references.
8. When mapping the **Value** and **Limit** key fields, the values are rounded off to the fourth decimal place and then mapped.
9. The alarm record that doesn't have the specific combination of design and runtime mapping will use the default runtime mapping.
10. Local node acting as a Hot Backup client cannot be configured as one of the Hot Backup Providers.
11. Alarm Provider Hot Backup pairs must not include the client node where the configuration is done from as one of the Alarm Providers. The Hot Backup utility is intended to work with pairs where Primary and Secondary Alarm Providers are different from the system that the configuration is done from.

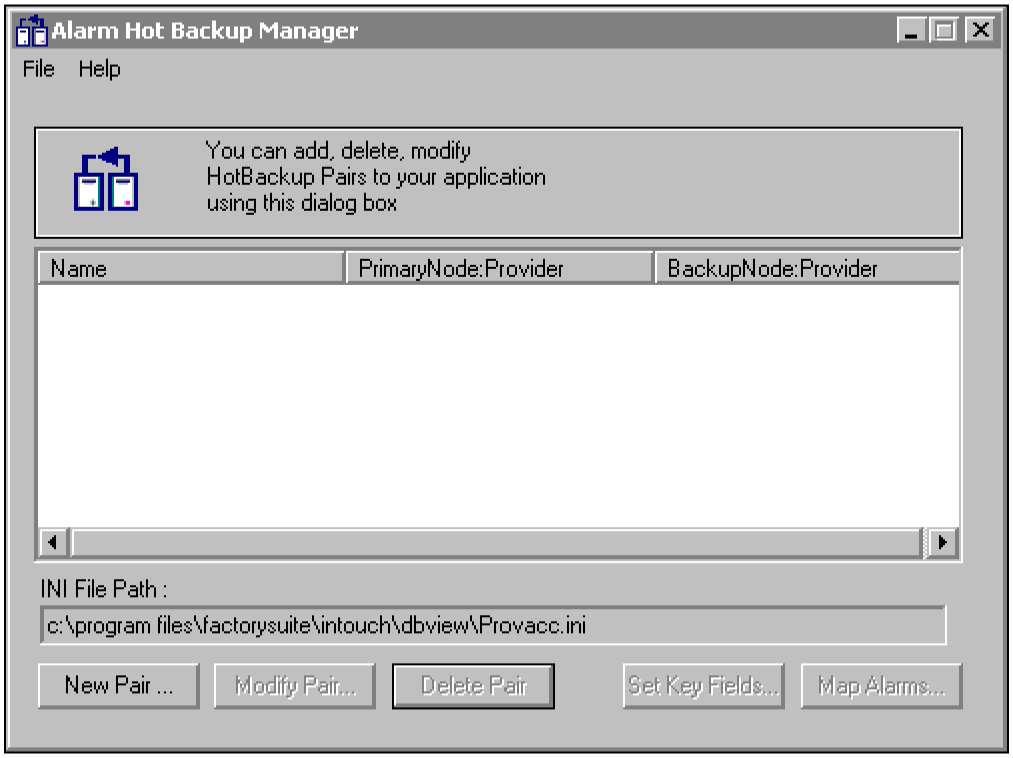
Hot Backup Configuration

The Hot Backup Synchronization Utility takes care of all necessary Hot Backup configurations. The Hot Backup Utility is automatically added to WindowMaker's Application Explorer when you install InTouch.

To configure Hot Backup

1. Start up WindowMaker.

2. Double-click the Alarm Hot Backup Manager item in the Application Explorer. The **Alarm Hot Backup Manager** dialog box appears.



You can add, delete, modify HotBackup Pairs to your application using this dialog box

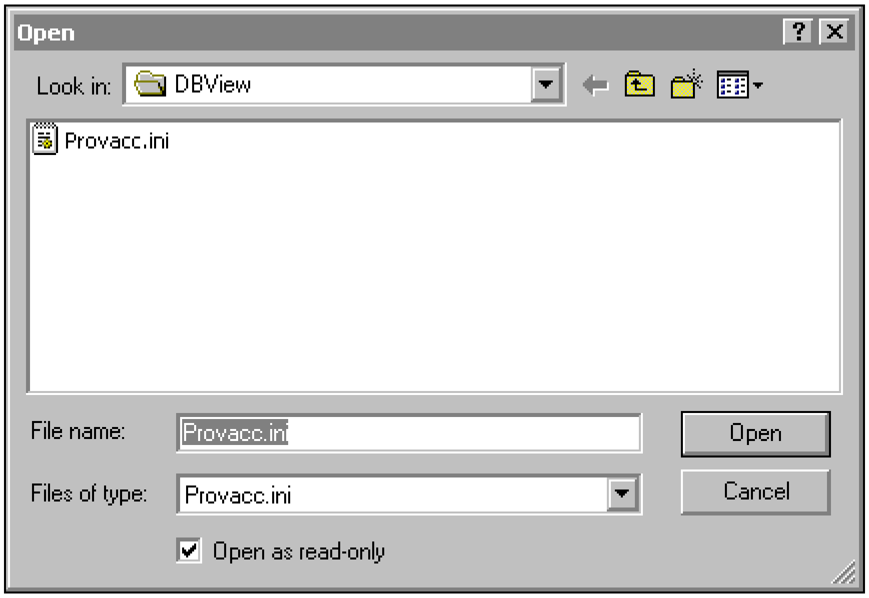
INI File Path : c:\program files\factorysuite\intouch\dbview\Provacc.ini

Note By default, the utility automatically checks for the provacc.ini file in the last opened InTouch application folder. If the file is available, the utility will configure itself with the file. Otherwise, the utility will create a new file in the required folder.

To change the default INI File

1. Start up WindowMaker.
2. Double-click the Alarm Hot Backup Manager icon in the Application Explorer. The **Alarm Hot Backup Manager** dialog box appears.

3. Select the **Open** command on the **File** menu. The **Open** dialog box appears listing all available provacc.ini files. (This dialog box allows you to select only a provacc.ini file.)

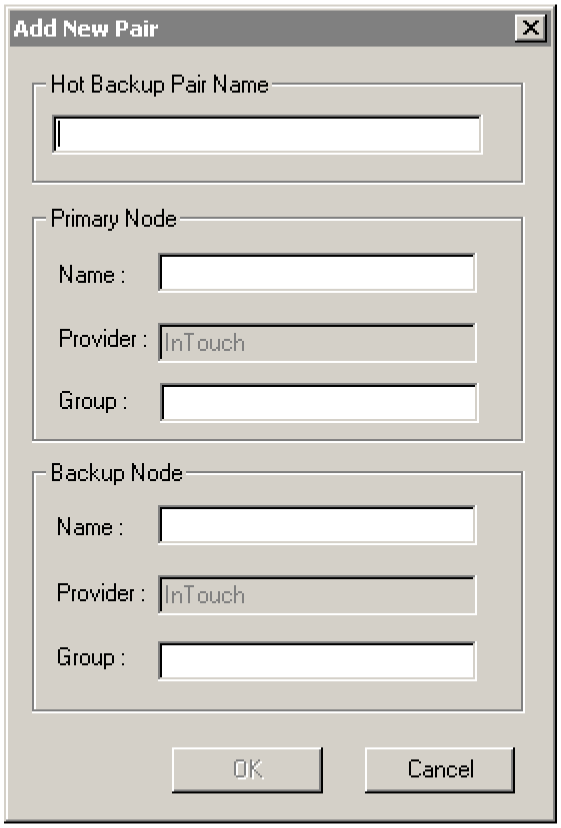


4. Change the path as necessary and then click **Open**. (It is recommended that the provacc.ini file is located in the application directory.)

To add a new Hot Backup pair

1. Start up WindowMaker.
2. Double-click the **Alarm Hot Backup Manager** item in the Application Explorer. The **Alarm Hot Backup Manager** dialog box appears.

3. Click **New Pair**. The **Add New Pair** dialog box appears.



4. In the **Hot Backup Pair Name** box, type a unique name for the new pair. Pair names can be up to 32 alphanumeric characters long.

Note Duplicate pair names are **not** allowed. Underscore is the only special character that is allowed at the beginning, middle or at the end of the Hot Backup pair name.

5. In the **Name** box in the **Primary Node** group, type the name of the computer where the primary provider application is running. Node names can be up 32 alphanumeric characters long.

   For the node name:

   - The node name must start with a letter
   - You can use any alphanumeric character
   - The only special character allowed is the underscore (_)
   - No other special characters are allowed in any place in the node name

   Note The node name for a particular application must be unique for the entire Hot Backup listing. An appropriate error message box appears if you enter wrong or duplicate entries.

Note InTouch is the constant provider name and cannot be changed.

6. In the **Group** box in the **Primary Node** group, type the name of the Alarm Group for querying alarms from the primary provider. For example, $System. Alarm Group names can be up 32 alphanumeric characters long.

   For the group name:

   - All valid characters are allowed (A to Z, a to z, 0 to 9, !, @, -, ?, #, $, %, _, \, &).
   - The group name must start with A to Z or a to z.
   - $ is the only other character a group name can start with and if it is used, the group name can only be $System.
   - You cannot use a backslash (\) at the end of a group name.

   Important! The "\" character is used instead of the "!" character for separating a pair name from a group name.

   - Consecutive backslashes (\\) are not allowed.

7. In the **Name** box in the **Backup Node** group, type the name of the computer where the backup provider application is running. Node names can be up 32 alphanumeric characters long.
8. In the **Group** box in the **Backup Node** group, type the name of the Alarm Group for querying alarms from the backup provider. For example, $System. Alarm Group names can be up 32 alphanumeric characters long.

   Note A hot backup pair name cannot be a node in another hot backup pair configuration.

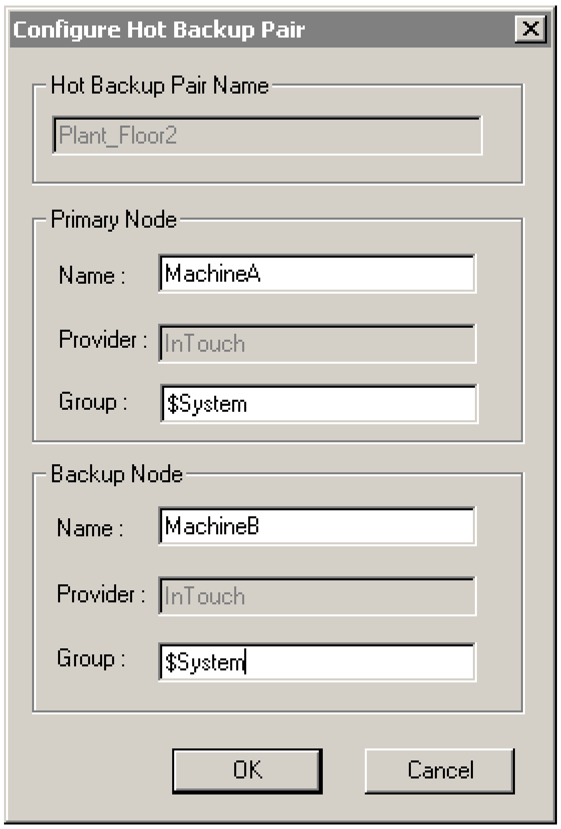
9. Click **OK**.

   Note You must execute the **Save** command on the **File** menu to save all your changes to the INI files. You will be prompted to save changes before exiting the utility.

To edit an existing Hot Backup pair

1. Start up WindowMaker.
2. Double-click the **Alarm Hot Backup Manager** item in the Application Explorer. The **Alarm Hot Backup Manager** dialog box appears.

3. Select the Hot Backup Pair Name that you want to modify and then click **Modify Pair**. The **Configure Hot Backup Pair** dialog box appears.



Configure Hot Backup Pair

Hot Backup Pair Name: Plant_Floor2

Primary Node
Name : MachineA
Provider : InTouch
Group : $System

Backup Node
Name : MachineB
Provider : InTouch
Group : $System

Note The node name for a particular application must be unique for the entire Hot Backup listing. An appropriate error message box appears if you enter wrong or duplicate entries.

For the node name:

- The node name must start with a letter
- You can use any alphanumeric character
- The only special character allowed is the underscore (_)
- No other special characters are allowed in any place in the node name

For the provider name:

- InTouch is the constant provider name and cannot be changed.

For the group name:

- All valid characters are allowed (A to Z, a to z, 0 to 9, !, @, -, ?, #, $, %, _, \, &).
- The group name must start with A to Z or a to z.
- $ is the only other character a group name can start with and if it is used, the group name can only be $System.

- You cannot use a backslash (\) at the end of a group name.
- The "\" character is used instead of the "!" character for separating a pair name from a group name.
- Consecutive backslashes (\\) are not allowed.

For more information on the fields in this dialog box, see the "To add a new Hot Backup pair" procedure.

4. Edit the necessary fields and then click **OK**.

Note You must execute the **Save** command on the **File** menu to save all your changes to the INI files. You will be prompted to save changes before exiting the utility.

To delete an existing Hot Backup pair

1. Start up WindowMaker.
2. Double-click the **Alarm Hot Backup Manager** item in the Application Explorer. The **Alarm Hot Backup Manager** dialog box appears.
3. Select the Hot Backup pair that you want to delete, then click **Delete Pair**. A message box appears asking you to confirm the deletion.
4. Click **Yes**.

Note You must execute the **Save** command on the **File** menu to save all your changes to the INI files. You will be prompted to save changes before exiting the utility.

Setting Key Fields for Alarm Records

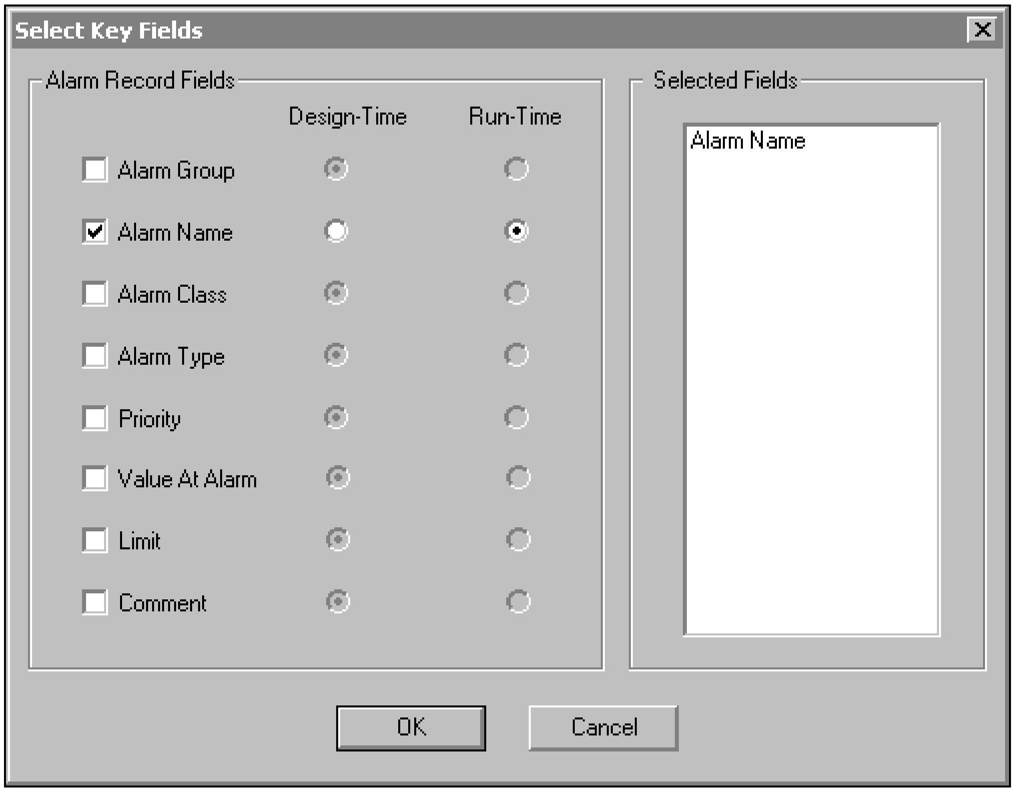
To achieve synchronization, you must identify a combination of alarm record fields such that the composite value of these fields uniquely generates a mapping key. The composite mapping key fields can be a mix of both design-time and runtime.

To select alarm record key fields

1. Start up the Alarm Hot Backup Manager Utility.

Tip You can start it up by double-clicking it in the Application Explorer in WindowMaker.

2. Select the Hot Backup Pair and then, click **Set Key Fields**. The **Select Key Fields** dialog box appears.



Alarm Record Fields	Design-Time	Run-Time
☐ Alarm Group	●	○
☑ Alarm Name	○	●
☐ Alarm Class	●	○
☐ Alarm Type	●	○
☐ Priority	●	○
☐ Value At Alarm	●	○
☐ Limit	●	○
☐ Comment	●	○

Selected Fields: Alarm Name

3. In the **Alarm Record Fields** group, select the fields that you want to include in the mapping key list. (The selected fields are added to the **Selected Fields** list box.)
4. Select the **Design-Time** option for the respective alarm record fields whose possible values are different and known at design time of the application. For example, in the case of an InTouch Provider, the "Name" field value is known at design time because it takes on the value of the tagnames defined in the primary and backup node applications. By default, **Alarm Name** as **Run-Time** is pre-selected.

   Note The **Alarm Record Fields** can be configured during runtime or design time. All design time fields are carried forward to the **Map Alarms Records** dialog box for any tagname mappings.

   For more information, see "Mapping Alarm Records."
5. Select the **Run-Time** option for the respective alarm record fields whose possible values are identical for both primary and backup providers. By default, **Alarm Name** as **Run-Time** is pre-selected.

Note At runtime, if the *design time key* is a part of the composite key, and the primary provider cannot find the mapping entry for that key in the mapping table, Hot Backup will automatically find the alarm with the same tagname in the backup provider and send an acknowledgement to the corresponding alarm and vice versa.

6. Click **OK**.

Note You must execute the **Save** command on the **File** menu to save all your changes to the INI files. You will be prompted to save changes before exiting the utility.

Mapping Alarm Records

You will need to map your Alarm Providers whenever the primary and secondary Alarm Providers are not identical applications. For example in a case where the primary is a "full-blown" application and the secondary is a "minimal" application, the names of the alarms might not be the same. Alarm Provider mapping establishes a correspondence between the primary and the secondary, so that when an alarm is acknowledged on one Provider, the Distributed Alarm System knows which alarm to acknowledge on the other.

Note The mappings are imported from a .csv (comma separated variable) file that you will need to create using any simple text file editor or Microsoft Excel. The alarm field names must match those contained in the **Map Alarms Records** dialog box. Otherwise, the import will abort and display a warning message box.

Each Hot Backup pair is identified by an application level unique name. For example:

Hot BackupPair Name	PrimaryNode!Provider	BackupNode!Provider
BoilerRoom1	NodeA!InTouch	NodeX!InTouch

Note To simplify the work for you, if no map is provided, the Distributed Alarm System assumes that the same alarms are on both the primary and the secondary.

You can combine the field values of alarm records - such as Group, name, and Priority - to generate a "composite mapping key" that uniquely identifies alarm records.

An InTouch Alarm Provider equates the "Name" field to the name of the tagname that generated the alarm. Therefore, when given Hot Backup pair, a mapping key can be generated using the combination of Group and "Name" field.

For example:

Provider Node	Backup Node
$System!TagA	$System!TagB

If a provider has a name field and comment field together as a unique field then the mapping key can be a combination of name and comment.

Provider Node	Backup Node
tagA!CommentA	tagB!CommentB

This could be true for any other field combination for a third provider.

There are different terms used in the Alarm Hot Backup Manager Utility "Selected Key Fields" dialog box and for the header names in the .csv file (the .csv file is used to create alarm mappings between primary and secondary alarm providers). The table below outlines the relationship: .

Selected "Key Fields" dialog box	.csv header names
Alarm Group	Group
Alarm Name	Name
Alarm Class	Class
Alarm Type	Type
Priority	Priority
Value at Alarm	Value
Limit	Limit
Comment	Comment

Note The terms shown in the right column should be used as header names when creating the .csv file for importing alarm mapping.

Important! "~" (tilde) should not be used in the .csv file for alarm mapping.

Important! When creating a .csv file for Excel, do not include "," since Excel will generate it automatically when the .csv file is being created.

Details Required for the Import File

The following situations will prevent a file from importing:

- The required number of columns should be filled with values for all the records at the import file. There should never be fewer values or more values for any record.
- The headings at the import file should be the same as that of the headings at the Map Alarm Records dialog box and should be in the same order.

For the following cases, if a record that is imported has a wrong entry, the user will be prompted to skip that particular record number, or to abort the importing process itself.

- The "group" column values should not have blank spaces.
- The "name" column values should not have blank spaces.

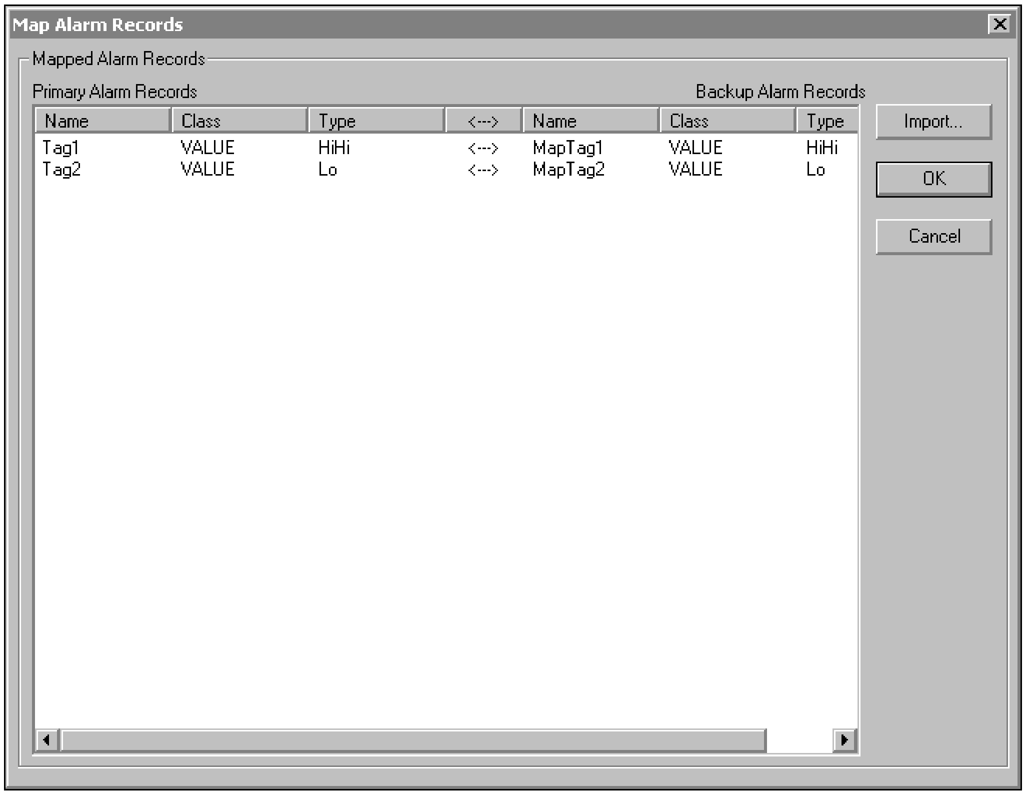
- The "Class" column values can never be any value other than VALUE, DEV, ROC, and DSC.
- The "Type" column values can never be any value other than LOLO, LO, HI, HIHI, MinDev, MajDev, ROC, and DSC.
- The "Priority" column values should be numbers from 1 to 999.
- The "Value" and "Limit" column values can be anything other than Null, when the "Class" or "Type" values for that particular record in that particular node are not known.
- The "Value" and "Limit" column values can accept only 1234567890.-+eE characters when the Class value for that particular record in that particular node is known as Value, Dev or ROC.
- The "Value" and "Limit" column values can accept only 1234567890.-+eE characters when the Type value for that particular record in that particular node is known as LOLO, LO, HI, HIHI, MinDev, MajDev or ROC.
- The "Value" and "Limit" column values can be anything other than Null, when any one of the "Class" or "Type" values for that particular record in that particular node is known as DSC.
- The Comment column values have no limitations.
- Duplicate records should not be present in the import file. If duplicate records are there, then they will be automatically skipped and the details will be given to the user after the import process is completed.

To map alarm records

1. Start up the **Alarm Hot Backup Manager** Utility.

Tip You can start it up by double-clicking it in the Application Explorer in WindowMaker.

2. Select the Hot Backup Pair and then, click **Map Alarms**. The **Map Alarm Records** dialog box appears listing your mapping information.



Name	Class	Type	<--->	Name	Class	Type
Tag1	VALUE	HiHi	<--->	MapTag1	VALUE	HiHi
Tag2	VALUE	Lo	<--->	MapTag2	VALUE	Lo

3. Click **Import**. The **Open** dialog box appears.
4. Locate and select the .csv file that you want to import.
5. Click **OK**.

Note You must execute the **Save** command on the **File** menu to save all your changes to the INI files. You will be prompted to save changes before exiting the utility.The user will be prompted for over-writing during a new import if there are records that have been already imported and displayed at this dialog.

Note No cross-validation between the fields "Alarm Class" and "Alarm Type" is done when CSV is imported in the Map Alarm Records dialog box.

Hot Backup Usage Example

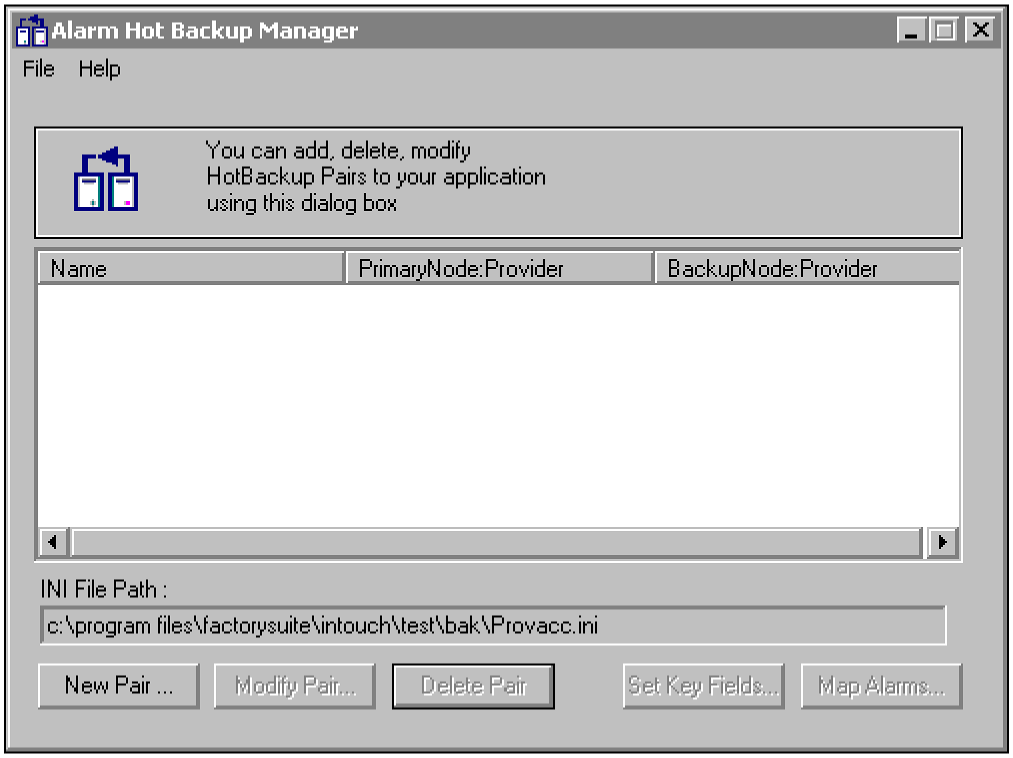
This section provides you with a sample user scenario for setting up and implementing Alarm Hot Backup Pairs.

Let's assume that you have three machines with InTouch installed on them. Two of the machines (**MachineA** and **MachineB**) are running "identical" InTouch Alarm Provider applications. **MachineA**'s application can generate two summary alarms with the names **Tag1** and **Tag2**. **MachineB**'s application can generate two logically identical alarms, **MapTag1** and **MapTag2**, respectively.

Your third machine is an InTouch Alarm Consumer that is configured for Hot Backup.

To configure the Hot Backup pairs

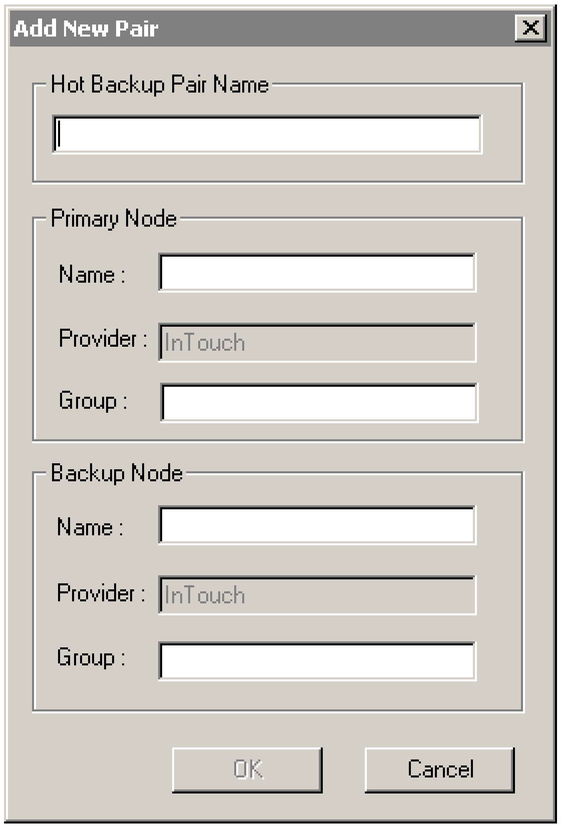
1. Create a new InTouch application on the client node.
2. Start up WindowMaker.
3. Double-click the **Alarm Hot Backup Manager** item in the Application Explorer. The **Alarm Hot Backup Manager** dialog box appears.



You can add, delete, modify HotBackup Pairs to your application using this dialog box

c:\program files\factorysuite\intouch\test\bak\Provacc.ini

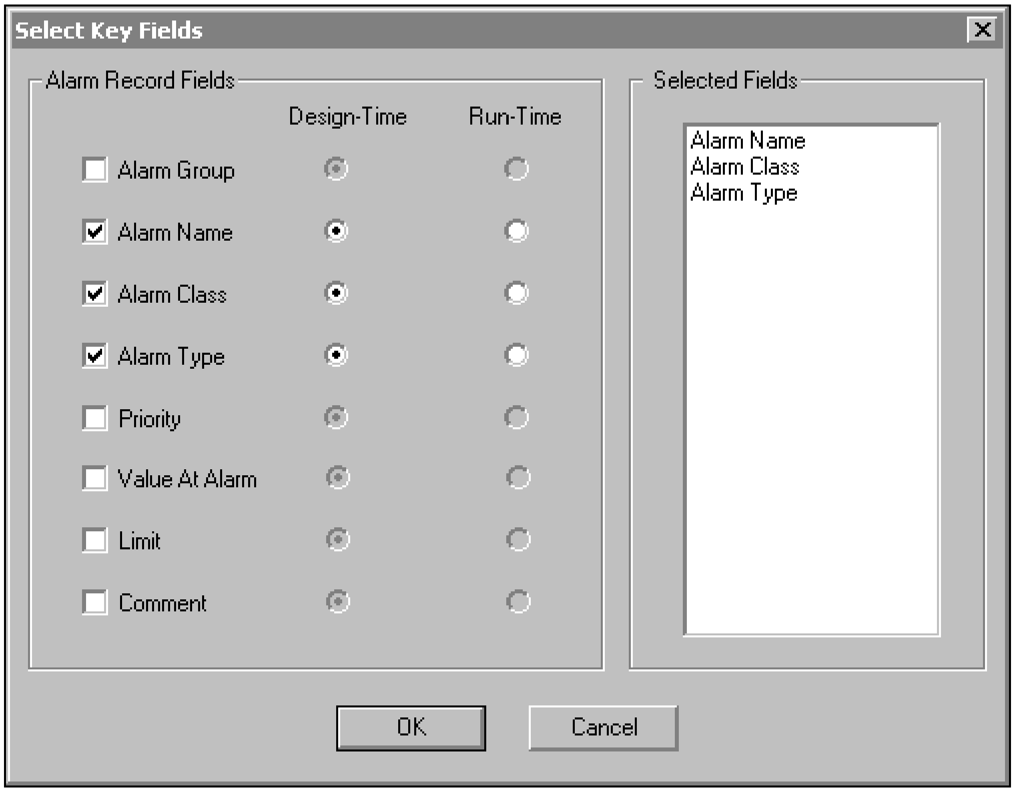
4. Click **New Pair.** The **Add New Pair** dialog box appears.



5. Type **Pair1** in **Hot Backup Pair Name** box.
6. Since our primary Alarm Provider InTouch application is running on **MachineA** with Alarm Group **$System** we configure the **Primary Node** group's fields as follows:
   - **Primary Node Name = MachineA**
   - **Primary Node Provider = InTouch**
   - **Primary Node Group is $System.**
7. Since our secondary Alarm Provider InTouch application is running on **MachineB** with Alarm Group **$System** we configure the **Backup Node** group's fields as follows:
   - **Backup Node Name = MachineB**
   - **Backup Node Provider = InTouch**
   - **Backup Node Group is $System.**
8. After this configuration is completed, we click **OK** to save our new backup pair. Next we need to map our key fields.

To map alarm record key fields

1. Double-click the Alarm **Hot Backup Manager** item in the Application Explorer. The **Alarms Hot Backup Manager** dialog box appears.
2. Select the Hot Backup pair, **Pair1** in the list and then, click **Set Key Fields**. The **Select Key Fields** dialog box appears.



	Design-Time	Run-Time
☐ Alarm Group	●	○
☑ Alarm Name	●	○
☑ Alarm Class	●	○
☑ Alarm Type	●	○
☐ Priority	●	○
☐ Value At Alarm	●	○
☐ Limit	●	○
☐ Comment	●	○

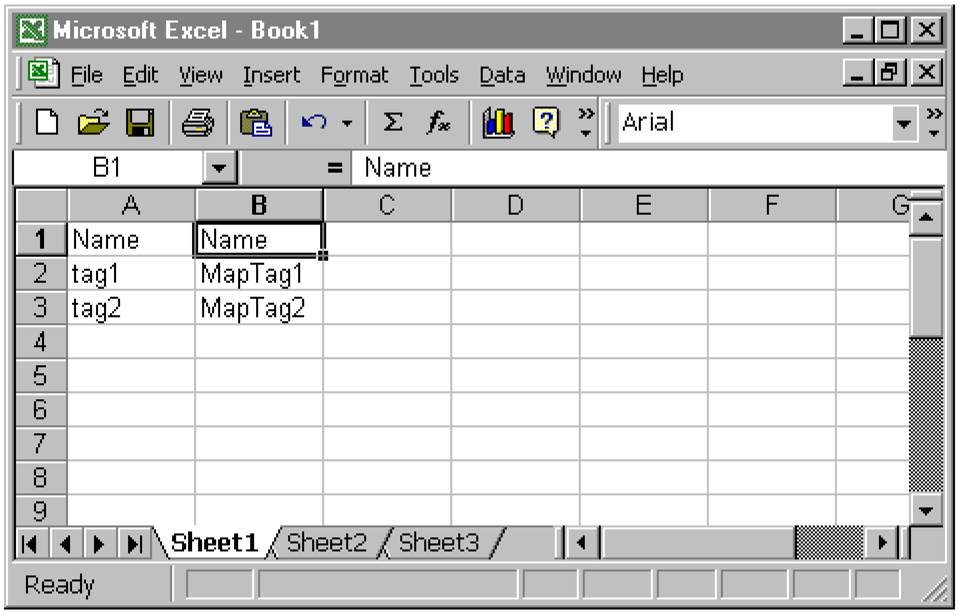
Selected Fields:
Alarm Name
Alarm Class
Alarm Type

3. Since we are mapping alarms via the alarm name because the values are different for the two providers, we select the **Alarm Name** and **Design-Time** options.
4. Next we click **OK** to save our configuration. A **Hot Backup Configuration** message box appears prompting us to save our configuration.
5. We click **Yes**. We are now ready to create our .csv file that will contain all of our alarm record mappings.

To create an alarm mapping .CSV file

In our user scenario we have three machines with InTouch installed on them. Two of the machines (**MachineA** and **MachineB**) are running "identical" InTouch Alarm Provider applications. **MachineA**'s application can generate two summary alarms with the names **Tag1** and **Tag2**. **MachineB**'s application can generate two logically identical alarms, **MapTag1** and **MapTag2**, respectively.

Therefore, we created a .csv file named **pair1.csv** that contains our alarm record mappings using Microsoft Excel(. (The mappings are imported from this .csv file.) The alarm record mapping ensures that when an alarm is acknowledged on one Alarm Provider, the Distributed Alarm System knows which alarm to acknowledge on the other Alarm Provider.



	A	B
1	Name	Name
2	tag1	MapTag1
3	tag2	MapTag2

Note The alarm field names must match those contained in the **Map Alarms Records** dialog box. Otherwise, the import will abort and display a warning message box.

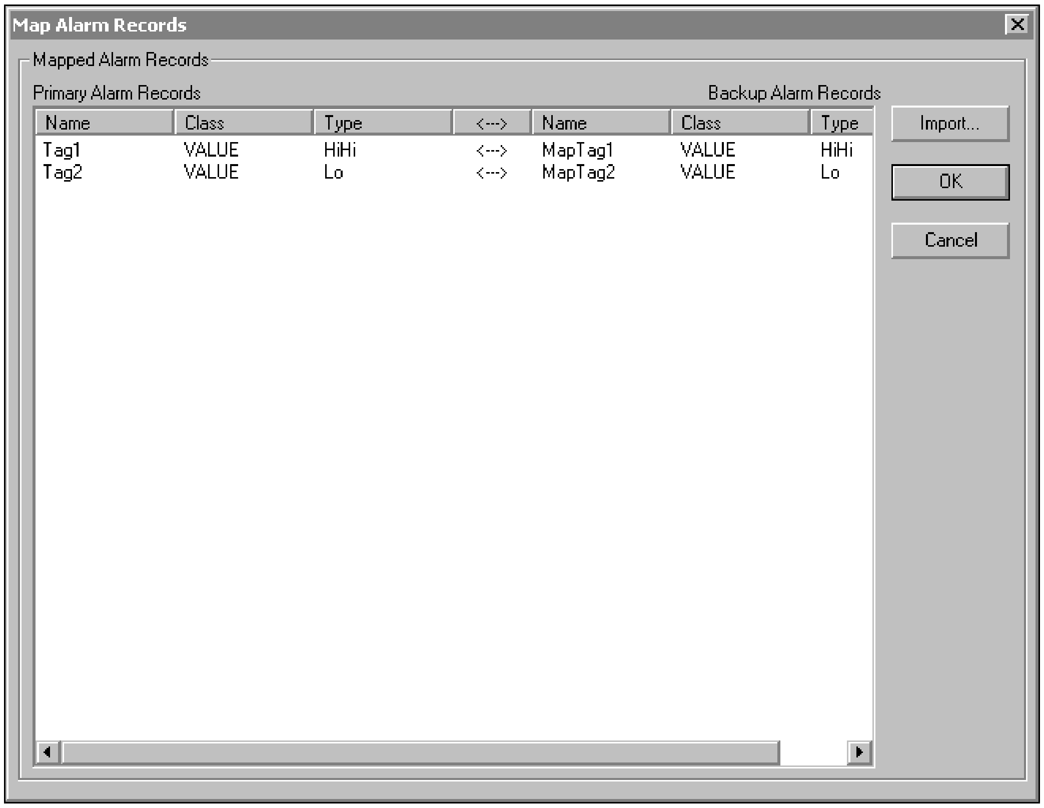
For more information, see "Mapping Alarm Records."

We are now ready to import our .csv file containing all of our alarm record mappings.

To import alarm record mapping .csv file

1. Start up WindowMaker.
2. Double-click the **Alarm Hot Backup Manager** item in the Application Explorer. The **Alarm Hot Backup Manager** dialog box appears.
3. We select **Pair1** in the list and then, click **Map Alarms**. A blank Map Alarm Records dialog box appears.

4. Next we click **Import**. The **Open** dialog box appears. We locate and select our .csv file (**pair1.csv**), and then click **Open**. The Distributed Alarm System begins the importing process.



Name	Class	Type	<--->	Name	Class	Type
Tag1	VALUE	HiHi	<--->	MapTag1	VALUE	HiHi
Tag2	VALUE	Lo	<--->	MapTag2	VALUE	Lo

5. Once the importing process is complete, click **OK**.

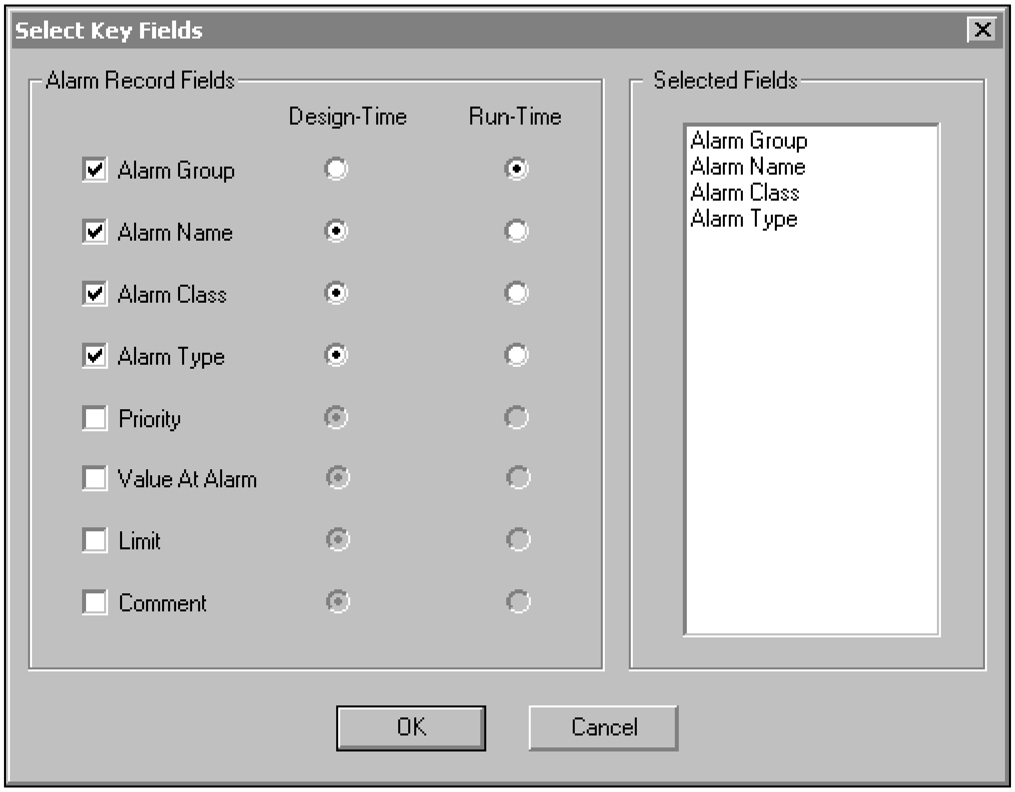
Now it's ready to run Hot Backup applications.

To run Hot Backup

1. Start up both Alarm Provider applications.
2. At runtime, generate the alarms **Tag1, Tag2, MapTag1, MapTag2**.
3. In the client application, create a Distributed Alarm Display object with a **Pair1** query (the name of our Hot Backup pair).
4. Run the client application displaying the window containing our Distributed Alarm Display object.
5. The Distributed Alarm Display object is now displaying alarms of Machine A.
6. At the primary Alarm Provider, **MachineA**, acknowledge an alarm such as **Tag2**.
7. The corresponding mapped alarm, **MapTag2**, on the secondary Alarm Provider, **MachineB** is automatically acknowledged.

Ack Synchronization Example

Let’s assume that **Alarm Name**, **Alarm Class** and **Alarm Type** have been selected as **Design-Time** Key Fields, and **Alarm Group** has been selected as a **Run-Time** field in the **Select Key Fields** dialog box.



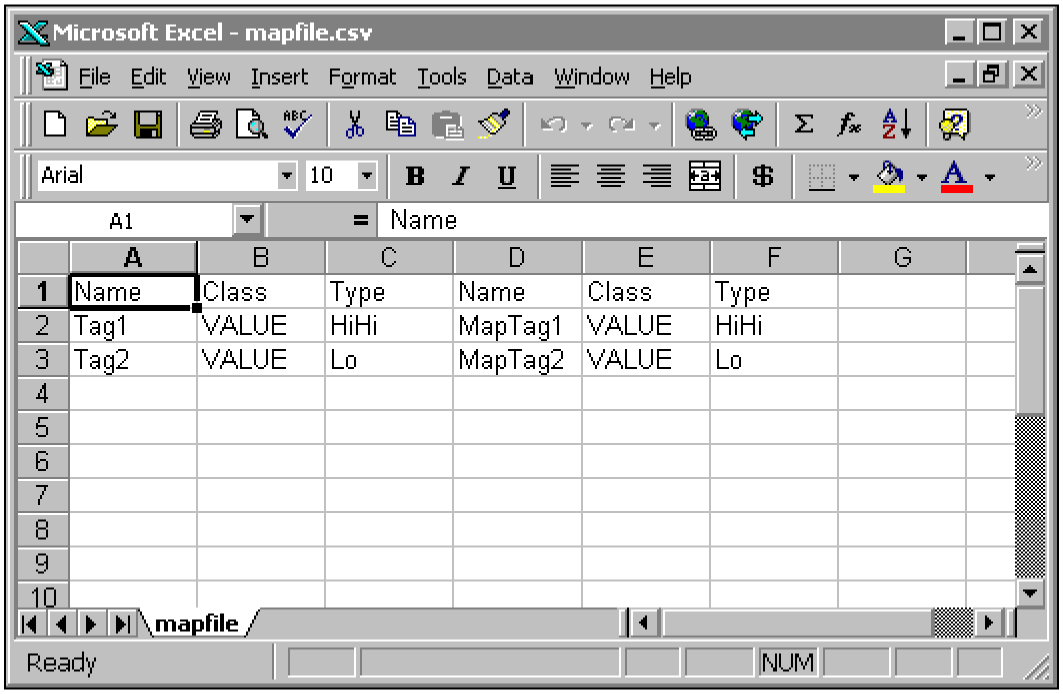
Alarm Record Fields

	Design-Time	Run-Time
✓ Alarm Group		●
✓ Alarm Name	●	
✓ Alarm Class	●	
✓ Alarm Type	●	
Priority		
Value At Alarm		
Limit		
Comment		

Selected Fields

Alarm Group
Alarm Name
Alarm Class
Alarm Type

And a corresponding alarm record mapping .csv file (**Mapfile.csv**) has been created using Microsoft Excel.


Microsoft Excel - mapfile.csv
File Edit View Insert Format Tools Data Window Help
Arial 10
A1 = Name
mapfile
Ready
NUM

	A	B	C	D	E	F	G
1	Name	Class	Type	Name	Class	Type	
2	Tag1	VALUE	HiHi	MapTag1	VALUE	HiHi	
3	Tag2	VALUE	Lo	MapTag2	VALUE	Lo	
4							
5							
6							
7							
8							
9							
10							

Upon importing the **Mapfile.csv** file via the **Map Alarm Records** dialog box, the alarm named **tag1** with a Class of **VALUE** and a Type of **HIHI** is mapped to the alarm name **maptag1** with a Class of **VALUE** and a Type of **HIHI**. Therefore, whenever **HiHi** alarm of **Tag1** is acknowledged at the Primary alarm node the acknowledgment also appears on the **HiHi** alarm of **MapTag1** at the secondary Alarm Provider node and the client node(s), provided the **Alarm Group** names (runtime mapped field) of both **Tag1** at the primary node, and **MapTag1** at the secondary node, remain the same.

Conversely, if **Lo** alarm of **MapTag2** is acknowledged on the secondary Alarm Provider node, **Lo** alarm of **Tag2** is automatically acknowledged on the client and the primary Alarm nodes, provided the Alarm Group names (runtime mapped field) of both **MapTag2** and **Tag2** remain the same.

Acknowledgement synchronization will happen only if the **Design-time** and **Run-time** mapping are matched.

Note You can choose any combination of **Design-Time** and **Run-time** alarm record fields for mapping. However, it is important that you ensure that the mappings do not result in multiple references.

For example, if two **Alarm Record Fields** such as **Class** and **Priority** are selected in the **Select Key Fields** dialog box, it is very possible that more than one alarm will match the criteria. In such a case, the Hot Backup synchronization is not guaranteed to work. In the process of propagating the acknowledgment, a random alarm that matches the criterion may also be picked up, while other matching alarms may be left unacknowledged.

Distributed Alarm Database Views

This section describes database views to be provided by InTouch 8.0. These views provide easy-to-query virtual tables that permit analysis of past and current alarm and event occurrences. The primary purpose of these views is to allow analysis of alarm and event history in a way that is supported by Wonderware long-term (through subsequent product releases), regardless of changes and differences in the actual database schemas over those releases. Finally, a view is provided for backward compatibility with the AlarmSuite product. All views are available after install of the product and creation of the alarm log database.

Introduction to Views and Stored Procedures

The InTouch Distributed Alarm system provides a set of tools to allow viewing and analysis of historical alarm and event information that has been logged to the relational database by the Alarm DB Logger. These capabilities are provided in the form of database views and stored procedures.

A database view can be thought of as single logical table that combines information from multiple, underlying database tables in an easy to use format. Views exist within the database. Database views are often referred to as virtual database tables because they can be queried using standard SQL just as if they were real tables. When a view is queried, it returns a set of records (or rows). Each row has several columns of information that contain the data for the record.

The database views for the distributed alarm system provide a way of viewing historical alarms and events that have occurred in your automation application. Perform complex analysis on these views by performing SQL commands on them, using sophisticated filter criteria. So, for example, you could retrieve alarm records for all HiHi alarms that occurred during a particular time span in a particular area of the plant. Or, you could retrieve all data change events that were logged by a certain alarm provider node.

A stored procedure is a collection of SQL statements that work together to perform a certain function and return data from the database. As such, they can be thought of as a subroutine or function. Stored procedures can accept input parameters that govern how they are to operate, and they return data in the form of a result set. The user of the stored procedures does not have to use SQL syntax. Instead, the name of the stored procedures and the input parameters are all that are required to execute the procedure. The returned results are a set of records and appear as a SQL record set very similar to a view. Stored procedures have the additional benefit of improving performance since they exist in the database with their embedded SQL statements and reduce roundtrips back to the client.

The stored procedures provided with the InTouch Distributed Alarm system allow the user to extract information from a database without having to know the details of SQL. Internally, the stored procedures do the hard work of formulating the proper SQL queries and joins and doing the complex logic.

Both the stored procedures and views have the advantage of shielding the user from the internal details of the database and making it very easy to retrieve information that is required for analysis and reporting of alarms and events.

The following sections define the provided views and stored procedures along with easy to follow samples that exhibit their flexibility and power.

View and Stored Procedure Column Definitions

The stored procedures and views return tables of information. Each table contains a set of columns. This section provides the definition of all the columns that are returned by the views and stored procedures to be described later.

Column name	Description
AlarmCount	Integer indicating the number of onsets of the alarm during the specified time span. If alarm already existed prior to and entering the selected time range, it is not counted.
AlarmState	Unicode string indicating the state of the alarm such as UNACK_ALM (unacknowledged-in-alarm), UNACK_RTN (unacknowledged-return-to-normal), ACK_ALM (acknowledged-in-alarm), ACK_RTN (acknowledged-return-to-normal). The ACK state represents acknowledge simultaneous with RTN.
AlarmType	See Type.
Area	Unicode string indicating the name of the alarm group or area to which the tagname or object that generated the alarm or event belongs.
Category	Unicode string indicating the class or category of the alarm or event, such as Value, Dev, ROC, etc.
CheckValue	Unicode string indicating the alarm limit value for alarms or the previous value for events.
Comment	Unicode string indicating the comment for the alarm.
Description	Unicode string value indicating the description string of the alarm or of the object generating the alarm. For acknowledge events, it is the acknowledge comment.
Domain Name	Name of domain.
UNACKDuration	Number of milliseconds for UNACK Duration. For UNACK Duration, this represents the time between the most recent alarm transition (ALM or sub-state) and the ACK, if any. For Alarm Duration, this represents the time between onset of the alarm and return to normal.
EngUnits	Engineering units value. Future. Returned as "" in 8.0.
EventCount	Number of times the event occurred in the specified time span.
EventStamp	Datetime value indicating the date and time of the alarm or event, provided in local time.
EventStampUTC (GMT)	Datetime value indicating the date and time of the alarm or event, provided in coordinated universal time.
GroupName	See Area.
LastEvent (Future)	Datetime value indicating the date and time of last transition for this alarm, in local time.

Column name	Description
Limit	Unicode string indicating the alarm limit value of the alarm variable at the time of the alarm or event. Returned as float type for view v_AlarmSuiteAlarmLog.
NodeName	Unicode string indicating the node name that provided the alarm or event to the distributed alarm system
Operator	Unicode string indicating the name of the operator associated with the alarm or event.
Priority	Integer indicating the alarm priority value (1-999).
Provider	Unicode string indicating the application or component that provided the alarm or event to the distributed alarm system.
TagName	Unicode string indicating the name of the object generating the alarm or event.
TimeInState (Future)	Integer value indicating the time the alarm has been in the current alarm state in seconds.
Type	Unicode string indicating the type of the alarm within the category, such as HiHi, LoLo, etc.
Units	Engineering units string. Future. Returned as "" in 8.0.
User Full Name	Full name of user in operator (eg. Joseph P. Smith).
Value	Unicode string indicating the current value of the variable at the time of the alarm or event. Returned as float type for view v_AlarmSuiteAlarmLog.
ValueString	Unicode string indicating the current value of the variable at the time of the alarm or event.

To View the Definition of a View in Enterprise Manager

1. Expand a server group, and then expand a server.
2. Expand **Databases**.
3. Expand the database in which the View belongs.
4. Click **Views**.
5. In the details pane, right-click the stored procedure.
6. Click **Properties**

To Call a View

Unlike Stored Procedures, Views do not take parameters. To view data in a View, do the following:

1. In the Enterprise Manager, expand a server group.
2. Expand a server.
3. Expand **Databases**.
4. Expand the database in which the View belongs.
5. Click **Views**.
6. In the details pane, right-click the View, and then select **Open Rows**.
7. Click on **All Rows.**

Alarm History Database Views

This view provides a historical list of all alarms and alarm transition events that occurred over a selected time range. The query specifies start and end date and time (by column EventStamp or EventStampUTC). The returned records include alarm origination, alarm acknowledge, alarm enable, alarm disable, and alarm return-to-normal events. Notice that all strings are Unicode in the views.

v_AlarmHistory

Column Name	Datatype	Description
EventStamp	Datetime	Date and time of alarm event (in local time of database)
AlarmState	nChar	State of alarm: one of UNACK, UNACK_RTN, ACK, ACK_RTN, DISABLED (future), ENABLED (future)
TagName	nChar	Name of the object that generated the alarm: such as TIC101
Description	nVarchar	Description string of the alarm. Can default to object description (or comment in InTouch). Or Acknowledge comment for ack records.
Area	nChar	Name of the Area or Group for the alarm.
Type	nChar	Type of alarm: such as Hi, HiHi, ROC, PV.HiAlarm
Value	nChar	Value of alarm variable at time of alarm.
CheckValue	nChar	Value of alarm limit at time of alarm.
Priority	Integer	Alarm priority.
Category	nChar	Alarm class or alarm category. Such as Value, Dev, ROC, Process, Batch, System, etc.
Provider	nChar	Provider of alarm: node/InTouch, or GalaxyName.
Operator	nChar	Name of operator: JoeR (if any).
Domain Name	nChar	Name of domain.
User Full Name	nChar	full name of user in operator (e.g., Joseph P. Smith).
UNACK Duration	Float	The time between the most recent alarm transition (ALM or sub-state) and the ACK, if any.
User1	Float	User-defined field number 1.
User2	Float	User-defined field number 2.
User 3	nChar	User-defined field, string.
EventStampUTC	DateTime	UTC date/ time of alarm event
Millisec	Small Int	Fractional seconds for event stamp in increments of 0.1 msec.

SQL Statements in ’SQL Query Analyzer’ can also be written to view data in a View. For example, Query window, type

```
Select top 100 * from v_AlarmHistory where Priority>10
   AND((Provider LIKE ’%adelphi%’)AND (Area LIKE ’%$s%’))
```

-- select all records from v_AlarmHistory view

```
Select * from v_AlarmHistory
```

-- select all records from v_AlarmHistory view with Priority greater than 100

```
Select * from v_AlarmHistory WHERE Priority >=100
```

Highlight (Select) the above text and press **F5**. The resultant data will be displayed in the bottom portion of the Window.

For more information on SQL Statements, refer SQL Server Manuals.
v_AlarmHistory2

Column Name	Datatype	Description
EventStamp	Datetime	Date and time of alarm event (in local time of database)
AlarmState	nChar	State of alarm: one of UNACK, UNACK_RTN, ACK, ACK_RTN, DISABLED (future), ENABLED (future), ACK_ALM, UNACK_ALM
TagName	nChar	Name of the object that generated the alarm: such as TIC101
Description	nVarchar	Description string of the alarm. Can default to object description (or comment in InTouch). Or Acknowledge comment for ack records.
Area	nChar	Name of the Area or Group for the alarm.
Type	nChar	Type of alarm: such as Hi, HiHi, ROC, PV.HiAlarm
Value	nChar	Value of alarm variable at time of alarm.
CheckValue	nChar	Value of alarm limit at time of alarm.
Priority	Integer	Alarm priority.
Category	nChar	Alarm class or alarm category. Such as Value, Dev, ROC, Process, Batch, System, etc.
Provider	nChar	Provider of alarm: node/InTouch, or GalaxyName.
Operator	nChar	Name of operator: JoeR (if any).
Domain Name	nChar	Name of domain.
User Full Name	nChar	full name of user in operator (e.g., Joseph P. Smith).
Alarm Duration	Float	The time between onset of the alarm and return to normal.
User1	Float	User-defined field number 1.
User2	Float	User-defined field number 2.
User 3	nChar	User-defined field, string.
EventStampUTC	DateTime	UTC date/ time of alarm event
Millisec	Small Int	Fractional seconds for event stamp in increments of 0.1 msec.

SQL Statements in 'SQL Query Analyzer' can also be written to view data in a View. For example, Query window, type

```
Select top 100 * from v_AlarmHistory2 where Priority>10
   AND((Provider LIKE '%adelphi%')AND (Area LIKE '%$s%'))
```

-- select all records from v_AlarmHistory2 view

```
Select * from v_AlarmHistory2
```

-- select all records from v_AlarmHistory2 view with Priority greater than 100

```
Select * from v_AlarmHistory2 WHERE Priority >=100
```

Highlight (Select) the above text and press **F5**. The resultant data will be displayed in the bottom portion of the Window.

For more information on SQL Statements, refer SQL Server Manuals.

Event History Database Views

This database view provides a historical list of all events, excluding alarms that occurred over a selected time range. The query client specifies start and end date and time. The returned records include all non-alarm events.

v_EventHistory

Column Name	Datatype	Description
EventStamp	Datetime	Date and time of event.
TagName	nChar	Name of the object that generated the event, such as Pump1.
Description	nVarChar	Description string of the event. Can default to object description, or comment in InTouch.
Area	nChar	Name of the Area or Group for the event.
Type	nChar	Type of event, such as "Operator data change", "Startup", etc.
Value	nChar	New Value (if any).
CheckValue	nChar	Old Value (if any).
Category	nChar	Event category or class, such as Value, Process, Batch, System, etc.
Provider	nChar	Generator of event, such as node/InTouch, or View Engine name for user change.
Operator	nChar	Name of operator1: JoeR (if any).
Domain Name	nChar	Name of domain.
User Full Name	nChar	full name of user in operator (e.g., Joseph P. Smith).
User1	Float	User-defined field number 1.
User2	Float	User-defined field number 2.
User 3	nChar	User-defined field, string.
EventStampUTC	DateTime	UTC date/ time of event.
Millisec	Small Int	Fractional seconds for event stamp in increments of 0.1 msec.

Alarm-Event History Database View

This database view provides a historical list of all events and alarms that occurred over a selected time range. The query client specifies start and end date and time. The returned records include all alarms and events. This view combines the alarm view and event view into one and represents the union of the records from those views.

v_AlarmEventHistory

Column Name	Datatype	Description
EventStamp	Datetime	Date and time of event.
AlarmState	nChar	State of alarm: one of UNACK, UNACK_RTN, ACK, ACK_RTN, DISABLED (future), ENABLED (future). Does not apply for events.
TagName	nChar	Name of the object that generated the alarm, such as TIC101.
Description	nVarchar	Description string of the alarm/event. Can default to object description (or comment in InTouch). Or acknowledge comment for ack records.
Area	nChar	Name of the Area or Group for the alarm.
Type	nChar	Type of alarm or event, such as Hi, HiHi, ROC, PV.HiAlarm, Operator data change, etc.
Value	nChar	Value of alarm variable at time of alarm.
CheckValue	nChar	Value of alarm limit at time of alarm, or old value for event.
Priority	Integer	Alarm priority.
Category	nChar	Alarm or event class, or alarm category, such as Value, Process, Batch, System, etc.
Provider	nChar	Provider of alarm, such as node/InTouch, or GalaxyName.
Operator	nChar	Name of ack operator or data change operator.
Domain Name	nChar	Name of domain.
User Full Name	nChar	full name of user in operator (e.g., Joseph P. Smith).
UNACK Duration	Float	Number of milliseconds from the most recent alarm transition to ACK.
User1	Float	User-defined field number 1.
User2	Float	User-defined field number 2.
User 3	nChar	User-defined field, string.
EventStampUTC	DateTime	UTC date/ time of event
Millisec	Small Int	Fractional seconds for event stamp in increments of 0.1 msec.

SQL Statements in ’SQL Query Analyzer’ can also be written to view data in a View. For example:

To:

Select TagName, Area and Type columns of all records from v_AlarmEventHistory view with TagName "MyTag1" and AlarmState ACK_RTN or ACK and order by Provider

Type in the Query window as follows:

```
Select TagName,Area,Type FROM v_AlarmEventHistory WHERE
   TagName='MyTag1' AND (AlarmState='ACK_RTN' OR
   AlarmState='ACK') ORDER BY Provider
```

Highlight (select) the previous text and press **F5**. The resulting data will be displayed in the bottom portion of the Window. For more information on SQL Statements, refer to the SQL Server Manuals.

v_AlarmEventHistory2

Column Name	Datatype	Description
EventStamp	Datetime	Date and time of event.
AlarmState	nChar	State of alarm: one of UNACK, UNACK_RTN, ACK, ACK_RTN, DISABLED (future), ENABLED (future). Does not apply for events.
TagName	nChar	Name of the object that generated the alarm, such as TIC101.
Description	nVarchar	Description string of the alarm/event. Can default to object description (or comment in InTouch). Or acknowledge comment for ack records.
Area	nChar	Name of the Area or Group for the alarm.
Type	nChar	Type of alarm or event, such as Hi, HiHi, ROC, PV.HiAlarm, Operator data change, etc.
Value	nChar	Value of alarm variable at time of alarm.
CheckValue	nChar	Value of alarm limit at time of alarm, or old value for event.
Priority	Integer	Alarm priority.
Category	nChar	Alarm or event class, or alarm category, such as Value, Process, Batch, System, etc.
Provider	nChar	Provider of alarm, such as node/InTouch, or GalaxyName.
Operator	nChar	Name of ack operator or data change operator.
Domain Name	nChar	Name of domain.
User Full Name	nChar	full name of user in operator (eg, Joseph P. Smith).
Alarm Duration	Float	Number of milliseconds from the onset of alarm to RTN.
User1	Float	User-defined field number 1.
User2	Float	User-defined field number 2.
User 3	nChar	User-defined field, string.
EventStampUTC	DateTime	UTC date/ time of event
Millisec	Small Int	Fractional seconds for event stamp in increments of 0.1 msec.

SQL Statements in 'SQL Query Analyzer' can also be written to view data in a View. For example:

To:

Select TagName, Area and Type columns of all records from v_AlarmEventHistory view with TagName "MyTag1" and AlarmState ACK_RTN or ACK and order by Provider

Type in the Query window as follows:

```
Select TagName,Area,Type FROM v_AlarmEventHistory WHERE
   TagName='MyTag1' AND (AlarmState='ACK_RTN' OR
   AlarmState='ACK') ORDER BY Provider
```

Highlight (select) the previous text and press **F5**. The resulting data will be displayed in the bottom portion of the Window. For more information on SQL Statements, refer to the SQL Server Manuals.

AlarmCounter Database Stored Procedure

This database stored procedure provides the count of the number of alarm occurrences for each alarm that occurred in a time interval. The query client must specify start and end date and time. It has five optional parameters: TagName, Class, Type, Provider and Comment. The counter only applies to the number of alarm originations (not transitions such as acknowledges or returns-to-normal). So, for example, if an alarm occurred and then was acknowledged and then returned to normal, the count for that alarm is only 1 (not 3). The purpose of this view is to show frequency of alarms. A unique alarm is identified by its TagName, Provider, Type, and Category. An example question answered by this view: How many times did object TIC101 (TagName) on provider Node1|InTouch (Provider) go into a Value alarm (Category) that was HiHi (Type) during a time span?

sp_AlarmCounter

Note The alarm counter is applicable to only the Detailed Mode of logging and is not supported for the Consolidated mode of logging.

Column Name	Datatype	Description
TagName	Nchar	Name of the object that generated the alarm, such as TIC101.
Area	Nchar	Name of the Area or Group for the alarm.
Type	Nchar	Type of alarm, such as Hi, HiHi, ROC, PV.HiAlarm.
Category	Nchar	Alarm class or alarm category, such as Value, Process, Batch, etc.
AlarmCount	Integer	Number of onsets of the alarm during the time range. If alarm existed prior to entering the selected time range, do not count that one.
Priority	Integer	Alarm priority.
Provider	Nchar	Provider of alarm, such as node/InTouch, or GalaxyName.
Comment	Nchar	

Viewing the Definition of a Stored Procedure in Enterprise Manager

To view the definition of a stored procedure in Enterprise Manager:

1. Expand a server group, and then expand a server.
2. Expand **Databases**.
3. Expand the database in which the stored procedure belongs.
4. Click **Stored Procedures**.
5. In the details pane, right-click the stored procedure, then click **Properties**.

Calling a Stored Procedure

A stored procedure can be called from SQL Server using the Transact-SQL statement EXECUTE.

An example of calling a stored procedure:

In the ’SQL Query Analyzer’ Query window, type:

```
EXECUTE sp_AlarmCounter @StartDate=’2001-01-01’,
   @EndDate=’2001-03-31’, @Tagname = ’tag1’, @Type = ’LO’,
   @Provider = ’WW21353\InTouch’, @Comment = ’SSAADD’
```

Highlight (Select) the text and press **F5**. The results will be displayed in the bottom portion of the Window.

StartDate and EndDate are the two parameters to the sp_AlarmCounter stored procedure that are compulsory. The rest of the five parameters are optional. This means you can skip any of the other five parameters apart from the StartDate and EndDate. When a particular parameter is skipped, it will not be used for filtering and a general result will be displayed which will contain all the matches for the parameter skipped.

StartDate and EndDate are SQL Server, ’datetime’ type variables which can take data in various formats as specified in the *SQL Server Users Guide*. The desired time duration within which the Stored Procedure has to be executed should be passed using these two parameters.

EventCounter Database Stored Procedure

This database stored procedure provides the count of the number of events of a certain type on a certain tagname that occurred in a time interval. The query client must specify start and end date and time. It has three optional parameters: TagName, Provider and Comment. The counter applies only to non-alarm events.   The purpose of this view is to show frequency of each event. For example, how many times did the pump turn on? The TagName, Provider, Category and Type are used to uniquely identify an event and do the counting.

sp_EventCounter

Column Name	Datatype	Description
TagName	NChar	Name of the object that generated the alarm, such as TIC101
Area	NChar	Name of the Area or Group for the alarm.
Type	NChar	Type of event.
Category	NChar	Alarm class or alarm category, such as Value, Process, Batch, etc.
EventCount	Integer	Number of times the event of this Type for the TagName has occurred in the specified time range.
Provider	NChar	Provider of event: node/InTouch, or GalaxyName.
Comment	NChar	

Similarly, the sp_EventCounter stored procedure also takes two parameters that are compulsory, namely StartDate and EndDate and three optional parameters Tagname, Provider and Comment.

```
EXECUTE sp_AlarmCounter @StartDate='2001-01-01 23:23:23',
   @EndDate='2001-03-31 23:23:23', @Tagname = '$NewAlarm'
```

AlarmSuite Alarm Log Database View

This database view provides a view of historical alarms and events that returns a table that has the same columns as the actual Alarm Suite table called AlarmLog. Queries that work against the AlarmSuite AlarmLog table also work against this view, except that the table name in those queries must be changed to the view name.

v_AlarmSuiteAlarmLog

Column Name	Datatype	Description
EventStamp	Datetime	Date and Time of the event.
EventType	NChar	Type of the event, as defined by AlarmSuite.
AlarmType	NChar	Type of the alarm.
AlarmState	NChar	Ack state of the alarm.
NodeName	NChar	Node of alarm.
TagName	NChar	Name of the object reporting alarm.

Column Name	Datatype	Description
GroupName	NChar	Name of alarm group or area.
Comment	NChar	Acknowledge comment (if any).
Value	Float	Value of alarm variable
Limit	Float	Value of alarm limit at time of alarm.
ValueString	NChar	See Alarm Suite.
Operator	NChar	Operator name
Priority	Integer	Priority
Units	NChar	Return as "" in 7.11.

SQL Statements in ’SQL Query Analyzer’ can also be written to view data in a View. For example, Query window, type:

```
-- select all distinct tagnames from the
   v_AlarmSuiteAlarmLog view
```

Select Distinct TagName From v_AlarmSuiteAlarmLog

Highlight (Select) the above text and press F5. The resulting data will be displayed in the bottom portion of the Window.

For more information on SQL Statements, refer SQL Server Manuals.

C H A P T E R 1 0

Alarm/Event Clients

InTouch provides an Alarm Viewer ActiveX Control with built-in scroll bars, sizable display columns, multiple alarm selections, an update status bar, dynamic display types, and display colors based on alarm priority. The Alarm Viewer ActiveX Control is the preferred control for viewing alarms. However, InTouch also includes a distributed alarm display, which facilitates viewing existing alarms without reconfiguring them to work with the AlarmViewer ActiveX Control.

InTouch provides an Alarm DB View ActiveX Control that allows you to visualize the alarm data from the new Alarm DB Logger database. This control is used to view all alarm and event information.

InTouch also provides InTouch QuickScript functions that provide dynamic control over the alarm display and alarm acknowledgment. The use of QuickScript functions is described for each control.

Contents

Alarm Viewer ActiveX Control Guidelines

InTouch provides an Alarm Viewer ActiveX Control that allows you to view all locally and remotely generated alarms. You design the appearance of the ActiveX Control and the data displayed by specifying the following attributes:

- Context sensitive menu features
- Display mode
- List control options
- Colors for different properties
- Font type, style and size

- Alarm selection (filters)
- Query filters
- Column management
- Sorting

Once the control format has been designed, a user may have the ability to make the following adjustments to manipulate the data they are viewing:

- Sort the information by column
- Update the display
- Perform a query
- Resize the width of a column

The Alarm Viewer ActiveX Control can be dropped into WindowMaker, resized, and positioned. Configuration of the Alarm Viewer ActiveX control is done using the Property sheets. The alarm data can then be viewed in the Alarm Viewer ActiveX Control view window.

Alarm Viewer ActiveX Display Guidelines

The Alarm Viewer ActiveX display is an additional alarm viewer control wizard. While it may appear like the old standard alarm display wizard, it uses the same mechanism as the Window Control wizards. This mechanism requires certain guidelines to be observed when using objects such as the Alarm Viewer ActveX display. These guidelines are as follows:

Each display must have an identifier so that any associated QuickScript function knows which display to modify. This identifier, entered as **Control Name** in the **Alarm Viewer ActiveX Properties** dialog box, must be unique for each display.

Displays should not overlap other InTouch objects such as window controls or graphic objects. You can easily verify this by clicking on the Alarm Viewer ActiveX display in WindowMaker, and checking the display's "handles." The handles should not touch other graphic objects on the screen.

Displays should be used sparingly. Placing numerous displays on one screen can result in reduced system performance. When possible, limit the number of displays on your screen and call further screens with additional displays if necessary.

Installing the Alarm Viewer ActiveX Control

This Alarm Viewer ActiveX control is installed when InTouch is installed.

To paste this control in a WindowMaker window

1. Open the Wizard selection dialog box.
2. Select **AlmViewerCtrl** under ActiveX Controls, then click **OK**.
3. Paste the control on the window and resize it to the required size.

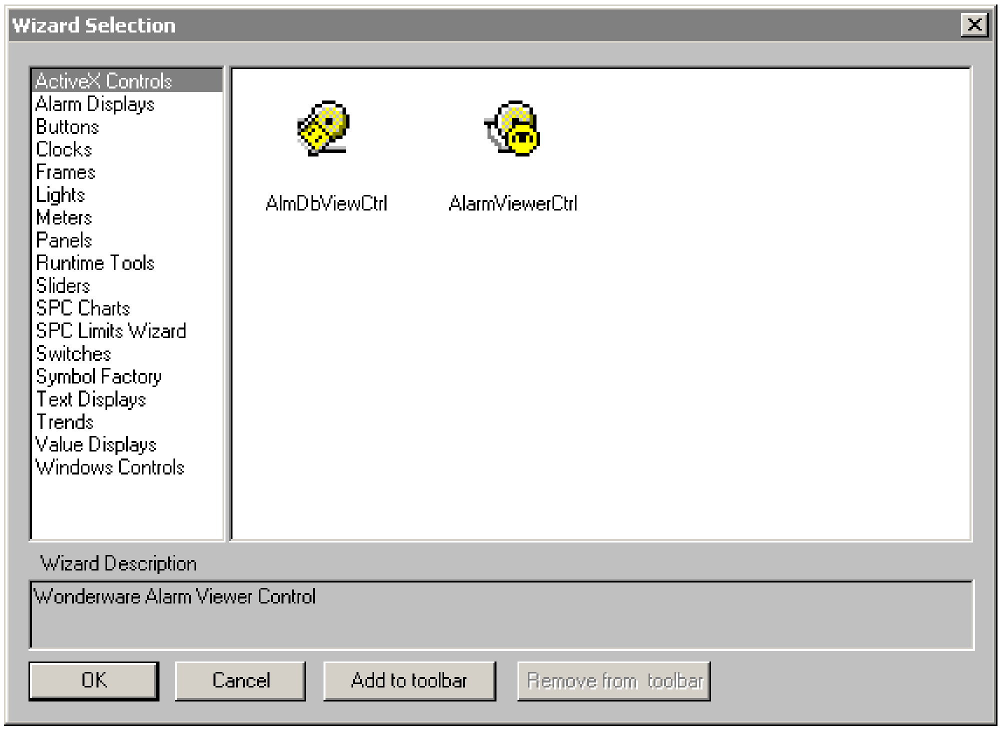
Uninstalling the Alarm Viewer ActiveX Control

1. Delete all of the Alarm Viewer controls pasted on the windows.
2. Select Configure from the **Special** menu.
3. Select **Wizard/ActiveX Installation**, then open the Wizard/ActiveX installation dialog box.
4. Select the **ActiveX Control** installation property page. The Wonderware Alarm Viewer Control name appears in the Installed ActiveX Controls text area.
5. Click Wonderware Alarm Viewer Control, then click **Remove**.
6. Click **Yes** for the warning message.
7. Click **Close**.

Creating an Alarm Viewer ActiveX Control

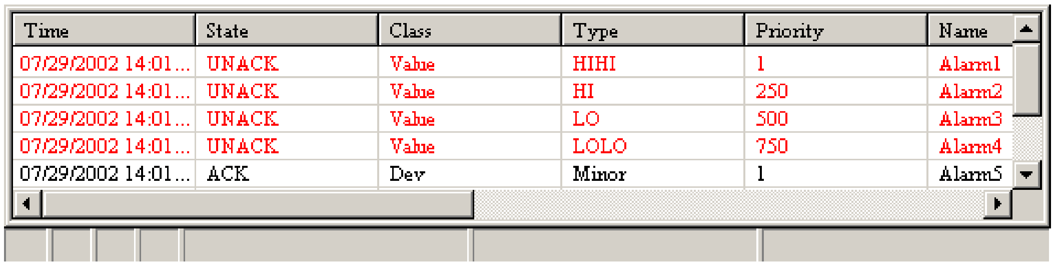
To create an Alarm Viewer ActiveX display

1. Click the **Wizard** tool in the **Wizard/ActiveX Toolbar**. The **Wizard Selection** dialog box appears.



Wonderware Alarm Viewer Control

2. Select **ActiveX Controls** in the list of wizards.
3. Double-click the **AlarmViewerCtrl** control, or select it and then click **OK**. The dialog box closes and your window reappears with the cursor in the *paste* mode.
4. Click in the window to paste the Alarm Viewer control.



Time	State	Class	Type	Priority	Name
07/29/2002 14:01...	UNACK	Value	HIHI	1	Alarm1
07/29/2002 14:01...	UNACK	Value	HI	250	Alarm2
07/29/2002 14:01...	UNACK	Value	LO	500	Alarm3
07/29/2002 14:01...	UNACK	Value	LOLO	750	Alarm4
07/29/2002 14:01...	ACK	Dev	Minor	1	Alarm5

Tip To size the wizard, point to one of its selection handles then drag it until the desired size is reached.

5. You are now ready to configure the display.

Accessing the AlmViewerCtrl Properties Dialog Box

You access the AlmViewerCtrl Properties Dialog Box by:

- Double clicking on the control, or by
- Right clicking over the control and selecting the Properties menu from the pop up menu.

Configuring an Alarm Viewer ActiveX Control

The **AlarmViewerCtrl** dialog box has eight property sheets. The property sheets are **Control Name**, **General**, **Color**, **Time Format**, **Query**, **Properties** and **Events**.

Note The configuration dialog box behaves like any standard Windows property sheet in that no settings are recorded until you click **OK** or **Apply**. The options are verified for proper entries, however, when you change from one property sheet (tab) to another. If an entry verification fails, the property sheet containing the failed entry is brought back into focus, and a message box appears indicating the error. When you change from one property sheet to another, the changes made to the previous property sheet are automatically applied. If you click **Cancel**, all input is ignored and the dialog box closes.

Alarm Viewer ActiveX Display Properties

To configure the Alarm Viewer ActiveX display

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrl** dialog box appears with the **Control Name** property sheet active.

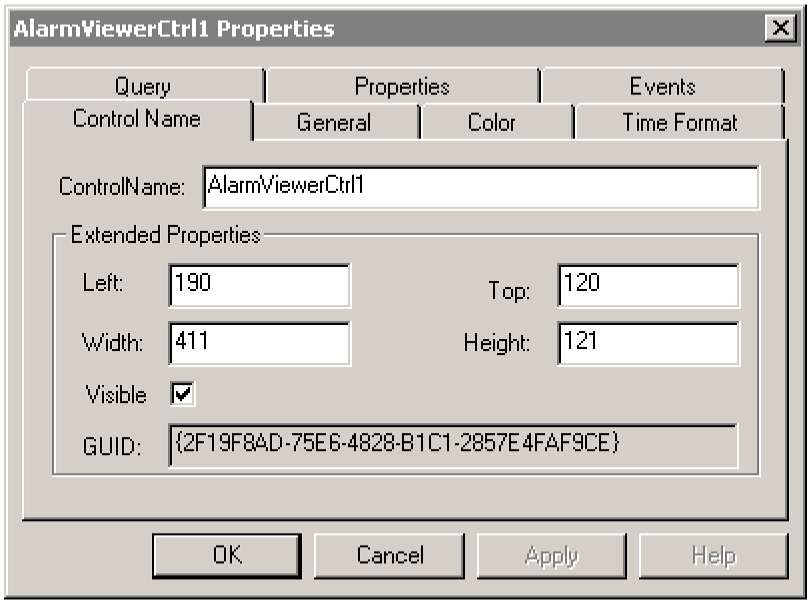
Tip If you right-click a text box in any alarm configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

2. Click the tab for the property sheet to configure.

Configuring the Control Name and Display Position

To configure the control name and display position

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrl** dialog box appears with the **Control Name** property sheet active.



ControlName: AlarmViewerCtrl1

Left: 190
Top: 120
Width: 411
Height: 121
Visible ✓
GUID: {2F19F8AD-75E6-4828-B1C1-2857E4FAF9CE}

2. In the **ControlName** box, type the name for the alarm display. This name must be unique for each alarm display used.

Note By default, the Control Name is determined by the ProgID for that control. ProgIDs are names that are entered into the system registry when ActiveX controls are installed on a computer. When an instance of that control is placed in an InTouch application, the control's ProgID is obtained from the system registry and an index number is appended to it, resulting in a Control Name, such as **AlarmViewerCtrl1**.

Tip The name you type here will be used throughout the system for referring to this object for execution of tasks such as alarm acknowledgment and queries.

3. Enter numerical values in the **Left**, **Width**, **Top** and **Height** dialog boxes to position the alarm viewer for the control.:

Option	Description
Left	Determines the distance between the Alarm Viewer display and the left edge of the window. A lower number positions the display closer to the left edge of the window. A higher number positions the display further from the left edge of the window.
Width	Determines the width of the Alarm Viewer display.
Top	Determines the distance between the Alarm Viewer display and the top edge of the window. A lower number positions the display closer to the top edge of the window. A higher number positions the display further from the top edge of the window.
Height	Determines the height of the Alarm Viewer display.
Visible	Unselect the **Visible** check box to make the control invisible during runtime.
GUID	Displays the unique ID for this ActiveX control.

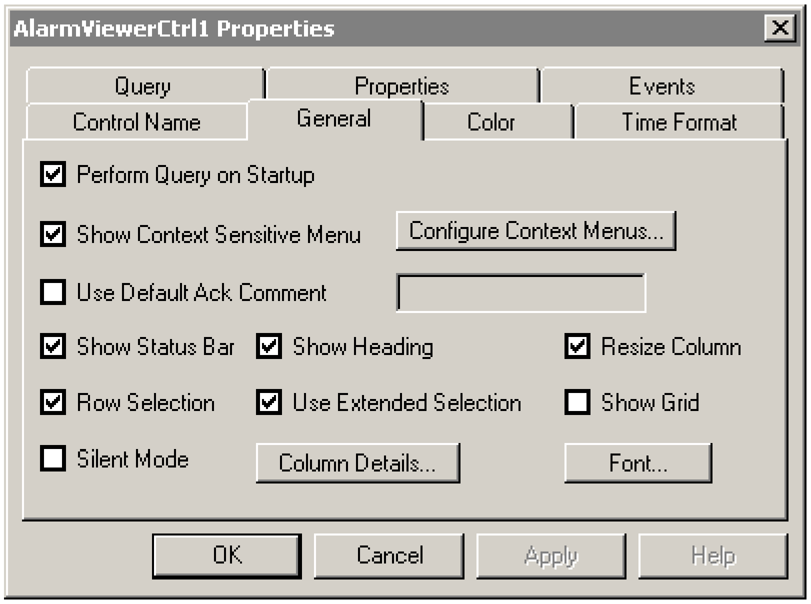
4. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Configuring the General Properties

To configure the general properties

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.

2. Click the **General** tab to activate the **General** property sheet.The **General** property sheets is shown with the default settings selected.



- [✓] Perform Query on Startup
- [✓] Show Context Sensitive Menu
- [ ] Use Default Ack Comment
- [✓] Show Status Bar
- [✓] Show Heading
- [✓] Resize Column
- [✓] Row Selection
- [✓] Use Extended Selection
- [ ] Show Grid
- [ ] Silent Mode

3. Check the boxes to select the general display options. Uncheck boxes to omit options from the alarm viewer. The check box options are described in the table below:

Property	Description
Perform Query on Startup	Automatically begins updating the display using default query properties, if selected. If not selected, you need to perform an **almDefQuery** or **almQuery** before the display will update. The **Requery** option is also available on the right-click shortcut menu.
Show Context Sensitive Menu	Enables the activation of the right-click popup menu. Selecting this option enables the **Configure Context Menus** button and the **Use Default Ack Comment** check box.
Use Default Ack Comment	Controls whether a default comment will be used when an operator ACKs an alarm. If this box is checked and a string is entered, the string will be used during runtime as a default comment. If this box is not checked, when the operator ACKs an alarm, a dialog box appears to let the operator enter a comment. The dialog box can be filled in or left blank. Selecting this option enables the Ack Comment field in which to enter the comment. This button is only available when the **Show Context Sensitive Menu** box is checked.

Property	Description
Show Status Bar	Toggles the display of the status bar.
Show Heading	Toggles the display of the heading bar.
Row Selection	Selecting this option allows you to select records in the alarm viewer. This allows multiple selections of alarms and doesn't toggle previously made alarm selections based on new selections.
Use Extended Selection	Allows multiple alarms to be selected by holding down Ctrl or Shift in conjunction with a mouse. The default is to toggle selection of alarms by simply clicking on them (available only if **Row Selection** check box is selected).
Silent Mode	If **Silent Mode** is selected, the distributed alarm display will not pop up error messages in Runtime. If it is not selected, the alarm display will show pop up error messages. Error messages are always sent to the Log Viewer.
Resize Columns	If **Resize Columns** is selected, the user can resize columns in Runtime. If it is not selected, the user will not be able to resize columns in Runtime.
Show Grid	If **Show Grid** is selected, the Alarm Viewer shows a grid in the distributed alarm display. If it is not selected, no grid is visible.

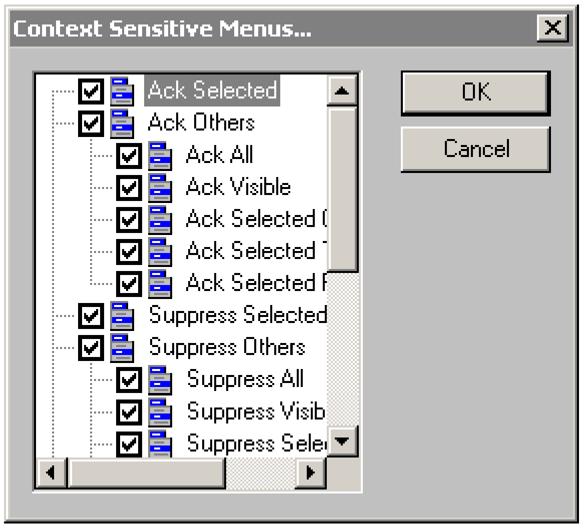
4. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Configuring the Context Sensitive Menus

To configure the context sensitive menus

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.
2. Click the **General** tab to activate the **General** property sheet.

3. Click the **Configure Context Menus** button to select the options to display on the context sensitive right mouse click menus in Runtime. The Context Sensitive Menus screen appears.



Note The **Configure Context Menus** button is only enabled if the **Show Context Sensitive Menu** option is selected.

4. Configure the context sensitive menu options and click **OK**. At least one context sensitive menu item should be selected if the **Show Context Sensitive Menu** box is checked.

5. The context sensitive menu options are described below:

Menu Option	Description
Ack Selected	Allows user to acknowledge selected alarms.
Ack Others	Allows user to acknowledge alarms by other methods. The user can select which alarms to acknowledge from the non-bolded options below. If **Ack Others** is selected, you must select at least one of the submenu items.
Ack All	Allows user to acknowledge all active alarms.
Ack Visible	Allows user to acknowledge visible alarms.
Ack Selected Groups	Allows user to acknowledge selected groups with the same provider name.
Ack Selected Tags	Allows user to acknowledge selected tags with the same provider, group and priority.
Ack Selected Priorities	Allows user to acknowledge selected priorities with the same provider, group and priority.

Menu Option	Description
Suppress Selected	Allows user to suppress selected alarms.
Suppress Others	Allows user to suppress alarms by other methods. The user can select which alarms to suppress from the non-bolded options below. If **Suppress Others** is selected, you must select at least one of the submenu items.
Suppress All	Allows user to suppress all alarms.
Suppress Visible	Allows user to suppress visible alarms.
Suppress Selected Groups	Allows user to suppress selected groups.
Suppress Selected Tags	Allows user to suppress selected tags.
Suppress Selected Priorities	Allows user to suppress selected priorities.
Unsuppress All	Allows user to unsuppress all suppressed alarms.
Query Favorites	Allows user to access the Query Favorites dialog.
Stats	Allows the user to access and view alarm statistics. Right-click the alarm viewer display in runtime, then click **Stats** on the shortcut menu. The **Alarm Statistics** dialog box indicates which control the alarm statistics are from in the dialog box header.
Suppression	Allows the user to access the suppression dialog.
Freeze	Allows the user to toggle the freeze/unfreeze mode of the display.
Requery	Allows the user to re-run an alarm query.
Sort	Allows the user to access the Sort dialog.

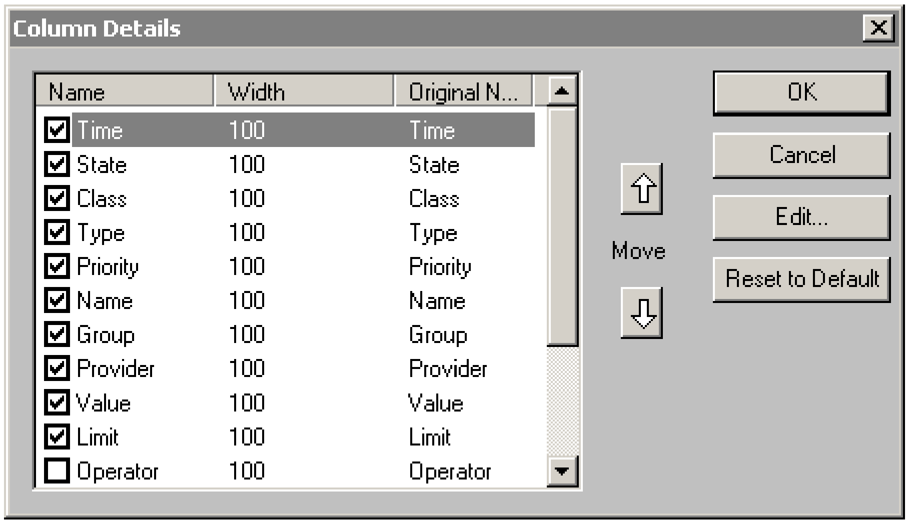
Note If **Ack Selected** and **Ack Others** menu items are both unchecked at design time in the **Context Sensitive Menus** dialog box, the Use **Default Ack Comment** check box and the text box are disabled. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Configuring the Display Column Details

To configure the display column details

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.
2. Click the **General** tab to activate the **General** property sheet.

3. Click the **Column Details** button. The Column Details dialog appears.



Name	Width	Original N...
☑ Time	100	Time
☑ State	100	State
☑ Class	100	Class
☑ Type	100	Type
☑ Priority	100	Priority
☑ Name	100	Name
☑ Group	100	Group
☑ Provider	100	Provider
☑ Value	100	Value
☑ Limit	100	Limit
☐ Operator	100	Operator

4. From the **Column Details** dialog box, select the checkbox next to the **Column Name** to display the column in the alarm object. The columns in the **Column Details** dialog box are **Name**, **Width** and **Original Name**. **Original Name** shows what the columns were named before any changes were made. The original column names that you can select to display are described below.

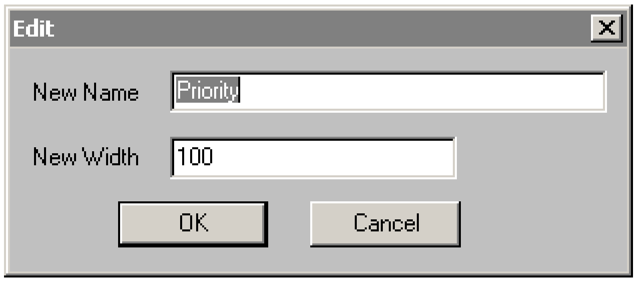
Note At least one column must be selected.

Column Name	Description
Time	Displays the time in the format selected from the Time Format properties sheet.
State	Displays the state of the alarm.
Class	Displays the category of the alarm.
Type	Displays the alarm type.
Priority	Displays the alarm priority.
Name	Displays the alarm/tagname.
Group	Displays the Alarm Group name.
Provider	Displays the name of the alarm provider.
Value	Displays the value of the tagname when the alarm occurred. The width of the column should be large enough to provide the desired level of precision.
Limit	Displays the alarm limit value of the tagname. The width of the column should be large enough to provide the desired level of precision.
Operator	Displays the logged-on operator's ID associated with the alarm condition.

Column Name	Description
Comment	Displays the tagname's comments. These comments were typed in the Alarm Comment box when the tagname's alarm was defined in the database. When ACK Comments are introduced for alarms, the new comments are updated in this comment column.
OperatorName	Displays the operator name associated with the alarm.
OperatorDomain	Displays the operator domain associated with the alarm.
Node Name	Displays the node name associated with the alarm.
Tag Comment	Displays the tag comment associated with the alarm.

Note All column names are selected by default except for **Operator** and **Comment**.

5. To rearrange the columns, select the column name and use the up and down arrow keys. The column name appearing at the top of the **Column Details** dialog box is the column displayed to the furthest left of the alarm display.
6. To edit the column name and width, double-click the column name or select a column name, then click **Edit.** The **Edit** dialog box appears for that column.



New Name: Priority

New Width: 100

7. Enter in a new name in the **New Name** text box if you want to display a column name other than the default column name.
8. Enter in a column width in the **New Width** text box. The column width is measured in pixels and can range from 1 to 999 pixels. The default column width is 100 pixels.
9. Click **OK** on the **Edit** dialog.

Note Click **Reset to Default** to return to the default **Column Details** settings.

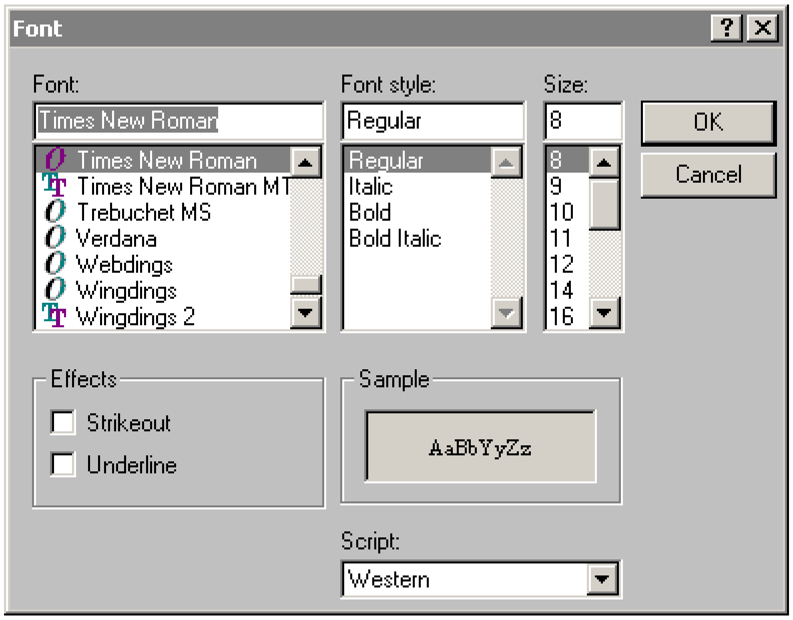
10. Click **OK** on the **Column Details** dialog.

11. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Configuring the ActiveX Alarm Viewer Font Properties

To configure the font properties

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.
2. Click the **General** tab, then click the **Font** button. The Font dialog box appears:



3. Scroll the Font list and select the desired font type.
4. Scroll the **Font Style** list to select a font style.
5. Scroll the **Size** list to select a font size.
6. In the **Effects** area, check the **Strikeout** box or **Underline** box to select the Strikeout or Underline attributes.
7. Click the **Script** drop-down arrow to select the desired script type.

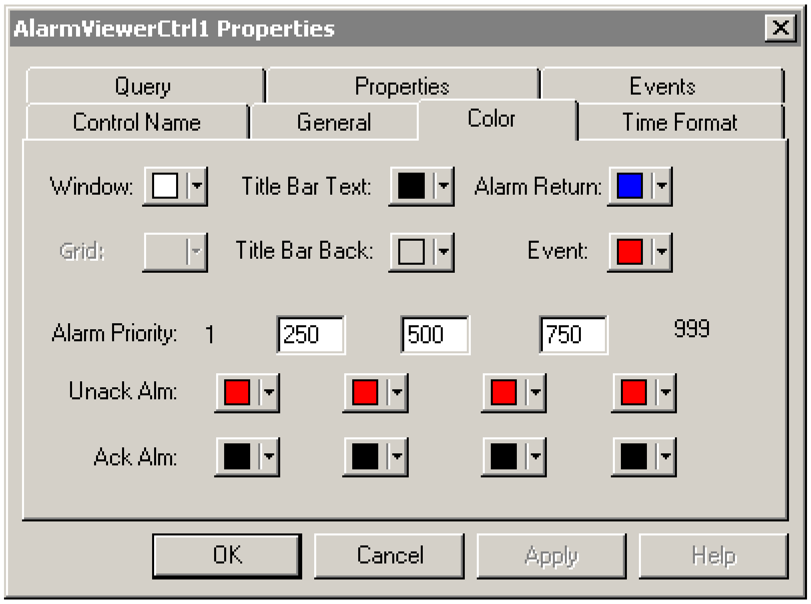
Note The **Sample** box shows a sample of the selected font attributes.

8. Click **OK**.

Configuring the ActiveX Alarm Viewer Color Properties

To configure the alarm display colors

1. Double-click on the Alarm Viewer ActiveX or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.
2. Click the **Color** tab to activate the **Color** property sheet.



Alarm Priority: 1 250 500 750 999

3. Click each color box to open the color palette. Click the color that you want to use in the palette for each of the following:

Property	Description
Window	Sets display background color.
Title Bar Text	Sets title bar text color (available only if Show Heading option is selected).
Alarm Return	Sets color of returned alarms (alarms that have returned to normal without being acknowledged).
Grid	Sets color of the grid. By default the grid is not shown. The default grid color is light gray. The color of the grid in the alarm object is automatically set to a contrasting color of the selected **Window** color.
Title Bar Background	Sets title bar background color (available only if Show Heading option is selected).
Event	Sets color of Event alarms.

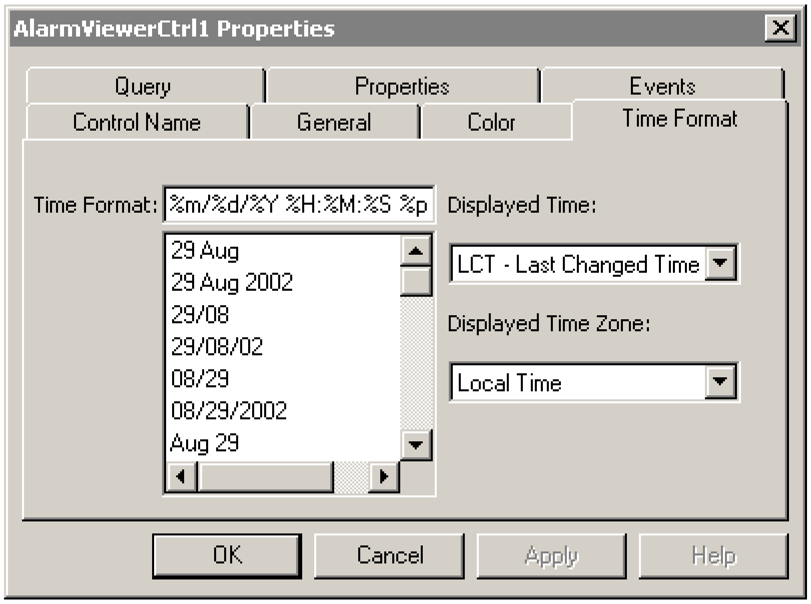
4. In the **Alarm Priority** boxes, type the breakpoint values for the alarm display.
5. Click the **UnAck Alarm** and **Ack Alarm** color boxes to open the color palette. Click the color in the palette that you want to use.
6. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Configuring the Alarm Viewer Time Format

To configure the Alarm Viewer time format

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.

2. Click the **Time Format** tab to activate the **Time Format** property sheet.



Time Format: %m/%d/%Y %H:%M:%S %p

29 Aug
29 Aug 2002
29/08
29/08/02
08/29
08/29/2002
Aug 29

Displayed Time: LCT - Last Changed Time

Displayed Time Zone: Local Time

Scroll the Time Format options to select the desired time format or configure additional time format options. The time format strings consist of string characters separated by the % symbol. The time format string characters are explained in the table below::

String character	Description
d	two digit date - 09
b	3 letter month abbreviation - Aug
Y	four digit year - 2002
m	two digit month - 08
y	two digit year - 02
#x	full day and date - Friday, August 09, 2002
B	full month name - August
H	24 hour time - 16:00
M	minute 00:41
p	PM
S	seconds - 16:41:07
s	fractions of a second - 16:41:07.390
I	12 hour time requiring AM/PM designation - 04:41 PM

Some sample time format character strings are shown below:

Time Format String	Display
%d %b	09 Aug
%m/%d/%Y	08/09/2002

Time Format String	Display
%#x	Friday, August 09, 2002
%Y-%m-%d	2002-08-09
%m/%d/%Y %H:%M %p	08/09/2002 16:56 PM
%m/%d/%Y %H:%M:%s %p	08/09/2002 16:56:38.07 PM
%I:%M %p	04:56 PM

3. Click the **Displayed Time** drop-down arrow to select the displayed time. The available options are:

OAT	Original Alarm Time - that is, the date/time stamp of the onset of the alarm.
LCT	Last Changed Time - that is, the date/time stamp of the most recent change of state for the instance of the alarm: onset of the alarm, change of sub-state, return to normal, or acknowledgment.
LCT But OAT on ACK	Last Changed Time, but Original Alarm Time on ACK - that is, LCT will be used while the alarm is UNACKed, then OAT will be used after the alarm has been ACKed.

4. Click the **Displayed Time Zone** drop-down arrow to select the time zone. The available options are:

GMT	Greenwich Mean Time, also known as Coordinated Universal Time, UTC, or Zulu.
Local Time	Alarm time adjusted for the local time zone.
Origin Time	Alarm time adjusted for the time zone of the alarm source.

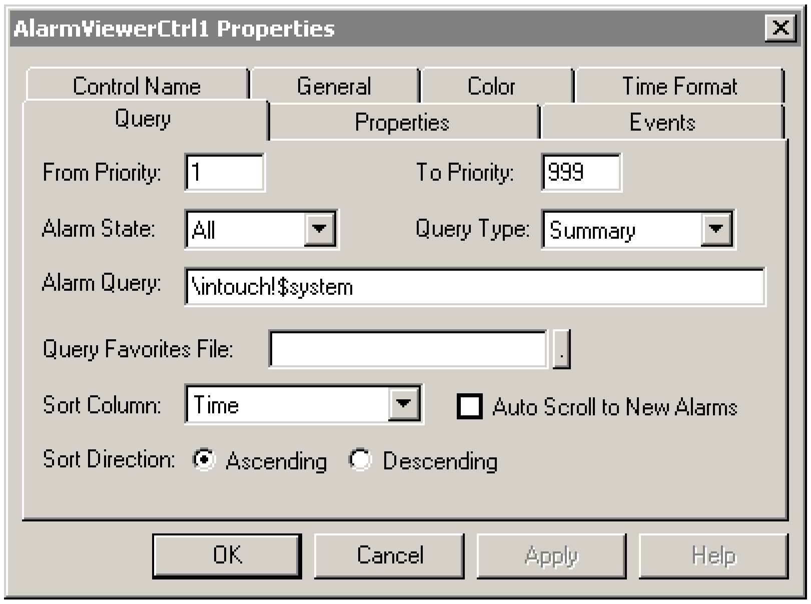
5. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Configuring the Display Alarm Query

To configure the alarm display query

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.

2. Click the **Query** tab to activate the **Query** property sheet.



From Priority: 1
To Priority: 999
Alarm State: All
Query Type: Summary
Alarm Query: \intouch!$system
Query Favorites File:
Sort Column: Time
☐ Auto Scroll to New Alarms
Sort Direction: [✓ Ascending] Descending

3. Select the desired query options. The query options are described in the table below:

Property	Description
From Priority	Minimum alarm priority set to 1 by default. This value should always be less than "To Priority" and the range of values is from 1 to 999.
To Priority	Maximum alarm priority set to 999 by default. This value should always be more than "From Priority" and the range of values is from 1 to 999.
Alarm State	Default alarm state to query set to All by default. ALL indicates a query of all alarms. UnAck indicates a query of unacknowledged alarms. Ack indicates a query of acknowledged alarms.
Query Type	Sets display type as either Summary or Historical.

Property	Description
Alarm Query	Sets the initial Alarm query. This field accepts text only; it does not accept tags. The default Alarm Query is **\intouch!$system** The valid syntax for these lists include: **\\\\Node\InTouch!Group** Full path to Alarm Group **\InTouch!Group** Full path to local Alarm Group **GroupList** Another Group List
Sort Column	Select the column to sort using the drop-down arrow. Options are alarm Time, State, Class, Type, Priority, Name, Group, Provider, Value, Limit, Operator or Comment.
Query Favorites File	The control will store alarm Queries. Click the button to browse for query favorites.The differences between Query Favorites and the Alarm Query are: • The Alarm Query files are XML. • A filename should be given with the proper folder name to enable the Query Favorites context sensitive menu option at runtime. • Existing filenames allow access to existing queries. New filenames are created when new queries are created. • The query file can be in any folder and does not need to be in the application folder.
Auto Scroll to new Alarms	This check box will be available only when the sort option selected is based on Time.
Sort Direction	Select the criteria by which to sort the query and click the Up or Down radio buttons to sort in ascending or descending order respectively. By default, the display is sorted by time in ascending order.

Note To perform multiple queries, separate each query with a space.

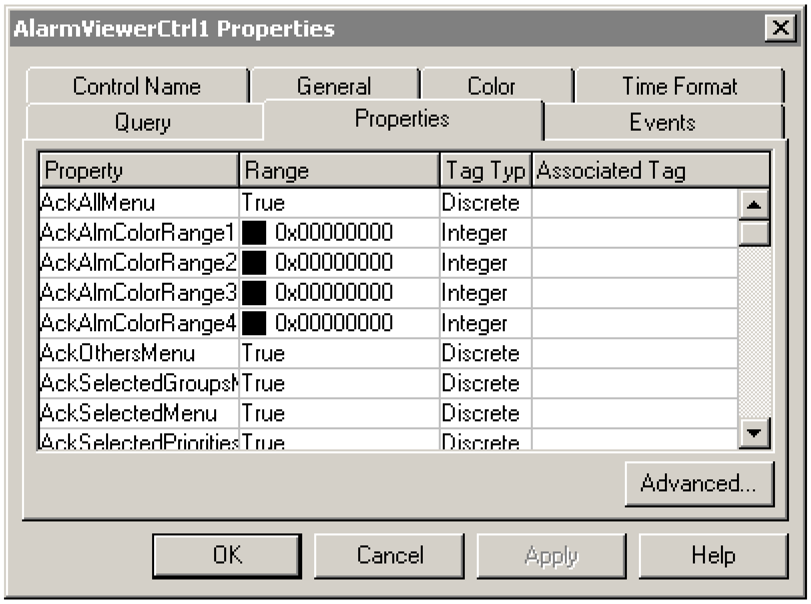
For example: \\Master\InTouch!MyGroup LocalGroupList

4. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Assigning Tagnames to Control Properties

To assign tagnames to control properties

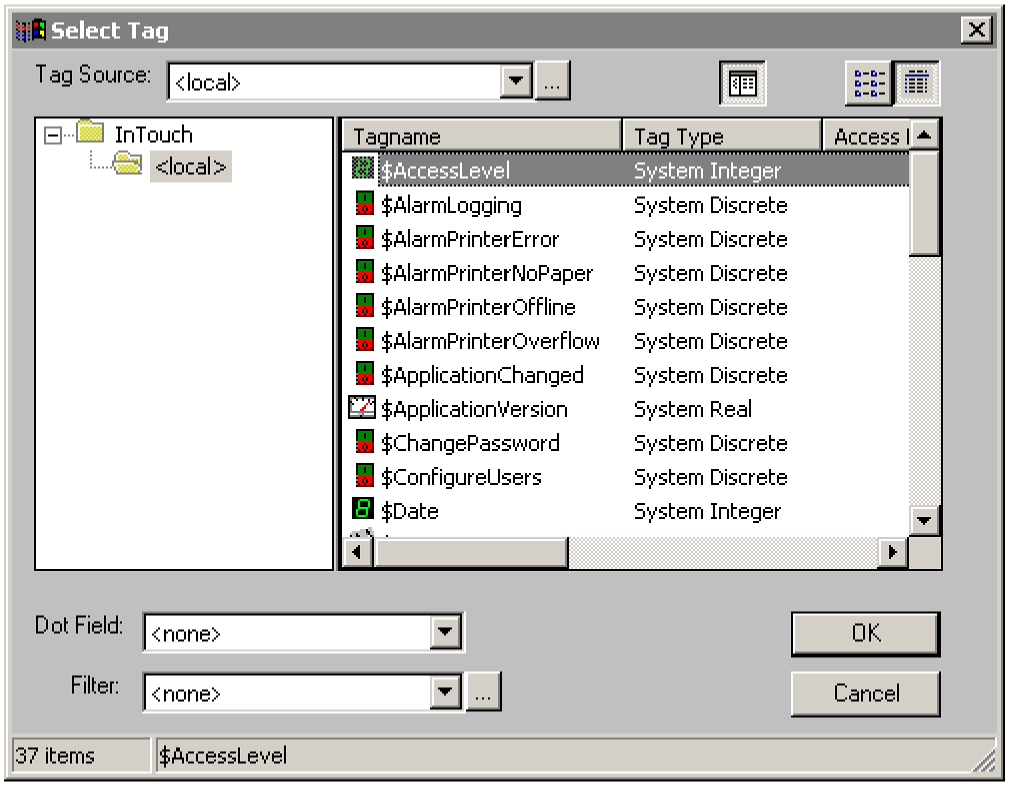
1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.
2. Click the **Properties** tab to activate the **Properties** sheet.



Property	Range	Tag Typ	Associated Tag
AckAllMenu	True	Discrete	
AckAlmColorRange1	■ 0x00000000	Integer	
AckAlmColorRange2	■ 0x00000000	Integer	
AckAlmColorRange3	■ 0x00000000	Integer	
AckAlmColorRange4	■ 0x00000000	Integer	
AckOthersMenu	True	Discrete	
AckSelectedGroups!	True	Discrete	
AckSelectedMenu	True	Discrete	
AckSelectedPriorities	True	Discrete	

3. Click the property for which to assign a tagname.

4. Click the button in the Associated Tag column to browse tagnames. The **Select Tag** window appears.



Tagname	Tag Type	Access I
$AccessLevel	System Integer	
$AlarmLogging	System Discrete	
$AlarmPrinterError	System Discrete	
$AlarmPrinterNoPaper	System Discrete	
$AlarmPrinterOffline	System Discrete	
$AlarmPrinterOverflow	System Discrete	
$ApplicationChanged	System Discrete	
$ApplicationVersion	System Real	
$ChangePassword	System Discrete	
$ConfigureUsers	System Discrete	
$Date	System Integer	

5. Click **OK**.
6. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

AlarmViewer Properties Tab Properties

To set properties, type #object.PropertyName = 1; or #object.PropertyName = tag1; where object is the name of the AlarmViewer and tag1 is a discrete tag. For example, to set the AckAllMenu property, type #AlarmViewer.AckAllMenu = 1.

To get properties, type tag1 = #object.PropertyName; where object is the name of the AlarmViewer) and tag1 is a discrete tag. For example, to get the AckAllMenu property, type tag1 = #AlarmViewer.AckAllMenu;.

InTouch accepts the words "True" and "False" within double quotes as values 1 and 0 respectively. An action script such as:

#AlarmViewerCtrl2.SortMenu = "False";

erases the “Sort Menu” from the right-click menu of the Alarm Viewer ActiveX Control.

The **Properties** tab properties are as follows:

Property:	**AckAllMenu**
Purpose:	Boolean Property. Enables/Disables "AckAll" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.AckAllMenu *[= discrete]*

Property:	**AckAlmColorRange1**
Purpose:	Sets color to be used to display ACKed alarms with priorities in the range 1 to ColorPriorityRange1. The default priority range is 1 to 250.
Type:	Integer
Default:	Black
Syntax:	*Object*.AckAlmColorRange1 *[= Integer]*

Property:	**AckAlmColorRange2**
Purpose:	Sets color to be used to display ACKed alarms with priorities in the range ColorPriorityRange1 to ColorPriorityRange2. The default priority range is 250 to 500.
Type:	Integer
Default:	Black
Syntax:	*Object*.AckAlmColorRange2 *[= Integer]*

Property:	**AckAlmColorRange3**
Purpose:	Sets color to be used to display ACKed alarms with priorities in the range ColorPriorityRange2 to ColorPriorityRange3. The default priority range is 500 to 750.
Type:	Integer
Default:	Black
Syntax:	*Object*.AckAlmColorRange3 *[= Integer]*

Property:	**AckAlmColorRange4**
Purpose:	Sets color to be used to display ACKed alarms with priorities in ColorPriorityRange3 to 999. The default priority range is 750 to 999.
Type:	Integer
Default:	Black
Syntax:	*Object*.AckAlmColorRange4 *[= Integer]*

Property:	**AckOthersMenu**
Purpose:	Boolean Property. Enables/Disables "AckOthers" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.AckOthersMenu *[= discrete]*

Property:	**AckSelectedGroupsMenu**
Purpose:	Boolean Property. Enables/Disables "AckSelected" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.AckSelectedGroupsMenu*[= discrete]*

Property:	**AckSelectedMenu**
Purpose:	Boolean Property. Enables/Disables "AckSelected" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.AckSelectedMenu *[= discrete]*

Property:	**AckSelectedPriorities**
Purpose:	Boolean Property. Enables/Disables "AckSelectedPriorities" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.AckSelectedPriorities *[= discrete]*

Property:	**AckSelectedTags**
Purpose:	Boolean Property. Enables/Disables "AckSelectedTags" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.AckSelectedTags *[= discrete]*

Property:	**AckVisibleMenu**
Purpose:	Boolean Property. Enables/Disables "AckVisible" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.AckVisibleMenu *[= discrete]*

Property:	**AlarmQuery**
Purpose:	Sets the initial Alarm query. This field accepts text only; it does not accept tags. The valid syntax for these lists include: **\\Node\InTouch!Group** Full path to Alarm Group **\InTouch!Group** Full path to local Alarm Group **GroupList** Another Group List
Type:	Message
Default:	\intouch!$system
Syntax:	*Object*.AlarmQuery *[= message]*

Property:	**AlarmState**
Purpose:	Default alarm state to query (All, UnAck, Ack).
Type:	Message
Default:	All
Syntax:	*Object*.AlarmState *[= message]*

Property:	**AlmRtnColor**
Purpose:	Sets color of returned alarms (alarms that have returned to normal without being acknowledged), whether ACKed or UNACKed.
Type:	Integer
Default:	All
Syntax:	*Object*.AlmRtnColor *[= Integer]*

Property:	**AutoScroll**
Purpose:	If the user scrolls the list from the beginning, this automatically jumps to the new alarm. (New alarms are defined as those that are not currently displayed within the display object.)
Type:	Discrete
Default:	False
Syntax:	*Object*.Autoscroll *[= discrete]*

Property:	**ColorPriorityRange1**
Purpose:	Sets the boundry of the priority range in which alarms are to be displayed. The value of this property must be greater than one and less than the value for ColorPriorityRange2.
Type:	Integer
Default:	250
Syntax:	*Object*.ColorPriorityRange1 *[= integer or priority]*

Property:	**ColorPriorityRange2**
Purpose:	Sets the boundry of the priority range in which alarms are to be displayed. The value of this property must be greater than the value for ColorPriorityRange1 and less than the value for ColorPriorityRange3.
Type:	Integer
Default:	500
Syntax:	*Object*.ColorPriorityRange2 *[= integer or priority]*

Property:	**ColorPriorityRange3**
Purpose:	Sets the boundry of the priority range in which alarms are to be displayed. The value of this property must be greater than the value of ColorPriorityRange2 and less than 999.
Type:	Integer
Default:	750
Syntax:	*Object*.ColorPriorityRange3 *[= integer or priority]*

Property:	**ColumnResize**
Purpose:	Returns or sets a value that determines whether the columns can be resized at runtime.
Type:	Discrete
Default:	True
Syntax:	*Object*.ColumnResize *[= discrete]*

Property:	**DefaultAckComment**
Purpose:	String Property. This will be used as a comment when the alarm is acknowledged when the "UseDefaultAckComment" is TRUE, else the user will prompted to enter a comment.
Type:	Message
Default:	None
Syntax:	*Object*.DefaultAckComment *[= message]*

Property:	**DisplayedTime**
Purpose:	Displays the alarm message time.
Type:	Message
Default:	LCT-Last Changed Time
Syntax:	*Object*.DisplayedTime *[= message]*

Property:	DisplayedTimeZone
Purpose:	String Property. Returns/Sets the current time zone string. The values can only be "GMT" or "Origin Time" or "Local Time."
Type:	Message
Default:	Local Time
Syntax:	*Object*.DisplayedTimeZone *[= message]*

Property:	EventColor
Purpose:	Sets color of Event alarms.
Type:	Integer
Default:	Red
Syntax:	*Object* EventColor *[= color]*

Property:	Extended Selection
Purpose:	Allows multiple alarms to be selected by holding down Ctrl or Shift in conjunction with a mouse. The default is to toggle selection of alarms by simply clicking on them (available only if the **Row Selection** check box is selected).
Type:	Discrete
Default:	True
Syntax:	*Object*.ExtendedSelection *[= discrete]*

Property:	Font
Purpose:	Sets the font for the display of records and the heading in the control.
Type:	None
Default:	Times New Roman
Syntax:	*Read-only*

Property:	**FreezeMenu**
Purpose:	Boolean Property. Enables/Disables "Freeze" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.FreezeMenu *[= discrete]*

Property:	**FromPriority**
Purpose:	Sets the low priority value of the default query.
Type:	Integer
Default:	1
Syntax:	*Object*.FromPriority *[= integer]*

Property:	**GridColor**
Purpose:	Sets the color of the background grid.
Type:	Integer
Default:	Gray
Syntax:	*Object*.GridColor *[= color]*

Property:	**QueryFavoritesFile**
Purpose:	String Property. Returns/Sets the query favorites file name.
Type:	Message
Default:	None
Syntax:	*Object*.QueryFavoritesFile *[= message]*

Property:	**QueryFavoritesMenu**
Purpose:	Enables/Disables "QueryFavorites" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.QueryFavoritesMenu *[= discrete]*

Property:	**QueryName**
Purpose:	Returns the current query name.
Type:	String (Read Only)
Default:	None
Syntax:	*Object*.QueryName *[= String]*

Property:	**QueryStartup**
Purpose:	Automatically begins updating the display using default query properties, if selected. If not selected, you need to perform an ApplyDefaultQuery or ApplyQuery before the display will update.
Type:	Discrete
Default:	True
Syntax:	*Object*.QueryStartup *[= discrete]*

Property:	**QueryType**
Purpose:	Sets display type as either Summary or Historical.
Type:	Message
Default:	Summary
Syntax:	*Object*.QueryType *[= message]*

Property:	**RequeryMenu**
Purpose:	Enables/Disables "Requery" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.RequeryMenu *[= discrete]*

Property:	**RetainSuppression**
Purpose:	Retains alarm suppression between alarm queries when the alarm query is changed.
Type:	Discrete
Default:	True
Syntax:	*Object*.RetainSuppression *[= discrete]*

Property:	**RowSelection**
Purpose:	Allows user to select alarms at runtime.
Type:	Discrete
Default:	True
Syntax:	*Object*.RowSelection *[= discrete]*

Property:	**SelectedCount**
Purpose:	Returns the total number of selected alarms.
Type:	Integer (Read Only)
Default:	None
Syntax:	*Object*.SelectedCount *[= integer]*

Property:	**ShowContextMenu**
Purpose:	Enables the activation of the right-click popup menu.
Type:	Discrete
Default:	True
Syntax:	*Object*.ShowContextMenu *[= discrete]*

Property:	**ShowGrid**
Purpose:	Returns or sets a value that determines whether the grid lines are displayed in the control.
Type:	Discrete
Default:	False
Syntax:	*Object*.ShowGrid *[= discrete]*

Property:	**ShowHeading**
Purpose:	Displays the title bar of the control.
Type:	Discrete
Default:	True
Syntax:	*Object*.ShowHeading *[= discrete]*

Property:	**ShowStatusBar**
Purpose:	Returns or sets a value that determines whether the status bar is shown. The status bar that contains three indicators: A status message, current alarm query, and a progress bar. These indicators provide an overview of the current state of the display query and provide details about the suppression available in the Distributed Alarm Object. The right pane of the status bar is red when freeze is in effect and the left pane of the status bar is red when suppression is in effect. The word "suppression" displays in the left pane when suppression is in effect.
Type:	Discrete

Property:	**ShowStatusBar**
Default:	True
Syntax:	*Object*.ShowStatusBar *[= discrete]*

Property:	**SilentMode**
Purpose:	Returns or sets a value that determines whether the control is in Silent mode.
Type:	Discrete
Default:	False
Syntax:	*Object*.SilentMode *[= discrete]*

Property:	**SortColumn**
Purpose:	String Property. Returns/Sets the current sort column.
Type:	Message
Default:	Time
Syntax:	*Object*.SortColumn *[= message]*

Property:	**SortOrder**
Purpose:	String Property. Returns/Sets the sort direction. Possible values are "Ascending" and "Descending," represented as 0 and 1 respectively.
Type:	Discrete
Default:	False
Syntax:	*Object*.SortOrder *[= discrete]*

Property:	**SortMenu**
Purpose:	Boolean Property. Enables/Disables "Sort" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SortMenu *[= discrete]*

Property:	**StatsMenu**
Purpose:	Boolean Property. Enables/Disables "Stats" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.StatsMenu *[= discrete]*

Property:	**SuppressAllMenu**
Purpose:	Boolean Property. Enables/Disables "SuppressAll" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SuppressAllMenu *[= discrete]*

Property:	**SuppressedAlarms**
Purpose:	Returns the total number of suppressed alarms.
Type:	Integer (Read Only)
Default:	None
Syntax:	*Object*.SuppressedAlarms *[= integer]*

Property:	**SuppressionMenu**
Purpose:	Boolean Property. Enables/Disables "Suppression" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SuppressionMenu *[= discrete]*

Property:	**SuppressOthersMenu**
Purpose:	Boolean Property. Enables/Disables "SuppressOthers" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SuppressOthersMenu *[= discrete]*

Property:	**SuppressSelectedGroupsMenu**
Purpose:	Boolean Property. Enables/Disables "SuppressSelectedGroups" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SuppressSelectedGroupsMenu *[= discrete]*

Property:	**SuppressSelectedMenu**
Purpose:	Boolean Property. Enables/Disables "SuppressSelected" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SuppressSelectedMenu *[= discrete]*

Property:	**SuppressSelectedPriority**
Purpose:	Boolean Property. Enables/Disables "SuppressSelectedPriority" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SuppressSelectedPriority *[= discrete]*

Property:	**SuppressSelectedTagsMenu**
Purpose:	Boolean Property. Enables/Disables "SuppressSelectedTags" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SuppressSelectedTagsMenu *[= discrete]*

Property:	**SuppressVisibleMenu**
Purpose:	Boolean Property. Enables/Disables "SuppressVisible" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.SuppressVisibleMenu *[= discrete]*

Property:	**TimeFormat**
Purpose:	Sets the alarm display time format.
Type:	Message
Default:	%b%y (e.g. 27 Aug)
Syntax:	*Object*.TimeFormat *[= message]*

Property:	**TitleBackColor**
Purpose:	Sets title bar background color (available only if the Show Titles option is selected).
Type:	Integer
Default:	Gray
Syntax:	*Object*.TitleBackColor *[= color]*

Property:	**TitleForeColor**
Purpose:	Sets title bar foreground color (available only if the Show Titles option is selected).
Type:	Integer
Default:	Black
Syntax:	*Object*.TitleForeColor *[= color]*

Property:	**ToPriority**
Purpose:	Sets the maximum priority for the alarm query.
Type:	Integer
Default:	999
Syntax:	*Object*.ToPriority *[= integer]*

Property:	**TotalAlarms**
Purpose:	Returns the total number of alarms.
Type:	Integer (Read Only)
Default:	None
Syntax:	*Object*.TotalAlarms *[= integer]*

Property:	**UnackAlarms**
Purpose:	Returns the total number of UNACKed alarms.
Type:	Integer (Read Only)
Default:	None
Syntax:	*Object*.UnackAlarms *[= integer]*

Property:	**UnAckAlmColorRange1**
Purpose:	Sets color of UnACKed alarms according to a value range. The value of this property must be greater than one and less than the value for UnAckAlmColorRange2.
Type:	Integer
Default:	Red
Syntax:	*Object*.UnAckAlmColorRange1 *[= color]*

Property:	**UnAckAlmColorRange2**
Purpose:	Sets the color to be used to display UNACKed alarms with priorities in the range ColorPriorityRange1 to ColorPriorityRange2.
Type:	Integer
Default:	Red
Syntax:	*Object*.UnAckAlmColorRange2 *[= color]*

Property:	**UnAckAlmColorRange2**
Purpose:	Sets the color to be used to display UNACKed alarms with priorities in the range ColorPriorityRange2 to ColorPriorityRange3.
Type:	Integer
Default:	Red
Syntax:	*Object*.UnAckAlmColorRange3 *[= color]*

Property:	**UnAckAlmColorRange3**
Purpose:	Sets the color to be used to display UNACKed alarms with priorities in the range ColorPriorityRange3 to 999.
Type:	Integer
Default:	Red
Syntax:	*Object*.UnAckAlmColorRange4 *[= color]*

Property:	**UnsuppressAllMenu**
Purpose:	Boolean Property. Enables/Disables "UnsuppressAll" menu item.
Type:	Discrete
Default:	True
Syntax:	*Object*.UnsuppressAllMenu *[= discrete]*

Property:	**UseDefaultAckComment**
Purpose:	String Property. The default ACK comment will be used when the alarm is acknowledged if the "UseDefaultAckComment" is TRUE. Otherwise the user will prompted to enter a comment.
Type:	Discrete
Default:	False
Syntax:	*Object*.UseDefaultAckComment *[= discrete]*

Property:	**WindowColor**
Purpose:	Sets display background color.
Type:	Integer
Default:	White
Syntax:	*Object*.WindowColor *[= color]*

Methods & Events

Method:	**AboutBox**
Purpose:	Shows the "About" dialog box.
Syntax:	*Object.*AboutBox
Example:	`#AlarmViewerCtrl1.AboutBox();` (where the name of the control is AlarmViewerCtrl1).

Method:	**AckSelected**
Purpose:	The Alarm Viewer allows alarms to be selected by clicking on them with the mouse at runtime. The **AckSelected** function can be used to acknowledge those alarms.
Syntax:	*Object*.AckSelected
Example:	`#AlarmViewerCtrl1.AckSelected();`

Method:	**AckAll**
Purpose:	Acknowledges all the alarms in the current alarm query. Since the Alarm Viewer has only a limited display area, the **almAckAll** function may acknowledge alarms that are not visible in the display.
Syntax:	*Object*.AckAll
Example:	`#AlarmViewerCtrl1.AckAll();`

Method:	**AckVisible**
Purpose:	Acknowledges only those alarms that are currently visible in the Alarm Viewer.
Syntax:	*Object*.AckVisible
Example:	`#AlarmViewerCtrl1.AckVisible();`

Method:	**AckSelectedGroup**
Purpose:	Acknowledges all alarms that have the same group name from the same provider as one or more of the selected alarms.

Method:	**AckSelectedGroup**
Syntax:	*Object*.AckSelectedGroup
Example:	`#AlarmViewerCtrl1.AckSelectedGroup();`

Method:	**AckSelectedTag**
Purpose:	Acknowledges all alarms that have the same Tagname from the same provider and group name and having the same priority as one or more of the selected alarms.
Syntax:	*Object*.AckSelectedTag
Example:	`#AlarmViewerCtrl1.AckSelectedTag();`

Method:	**AckSelectedPriority**
Purpose:	Acknowledges all alarms that have the same priority from the same provider and group name as one or more of the selected alarms.
Syntax:	*Object*.AckSelectedPriority
Example:	`#AlarmViewerCtrl1.AckSelectedPriority();`

Method:	**ShowSuppression**
Purpose:	Displays the suppression dialog box, which contains all suppressed alarms.
Syntax:	*Object*.ShowSuppression
Example:	`#AlarmViewerCtrl1.ShowSuppression();`

Method:	**SuppressSelected**
Purpose:	Suppress the display of current and future occurrences of the selected alarms.
Syntax:	*Object*.SuppressSelected
Example:	`#AlarmViewerCtrl1.SuppressSelected();`

Method:	**SuppressAll**
Purpose:	Suppress the display of current and future occurrences of all active alarms.
Syntax:	*Object*.SuppressAll
Example:	`#AlarmViewerCtrl1.SuppressAll();`

Method:	**SuppressVisible**
Purpose:	Suppress the display of current and future occurrences of any visible alarm.
Syntax:	*Object*.SuppressVisible
Example:	`#AlarmViewerCtrl1.SuppressVisible();`

Method:	**SuppressSelectedGroup**
Purpose:	Suppress the display of current and future occurrences of any alarm that belongs to the same groups of one or more selected alarms having the same Provider name.
Syntax:	*Object*.SuppressSelectedGroup
Example:	`#AlarmViewerCtrl1.SuppressSelectedGroup();`

Method:	**SuppressSelectedTag**
Purpose:	Suppress the display of current and future occurrences of any alarm that belongs to the same Tagname name of one or more selected alarms having the same Provider name, Group name and Priority range.
Syntax:	*Object*.SuppressSelectedTag
Example:	`#AlarmViewerCtrl1.SuppressSelectedTag();`

Method:	**SuppressSelectedPriority**
Purpose:	Suppress the display of current and future occurrences of any alarm that belongs to the same priorities of one or more selected alarms having the same Provider name and Group tag.
Syntax:	*Object*.SuppressSelectedPriority
Example:	`#AlarmViewerCtrl1.SuppressSelectedPriority();`

Method:	**UnSuppressAll**
Purpose:	Clear alarm suppression.

Method:	**UnSuppressAll**
Syntax:	*Object*.UnSuppressAll
Example:	`#AlarmViewerCtrl1.UnSuppressAll();`

Method:	**ShowQueryFavorites**
Purpose:	Displays the Query Favorites dialog.
Syntax:	*Object*.ShowQueryFavorites
Example:	`#AlarmViewerCtrl1.ShowQueryFavorites();`

Method:	**ShowStatistics**
Purpose:	Displays the Statistics dialog.
Syntax:	*Object*.ShowStatistics
Example:	
	`#AlarmViewerCtrl1.ShowStatistics();`

Method:	**FreezeDisplay**
Purpose:	Freezes the display.
Syntax:	*Object*.FreezeDisplay
Example:	`#AlarmViewerCtrl1.FreezeDisplay();`

Method:	**Requery**
Purpose:	Queries the alarm provider again.
Syntax:	*Object*.Requery
Example:	`#AlarmViewerCtrl1.Requery();`

Method:	**AckGroup**
Purpose:	Acknowledge all alarms that have a given Group name from the same provider.
Syntax:	*Object*.AckGroup
Example:	`#AlarmViewerCtrl1.AckGroup();`

Method:	**AckPriority**
Purpose:	Acknowledges all alarms within a specified priority range having same provider name and group name.
Syntax:	*Object*.AckPriority
Example:	`#AlarmViewerCtrl1.AckPriority();`

Method:	**AckTag**
Purpose:	Acknowledge all alarms that have the same Tagname, Provider name and Group name within the given Priority range.
Syntax:	*Object*.AckTag
Example:	`#AlarmViewerCtrl1.AckTag();`

Method:	**SuppressGroup**
Purpose:	Suppress the display of current and future occurrences of any alarm that belongs to a given Group name, having the same Provider name.
Syntax:	*Object*.SuppressGroup
Example:	`#AlarmViewerCtrl1.SuppressGroup();`

Method:	**SuppressPriority**
Purpose:	Suppress the display of current and future occurrences of any alarm of the specified priority range, having the same Provider name and Group name.
Syntax:	*Object*.SuppressPriority
Example:	`#AlarmViewerCtrl1.SuppressPriority();`

Method:	**SuppressTag**
Purpose:	Suppress the display of current and future occurrences of any alarm emitted by a given tagname name, having the same Provider, Group name and Priority range.
Syntax:	*Object*.SuppressTag
Example:	`#AlarmViewerCtrl1.SuppressTag();`

Method:	**SelectGroup**
Purpose:	Toggles the selection of all the alarms from a given Group within a given Provider.
Syntax:	*Object*.SelectGroup
Example:	`#AlarmViewerCtrl1.SelectGroup();`

Method:	**SelectPriority**
Purpose:	Toggles the selection of all alarms from a given Priority range within a given Group in a given Provider.
Syntax:	*Object*.SelectPriority
Example:	`#AlarmViewerCtrl1.SelectPriority();`

Method:	**SelectTag**
Purpose:	Toggles the selection of all alarms from a specific Provider/Group/Tag. You can also specify a Priority range, or use 1-999.
Syntax:	*Object*.SelectTag
Example:	`#AlarmViewerCtrl1.SelectTag();`

Method:	**ApplyQuery**
Purpose:	Performs a query for either summary or historical alarm information. All query properties are provided in this function.
Syntax:	*Object*.ApplyQuery
Example:	`#AlarmViewerCtrl1.ApplyQuery();`

Method:	**ApplyDefaultQuery**
Purpose:	Performs a query using the configuration dialog default properties. These properties include: From Priority, To Priority, Alarm List, and Display Type. The default properties can only be changed at development time and are not overwritten by other alarm queries.

Method:	**ApplyDefaultQuery**
Syntax:	*Object*.ApplyDefaultQuery
Example:	`#AlarmViewerCtrl1.ApplyDefaultQuery();`

Method:	**SelectAll**
Purpose:	Toggles the selection of all the alarms in a display. Since the alarm display has only a limited display area, the **SelectAll** function may select alarms that are not visible in the display.
Syntax:	*Object*.SelectAll
Example:	`#AlarmViewerCtrl1.SelectAll();`

Method:	**SelectItem**
Purpose:	Selects an alarm record at *nItem*.
Syntax:	*Object*.SelectItem
Example:	`#AlarmViewerCtrl1.SelectItem();`

Method:	**UnSelectAll**
Purpose:	Unselects all the selected records.
Syntax:	*Object*.UnSelectAll
Example:	`#AlarmViewerCtrl1.UnSelectAll();`

Method:	**SetQueryByName**
Purpose:	Start a new alarm query using the query parameters associated with a user defined query name.
Syntax:	*Object*.SetQueryByName
Example:	`#AlarmViewerCtrl1.SetQueryByName();`

Method:	**MoveWindow**
Purpose:	Provides commands to manipulate the display window. These commands include: Page Up, Page Down, Scroll Right, Scroll Left, Line Up, Line Down, Top, End, and more.
Syntax:	*Object*.MoveWindow
Example:	`#AlarmViewerCtrl1.MoveWindow();`

Method:	**GetItem**	
Purpose:	Returns the data at a specified row & column as string.	
Syntax:	*Object*.GetItem*(Integer, message)*	
	Integer	An integer expression that evaluates to a specific row in the control
	Message	A string expression that evaluates to the column name in the control
Example:	`tagname = #AlarmViewerCtrl1.GetItem(1, "Group");`	

Method:	**ShowContext**
Purpose:	Shows the context sensitive menu if any one of "RefreshMenu" or "ResetMenu" or "SortMenu" properties are enabled.
Syntax:	*Object*.ShowContext
Example:	`#AlarmViewerCtrl1.ShowContext();`

Method:	**ShowSort**
Purpose:	Shows the "Secondary Sort" dialog.
Syntax:	*Object*.ShowSort
Example:	`#AlarmViewerCtrl1.ShowSort();`

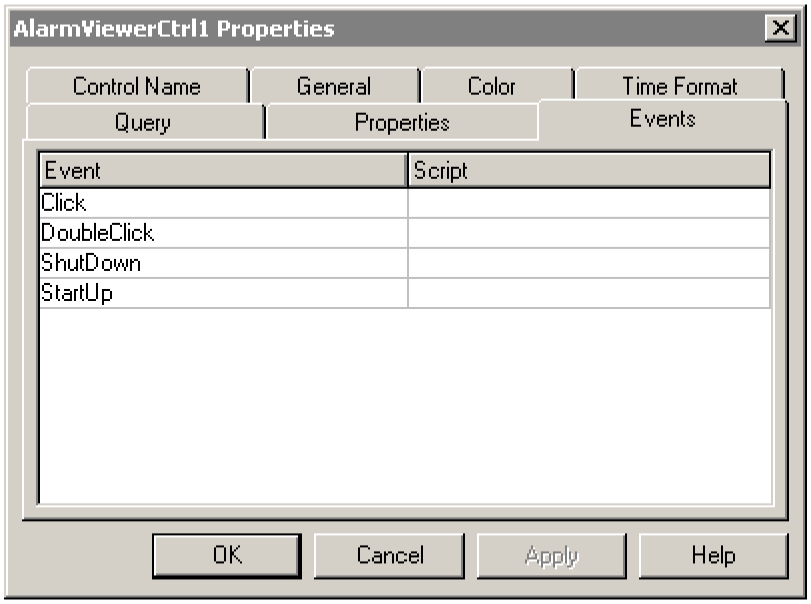
Method:	**SetSort**
Purpose:	Sets the sort criteria as specified by the "SortColumn" and "SortOrder" properties.
Syntax:	*Object*.SetSort
Example:	`#AlarmViewerCtrl1.SetSort();`

Assign ActiveX Scripts to the Alarm Viewer

To assign ActiveX Scripts to the Alarm Viewer

1. Double-click on the Alarm Viewer ActiveX display or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.

2. Click the **Events** tab to activate the **Events** property sheet.



Event	Script
Click	
DoubleClick	
ShutDown	
StartUp	

3. Click the event for which to assign a script.
4. Enter a script name in the script field or click the button to browse for an ActiveX script. If you elected to browse for an ActiveX script, click OK once you have selected the script.
5. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Event:	**Click**	
Purpose:	Occurs when the user clicks the left mouse button and releases it over the control.	
Syntax:	`*tagname* = *#Thisevent*.clicknRow`	
	`nRow`	An integer expression reflecting the row where this event happened.
	`#Thisevent`	Keyword

Note The ActiveX Alarm Viewer control ignores the UI methods when the method is called from OnStartup event, since the control is not visible yet. The UI methods listed below will be ignored if they are called from OnStartup event:
"ShowSort", "ShowContext", "GetSelectedItem", "GetNext", "GetPrevious" & "AboutBox".

Event:	DoubleClick	
Purpose:	Occurs when the user double-clicks the left mouse button and releases it over the control.	
Syntax:	*Tagname = #ThisEvent*.DoubleClicknRow	
	NRow	An integer expression reflecting the row where this event happened.
	#Thisevent	Keyword

Click and DoubleClick events are zero-based. When Click and/or DoubleClick events are published for the user, the row count in the display starts with 0

Right-click Menu

The ActiveX Alarm Viewer provides you with a right-click menu for quick access to the commands that you can apply to the display object and/or one or more selected alarms, Alarm Groups, Tagnames and Priorities contained in the display at runtime.

- **Ack Selected** - Acknowledges the selected alarm.
- **Ack Others -** When the operator right-clicks the display and then points to **Ack Others**, a sub-menu opens that contains ACK commands.
    - **Ack All** - Acknowledges all the alarms in the current alarm query. Since the alarm display has only a limited display area, the **AckAll** function may acknowledge alarms that are not visible in the display.
    - **Ack Visible** - Acknowledges only those alarms that are currently visible in the alarm display.
    - **Ack Selected Groups** - Acknowledges all alarms that have the same group name from the same provider as one or more of the selected alarms.
    - **Ack Selected Tags** - Acknowledges all alarms that have the same Tagname from the same provider and group name and having the same priority as one or more of the selected alarms.
    - **Ack Selected Priorities** - Acknowledges all alarms that have the same priority from the same provider and group name as one or more of the selected alarms.

Note An ACK on an alarm that has already been acknowledged has no effect.

- **Suppress Selected** - When the operator right-clicks the display and then clicks **Suppress Selected** during runtime, the operator can quickly Suppress the selected alarm.
- **Suppress Others** - When the operator right-clicks the display and then points to **Suppress Others**, a sub-menu opens that contains suppression commands.

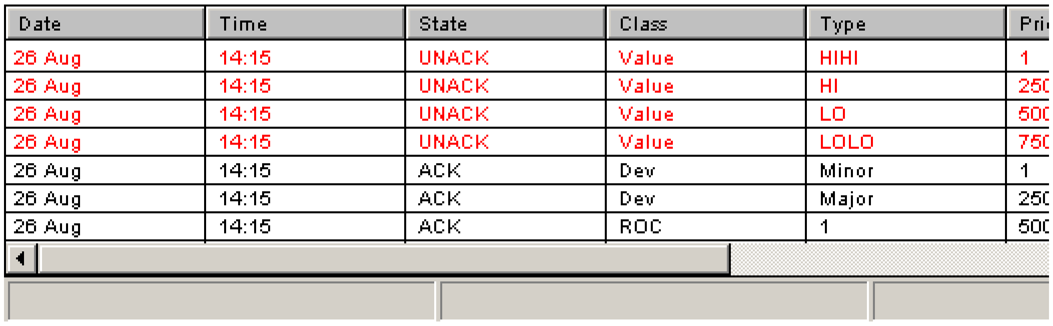
  - **Suppress All** - Suppress the display of current and future occurrences of all alarms.
  - **Suppress Visible** - Suppress the display of current and future occurrences of any visible alarm.
  - **Suppress Selected Groups** - Suppress the display of current and future occurrences of any alarm that belongs to the same groups of one or more selected alarms having the same Provider name.
  - **Suppress Selected Tags** - Suppress the display of current and future occurrences of any alarm that belongs to the same Tagname name of one or more selected alarms having the same Provider name, Group name and Priority range.
  - **Suppress Selected Priorities** - Suppress the display of current and future occurrences of any alarm that belongs to the same priorities of one or more selected alarms having the same Provider name and Group name.
  - **Unsuppress All** - Clears the suppression settings.
- **Query Favorites** - When you right-click the display and then click **Query Favorites**, the **Alarm Query** dialog box appears.

  For more information see Selecting and Configuring Alarm Query Favorites.
- **Stats** - Brings up the alarm statistics dialog box.
- **Suppression** - Brings up the Alarm Suppression dialog box.
- **Freeze** - Freezes the current display.
- **Requery** - Queries the alarm provider again.
- **Sort** - Brings up the Secondary Sort dialog box.

The Distributed Alarm Display

The Distributed Alarm System has a single display object to show both locally and remotely generated alarms. This display object's features include: built in scroll bars, resizable display columns, multiple selection of alarms, update status bar, query status bar, context sensitive right-click menu, and alarm display colors based on alarm priority.

InTouch allows you to modify the appearance of the alarm display (including the information that is displayed), the colors used for various alarm conditions, and the Alarm Group and alarm priority levels displayed.



Date	Time	State	Class	Type	Pri
26 Aug	14:15	UNACK	Value	HIHI	1
26 Aug	14:15	UNACK	Value	HI	250
26 Aug	14:15	UNACK	Value	LO	500
26 Aug	14:15	UNACK	Value	LOLO	750
26 Aug	14:15	ACK	Dev	Minor	1
26 Aug	14:15	ACK	Dev	Major	250
26 Aug	14:15	ACK	ROC	1	500

Scroll Bars

The distributed alarm display has built-in horizontal and vertical scroll bars that allow you to move through listed alarms. You can configure the display of these scroll bars.

Next/Prev Page Controls

The **.NextPage** and **.PrevPage** alarm display control properties are also supported for the distributed alarm display.

For more information see, "Alarm Display Control Properties."

Sizable Display Columns

The distributed alarm display uses a grid to hold the alarm messages. This grid allows for dynamic sizing of the column widths simply by selecting a column and dragging it to set the column width. This functionality is available only during runtime. You can configure whether or not the grid can be used to size the columns.

Tip Grid column changes are not saved; therefore, if you make grid column changes and close the window containing the alarm display, the grid columns will again be at their default width upon re-opening that window. Double-click on the vertical grid line to autosize the column.

Multiple Selection

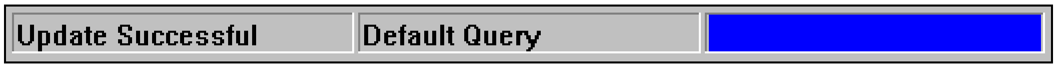
The grid allows you to select a single or multiple alarms in a list box. The selected alarms can be acknowledged by using the **almAckSelect()** QuickScript function described later in this chapter. When you configure the distributed alarm display, you can also define the selection behavior to allow either toggle selection (item by item), or multiple selection (holding down CTRL or SHIFT in conjunction with a mouse click to select multiple alarms). You can turn off runtime selection.

Alarm Message Colors

You can also configure up to eight different colors for each displayed alarm message, based on the priority of the alarm and whether it is acknowledged or not.

Update Status Bar

The distributed alarm display includes a status bar that contains three indicators: A status message, current alarm query, and a progress bar. These indicators provide an overview of the current state of the display query and provide details about the suppression available in the Distributed Alarm Object. The right pane of the status bar is red when freeze is in effect and the left pane of the status bar is red when suppression is in effect. The word "suppression" displays in the left pane when suppression is in effect. You can turn off the display of the status bar in runtime.



Feature	Description
Status Message	The status message at the left end of the status bar provides a more detailed description of the current query status.
Progress Bar	The update progress bar at the right end of the status bar provides a visual indication of the current query progress.
Alarm Query	The Alarm Query provides a visual indication of the current alarm query. **Note** If new alarms are received while the display is frozen, the words "New Alarm (s)" appears in the status bar.

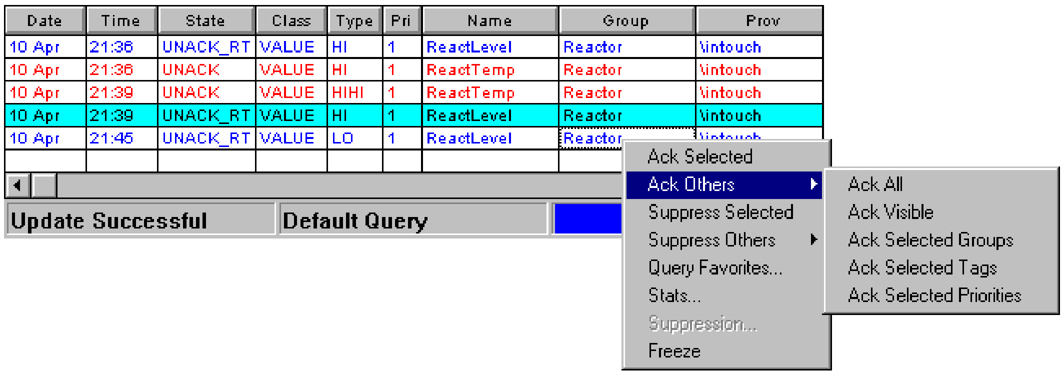
Status Message	State/Indicator	Progress Bar
None	No Query	None
Update Incomplete	Query Incomplete	Blue/Green
Update Successful	Query Complete	Solid Blue
Suppression	Query name	Solid Blue
Freeze	Query name	Red
New Alarm (s)	Query name	Red

Right-click Menu

The Distributed Alarm Display object provides you with a right-click menu for quick access to the commands that you can apply to the display object and/or one or more selected alarms, Alarm Groups, Tagnames and Priorities contained in the display at runtime.

Note By default, the right-click menu is disabled. It can be enabled by setting the appropriate value in the Property sheets. Also, script functions can be called to do anything you can do with the right-click menu.

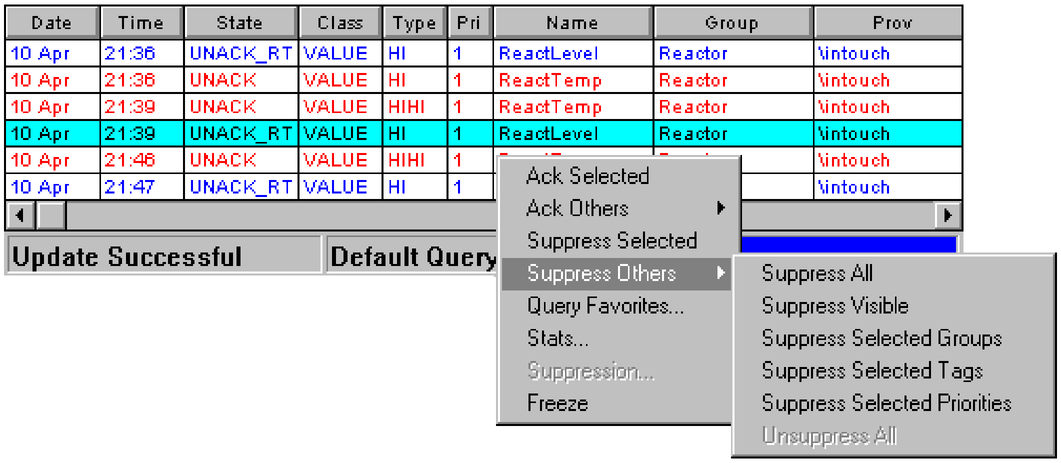
- **Ack Selected** - When the operator right-clicks the display and then points to **Ack Selected** during runtime, the operator can quickly acknowledge the selected alarm.
- **Ack Others** - When the operator right-clicks the display and then points to **Ack Others**, a sub-menu opens that contains acknowledgment commands. During runtime, the operator can quickly acknowledge all alarms in the display, or only visible alarms, selected groups, selected tagnames and selected priorities by using this right-click functionality.



Date	Time	State	Class	Type	Pri	Name	Group	Prov
10 Apr	21:36	UNACK_RT	VALUE	HI	1	ReactLevel	Reactor	\intouch
10 Apr	21:36	UNACK	VALUE	HI	1	ReactTemp	Reactor	\intouch
10 Apr	21:39	UNACK	VALUE	HIHI	1	ReactTemp	Reactor	\intouch
10 Apr	21:39	UNACK_RT	VALUE	HI	1	ReactLevel	Reactor	\intouch
10 Apr	21:45	UNACK_RT	VALUE	LO	1	ReactLevel	Reactor	

Update Successful | Default Query

- **Suppress Selected** - When the operator right-clicks the display and then points to **Suppress Selected** during runtime, the operator can quickly Suppress the selected alarm.
- **Suppress Others** - When the operator right-clicks the display and then points to **Suppress Others**, a sub-menu opens that contains suppression commands. During runtime, the operator can quickly suppress all alarms in the display, or only visible alarms, selected groups, selected tagnames and selected priorities by using this right-click functionality.



Date	Time	State	Class	Type	Pri	Name	Group	Prov
10 Apr	21:36	UNACK_RT	VALUE	HI	1	ReactLevel	Reactor	\intouch
10 Apr	21:36	UNACK	VALUE	HI	1	ReactTemp	Reactor	\intouch
10 Apr	21:39	UNACK	VALUE	HIHI	1	ReactTemp	Reactor	\intouch
10 Apr	21:39	UNACK_RT	VALUE	HI	1	ReactLevel	Reactor	\intouch
10 Apr	21:46	UNACK	VALUE	HIHI	1			\intouch
10 Apr	21:47	UNACK_RT	VALUE	HI	1			\intouch

Update Successful | Default Query

- **Query Favorites** - When you right-click the display and then point to **Query Favorites**, the **Alarm Query** dialog box appears.

  For more information see "Selecting and Configuring Alarm Query Favorites."
- **Stats** - Brings up the alarm statistics dialog box.
- **Suppression** - Brings up the Alarm Suppression dialog box.
- **Freeze** - Freezes the current display.

Selecting and Configuring Alarm Query Favorites

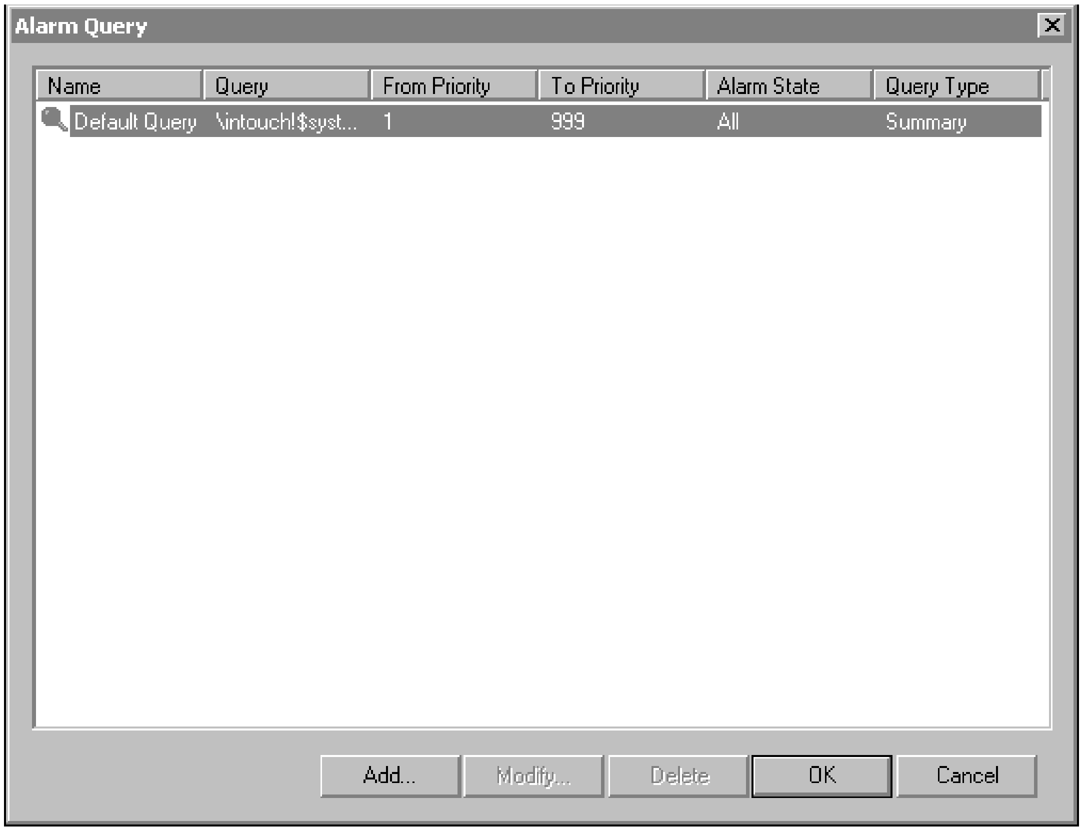
The **Query Favorites** command on the Distributed Alarm Display object's right-click menu provides you with the ability to quickly select an alarm query for display from a list of previously defined alarm queries. Additionally, by using the **Query Favorites** command, you can create new named queries, edit an existing query, or delete an existing query.

Note For multi-line alarm queries appearing in the Distributed Alarm Display, line separations display as "garbage" characters. This will not affect the function.

To select an alarm query for display

1. Right-click the Distributed Alarm Display object in **Runtime**.

2. Click **Query Favorites**. The **Alarm Query** dialog box appears.



Name	Query	From Priority	To Priority	Alarm State	Query Type
Default Query	\intouch!$syst...	1	999	All	Summary

3. Select the named query that you want to display in the list of currently defined queries.
4. Click **OK**. The Distributed Alarm Display object now displays the alarm information for the selected named query.

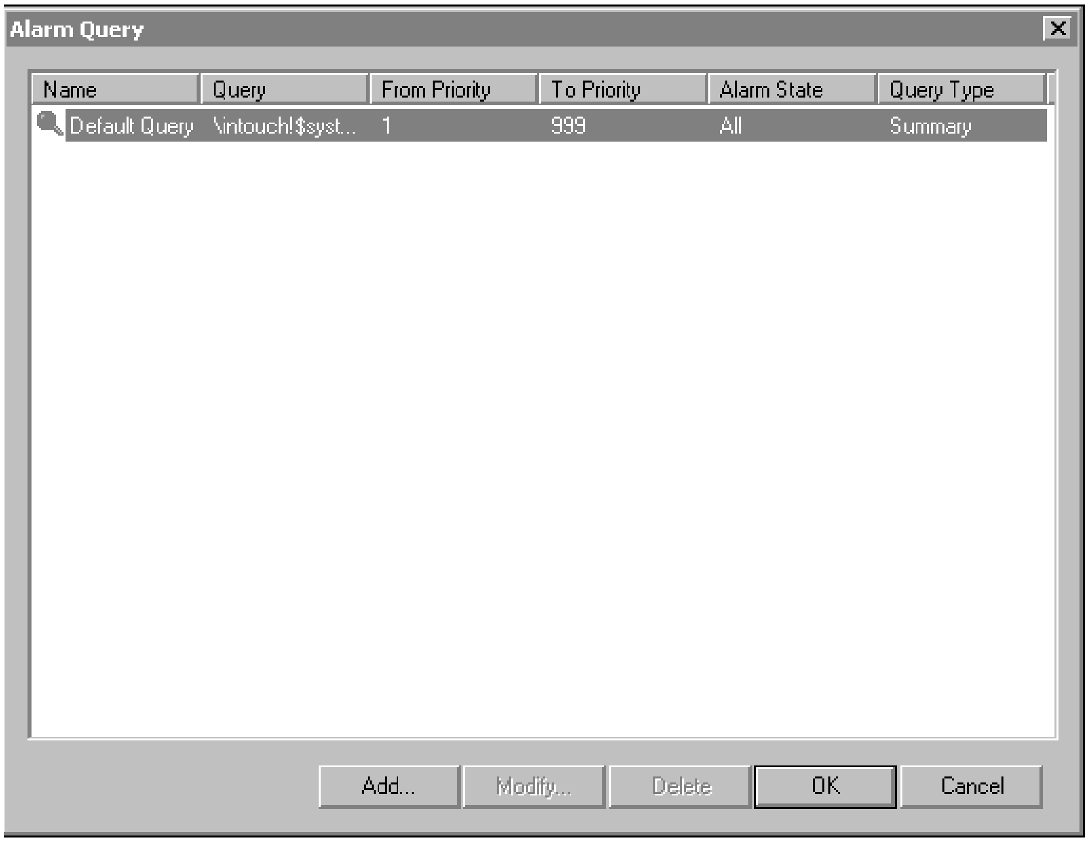
   For more information on dynamically controlling the Distributed Alarm Display object, see "Distributed Alarm Display Properties and Functions."

Also see "Alarm Query QuickScript Functions."

To add a new named query

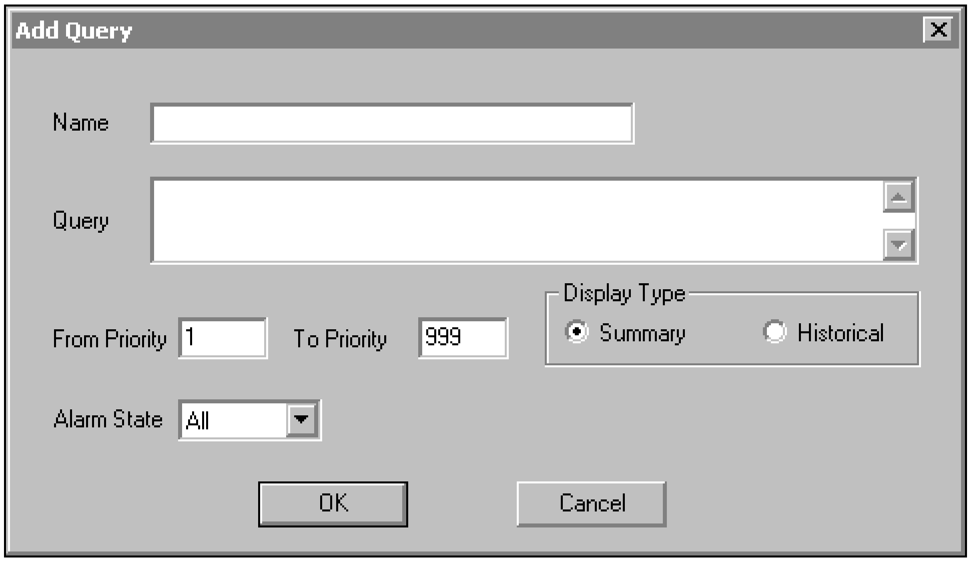
1. Right-click the Distributed Alarm Display object in **Runtime**.

2. Click **Query Favorites**. The **Alarm Query** dialog box appears.



Name	Query	From Priority	To Priority	Alarm State	Query Type
Default Query	\intouch!$syst...	1	999	All	Summary

3. Click **Add**. The **Add Query** dialog box appears.



From Priority: 1
To Priority: 999
Display Type: [✓ Summary] Historical
Alarm State: All

4. In the **Name** input box, type the name that you want to use for the query.

5. In the **Query** input box, type the sets of InTouch alarm queries that you want to perform. You can specify one or more Alarm Providers and groups, just as with the alarm query for the Alarm Printer Utility.

   For more information see "Alarm Viewer ActiveX Display Properties."
6. In the **From Priority** field, type the minimum alarm priority value (1 to 999).
7. In the **To Priority** field, type the maximum alarm priority value (1 to 999).

   Note Each alarm configured in InTouch has a priority value associated with it. This value represents the importance of the alarm and can range from 1 to 999 with 1 being the most important.

   For more information on alarm priorities, see "Alarm Priorities."
8. Click the **Alarm State** arrow and select the alarm state that you want to use in the alarm query.
9. Select the option for the **Display Type** that you want to use.

   For more information, see "Summary Alarms versus Historical Alarms."

   Also see "Configuring an Alarm Viewer ActiveX Control."
10. Click **OK** to close the **Add Query** dialog box.
11. Click **OK** on the **Alarm Query** dialog box to commit the addition.

To modify an existing named query

1. Right-click the Distributed Alarm Display object.
2. Click **Query Favorites**. The **Alarm Query** dialog box appears.
3. Select the named query that you want to modify in the list of currently defined queries.
4. Click **Modify**. The **Modify Query** dialog box appears.
5. Make the necessary modifications and then, click **OK** to close the **Modify Query** dialog box.
6. Click **OK** on the **Alarm Query** dialog box to commit the modification.

Note Modification will not be automatically applied to other Distributed Alarm Objects that are using the alarm query being modified.

To delete an existing named query

1. Right-click the Distributed Alarm Display object.
2. Click **Query Favorites**. The **Alarm Query** dialog box appears.
3. Select the named query that you want to delete in the list of currently defined queries.
4. Click **Delete**. A message box appears asking you to verify deletion of the selected named query.
5. Click **Yes**.
6. Click **OK** on the **Alarm Query** dialog box to commit the deletion.

Note Deletion will not be automatically applied to other Distribute Alarm Objects that are using the alarm query being deleted.

Distributed Alarm Display Guidelines

The distributed alarm display requires that certain guidelines to be observed when using objects such as the Distributed Alarm Display. These guidelines are as follows:

Each display must have an identifier so that any associated QuickScript functions know which display to modify. This identifier, entered as **Display Name** in the **Alarm Configuration** dialog box, must be unique for each display.

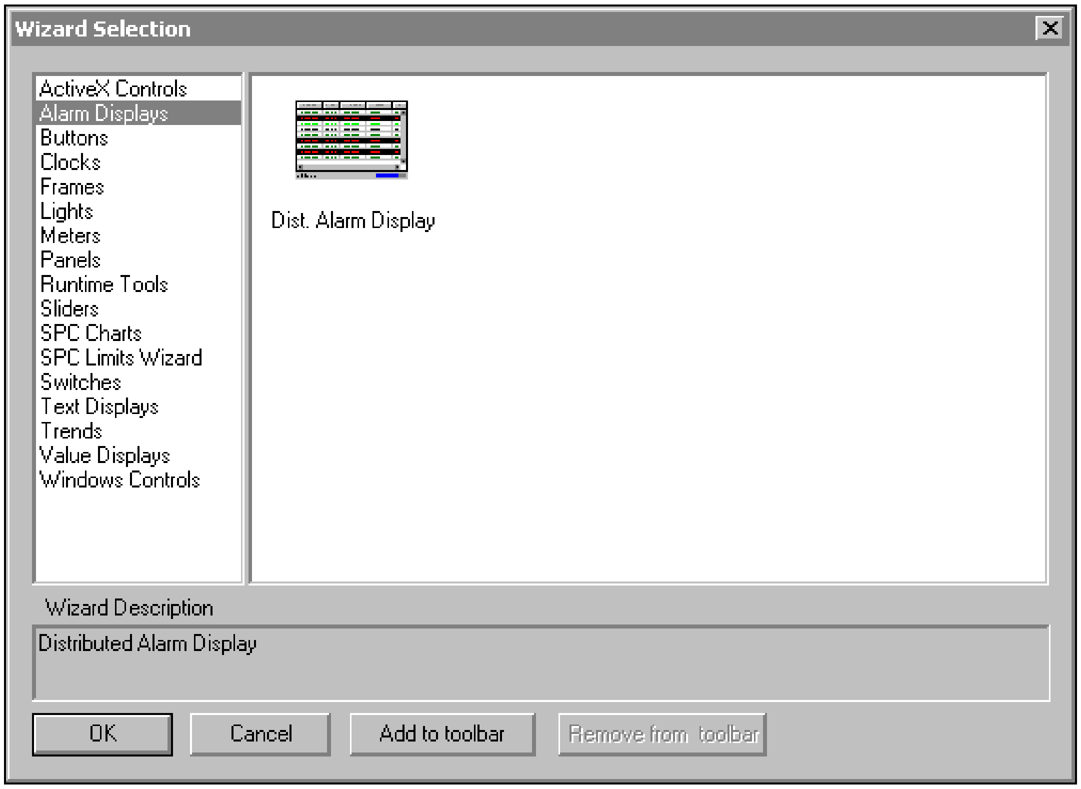
Displays should not overlap other InTouch objects such as window controls or graphic objects. You can easily verify this by clicking on the distributed alarm display in WindowMaker, and checking the display's "handles." The handles should not touch other graphic objects on the screen.

Displays should be used sparingly. Placing numerous displays on one screen can result in reduced system performance. When possible, limit the number of displays on your screen and call further screens (dialog boxes) with additional displays if necessary.

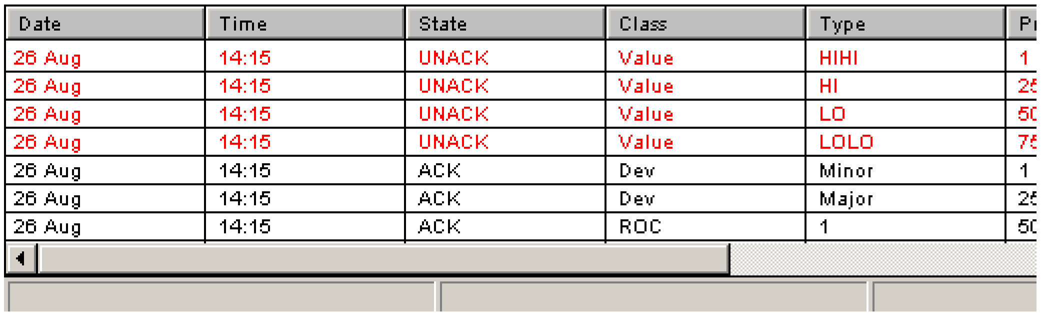
Creating a Distributed Alarm Display

To create a distributed alarm display

1. Click the **Wizard** tool in the **Wizard/ActiveX Toolbar**. The **Wizard Selection** dialog box appears.



2. Select **Alarm Displays** in the list of wizards.
3. Double-click the **Dist. Alarm Display** wizard, or select it and then click **OK**. The dialog box closes and your window reappears with the cursor in the "paste" mode.
4. Click in the window to paste the Alarm Display wizard.



Date	Time	State	Class	Type	P
26 Aug	14:15	UNACK	Value	HIHI	1
26 Aug	14:15	UNACK	Value	HI	25
26 Aug	14:15	UNACK	Value	LO	50
26 Aug	14:15	UNACK	Value	LOLO	75
26 Aug	14:15	ACK	Dev	Minor	1
26 Aug	14:15	ACK	Dev	Major	25
26 Aug	14:15	ACK	ROC	1	50

Tip To size the wizard, point to one of its selection handles then drag it until the desired size is reached.

5. You are now ready to configure the display.

Configuring a Distributed Alarm Display

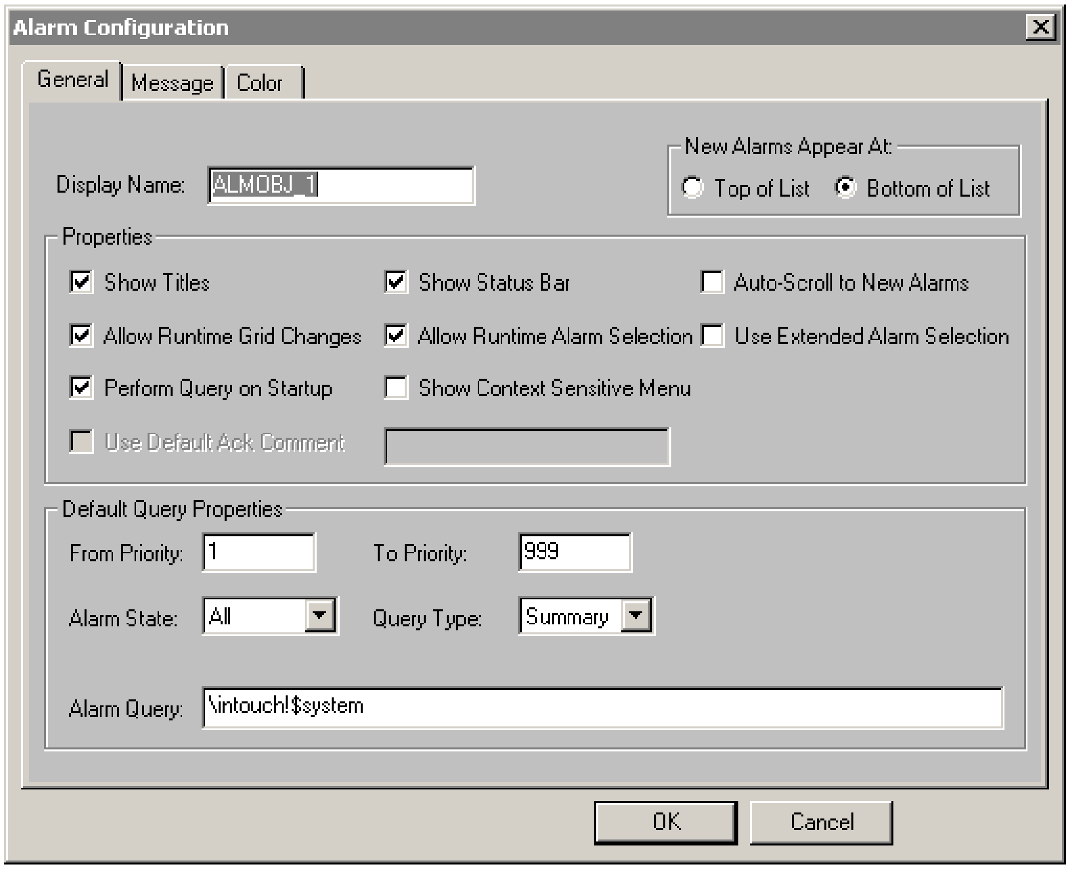
The **Alarm Configuration** dialog box has three property sheets that contain the options for **General**, **Message**, and **Color** configuration.

Note The configuration dialog box behaves like any standard Windows property sheet in that no settings are recorded until you click **OK**. The options are verified for proper entries, however, when you change from one property sheet (tab) to another, if an entry verification fails, the property sheet containing the failed entry is brought back into focus, and a message box appears indicating the error. If you click **Cancel**, all input is ignored and the dialog box closes.

Distributed Alarm Display General Properties

To configure a distributed alarm display

1. Double-click on the distributed alarm display or right-click it, and then click **Properties**. The **Alarm Configuration** dialog box appears with the **General** property sheet active:



Display Name: ALMOBJ_1

New Alarms Appear At: Top of List [✓ Bottom of List]

Properties

- ✓ Show Titles
- ✓ Show Status Bar
- ☐ Auto-Scroll to New Alarms
- ✓ Allow Runtime Grid Changes
- ✓ Allow Runtime Alarm Selection
- ☐ Use Extended Alarm Selection
- ✓ Perform Query on Startup
- ☐ Show Context Sensitive Menu
- ☐ Use Default Ack Comment

Default Query Properties

From Priority: 1
To Priority: 999
Alarm State: All
Query Type: Summary
Alarm Query: \intouch!$system

Tip If you right-click a text box in any alarm configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

2. In the **Display Name** box, type the name for the alarm display. This name must be unique for each alarm display used.

Tip The name you type here will be used throughout the system for referring to this object for execution of tasks such as alarm acknowledgment and queries.

3. Select the **New Alarms Appear At** option for where you want new alarms to appear in the object:

Option	Description
Top of List	Displays the most recent alarm at the top of the list.
Bottom of List	Displays most recent alarm at the bottom of the list.

4. Select the **Properties** as described below:

Property	Description
Show Titles	Displays alarm message title bar.
Use Default Ack Comment	Controls whether a default comment will be used when an operator ACKs an alarm. If this box is checked and a string is entered, the string will be used during runtime as a default comment. If this box is not checked, when the oprator ACKs an alarm, a dialog box appears to let the operator enter a comment. The dialog box can be filled in or left blank.
Show Status Bar	Displays status bar.
Allow Runtime Grid Changes	Allows the user to change column settings in runtime.
Perform Query on Startup	Automatically begins updating the display using default query properties, if selected. If not selected, you need to perform an **almDefQuery** or **almQuery** before the display will update.
Auto-Scroll to New Alarms	If the user scrolls the list from the beginning, this automatically jumps to the new alarm. (New alarms are defined as those that are not currently displayed within the display object.)
Allow Runtime Alarm Selection	Allows user to select alarms at runtime.
Use Extended Alarm Selection	Allows multiple alarms to be selected by holding down Ctrl or Shift in conjunction with a mouse. The default is to toggle selection of alarms by simply clicking on them (visible only if **Allow Runtime Alarm Selection** check box is selected).
Show Context Sensitive Menu	Enables the activation of the right-click popup menu.

5. Select the **Default Query Properties** options as described below:

Tip The **Default Query Properties** are used if you select the **Perform Query on Startup** option or, if the **almDefQuery** QuickScript function is executed.

Property	Description
From Priority	Default minimum alarm priority.
To Priority	Default maximum alarm priority. For more information on alarm priorities, see the "Alarm Priorities" section of this chapter.
Alarm State	Default alarm state to query (All, UnAck, Ack).
Query Type	Sets display type as either Summary or Historical.
Alarm Query	Sets the initial Alarm query. This field accepts text only; it does not accept tags. The valid syntax for these lists include: **\\Node\InTouch!Group** Full path to Alarm Group **\InTouch!Group** Full path to local Alarm Group **GroupList** Another Group List

Note To perform multiple queries, separate each query with a space.

For example: \\Master\InTouch!MyGroup LocalGroupList

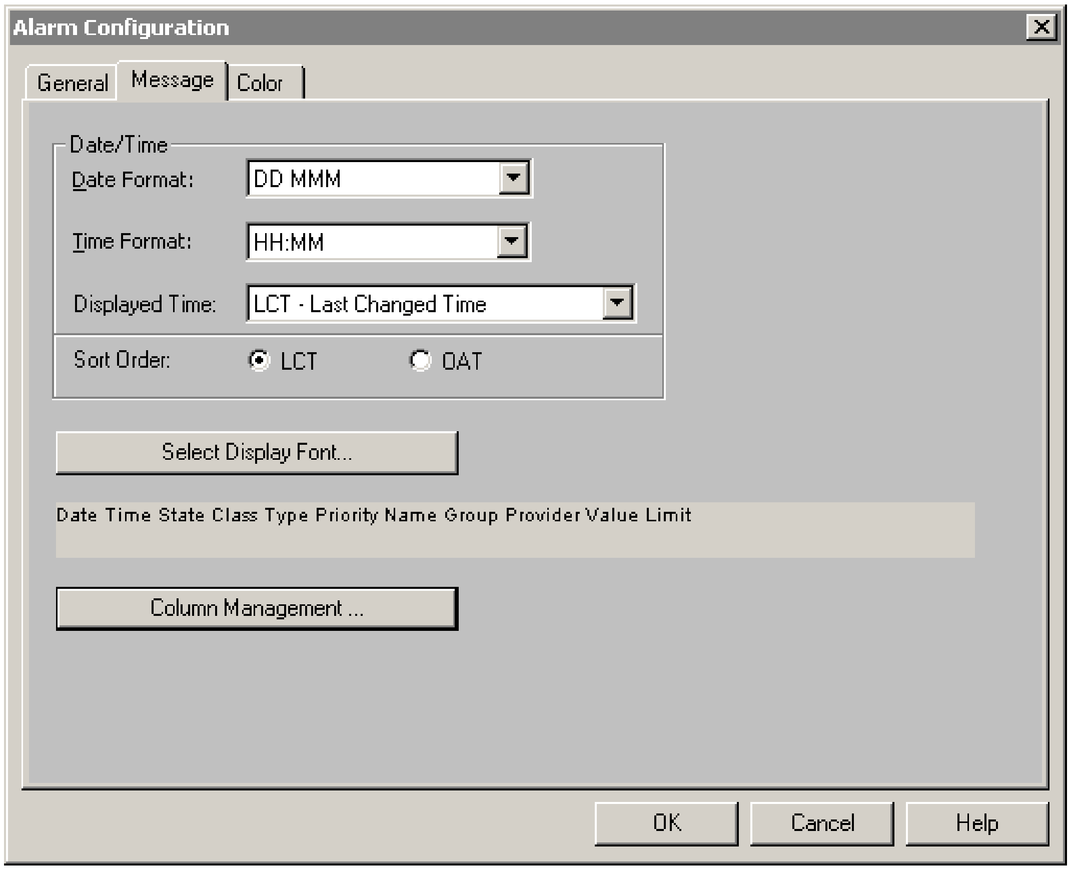
Distributed Alarm Message Format

The information shown in a distributed alarm display object is configurable. For example, information you want displayed and in some cases, how many characters you want displayed for an item.

To configure the alarm display message format

1. Double-click the desired DAO. The **Alarm Configuration** dialog box will appear.

2. Click the **Message** tab to activate the **Message** property sheet:



Date/Time
Date Format: DD MMM
Time Format: HH:MM
Displayed Time: LCT - Last Changed Time
Sort Order: [✓ LCT] OAT

Date Time State Class Type Priority Name Group Provider Value Limit

Tip If you right-click a text box in any alarm configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

The preview area (at the bottom of the dialog) displays an example of the alarm message as currently configured. This example will show the message using the font selected, but not color.

3. Click the **Date Format** drop-down arrow to select the format for the date. The available formats are:

Selection	Display	Selection	Display
DD MMM	28 Feb	**MM/DD**	02/28
DD MM YYYY	28 Feb 2002	**MM/DD/YY**	02/28/02
DD/MM	28/02	**MMM DD**	Feb 28
DD/MM/YY	28/02/02	**MMM DD YYYY**	Feb 28 2002
YY/MM/DD	02/02/28	**YYYY/MM/DD**	2002/02/28

4. Click the **Time Format** arrow to select format for the time. The values in this field are used as a template to specify the format of the time. For example, to specify the time as **10:24:30 AM**, use **HH:MM:SS AP**. The template characters are as follows:

AP	Selects the AM/PM format. For example, three o'clock in the afternoon is displayed as 3:00 PM. A time without this designation defaults to 24 hour military time format. For example, three o'clock in the afternoon is displayed as 15:00.
HH	Displays the hour the alarm/event occurred.
MM	Displays the minute the alarm/event occurred.
SS	Displays the second the alarm/event occurred.
SSS	Displays the millisecond the alarm/event occurred.

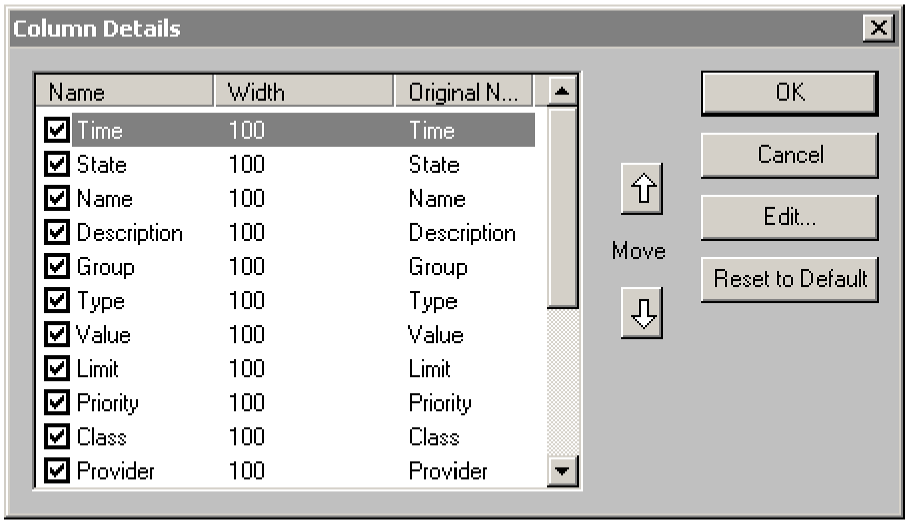
5. Using the radio buttons below the **Displayed Time** drop-down menu, select the order in which you want the alarms to be sorted in the alarm object. There are two choices:

OAT	Original Alarm Time - that is, the date/time stamp of the onset of the alarm.
LCT	Last Changed Time - that is, the date/time stamp of the most recent change of status for the instance of the alarm: onset of the alarm, change of sub-state, return to normal, or acknowledgment.

6. Click **OK**.

Column Management Button

The Column Details dialog box is invoked by clicking the **Column Management** button on the **Message** property sheet. The Column Details dialog box is used to select the columns to display, specify the display order, and to set the column names and widths.



Column Details

Name	Width	Original N...
✓ Time	100	Time
✓ State	100	State
✓ Name	100	Name
✓ Description	100	Description
✓ Group	100	Group
✓ Type	100	Type
✓ Value	100	Value
✓ Limit	100	Limit
✓ Priority	100	Priority
✓ Class	100	Class
✓ Provider	100	Provider

To configure the display column details

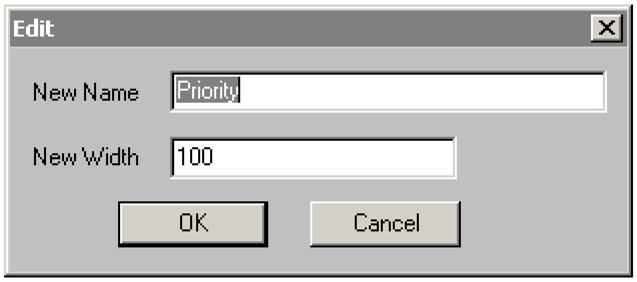
1. Double-click on the distributed alarm display or right-click it, and then click **Properties**. The **Alarm Configuration** dialog box appears with the **General** property sheet active.
2. Click the **Message** tab.
3. Click the **Column Management** button. The Column Details dialog appears.
4. From the **Column Details** dialog box, select the checkbox next to the **Column Name** to display the column in the alarm object. The column names that you can select to display are described below.

Note At least one column must be selected.

Column	Description
Date	Displays the date in the format selected from the **Message** tab.
State	Displays the state of the alarm.
Time	Displays the time in the format selected from the **Message** tab.
Name	Displays the alarm/tagname.
Description	Displays the decription of the alarm.
Group	Displays the Alarm Group name.
Type	Displays the alarm type.
Value	Displays the value of the tagname when the alarm occurred. The width of the column should be large enough to provide the desired level of precision.
Limit	Displays the alarm limit value of the tagname. The width of the column should be large enough to provide the desired level of precision.
Priority	Displays the alarm priority.
Class	Displays the category of the alarm.
Provider	Displays the name of the alarm provider.
Operator	Displays the logged-on operator’s ID associated with the alarm condition.
UTC Time	Displays the UTC Time, also known as Greenwich Mean Time, Coordinated Universal Time or Zulu, for an alarm.
Comment	Displays the tagname comments. These comments were typed in the Alarm Comment box when the tagname’s alarm was defined in the database.

Note All column names are selected by default except for **Operator** and **Comment**.

5. To rearrange the columns, select the column name and use the **Move** Up and Down arrow keys. The column name appearing at the top of the **Column Details** dialog box is the column displayed to the furthest left of the alarm display.
6. To edit the column name and width, select a column name and then click **Edit.** The **Edit** dialog box appears for that column.



New Name: Priority

New Width: 100

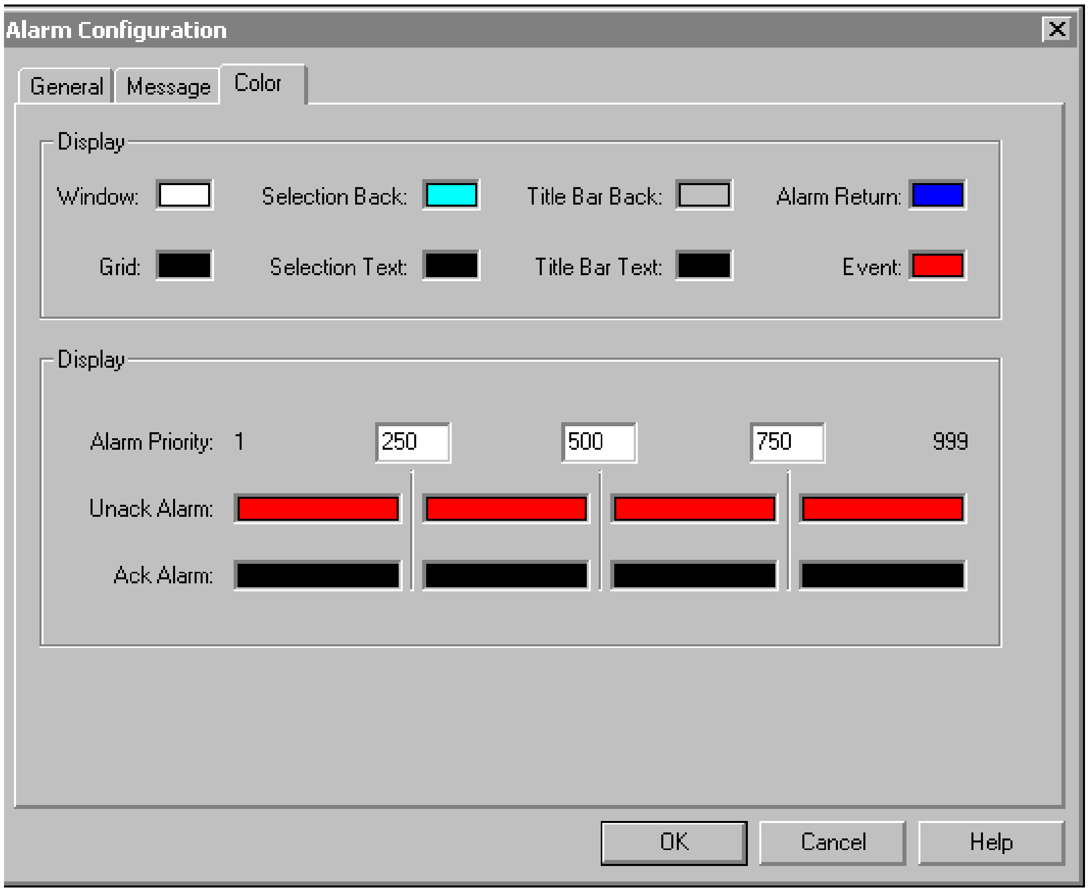
7. Enter in a new name in the **New Name** text box if you want to display a column name other than the default column name.
8. Enter in a column width in the **New Width** text box. The column width is measured in pixels and can range from 1 to 999 pixels. The default column width is 100 pixels.
9. Click **OK** on the **Edit** dialog.
10. Click **OK** on the **Column Details** dialog.
11. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Distributed Alarm Display Color Properties

To configure the alarm display colors

1. Double-click the distributed alarm display object or right-click it, and then click **Properties**. The **Alarm Configuration** dialog box will appear.

2. Click the **Color** tab to activate the **Color** property sheet:



Tip If you right-click a text box in any alarm configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

3. In the top **Display** group, click each color box to open the InTouch Palette. Click the color that you want to use in the palette for each of the following:

Option	Description
Window	Sets display background color.
Grid	Sets display grid color.
Selection Back	Sets highlighted text background color.
Selection Text	Sets highlighted text color.
Title Bar Back	Sets title bar background color (visible only if Show Titles option is on).
Title Bar Text	Sets title bar text color (visible only if Show Titles option is on).
Alarm Return	Sets color of returned alarms (alarms that have returned to normal without being acknowledged).
Event	Sets color of Event alarms.

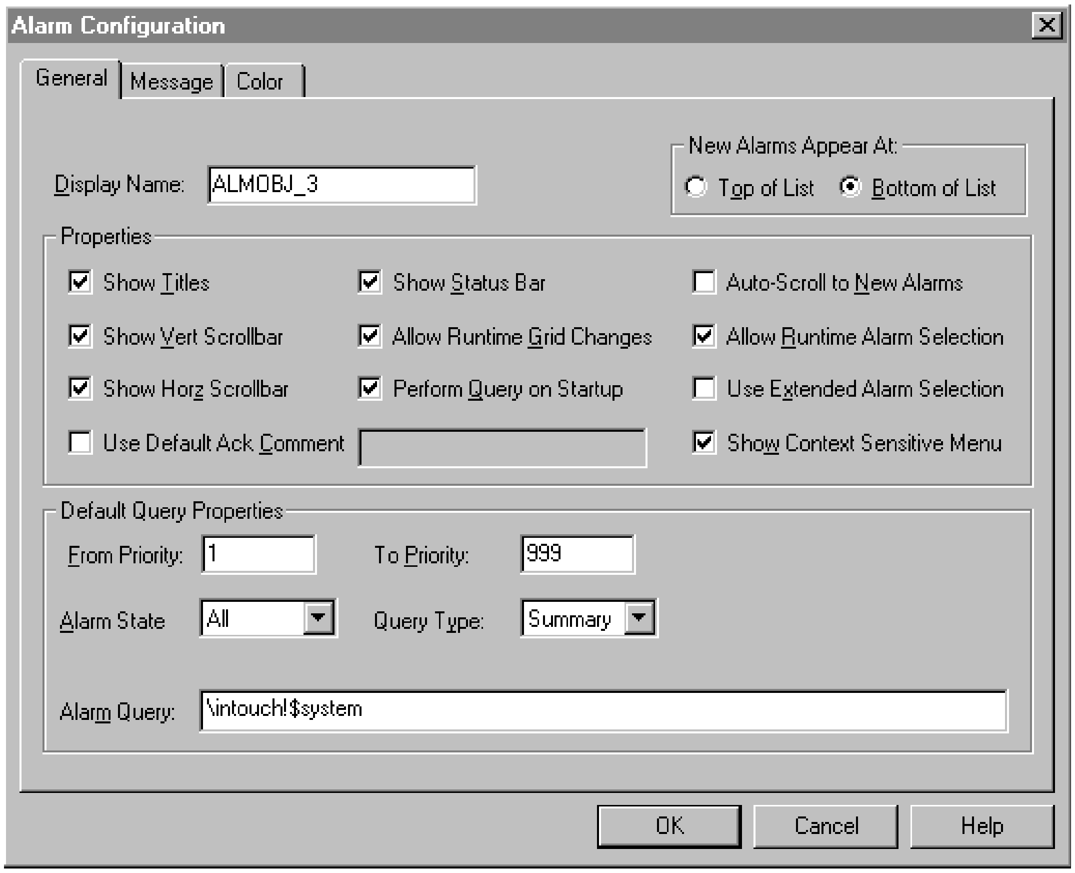
4. In the **Alarm Priority** boxes, type the breakpoint values for the alarm display.
5. Click the **UnAck Alarm** and **Ack Alarm** color boxes to open the InTouch palette. Click the color in the palette that you want to use.
6. Click **OK**.

Configuring the Display Alarm Query

The distributed alarm display can show summaries of active alarms or listings of historical alarms. The selection of whether to show summaries or historical alarms can be changed dynamically.

To configure the distributed alarm object query default

1. Double-click the distributed alarm object or right-click it, and then click **Properties**. The **Alarm Configuration** dialog box appears with the **General** property sheet active:



Display Name: ALMOBJ_3

New Alarms Appear At: Top of List [✓ Bottom of List]

Properties

- [✓] Show Titles
- [✓] Show Status Bar
- [ ] Auto-Scroll to New Alarms
- [✓] Show Vert Scrollbar
- [✓] Allow Runtime Grid Changes
- [✓] Allow Runtime Alarm Selection
- [✓] Show Horz Scrollbar
- [✓] Perform Query on Startup
- [ ] Use Extended Alarm Selection
- [ ] Use Default Ack Comment
- [✓] Show Context Sensitive Menu

Default Query Properties

From Priority: 1
To Priority: 999
Alarm State: All
Query Type: Summary
Alarm Query: \intouch!$system

Tip If you right-click a text box in any alarm configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

---

2. Click the **Query Type** arrow and select the type of alarm display that you want to use for the runtime default.

   For example, in runtime, the display type is determined by the query QuickScript function that you use with it. If you run an *almQuery( )* QuickScript against display *AlarmObj_2* with the *Type* parameter set to "Summary," then the display will show summaries of current alarms. Conversely, if the same display has an *almQuery( )* run against it with the *Type* parameter set to "History," it will show historical alarms. The property, *QueryType* reflects the current state of the alarm display.

3. Click **OK**.

Using the Distributed Display to Monitor Local Alarms

The distributed alarm display can be used to display and acknowledge both local and remote alarms.

To set up a display to monitor just local alarms

1. Paste a distributed alarm display object to your window. (Click the wizard tool, select the **Alarm Displays** category. Double-click the **Dist. Alarm Display** wizard, and then click in the window to paste it.)
2. Double-click on the display or right-click it, and then click **Properties**. The **Alarm Configuration** dialog box will appear with the **General** property sheet active.
3. In the **Alarm Query** box, type **\InTouch!$System**.

   Tip You may substitute any valid Alarm Group for **$System**. You can also define an Alarm Group List containing just **\InTouch!$System**, and then use this Group List in step 3 instead of a direct reference.

4. Configure the other parameters of the **Default Query Properties** for the type of display and any filtering your application requires.
5. Switch to WindowViewer to run the application.

Distributed Alarm Display Properties and Functions

The Distributed Alarm Display includes multiple tagname **dotfields** and QuickScript functions. The following section briefly describes how you can use the QuickScript functions.

For more information and examples of how to use the QuickScript functions, see your online *InTouch Reference Guide.*

Alarm Display Monitoring Properties

The alarm display has several QuickScript-exposed properties that can be used to monitor the status of the display at runtime. These properties are accessible through the *GetPropertyX( )* function, where X is the data type (D for Discrete, I for Integer, and M for Message).

Query Properties	Description
.AlarmGroup	Message property containing the current query list.
.PriFrom	Integer property containing the current query priority low filter value.
.PriTo	Integer property containing the current query priority high filter value.
.QueryType	Integer property containing the current query type: **1 = History 2 = Summary**
.QueryState	Integer property containing the current query filter: **0 = All 1 = Unack 2 = Ack**

For more information, see "Configuring an Alarm Viewer ActiveX Control"

Query Status Properties	Description
.Successful	Discrete property containing the current query status: **0 = Error 1 = OK**
.ProvidersReq	Integer property containing the number of alarm providers in the current query.
.ProvidersRet	Integer property containing the number of alarm providers that have successfully returned their query results.
.NumAlarms	Integer property containing the number of alarms in the current query.
.PageNum	Integer property containing the current page number displayed in the alarm display.
.TotalPages	Integer property containing the total number of pages in the alarm display.

Alarm Display Control Properties

The alarm display also has two QuickScript-exposed properties that may be used to control the movement of the display's screen in **Runtime**. These properties are controllable through the *SetPropertyD* function.

Control Properties	Description
.NextPage	Scrolls the alarm display one page down when this property transitions from 0 to 1.
.PrevPage	Scrolls the alarm display one page up when this property transitions from 0 to 1.

Whenever the value of this discrete variable transitions from Off (0, False) to On (1, True), the alarm display object will display the page that corresponds to that QuickScript (Next or Prev.). Once that page is displayed, the discrete variable will automatically reset to Off (0, False).

Note These functions are provided to ease the conversion of the earlier InTouch standard display to the current distributed display. Their functionality has been replaced with the scroll bars and the *almMoveWindow* QuickScript function.

Alarm Acknowledgment QuickScript Functions

The Distributed Alarm System is capable of acknowledging any alarms that it can query (summary display only). To provide this capability, the Distributed Alarm System includes alarm acknowledgment QuickScript functions. These functions supplement the **.Ack** dot field that the InTouch alarm system uses to acknowledge local alarms, and Alarm Groups. The specific syntax of these functions is addressed in the *InTouch Reference Guide*.

Function	Description
almAckAll	Acknowledges all the alarms in the current alarm query. Since the alarm display has only a limited display area, the **almAckAll** function may acknowledge alarms that are not visible in the display.
almAckDisplay	Acknowledges only those alarms that are currently visible in the alarm display.
almAckGroup	Acknowledge all alarms that have a given Group name from the same provider.
almAckPriority	Acknowledges all alarms within a specified priority range having same provider name and group name.
almAckRecent	Acknowledges only the most recent alarm that has occurred in the current alarm query.
almAckSelect	The distributed alarm display allows alarms to be selected by clicking on them with the mouse at runtime. The **almAckSelect** function can be used to acknowledge those alarms.
almAckSelectedGroup	Acknowledges all alarms that have the same group name from the same provider as one or more of the selected alarms.
almAckSelectedPriority	Acknowledges all alarms that have the same priority from the same provider and group name as one or more of the selected alarms.

Function	Description
almAckSelectedTag	Acknowledges all alarms that have the same Tagname from the same provider and group name and having the same priority as one or more of the selected alarms.
almAckTag	Acknowledge all alarms that have the same Tagname, Provider name and Group name within the given Priority range.

Alarm Suppression QuickScript Functions

The Distributed Alarm System is capable of suppressing one or more alarms at an Alarm Consumer by identifying a set of exclusion criteria. If an alarm matches the exclusion criteria, it will not be visible at the Alarm Consumer; it will not appear on a display, cannot be printed, or will not be logged at that particular Alarm Consumer. To provide this capability, the Distributed Alarm System includes alarm suppression QuickScript functions. The specific syntax of these functions is detailed in the *InTouch Reference Guide.*

Function	Description
almSuppressAll	Suppress the display of all current and future occurrences of all alarms currently active.
almSuppressSelected	Suppress the display of current and future occurrences of the selected alarms.
almSuppressDisplay	Suppress the display of current and future occurrences of any visible alarm.
almSuppressSelectedGroup	Suppress the display of current and future occurrences of any alarm that belongs to the same groups of one or more selected alarms having the same Provider name.
almSuppressSelectedPriority	Suppress the display of current and future occurrences of any alarm that belongs to the same priorities of one or more selected alarms having the same Provider name and Group tag.
almSuppressSelectedTag	Suppress the display of current and future occurrences of any alarm that belongs to the same Tagname name of one or more selected alarms having the same Provider name, Group name and Priority range.
almSuppressGroup	Suppress the display of current and future occurrences of any alarm that belongs to a given Group name, having the same Provider name.
almSuppressPriority	Suppress the display of current and future occurrences of any alarm of the specified priority range, having the same Provider name and Group name.

Function	Description
almSuppressTag	Suppress the display of current and future occurrences of any alarm emitted by a given tagname name, having the same Provider, Group name and Priority range.
almUnSuppressAll	Clear alarm suppression.
almSuppressRetain	Retain alarm suppression between alarm queries when the alarm query is changed.

Alarm Display Manipulation QuickScript Functions

The Distributed Alarm System provides several QuickScript functions to manipulate the display object. These functions allow movement of the display window, selection of alarms within the display, display of the number of selected alarms, and display of the statistics window.

The specific syntax of these functions is addressed in the online *InTouch Reference Guide*.

Function	Description
almMoveWindow	Provides commands to manipulate the display window. These commands include: Page Up, Page Down, Scroll Right, Scroll Left, Line Up, Line Down, Top, End, and more.
almSelectAll	Toggles the selection of all the alarms in a display. Since the alarm display has only a limited display area, the **almSelectAll** function may select alarms that are not visible in the display.
almSelectionCount	Returns integer value containing the number of alarms selected by operator in the Distributed Alarm Object.
almSelectGroup	Toggles the selection of all the alarms from a given Group within a given Provider.
almSelectItem	Toggles the selection of the item that is highlighted in an alarm display.
almSelectPriority	Toggles the selection of all alarms from a given Priority range within a given Group in a given Provider.
almShowStats	Displays the alarm statistics dialog box.
almSelectTag	Toggles the selection of all alarms from a specific Provider/Group/Tag. You can also specify a Priority range, or use 1-999.
almUnSelectAll	Unselects all the selected records.

Alarm Query QuickScript Functions

The distributed display retrieves alarm information by submitting an alarm query. The parameters of this query and the query type are specified in one of three QuickScript functions. The specific syntax of these functions is detailed in the *InTouch Reference Guide.*

Function	Description
almDefQuery	Performs a query using the configuration dialog default properties. These properties include: From Priority, To Priority, Alarm List, and Display Type. The default properties can only be changed at development time and are not overwritten by other alarm queries.
almQuery	Performs a query for either summary or historical alarm information. All query properties are provided in this function.
almSetQueryByName	Start a new alarm query using the query parameters associated with a user defined query name.

Note In Windows 2000, if the Alarm Provider is in a different domain from the Alarm Consumer, the Alarm Consumer will not be able to see the alarms unless the query has the fully qualified name of the provider machine or the IP address of the provider name. For example, an Alarm provider in a different domain, can be specified in the query as:
\\provider1.b3.wonderware.com\intouch!$system
where "provider1" is the machine name and the "b3.wonderware.com" is the Primary DNS Suffix for the domain.

Alarm DB View ActiveX Control

InTouch provides an Alarm DB View ActiveX Control that allows you to visualize the alarm data from the new Alarm DB Logger database. This control is used to view all alarm and event information. You design the appearance of the ActiveX Control and the data displayed by specifying the following attributes:

- Context sensitive menu features
- Display mode
- List control options
- Colors for different properties
- Font type, style and size
- Database specifications (server name, User ID and Password)
- Query filters
- Column management
- Sorting

Once the control format has been designed, a user may have the ability to make the following adjustments to manipulate the data they are viewing:

- Sort the information within a column
- Update the display
- Perform a query
- Resize the width of a column

The Alarm DB View ActiveX Control can be dropped into WindowMaker, resized, and positioned; you configure the control using the Property sheets. The data from the database can then be viewed in the Alarm DB View ActiveX Control view window. The type and format of the data viewed depends on the properties set up on the Property sheet.

Installation

The Alarm DB View ActiveX control is installed when InTouch is installed.

To paste the Alarm DB View ActiveX Control in a WindowMaker window

1. Open the Wizard selection dialog box.
2. Select **AlmDbViewCtrl** under ActiveX Controls, then click **OK**.
3. Paste the control on the window and resize it to the required size.

Uninstall

1. Delete all the Alarm DB View controls pasted on the windows.
2. Select Configure from the **Special** menu.
3. Select **Wizard/ActiveX Installation**, then open the Wizard/ActiveX installation dialog box.
4. Select the **ActiveX Control** installation property page. The Wonderware Alarm Database View Control name appears in the Installed ActiveX Controls text area.
5. Click Wonderware Alarm Database View Control, then click **Remove**.
6. Click **Yes** for the warning message.
7. Click **Close**.

Accessing the AlmDbViewCtrlX Properties Dialog Box

You access the AlmDbViewCtrlX Properties Dialog Box by:

- Double clicking on the control, or by
- Right clicking over the control and selecting the Properties menu from the pop up menu.

AlmDBViewCtrlX Properties Dialog Box

The tabs available on the AlmDbViewCtrlX Properties dialog box provide the options used to design the properties for a control. Click on the appropriate Tab ID below to view the instructions for completing the fields on the tab.

Note The instructions provided describe the various properties of the specified control by relating those available at the AlmDbViewCtrlX Properties dialog box with those that can be configured at the script editors.

For Example:
#AlmDBView1.RefreshMenu = False;

This expression refers to setting a property value for the object AlmDBView1.

In the following sections, the syntax is described using notations such as Object.RefreshMenu. In all such cases, "Object" refers to an object expression that evaluates to an object or control.

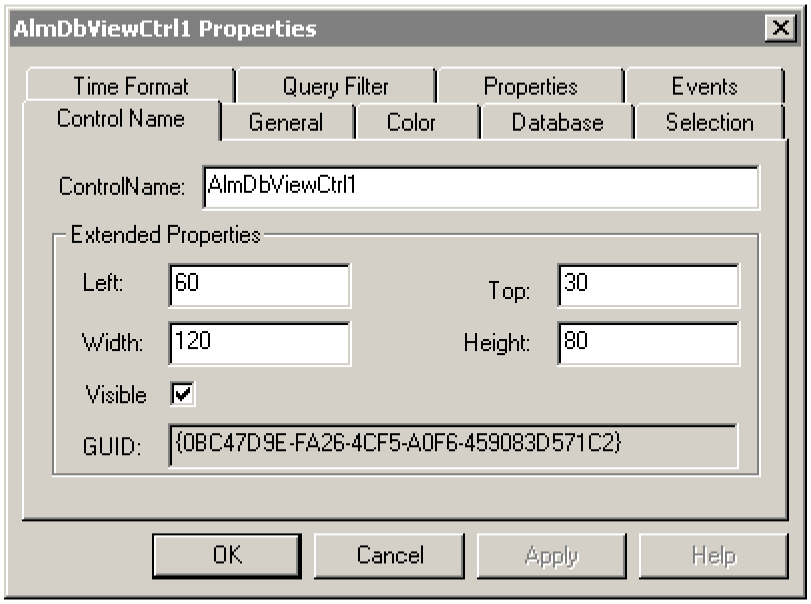
Control Name Page

- The **Control Name** tab of the Properties dialog provides the details about the control with respect to the application.

  Note By default, the Control Name is determined by the ProgID for that control. ProgIDs are names that are entered into the system registry when ActiveX controls are installed on a computer. When an instance of that control is placed in an InTouch application, the control's ProgID is obtained from the system registry and an index number is appended to it, resulting in a Control Name, such as **AlarmDBViewCtrl1**.

- The **Extended Properties** section provides details about the Control's configuration in the window (Left, Top, Width and Height); you can alter these properties.
- Unselect the **Visible** check box to make the control invisible during runtime.

- The **GUID** text box displays the unique ID for this ActiveX control.


AlmDbViewCtrl1 Properties

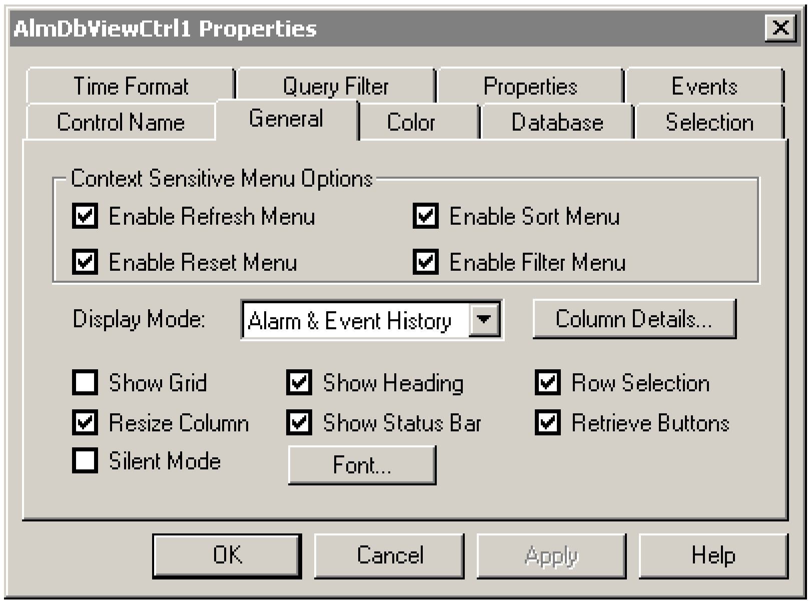
ControlName: AlmDbViewCtrl1

Extended Properties

Left: 60
Top: 30
Width: 120
Height: 80
Visible ☑
GUID: {0BC47D9E-FA26-4CF5-A0F6-459083D571C2}

General Property Page

The **General** tab provides the options for the features included at the user interface of this control.


AlmDbViewCtrl1 Properties

Context Sensitive Menu Options

☑ Enable Refresh Menu
☑ Enable Sort Menu
☑ Enable Reset Menu
☑ Enable Filter Menu

Display Mode: Alarm & Event History

☐ Show Grid
☑ Show Heading
☑ Row Selection
☑ Resize Column
☑ Show Status Bar
☑ Retrieve Buttons
☐ Silent Mode

Context Sensitive Menu Options

The Context Sensitive Menu Options section provides the options to enable and disable the context sensitive menu features available at this control.

The following descriptions of properties and methods of the AlmDbViewCtrl are based on the following assumption when used in InTouch scripting:

When an integer tag with a Boolean value is assigned a value "True" and is read to (or received from) an integer type tagname:

To Set: assign a value not equal 0 or "True"
To Get: you will receive the integer value -1

When an integer tag with a Boolean value is assigned a value "False" and is read to (or received from) an integer type tagname:

To Set: assign 0 or "False"
To Get: you will receive the integer value 0

Examples To Set:

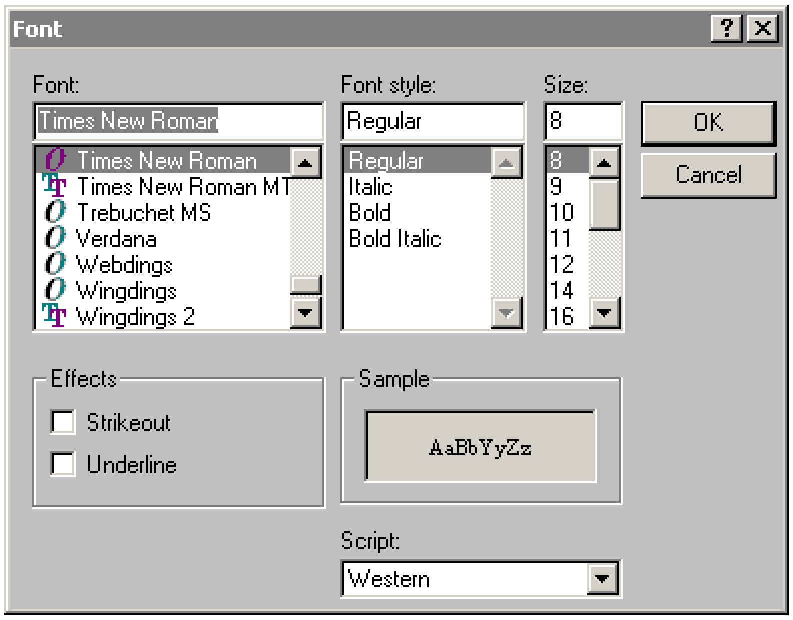
```
#AlmDbViewCtrl6.SortMenu = "True"; {enables the Sort Menu
in the context Menu of the Control}

#AlmDbViewCtrl6.SortMenu = 1; {enables the Sort Menu in the
context Menu of the Control}

#AlmDbViewCtrl6.SortMenu = "False"; {disables the Sort Menu
in the context Menu of the Control}

#AlmDbViewCtrl6.SortMenu = 0; {disables the Sort Menu in
the context Menu of the Control}
```

Examples To Get:

```
IF #AlmDbViewCtrl6.ShowFetch == 0 THEN {checks for the
actual state of the property}

#AlmDbViewCtrl6.ShowFetch = 1; {toggles the property into
the "True" state if it was previously in the "False" state}

ELSE
#AlmDbViewCtrl6.ShowFetch = 0; {toggles the property into
the "False" state if it was previously in the "True" state}

ENDIF;
Int_Tag = #AlmDbViewCtrl6.ShowGrid {will assign -1 to
Int_Tag   when Grid is shown, will assign 0 when Grid is
not shown}
```

Font Button

The **Font** button is used to set the font for the display of records and the heading in the control.



To configure the font properties

1. Double-click on the AlarmDBView ActiveX control or right-click it, and then click **Properties**. The **AlarmViewerCtrlX** dialog box appears with the **Control Name** property sheet active.
2. Click the **General** tab, then click the **Font** button. The Font dialog box appears.
3. Scroll the **Font** list and select the desired font type.
4. Scroll the **Font Style** list to select a font style.
5. Scroll the **Size** list to select a font size.
6. In the **Effects** area, check the **Strikeout** box or **Underline** box to select the Strikeout or Underline attributes.
7. Click the **Script** drop-down arrow to select the desired script type.

Note The **Sample** box shows a sample of the selected font attributes.

8. Click **OK**.

Enable Refresh Menu

Select this check box to enable the **Refresh** menu option at the right click menu of the control. The Refresh menu refreshes the control to the database, and if the connection is successful, it displays the set of records in the range 1 to MaxRecords.

Property:	**RefreshMenu**		
Purpose:	Returns or sets a value that determines whether the "Refresh" menu item is displayed in the context-sensitive menu.		
Type:	Discrete		
Default:	`True`		
Syntax:	*Object.*RefreshMenu *[= Integer]*		
	Discrete		
		True	(Default) "Refresh" menu item is shown
		False	"Refresh" menu item is not shown.

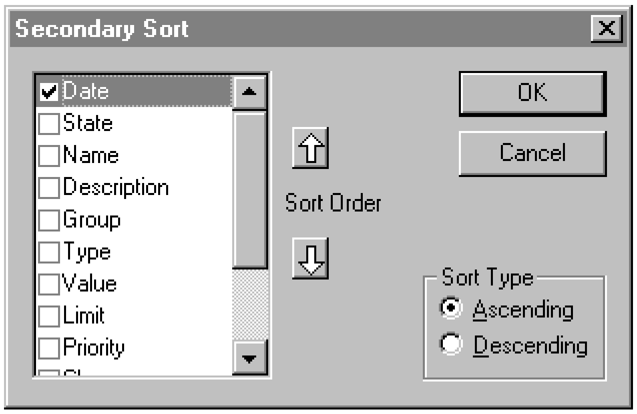
Enable Sort Menu

Select this check box to enable the **Sort menu** option on the right click menu of the control. This menu displays the Secondary Sort menu used to set the user-defined sorting of columns.

Property:	**SortMenu**		
Purpose:	Returns or sets a value that determines whether the "Sort" menu item is displayed in the context-sensitive menu.		
Type:	Integer		
Default:	`True`		
Syntax:	*Object.*SortMenu *[= Integer]*		
	Integer	An Integer expression specifying whether "Sort" menu item is displayed as described in Settings.	
		True	(Default) "Sort" menu item is shown
		False	"Sort" menu item is not shown.

Secondary Sort

The **Secondary Sort** dialog box shows up when this popup menu is clicked.



This dialog box displays the list of columns that are currently displayed at the control. This dialog box is used to do single and multiple column sorting, in ascending or descending order.

To specify the columns to be sorted, select the check box beside the column name. Use the **Sort Order** arrow keys to rearrange the columns.

For example, if the desired sorting is in descending order based on the alarm state first, then date:

1. Select both these check boxes (Date and State).
2. Select the **State** row.
3. Click the UP **Sort Order** arrow key.
4. Select the **Descending** radio button from the **Sort Type** section.
5. Click **OK**.

Enable Filter Menu

Select this check box to enable the **Filter** menu option on the right click menu of the control. This menu displays the filter menu used to set the user-defined filtering criteria.

Property:	**FilterMenu**		
Purpose:	Returns or sets a value that determines whether the "Filter" menu item is displayed in the context-sensitive menu.		
Type:	Discrete		
Default:	`True`		
Syntax:	*Object*.FilterMenu *[= Discrete]*		
		True	(Default) "Filter" menu item is shown
		False	"Filter" menu item is not shown.

Enable Reset Menu

This check box is used to enable the **Reset menu** option at the right click menu of the control. The Reset menu arranges all the columns to the settings saved at design time.

Property:	**ResetMenu**		
Purpose:	Returns or sets a value that determines whether the "Reset" menu item is displayed in the context-sensitive menu.		
Type:	Discrete		
Default:	`True`		
Syntax:	*Object*.ResetMenu *[= Discrete]*		
		True	(Default) "Reset" menu item is shown
		False	"Reset" menu item is not shown.

Display Mode

This drop down list box displays a listing of the available view options. The contents of the list are:

- Alarms & Events History
- Alarms History
- Events History.

Property:	**DisplayMode**
Purpose:	Returns the display mode of the control. This property is **Read Only**.
Type:	String
Default:	Alarms & Events History
Syntax:	*Object*.DisplayMode
	`tagname = #AlmDbView1.DisplayMode;`
	where the name of the control is AlmDbView1 and tagname is defined as a Message tagname.

Runtime Features Check Boxes

Show Grid check box

The **Show Grid** check box is used to enable or disable the display of the grid at the control.

Property:	**ShowGrid**		
Purpose:	Returns or sets a value that determines whether the grid lines are displayed in the control.		
Type:	Discrete		
Default:	`False`		
Syntax:	*Object*.ShowGrid *[= Discrete]*		
		True	Grid lines will be displayed in the control.
		False	(Default) Grid lines will not be displayed.

Show Heading check box

The **Show Heading** check box is used to enable or disable the display of the heading of the control.

Property:	**ShowHeading**		
Purpose:	Returns or sets a value that determines whether the column headings are displayed in the control.		
Type:	Discrete		
Default:	`True`		
Syntax:	*Object*.ShowHeading *[= Discrete]*		
		True	(Default) Column headings will be displayed in the control.
		False	Column headings will not be displayed.

- Primary Sort

  The heading provides the user with the option of primary sorting. Clicking on the heading will do a sorting of all the rows that got satisfied with the last query and the first record based on the sorting is displayed to the user.

- Column Movement

  The columns can be moved across the control and can be placed near some other columns by clicking over the heading and dragging them.

Silent Mode

Select this check box to enable the Silent Mode option. This property determines whether the control is in silent mode or not. When this check box is selected, no error messages are displayed. To see the error, call the method "GetLastError" to return the error message.

Property:	**SilentMode**		
Purpose:	Returns or sets a value that determines whether the control is in Silent mode.		
Type:	Discrete		
Default:	`False`		
Syntax:	*Object*.SilentMode *[= Discrete]*		
		True	Silent mode is on.
		False	(Default) Silent mode is off.

Row Selection check box

The **Row Selection** check box is used to enable or disable the row selection option of the control.

Property:	**RowSelection**		
Purpose:	Returns or sets a value that determines whether the row selection will be allowed at runtime.		
Type:	Discrete		
Default:	`True`		
Syntax:	*Object*.RowSelection *[= Discrete]*		
		True	(Default) Row Selection will be allowed in the control.
		False	Row Selection will not be allowed.

Note If the row selection is not allowed - i.e. the property is set to `False` - then there will be no "Click" or "Double Click" event generated.

Resize Column check box

The **Resize Column** check box is used to enable or disable the column resize option of the control.

Property:	**ColumnResize**
Purpose:	Returns or sets a value that determines whether the columns can be resized at runtime.
Type:	Discrete

Property:	**ColumnResize**		
Default:	`True`		
Syntax:	*Object*.ColumnResize *[= Discrete]*		
		True	(Default) Columns can be resized at runtime.
		False	Columns cannot be resized.

Show Status Bar check box

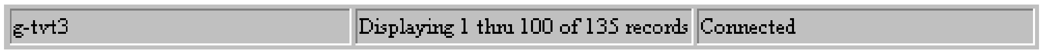
The **Show Status Bar c**heck box is used to enable or disable the display of the status bar at the bottom of the control.

This GUI feature is related to the property called:

Property:	**ShowStatusBar**		
Purpose:	Returns or sets a value that determines whether the status bar is shown.		
Type:	Discrete		
Default:	`True`		
Syntax:	*Object*.ShowStatusBar *[= Discrete]*		
		True	(Default) Status bar is shown.
		False	Status bar is not shown.

Status Bar

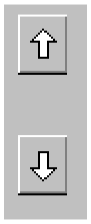
The status bar displays the current status of the control.


g-tvt3 | Displaying 1 thru 100 of 135 records | Connected

The left frame displays the server name (property called ServerName) and the right side of the frame displays the connection status with the server (property called ConnectStatus). The middle frame displays the number of the records (property called RowCount) that is displayed out of the total number of records (property called TotalRowcount) that has satisfied the query.

Retrieve Buttons check box

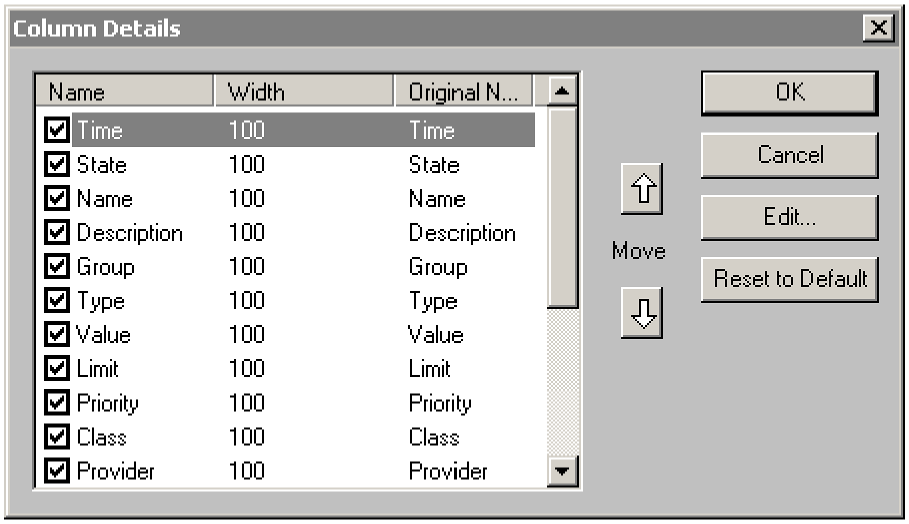
The **Retrieve Buttons** check box is used to enable or disable the display of the retrieve buttons at the right side of the control.



Property:	**ShowFetch**		
Purpose:	Returns or sets a value that determines whether the Fetch buttons are displayed.		
Type:	Discrete		
Default:	`True`		
Syntax:	*Object*.ShowFetch *[= Discrete]*		
		True	(Default) Fetch buttons are shown.
		False	Fetch buttons are not shown.

Column Details Button

The Column Details dialog box is invoked by clicking the **Column Details** button on the **General** property sheet. The Column Details dialog box is used to select the columns to display, specify the display order, and to set the column names and widths.



Name	Width	Original N...
✓ Time	100	Time
✓ State	100	State
✓ Name	100	Name
✓ Description	100	Description
✓ Group	100	Group
✓ Type	100	Type
✓ Value	100	Value
✓ Limit	100	Limit
✓ Priority	100	Priority
✓ Class	100	Class
✓ Provider	100	Provider

To configure the display column details

1. Double-click on the distributed alarm display or right-click it, and then click **Properties**. The **AlarmDBViewCtrl** dialog box appears with the **Control Name** property sheet active.
2. Click the **General** tab to activate the **General** property sheet.
3. Click the **Column Details** button. The Column Details dialog appears.
4. From the **Column Details** dialog box, select the checkbox next to the **Column Name** to display the column in the alarm object. The columns in the **Column Details** dialog box are **Name**, **Width** and **Original Name**. **Original Name** shows what the columns were named before any changes were made. The original column names that you can select to display are described below.

Note At least one column must be selected.

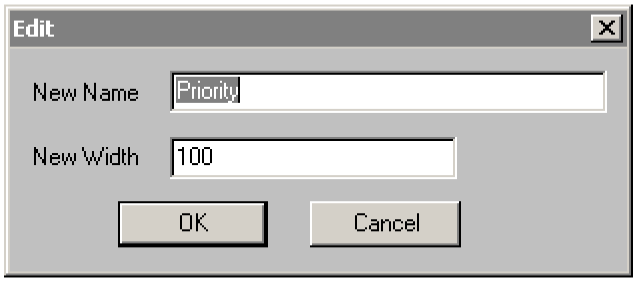
Column	Description
Time	Displays the time in the format selected from the Time Format properties sheet.
State	Displays the state of the alarm.
Name	Displays the alarm/tagname.
Description	Displays the decription of the alarm.
Group	Displays the Alarm Group name.
Type	Displays the alarm type.
Value	Displays the value of the tagname when the alarm occurred. The width of the column should be large enough to provide the desired level of precision.
Limit	Displays the alarm limit value of the tagname. The width of the column should be large enough to provide the desired level of precision.
Priority	Displays the alarm priority.
Class	Displays the category of the alarm.
Provider	Displays the name of the alarm provider.
Operator	Displays the logged-on operator’s ID associated with the alarm condition.
Domain Name	Displays the domain name for the alarm.
User Full Name	Displays the logged-on user’s full name.
Duration	Displays the alarm duration.
User1	Displays the numerical values of User Defined Number 1 corresponding to the alarm.
User2	Displays the numerical values of User Defined Number 2 corresponding to the alarm.

Column	Description
User3	Displays the string values of User Defined string corresponding to the alarm.
Comment	Displays the tagname's comments. These comments were typed in the Alarm Comment box when the tagname's alarm was defined in the database. These comments are also updated when ACK comments are given.

Note All column names are selected by default except for **Operator** and **Comment**.

Tip The column names are reset to default column names if the display mode is changed. It is better to select the display mode first before changing the column names.

5. To rearrange the columns, select the column name and use the Move Up and Down arrow keys. The column name appearing at the top of the **Column Details** dialog box is the column displayed to the furthest left of the alarm display.
6. To edit the column name and width, double-click a column name or select a column name, then click **Edit.** The **Edit** dialog box appears for that column.



New Name: Priority

New Width: 100

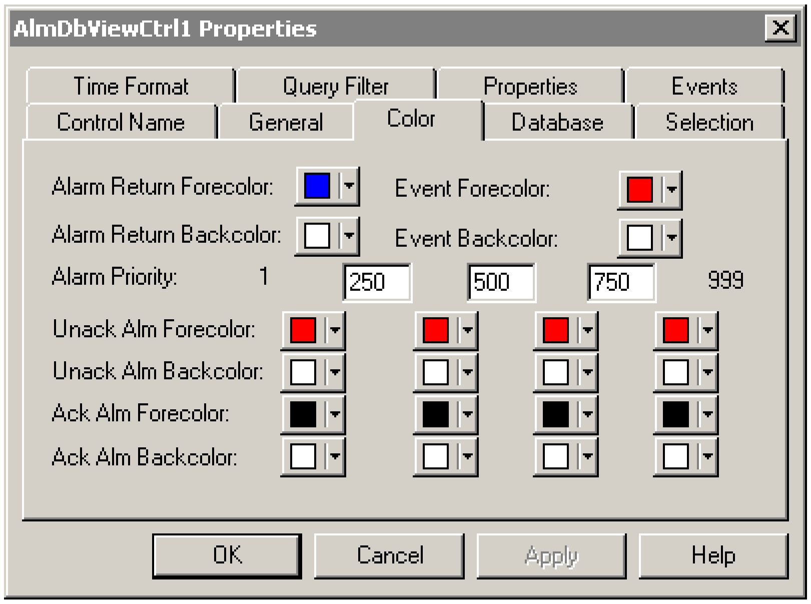
7. Enter in a new name in the **New Name** text box if you want to display a column name other than the default column name.
8. Enter in a column width in the **New Width** text box. The column width is measured in pixels and can range from 1 to 999 pixels. The default column width is 100 pixels.
9. Click **OK** on the **Edit** dialog.

Note Click **Reset to Default** to return to the default **Column Details** settings.

10. Click **OK** on the **Column Details** dialog.
11. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Colors Page

The **Colors** tab is used to set the colors for the various alarm and event records.



Alarm Return Forecolor: 🟦 Event Forecolor: 🟥

Alarm Return Backcolor: ⬜ Event Backcolor: ⬜

Alarm Priority:	1	250	500	750	999
Unack Alm Forecolor:	🟥	🟥	🟥	🟥	
Unack Alm Backcolor:	⬜	⬜	⬜	⬜	
Ack Alm Forecolor:	⬛	⬛	⬛	⬛	
Ack Alm Backcolor:	⬜	⬜	⬜	⬜	

Properties

The **Properties** list displays the set of record details for which color can be set.

To set properties, type #object.PropertyName = 1; or #object.PropertyName = tag1; where object is the name of the AlarmDBViewCtrl and tag1 is a discrete tag. For example, to set the AckRtnForeColor1 property, type #AlarmDBView1.AckRtnForeColor1 = 1.

To get properties, type tag1 = #object.PropertyName; where object is the name of the AlarmViewer) and tag1 is a discrete tag. For example, to get the AckRtnForeColor1 property, type tag1 = #AlarmDBView1.AckRtnForeColor1;.

InTouch accepts the words "True" and "False" within double quotes as values 1 and 0 respectively. An action script such as:

#AlmDbViewCtrl17.FilterMenu = "False";

erases the “Filter Menu” from the right-click menu at Alarm DB View Control.

The list contains the following properties:

Property:	**AckOrAlarmDuration**
Purpose:	The duration column is populated with either Ack Duration or Alarm Duration. FALSE (0) is Ack Duration and TRUE (1) is Alarm Duration.
Type:	Integer
Default:	False
Syntax:	*Object*.AckOrAlarmDuration *[= integer]*

Property:	**AlmRtnForeColor**	
Purpose:	Returns or sets the Returned alarm foreground color. This color applies to the records shown in the control with state ALM_RTN with priorities in the range 1 to ColorPriorityRange1.	
Type:	Integer	
Default:	Green	
Syntax:	*Object*.AckRtnForeColor1 *[= color]*	
	color	A value or constant that determines the color of the specified object.

Property:	**AlmRtnBackColor**	
Purpose:	Returns or sets the Returned alarm background color. This color applies to the records shown in the control with state ALM_RTN with priorities in the range 1 to ColorPriorityRange1.	
Type:	Integer	
Default:	Green	
Syntax:	*Object*.AckRtnForeColor1 *[= color]*	
	color	A value or constant that determines the color of the specified object.

Property:	**ColorPriorityRange1**
Purpose:	Sets the boundry of the priority range in which alarms are to be displayed. The value of this property must be greater than one and less than the value for ColorPriorityRange2.
Type:	Integer

Property:	**ColorPriorityRange1**
Default:	250
Syntax:	*Object*.ColorPriorityRange1 *[= integer or priority]*

Property:	**ColorPriorityRange2**
Purpose:	Sets the boundry of the priority range in which alarms are to be displayed. The value of this property must be greater than the value for ColorPriorityRange1 and less than the value for ColorPriorityRange3.
Type:	Integer
Default:	500
Syntax:	*Object*.ColorPriorityRange2 *[= integer or priority]*

Property:	**ColorPriorityRange3**
Purpose:	Sets the boundry of the priority range in which alarms are to be displayed. The value of this property must be greater than the value of ColorPriorityRange2 and less than 999.
Type:	Integer
Default:	750
Syntax:	*Object*.ColorPriorityRange3 *[= integer or priority]*

Property:	**EventForeColor**	
Purpose:	Returns or sets the event alarm foreground color. This color applies to the records shown in the control with state EVT_EVT.	
Type:	Integer	
Default:	Magenta	
Syntax:	*Object*.EventForeColor *[= color]*	
	color	A value or constant that determines the color of the specified object.

Property:	**EventBackColor**
Purpose:	Returns or sets the Event alarm background color. This color applies to the records shown in the control with state EVT_EVT.
Type:	Integer
Default:	Window Background Color

Property:	**EventBackColor**	
Syntax:	*Object*.EventBackColor *[= color]*	
	color	A value or constant that determines the color of the specified object.

Property:	**FilterFavoritesFile**
Purpose:	String property. Sets / returns the filter favorites file. This file is used by the Filter Favorites dialog to read / write filter favorites.
Type:	Discrete
Default:	`Null`
Syntax:	*Object*.FilterFavoritesFile *[= Discrete]*

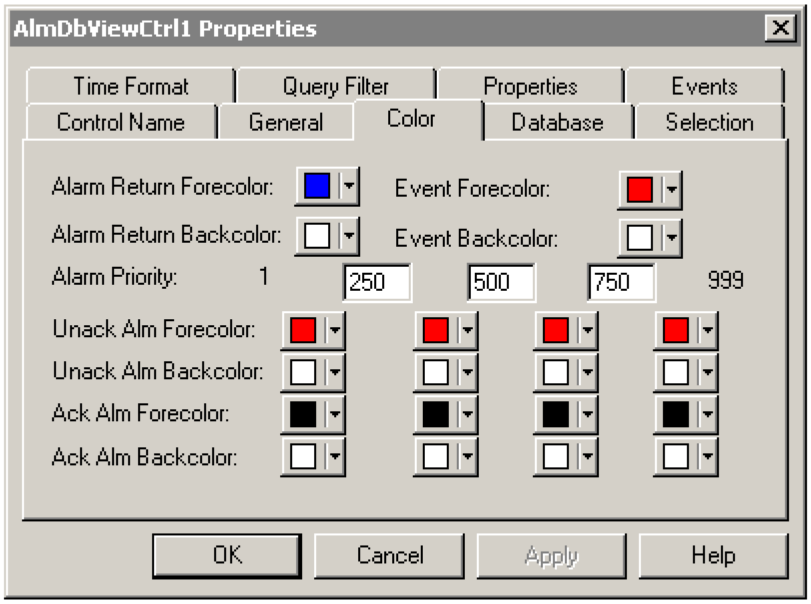
Property:	**FilterMenu**		
Purpose:	Returns or sets a value that determines whether the "Filter" menu item is displayed in the context-sensitive menu.		
Type:	Discrete		
Default:	`True`		
Syntax:	*Object*.FilterMenu *[= Discrete]*		
		True	(Default) "Filter" menu item is shown
		False	"Filter" menu item is not shown.

Property:	**FilterName**
Purpose:	Returns the name of the current filter (if any).
Type:	String (Read Only)
Default:	`Null`
Syntax:	*Object*.FilterName *[= String]*

Configuring the Alarm Display Colors

To configure the alarm display colors

1. Double-click on the alarm DB View Control or right-click it, and then click **Properties**. The **AlarmDBViewCtrlX** dialog box appears with the **Control Name** property sheet active.
2. Click the **Color** tab to activate the **Color** property sheet.



Alarm Return Forecolor: 🟦 Event Forecolor: 🟥

Alarm Return Backcolor: ⬜ Event Backcolor: ⬜

Alarm Priority: 1	250	500	750	999
Unack Alm Forecolor:	🟥	🟥	🟥	🟥
Unack Alm Backcolor:	⬜	⬜	⬜	⬜
Ack Alm Forecolor:	⬛	⬛	⬛	⬛
Ack Alm Backcolor:	⬜	⬜	⬜	⬜

3. Click each color box to open the InTouch Palette. Click the color that you want to use in the palette for each of the following:

   Alarm Return Forecolor
   Sets the foreground color for the alarm return display.

   Alarm Return Backcolor

   Sets the background color for the alarm return display.

   Event Forecolor
   Sets the foreground color for the events display.

   Event Backcolor
   Sets the background color for the events display.

4. In the **Alarm Priority** boxes, type the breakpoint values for the alarm display. You can assign breakpoint values so that alarms will appear in different colors depending on the Alarm Priority. The predetermined minimum and maximum alarm priority values are 1 and 999 respectively.

Color Priority	Range
1 to ColorPriorityRange1	Alarms with priorities in the range 1 to the breakpoint value in the first alarm priority box. The default value is set at 250.
ColorPriorityRange1 to ColorPriorityRange2	Alarms with priorities in the range between the breakpoint value in the first alarm priority box and the breakpoint value in the second box. The default values are set at 250 in the second alarm priority box and 500 in the third box.
ColorPriorityRange2 to ColorPriorityRange3	Alarms with priorities in the range between the breakpoint value in the second alarm priority box and the breakpoint value in the third box. The default values are set at 500 in the first alarm priority box and 750 in the second box.
ColorPriorityRange3 to 999	Alarms with priorities in the range between the breakpoint value in the third alarm priority box and 999. The default values is set at 750 in the third alarm priority box. The maximum value is 999.

5. Set the color for each ColorPriorityRange for the Unack Alm Forecolor. Click each color box to open the InTouch Palette.
6. Set the color for each ColorPriorityRange for the Unack Alm Backcolor.
7. Set the color for each ColorPriorityRange for the Ack Alm Forecolor.
8. Set the color for each ColorPriorityRange for the Unack Alm Backcolor.
9. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Property:	**UnAckAlmForeColorRange1**	
Purpose:	Returns or sets the Unacknowledged alarm foreground color. This color applies to the records shown in the control with state UNACK_ALM with priorities in the range 1 to ColorPriorityRange1.	
Type:	Integer	
Default:	Red	
Syntax:	*Object*.UnAckAlmForeColorRange1 *[= color]*	
	color	A value or constant that determines the color of the specified object.

Property:	**UnAckAlmForeColorRange2**	
Purpose:	Returns or sets the Unacknowledged alarm foreground color. This color applies to the records shown in the control with state UNACK_ALM with priorities in the range ColorPriorityRange1 to ColorPriorityRange2.	
Type:	Integer	
Default:	Red	
Syntax:	*Object*.UnAckAlmForeColorRange2 *[= color]*	
	color	A value or constant that determines the color of the specified object.

Property:	**UnAckAlmForeColorRange3**	
Purpose:	Returns or sets the Unacknowledged alarm foreground color. This color applies to the records shown in the control with state UNACK_ALM with priorities in the range ColorPriorityRange2 to ColorPriorityRange3.	
Type:	Integer	
Default:	Red	
Syntax:	*Object*.UnAckAlmForeColor3 *[= color]*	
	color	A value or constant that determines the color of the specified object.

Property:	**UnAckAlmForeColorRange4**	
Purpose:	Returns or sets the Unacknowledged alarm foreground color. This color applies to the records shown in the control with state UNACK_ALM with priorities in the range ColorPriorityRange3 to 999.	
Type:	Integer	
Default:	Red	
Syntax:	*Object*.UnAckAlmForeColorRange4 *[= color]*	
	color	A value or constant that determines the color of the specified object.

Property:	**UnAckAlmBackColorRange1**	
Purpose:	Returns or sets the Unacknowledged alarm background color. This color applies to the records shown in the control with state UNACK_ALM with priorities in the range 1 to ColorPriorityRange1.	
Type:	Integer	
Default:	Window Background Color	
Syntax:	*Object*.UnAckAlmBackColorRange1 *[= color]*	
	Color	A value or constant that determines the color of the specified object.

Property:	**UnAckAlmBackColorRange2**	
Purpose:	Returns or sets the Unacknowledged alarm background color. This color applies to the records shown in the control with state UNACK_ALM with priorities in the range ColorPriorityRange1 to ColorPriorityRange2.	
Type:	Integer	
Default:	Window Background Color	
Syntax:	*Object*.UnAckAlmBackColorRange2 *[= color]*	
	Color	A value or constant that determines the color of the specified object.

Property:	**UnAckAlmBackColorRange3**	
Purpose:	Returns or sets the Unacknowledged alarm background color. This color applies to the records shown in the control with state UNACK_ALM with priorities in the range ColorPriorityRange2 to ColorPriorityRange3.	
Type:	Integer	
Default:	Window Background Color	
Syntax:	*Object*.UnAckAlmBackColorRange3 *[= color]*	
	Color	A value or constant that determines the color of the specified object.

Property:	**UnAckAlmBackColorRange4**	
Purpose:	Returns or sets the Unacknowledged alarm background color. This color applies to the records shown in the control with state UNACK_ALM with priorities in the range ColorPriorityRange3 to 999.	
Type:	Integer	
Default:	Window Background Color	
Syntax:	*Object*.UnAckAlmBackColorRange4 *[= color]*	
	Color	A value or constant that determines the color of the specified object.

Property:	**AckAlmBackColorRange1**	
Purpose:	Returns or sets the Acknowledged alarm background color. This color applies to the records shown in the control with state ACK_ALM with priorities in the range 1 to ColorPriorityRange1.	
Type:	Integer	
Default:	Window Background Color	
Syntax:	*Object*.AckAlmBackColorRange1 *[= color]*	
	color	A value or constant that determines the color of the background.

Property:	**AckAlmBackColorRange2**	
Purpose:	Returns or sets the Acknowledged alarm background color. This color applies to the records shown in the control with state ACK_ALM with priorities in the range ColorPriorityRange1 to ColorPriorityRange2.	
Type:	Integer	
Default:	Window Background Color	
Syntax:	*Object*.AckAlmBackColorRange2 *[= color]*	
	color	A value or constant that determines the color of the background.

Property:	**AckAlmBackColorRange3**	
Purpose:	Returns or sets the Acknowledged alarm background color. This color applies to the records shown in the control with state ACK_ALM with priorities in the range ColorPriorityRange2 to ColorPriorityRange3.	
Type:	Integer	
Default:	Window Background Color	
Syntax:	*Object*.AckAlmBackColorRange3 *[= color]*	
	color	A value or constant that determines the color of the background.

Property:	**AckAlmBackColorRange4**	
Purpose:	Returns or sets the Acknowledged alarm background color. This color applies to the records shown in the control with state ACK_ALM with priorities in the range ColorPriorityRange3 to 999.	
Type:	Integer	
Default:	Window Background Color	
Syntax:	*Object*.AckAlmBackColorRange4 *[= color]*	
	color	A value or constant that determines the color of the background.

Property:	**AckAlmForeColorRange1**	
Purpose:	Returns or sets the Acknowledged alarm foreground color. The color applies to the records shown in the control with state ACK_ALM with priorities in the range 1 to ColorPriorityRange1.	
Type:	Integer	
Default:	Blue	
Syntax:	*Object*.AckAlmForeColorRange1 *[= color]*	
	Color	A value or constant that determines the color of the specified object.

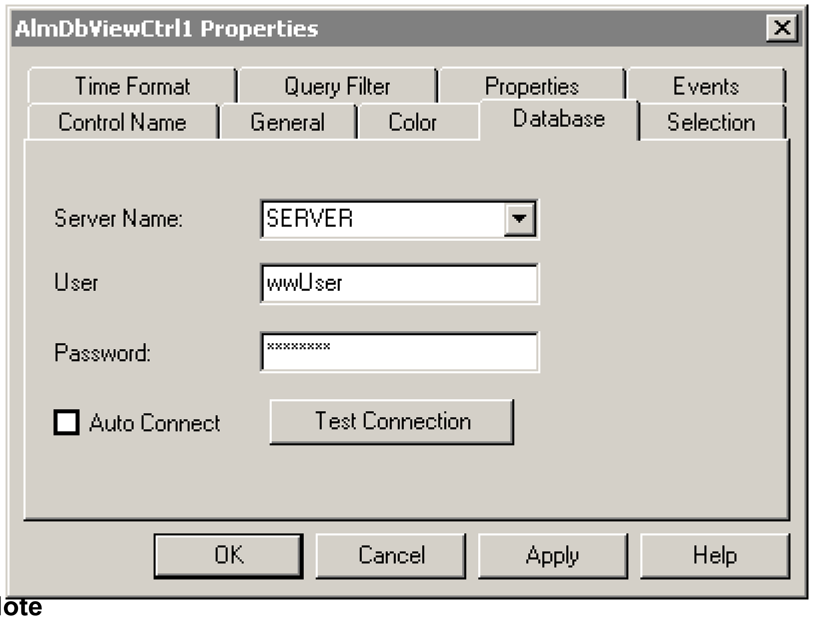
Property:	**AckAlmForeColor2**	
Purpose:	Returns or sets the Acknowledged alarm foreground color. The color applies to the records shown in the control with state ACK_ALM with priorities in the range ColorPriorityRange1 to ColorPriorityRange2.	
Type:	Integer	
Default:	Blue	
Syntax:	*Object*.AckAlmForeColorRange2 *[= color]*	
	Color	A value or constant that determines the color of the specified object.

Property:	**AckAlmForeColorRange3**	
Purpose:	Returns or sets the Acknowledged alarm foreground color. The color applies to the records shown in the control with state ACK_ALM with priorities in the range ColorPriorityRange2 to ColorPriorityRange3.	
Type:	Integer	
Default:	Blue	
Syntax:	*Object*.AckAlmForeColorRange3 *[= color]*	
	Color	A value or constant that determines the color of the specified object.

Property:	**AckAlmForeColorRange4**	
Purpose:	Returns or sets the Acknowledged alarm foreground color. The color applies to the records shown in the control with state ACK_ALM with priorities in the range ColorPriorityRange3 to 999.	
Type:	Integer	
Default:	Blue	
Syntax:	*Object*.AckAlmForeColorRange4 *[= color]*	
	Color	A value or constant that determines the color of the specified object.

Database Page

The Database tab is used to set the server name and its relevant properties, so that the control can be made to connect to the database, when required.


AlmDbViewCtrl1 Properties

Server Name: SERVER

User wwUser

Password: ********

Auto Connect

Note
It is good practice to use a user ID with appropriate access to the alarm database and not a system administration account.

Server Name

The **Server Name** drop down list box consists of the list of SQL Server and MSDE Server names available at the network. Users can edit this list box. This GUI feature is related to the property called:

Property:	**ServerName**	
Purpose:	Returns or sets the server name to which the control connects to fetch the data.	
Type:	Message	
Syntax:	*Object*.ServerName *[= text]*	

User

The **User** field is used to enter the user name for the database server so that the required connection can be established. This GUI feature is related to the property called:

Property:	**UserID**	
Purpose:	Returns or sets the User ID used as the control to connect to the SQL Server to fetch the data.	
Type:	Message	
Syntax:	*Object*.UserID *[= text]*	
	text	A string expression that evaluates the User ID.

Password

The **Password** field is used to enter the password for the user name of the database server so that the required connection can be established. This GUI feature is related to the property called:

Property:	**Password**	
Purpose:	Returns or sets the password used as the control to connect to the SQL Server to fetch data.	
Type:	Message	
Syntax:	*Object.*Password *[= text]*	
	text	A string expression that evaluates to the password.

Test Connection

The **Test Connection** button is used to test the connection to the WWALMDB database at design time with the given server name, user name and password.

Auto Connect check box

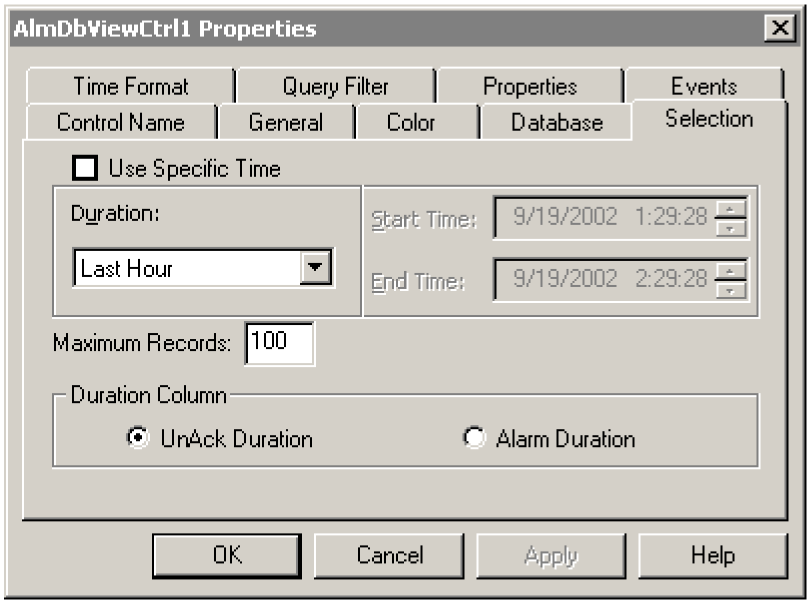
The **Auto Connect** check box is disabled by default. Once the server name and user id fields are filled, this check box becomes enabled. This check box is used to enable or disable the auto connection to the database by the control to the server at runtime. This GUI feature is related to the property called:

Property:	**AutoConnect**		
Purpose:	Returns or sets a value that determines whether the control connects to the database as soon as it is in runtime mode.		
Type:	Integer		
Default:	`False`		
Syntax:	*Object*.AutoConnect *[= Integer]*		
	Integer	An Integer expression specifying whether control connects to the database as soon as it is in run mode as described in Settings.	
		True	Connects to the database.
		False	(Default) Does not connect to the database.

Note If this property is set to `False` at design time, then the control doesn’t connect to the database when it is in run mode. An explicit call to "Connect" method has to be made to connect to the database.

Selection Tab

The **Selection** tab gives the details about the set of records that need to be retrieved based on the time. It also gives the number for the maximum number of records that get displayed, whether to display the date, time and how to display them.



Use Specific Time [ ]

Duration: Last Hour

Start Time: 9/19/2002 1:29:28

End Time: 9/19/2002 2:29:28

Maximum Records: 100

Duration Column: [✓ UnAck Duration] Alarm Duration

Use Specific Time check box

This checkbox is used to enable or disable whether the user can enter specific Start and End times for the records to be retrieved, or whether these are to be computed based on the **Duration** property. This GUI feature is related to the property called:

Property:	**SpecificTime**		
Purpose:	Returns or sets a value that determines whether the control uses the "start time" and "end time" properties, or computes the start time and end time based on the value of the Duration property.		
Type:	Discrete		
Default:	`False`		
Syntax:	*Object.*SpecificTime *[= Discrete]*		
		True	"Start time" and "End time" properties are used.
		False	(Default) "Start time" and "End time" are computed based on the "Duration" property.

Duration

Once the **Use Specific Time** check box is unchecked, the **Duration** dropdown list box is enabled. This gives a list of pre defined time intervals relative to the current time. This GUI feature is related to the property called:

Property:	**Duration**	
Purpose:	Returns or sets the duration used by the control to set the "Start Time" and "End Time".	
Type:	Message	
Default:	"Last Hour"	
Syntax:	*Object*.Duration *[= text]*	
	text	A String expression that evaluates to the duration.

Note This property must have one of the following strings:
Last Minute
Last 5 Minutes
Last 15 Minutes
Last Half Hour
Last Hour
Last 2 Hours
Last 4 Hours
Last 8 Hours
Last 12 Hours
Last Day
Last 2 Days
Last 3 Days
Last Week
Last 2 Weeks
Last 30 days
Last 90 days

Start Time

This Date-time control is enabled only when the **Use Specific Time** checkbox is checked. This is used to set the start time of the time interval within which the alarm records are to be retrieved. This GUI feature is related to the property called:

Property:	**StartTime**	
Purpose:	Returns or sets the start time.	
Type:	Message	
Syntax:	*Object.*StartTime *[= text]*	
	text	A string expression that evaluates to the Start Time.

Note The string returned is in the format as set in the **Date** and **Time** properties. To set the value the string has to be in MM/DD/YYYY HH:MM:SS format. This property handles date in any time zone from midnight, January 1, 1970, to January 18, 19:14:07, 2038.

End Time

This Date-time control is enabled only when the **Use Specific Time** check box is checked. This is used to set the end time of the time interval within which the alarm records are to be retrieved. This GUI feature is related to the property called:

Property:	**EndTime**	
Purpose:	Returns or sets the End time.	
Type:	Message	
Syntax:	*Object*.EndTime *[= text]*	
	text	A string expression that evaluates to the end time.

Note The string returned is in the format as set in the **Date** and **Time** properties. To set the value the string has to be in MM/DD/YYYY HH:MM:SS format. This property handles date in any time zone from midnight, January 1, 1970, to January 18, 19:14:07, 2038.

Maximum Records

The **Maximum Records** field is used to edit the number of records that can be displayed at the control at one instance. This GUI feature is related to the property called:

Property:	**MaxRecords**	
Purpose:	Returns or sets a value that specifies the maximum records to be fetched at a given time.	
Type:	Integer	
Default:	100	
Syntax:	*Object*.MaxRecords *[=integer]*	
	integer	An integer expression specifying the number of records to be fetched at a given time.

Note The maximum records can be in the range from 1 to 1000. For best performance keep this value as small as needed.

Duration Column

The Duration Column displays the number of milliseconds for UNACK/Alarm Duration.

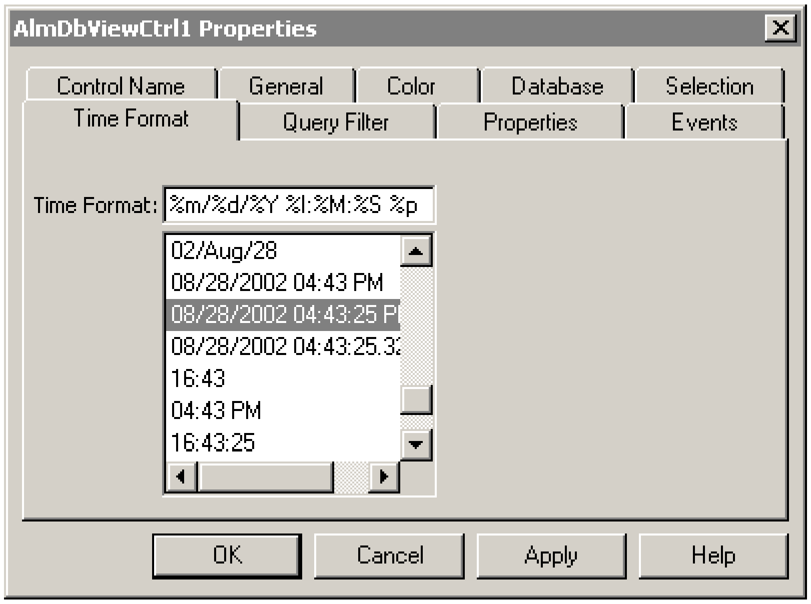
- Select **UnAck Duration** to display the time between the most recent alarm transition (ALM or sub-state) and the ACK, if any.
- Select **Alarm Duration** to display the amount of time elapsed between the initial occurence of an alarm and the time at which it returned to a normal state.

Time Format Tab

To configure the Alarm DBView control time format

1. Double-click on the Alarm DBView ActiveX control or right-click it, and then click **Properties**. The **AlarmDBViewCtrlX Properties** dialog box appears with the **Control Name** property sheet active.

2. Click the **Time Format** tab to activate the **Time Format** property sheet.



Time Format: %m/%d/%Y %I:%M:%S %p

02/Aug/28
08/28/2002 04:43 PM
08/28/2002 04:43:25 P
08/28/2002 04:43:25.3
16:43
04:43 PM
16:43:25

Scroll the Time Format options to select the desired time format. The available options are:::

String character	Description
d	two digit date - 09
b	3 letter month abbreviation - Aug
Y	four digit year - 2002
m	two digit month - 08
/	date division function as in 08/09
y	two digit year - 02
#x	full day and date - Friday, August 09, 2002
B	full month name - August
-	date division punctuation as in 08-09
.	date division punctuation as in 08.09
,	date division punctuation as in Aug 09, 2002
H	24 hour time - 16:00
:	time division punctuation as in 4:41
M	minute 00:41
p	PM
S	seconds - 16:41:07
s	fractions of a second - 16:41:07.390
I	12 hour time requiring AM/PM designation - 04:41 PM

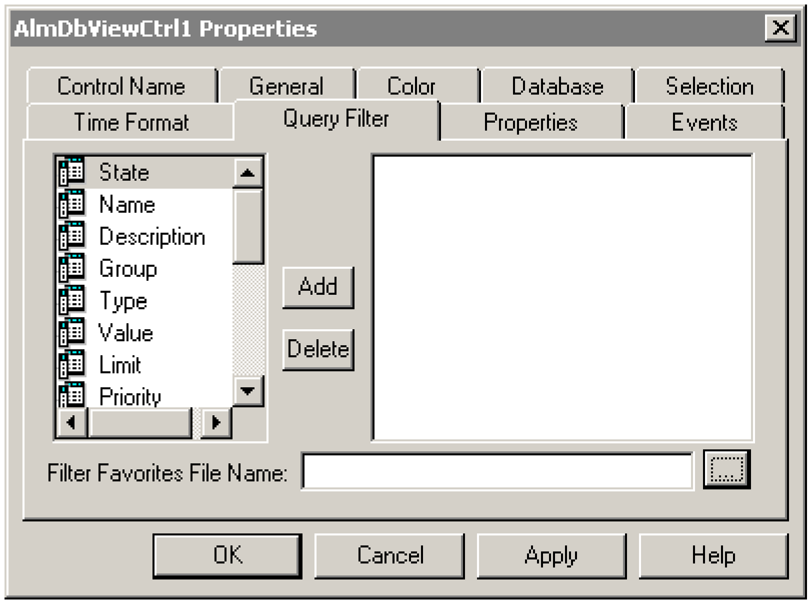
Some sample time format character strings are shown below:

Time Format String	Display
%d %b	09 Aug
%m/%d/%Y	08/09/2002
%#x	Friday, August 09, 2002
%Y-%m-%d	2002-08-09
%m/%d/%Y %H:%M %p	08/09/2002 16:56 PM
%m/%d/%Y %H:%M:%s %p	08/09/2002 16:56:38.07
%I:%M %p	04:56 PM

3. Click **Apply**. You can proceed to configure the next property or click **OK** to exit the properties sheets.

Query Filter Tab

The **Query Filter** tab allows you to define which records will be included in your query results by choosing columns to use as filters. For example, you can choose to filter by the date of a record or the state of the alarm. You can choose multiple fields to limit or expand your query results..

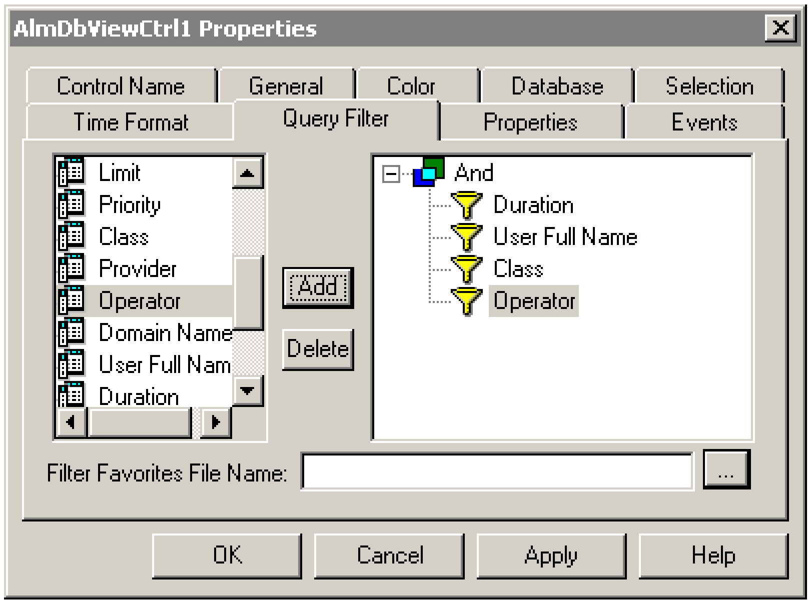


Selecting Query Filters

All of the filter columns are listed in the left pane. Columns that are selected as filters are listed in the right pane.

To select query filters

- Double-click on the column or select a column in the left pane and click **Add** to move it into the filters pane.



About the Filters Pane

The filters (right) pane contains the list of columns that are part of the query. When more than one column is defined, the columns are combined using either the "And" or the "Or." Select the "And" operator to return only the records that meet all values of the listed columns. Select the "Or" to return records that meet the values of any of the columns. By default multiple columns are always grouped as an "And" expression. To use an And/Or statement, the respective columns must be grouped together. Only a single filter expression can be created on an item in the filters pane. If multiple expressions are needed, then the item must be added to the filters pane again. To remove a column from the filters pane, click the coulmn to delete, then click **Delete**.

Alarm Columns

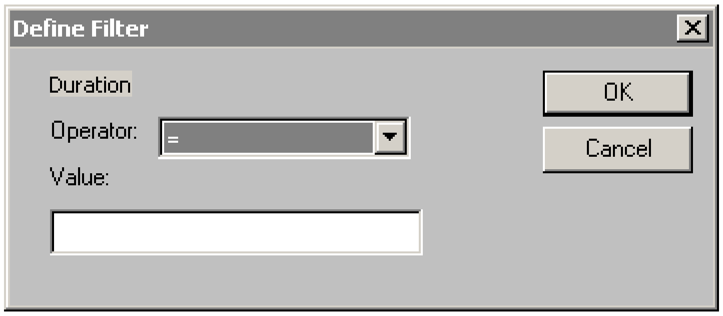
The alarm columns in the left pane of the **Query Filters** tab are described in the table below:

Column	Description
State	Filters query by alarm state. Select the desired alarm state from the drop-down menu.
Name	Filters query by alarm name.
Description	Filters query by alarm description.
Group	Filters query by alarm group.
Type	Filters query by alarm type.

Column	Description
Value	Filters query by alarm value. Values in the Value column are displayed as alphanumeric values. The comparisons of these values in the Query Filter are done as string comparisons.
Limit	Filters query by alarm limit. Values in the Limit column are displayed as alphanumeric values. The comparisons of these values in the Query Filter are done as a string comparisons.
Priority	Filters query by alarm priority.
Class	Filters query by alarm class.
Provider	Filters query by alarm provider.
Operator	Filters query by operator.
Domain Name	Filters query by alarm domain name.
User Full Name	Filters query by the user's full name.
Duration	Filters query by UNACK duration.
User1	Filters query by alarm user-defined value 1.
User2	Filters query by alarm user-defined value 2.
User3	Filters query by alarm user-defined value 3.

To define a filter for a column

1. Double-click the column in the filters pane or right-click the selected column and click **Edit Filter** from the shortcut menu. the **Define Filter** dialog box appears.



2. Click the **Operator** drop-down arrow and select the desired operator. The list of available operators are equal =, not equal to !=, less than or equal to <=, greater than or equal to >=, less than <, greater than >, **Like** and **Not Like**.

3. Enter a value in the **Value** box and click **OK**.

Note The value box will not accept data that cannot be processed for the selected query. The Value box accepts the following as SQL server wild card characters when "Like" and "Not Like" operators are used for alphanumeric column names:
% - any string of zero or more characters.
_ - any single character.
[ ] - Any single character not within the specified range ([^a-f]) or set ([^abcdef]).
[^] - Any single character not within the specified range ([^a-f]) or set ([^abcdef]).
All columns except Duration, User1, User2, State and Priority columns, accept all alphanumeric characters in the Value text box.
For the Prioritycolumn, the value text box accepts only integer values from 1 to 999.
The maximum value for Duration Days text box is 99999 and the maximum value for the Milliseconds text box is 999. No negative values are accepted.
The User1 and User2 columns accept only numbers, which can be negative, positive, or fractional.

Values for the State Column

When you add the **State** column to your filter query, you may assign values from the Value drop-down menu in the Define Filter dialog box. The values available are described in the following table:

Value	Description
ACK	Produces a query of all system ACKs.
ACK_ALM	Produces a query of all ACKed alarms.
UNACK_ALM	Produces a query of all UNACKed alarms.
ACK_RTN	Produces a query of all ACK_RTNs.
UNACK_RTN	Produces a query of all UNACK_RTNs.
All UNACK Records	Produces a query of all UNACK records.
All ACK Records	Produces a query of all ACK records.
All ALM Records	Produces a query of all alarm records.
All RTN Records	Produces a query of all RTN records.

Note When a tag in expanded summary alarm mode is used to create an alarm, which returns to normal when the main alarm is ACKed, two records are created. The first record is the ACK_RTN record as the new alarm is already in a returned to normal state. The second record is ACK, which corresponds to ACKing the main alarm. The previous implementation of ACK_ALM has been changed to ACK.

To group columns

1. Right-click the column and select **Group** from the shortcut menu.
2. Drag and drop a column onto another column.

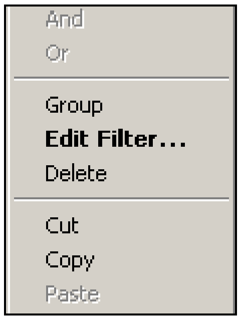
Note The "And," and "Or" the operators are parent nodes. The columns selected under each parent node are child nodes. You cannot drag and drop parent nodes to child nodes.

By default the columns that are grouped will have the "And" operator. To change it to "Or," right-click the operator and select the appropriate operator from the shortcut menu.

Note When no filters have been defined at design time, a record called Default Filter displays in the Filter Favorites dialog. If you have not defined any filters in the **Query Filter** tab, the Default Filter that appears in the **Filter Favorites** dialog box in Runtime will query all records.

About the Shortcut Menu

A shortcut menu is provided in the filters pane.



Right-click a column in the Filters pane to invoke the shortcut menu. The menu options are:

Menu Options	Function
And	Changes the "Or" nodes to "And." This option is disabled for all menu items except for "Or."
Or	Changes the "And" nodes to "Or." This option is disabled for all menu items except for "And."
Group	Creates a drag image of the item and starts the drag/drop operation.
Edit Filter	Pops up the **Define Filter** dialog box.
Cut	Cuts the selected filter.
Copy	Copies the selected filter.
Paste	Pastes the cut or copied filter.
Delete	Deletes a filter expression.

Copying or Moving Query Filters

If you have more than one instance of Alarm DBView and want to use the same filters for multiple instances, you can copy (or cut) the defined filters from one instance and paste them to another.

To copy filters from one instance of Alarm DB View to another

1. Define the desired filters in the first instance of Alarm DBView.
2. Right-click the filter(s) and click **Copy** (to move the filter(s), click **Cut** instead).
3. Click **OK** or **Cancel** to close the first instance of Alarm DBView.
4. Open the next instance of Alarm DBView and click the **Query Filter** tab.
5. Position the arrow in the right pane, right-click on a selected filter and click **Paste**.

Note You must add at least one filter property to the right pane in order to activate the shortcut menu.

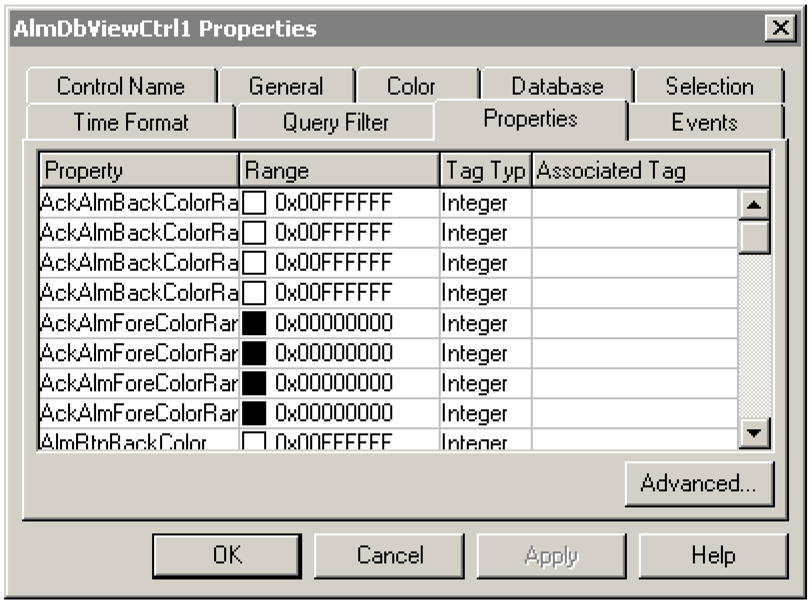
Loading Query Filter Favorites

To load query filter favorites

1. In the **Query Filter** tab, enter the network path and filename in the **Filter Favorites Name** box or click the button to browse for the file.
2. If you chose to browse for the file, select the from the network and click **Open**.
3. Click **Apply**.

Properties Page

The **Properties** tab is used to assign tagname names to the various properties available for this control.



Property	Range	Tag Typ	Associated Tag
AckAlmBackColorRa	□ 0x00FFFFFF	Integer	
AckAlmBackColorRa	□ 0x00FFFFFF	Integer	
AckAlmBackColorRa	□ 0x00FFFFFF	Integer	
AckAlmBackColorRa	□ 0x00FFFFFF	Integer	
AckAlmForeColorRar	■ 0x00000000	Integer	
AckAlmForeColorRar	■ 0x00000000	Integer	
AckAlmForeColorRar	■ 0x00000000	Integer	
AckAlmForeColorRar	■ 0x00000000	Integer	
AlmBtnBackColor	□ 0x00FFFFFF	Integer	

General Properties

Property:	**RowCount**
Purpose:	Returns the number of records currently being displayed in the control. This property is **READ ONLY**.
Type:	Integer
Syntax:	*Object*.RowCount
Example:	`tagname = #AlmDbView1.RowCount;` (where the name of the control is AlmDbView1 and tagname is defined as an Integer tagname.)

Property:	**TotalRowCount**
Purpose:	Returns the total number of records for the current query. This property is **READ ONLY.**
Type:	Integer
Syntax:	*Object*.TotalRowCount
Example:	`tagname = #AlmDbView1.TotalRowCount;` (Where the name of the control is AlmDbView1 and tagname is defined as an Integer tagname).

Note RowCount is the number of rows returned in the current query, which usually would be same as MaxRecords property except for the case when number of records fetched are less than MaxRecords property. For Example, if there are 950 records for a specific criterion and Max Records is 100, then in the last page there would be 50 records and the Row count would be 50. In the same example TotalRowCount would always be 950.

Methods & Events

Property:	**ConnectStatus**
Purpose:	Returns the status of the connection. This property is **READ ONLY.**
Type:	Message
Syntax:	*Object.*ConnectStatus

Example:	`*Tagname* = **#AlmDbView1.ConnectStatus;**`
	
	`(Where the name of the control is AlmDbView1 and tagname is defined as a Message tagname).`

Return Value	
Connected	The control is connected to the database.
Not Connected	The control is not connected to the database.
In Progress	The control is connecting to the database.

Method:	**Connect**
Purpose:	Connects the control to the database and if the connection is successful, displays the set of records in the range 1 to MaxRecords.
Syntax:	*Object.*Connect
Example:	**`#AlmDbView1.Connect();`** (Where the name of the control is AlmDbView1).

Method:	**Disconnect**
Purpose:	Disconnects the control from the database.
Syntax:	*Object.*Disconnect
Example:	**`#AlmDbView1.Disconnect();`** (where the name of the control is AlmDbView1)

Method:	**Refresh**
Purpose:	Refreshes the control from the database, and if the connection is successful, displays the set of records in the range 1 to MaxRecords.

Method:	**Refresh**
Syntax:	*Object*.Refresh
Example:	`#AlmDbView1.Refresh();` (where the name of the control is AlmDbView1)

Method:	**Reset**
Purpose:	Resets all the columns to the settings saved at design time.
Syntax:	*Object*.Reset
Example:	`#AlmDbView1.Reset();` (where the name of the control is AlmDbView1).

Method:	**ShowSort**
Purpose:	Shows the "Secondary Sort" dialog if the "SortMenu" property is enabled.
Syntax:	*Object*.ShowSort
Example:	`#AlmDbView1.ShowSort();` (where the name of the control is AlmDbView1).

Method:	**ShowContext**
Purpose:	Shows the context sensitive menu if any one of "RefreshMenu" or "ResetMenu" or "SortMenu" property is enabled.
Syntax:	*Object*.ShowContext
Example:	`#AlmDbView1.ShowContext();` (where the name of the control is AlmDbView1).

Method:	**ShowFilter**
Purpose:	Displays the Filter Favorites dialog box.
Syntax:	*Object*.ShowFilter
Example:	`#AlmDbView1.ShowFilter();` (where the name of the control is AlmDbView1).

Method:	**GetItem**	
Purpose:	Returns the data at a specified row & column as string.	
Syntax:	*Object*.GetItem*(Integer, message)*	
	Integer	An integer expression that evaluates to a specific row in the control
	Message	A string expression that evaluates to the column name in the control
Example:	`tagname = #AlmDbView1.GetItem(1, "Group");` (where the name of the control is AlmDbView1 and tagname is defined as a Message tagname).	

Method:	**GetSelectedItem**	
Purpose:	Returns the data for the selected row, given column as string	
Syntax:	*Object*.GetSelectedItem*(message)*	
	Message	An string expression that evaluates to the column name in the control
Example:	`tagname = #AlmDbView1.GetSelectedItem("State");` (where the name of the control is AlmDbView1 and tagname is defined as Message tagname).	

Method:	**GetNext**
Purpose:	Retrieves the next set of records from the database (if any).
Syntax:	*Object.*GetNext
Example:	`#AlmDbView1.GetNext();` (Where the name of the control is AlmDbView1).

Method:	**GetPrevious**
Purpose:	Retrieves the previous set of records from the database (if any).

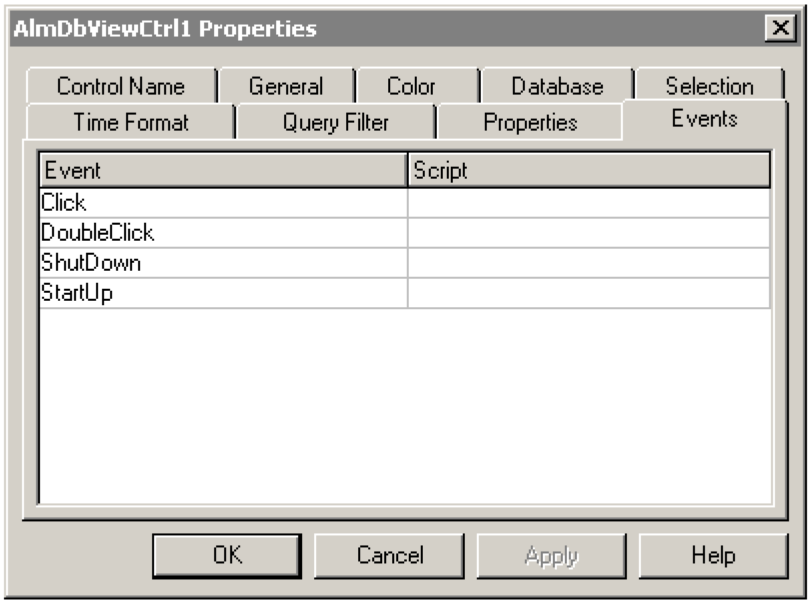
Method:	**GetPrevious**
Syntax:	*Object.*GetPrevious
Example:	**`#AlmDbView1.GetPrevious();`** (where the name of the control is AlmDbView1).

Method:	**GetLastError**	
Purpose:	Returns the last error message if in Silent mode.	
Syntax:	Object. GetLastError()	
Example:	**To Get:**	**`Tagname = #AlmDbView1.GetLastError();`**
	where the name of the control is AlmDbView1 and tagname is defined as variant or string.	

Events Page

Method:	**AboutBox**
Purpose:	Shows the "About" dialog box.
Syntax:	*Object.*AboutBox
Example:	`#AlmDbView1.AboutBox();`
	where the name of the control is AlmDbView1.

The **Events** tab is used to assign some script functions to the various events that can take place at this control.



Event	Script
Click	
DoubleClick	
ShutDown	
StartUp	

Event:	**Click**	
Purpose:	Occurs when the user clicks the left mouse button and releases it over the control.	
Syntax:	`tagname = #Thisevent.clicknRow`	
	`nRow`	An integer expression reflecting the row where this event happened.
	`#Thisevent`	Keyword

Note AlarmDB View control ignores the UI methods when the method is called from OnStartup event, since the control is not visible yet. The UI methods listed below will be ignored if they are called from OnStartup event: "ShowSort", "ShowContext", "GetSelectedItem", "GetNext", "GetPrevious" & "AboutBox".

Event:	**DoubleClick**	
Purpose:	Occurs when the user double-clicks the left mouse button and releases it over the control.	
Syntax:	`Tagname = #ThisEvent.DoubleClicknRow`	
	`nRow`	An integer expression reflecting the row where this event happened.
	`#Thisevent`	Keyword

Click and DoubleClick events are zero-based. When Click and/or DoubleClick events are published for the user, the row count in the display starts with 0.

C H A P T E R 1 1

Alarm Utilities

The InTouch notification system described in Chapter 9, "Alarms/Events," supports the displaying, logging, and printing of process alarms and system events. This chapter describes the operation of the Alarm Printer Utility and the Alarm DB Logger Utility.

Contents

The Alarm Printer Utility

The Alarm Printer utility provides a centralized utility to printout distributed alarm information across multiple nodes in a separate, standalone utility. With the Alarm Printer utility, you can print alarm status changes on an event-by-event basis using a dedicated line printer. You can define different printing options and save these definitions in the Alarm Printer utility configuration files (*.ALC) for later retrieval.

Every transition of an alarm constitutes an event that can be helpful in diagnosing a problem. The Distributed Alarm System can be configured to print certain events on a line printer as they occur. Usually, customers want this printout as a last resort in case of a catastrophic system failure and/or loss of power. Generally, this means printing via a serial port or parallel port to a dot matrix printer. Windows network printers and laser printers are usually inappropriate for this function as they cache entire pages in memory before actually printing them out - which means the information could be lost in a system crash or power outage.

Note Alarm Printer utility queries only historical data, not summary data.

Alarm Printing Date/Time Stamps

The default date format for Alarm Printer is:

DD MMM YYYY

Where **DD** is the day of the month, **MMM** is the month and **YYYY** is the year.

The default time format for Alarm Printer is:

HH:MM:SS.SSS

Where **HH** is the hour, **MM** is the minute, **SS** are seconds and **SSS** are milliseconds.

Note The time field of alarm records is always in **GMT** regardless of the time zone settings for the computer.

You can change the default format for the date and/or time on the **Message** property sheet in the **Configuration Settings** dialog box.

For more information, see "Configuring the Alarm Printer."

Using One or More Alarm Printers

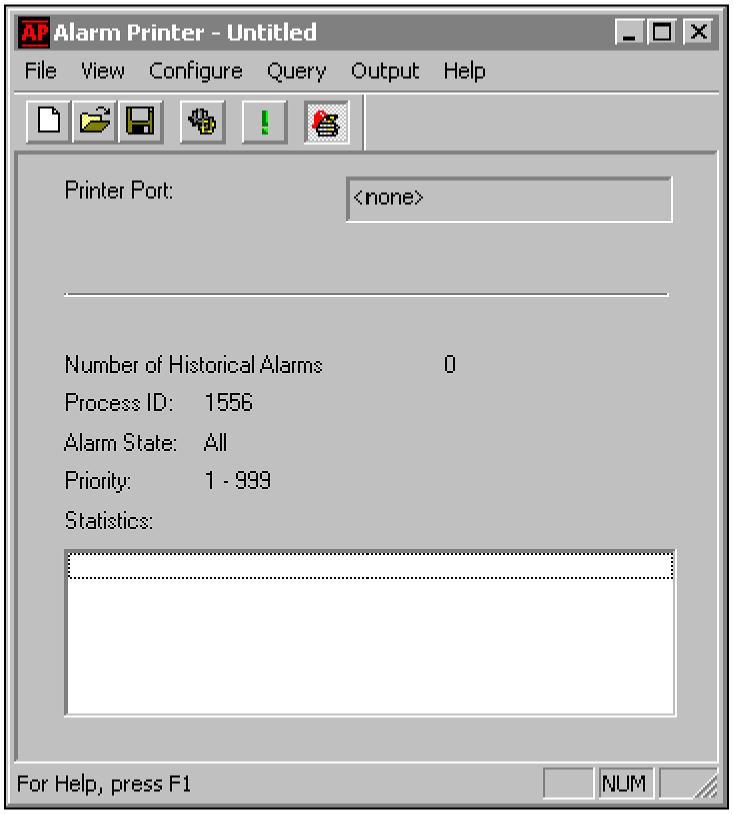
You can simultaneously run multiple instances of the Alarm Printer (AlmPrt.exe). Each instance of the Alarm Printer must be configured to print to a different printer, and must be configured with its own separate alarm query.

Tip You can configure separate instances of Alarm Printer to print alarms in specific priority ranges. For example, one Alarm Printer instance may only print high priority alarms, while another instance only prints low priority alarms. Likewise, you can use one instance of the Alarm Printer to print alarms from one area of the factory while another instance prints alarms from a different area.

Working with Alarm Printer

When you initially start the Alarm Printer, the **Alarm Printer** window appears displaying the Alarm Printer default configuration settings.

Note A specific alarm configuration file may be displayed if it is opened in runtime from the command prompt or by double-clicking its *.ALC filename.



Printer Port: <none>

Number of Historical Alarms 0
Process ID: 1556
Alarm State: All
Priority: 1 - 999
Statistics:

For Help, press F1 NUM

To create a new Alarm Printer configuration file

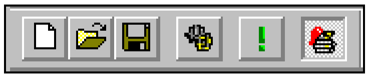
1. On the **File** menu, select **New**. The Alarm Printer defaults to **Alarm State** ALL, **Priority** 1-999 and **Printer Port** <none>.
2. Click the **Configure** menu. The **Configuration Settings** dialog box appears. Enter your desired settings.
3. On the file menu, select **Save**.

To edit an existing Alarm Printer configuration file

1. On the **File** menu, select **Open**.
2. Select the Alarm Printer Configuration file that you want to edit.
3. To save your changes, on the **File** menu, select **Save**.
4. Or, on the **File** menu, select **Save as** to save the changes to a new file without changing the existing file.

For more information on Alarm Printer configuration, see "Configuring the Alarm Printer."

The Alarm Printer Toolbar



The tools on the Alarm Printer toolbar provide you with quick access to all Alarm Printer commands.

The following illustrates and describes each tool:

Tool	Description
🗋	Create a new Alarm Printer configuration file.
📂	Open an existing Alarm Printer configuration file.
💾	Save the current Alarm Printer configuration file to disk.
⚙	Configure Alarm Printer.
!	Start/stop a Query.
🖨	Enable/disable alarm printing.

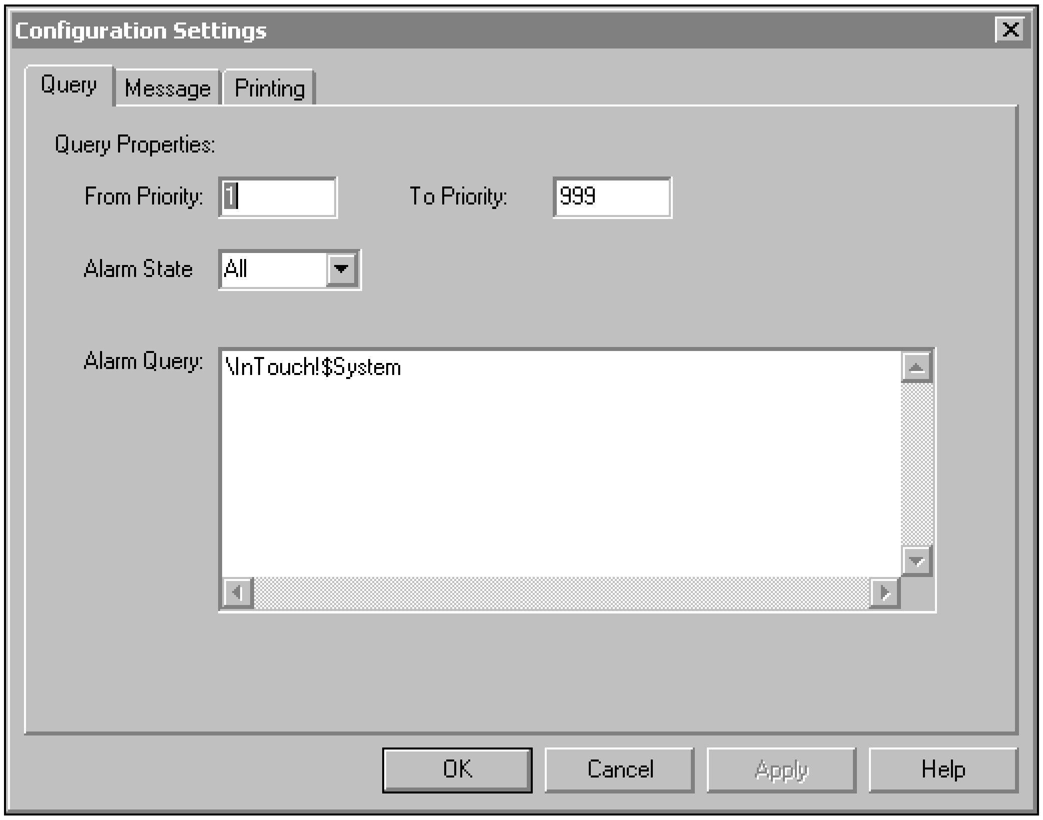
Configuring the Alarm Printer

Alarm Printer configurations are saved in Alarm Printer Configuration Files (*.alc). There is no limit to the number of Configuration files saved. Alarm Printer can use only one configuration file for each instance of Alarm Printer that is running.

To configure the Alarm Printer query properties

1. Select **Configure**, or click the Configure Alarm Query tool on the Alarm Printer toolbar.

2. The **Configuration Settings** dialog box appears.



Query Properties:

From Priority: 1

To Priority: 999

Alarm State: All

Alarm Query: \InTouch!$System

3. In the **From Priority** field, type the highest priority alarm value (1 to 999).
4. In the **To Priority** field, type the lowest priority alarm value (1 to 999).

Note Each alarm configured in InTouch has a priority value associated with it. This value represents the severity of the alarm and can range from 1 to 999 with 1 being the most severe.

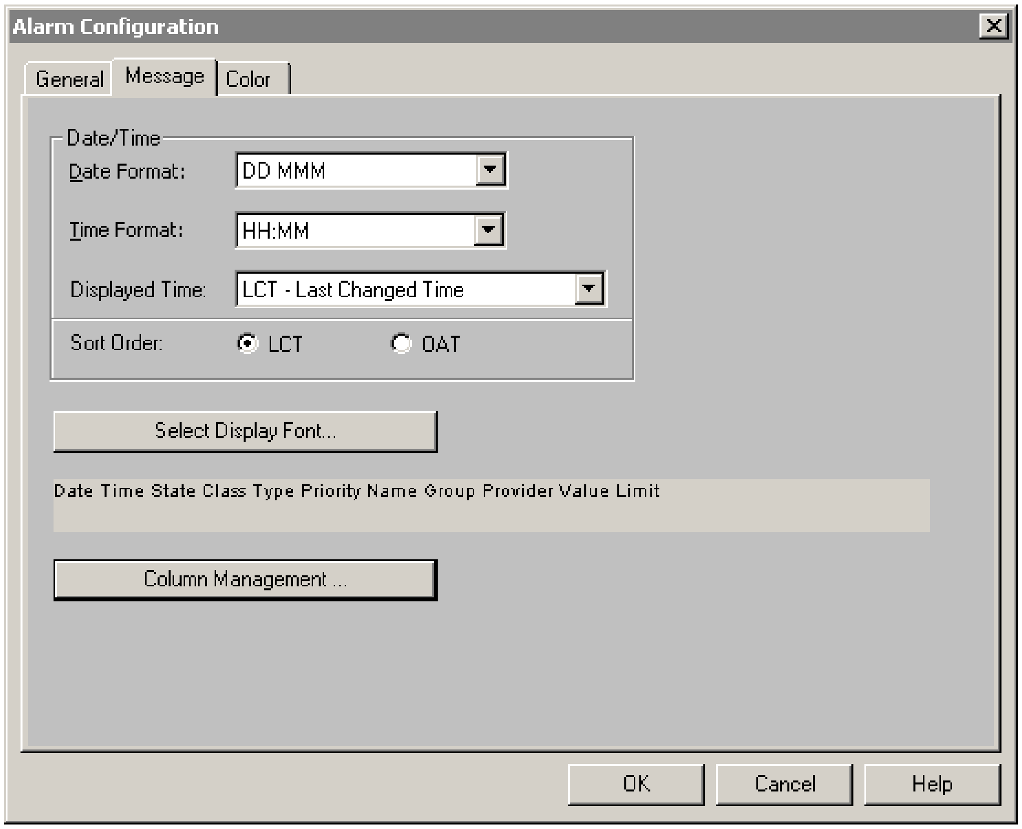
For more information on alarm priorities, see "Alarm Priorities."

5. Click the **Alarm State** arrow and select the alarm state that you want to use in the list
6. In the **Alarm Query** entry box, type the sets of InTouch alarm queries that you want to perform. You can specify one or more Alarm Providers and groups, just as with the alarm query for the Distributed Alarm Display. For more information see "Configuring a Distributed Alarm Display."

To configure the alarm record properties

1. On the **Output** menu, select **Configure**, or click the Configure Alarm Query tool on the Alarm Printer toolbar.
2. The **Configuration Settings** dialog box appears.

3. Select the **Message** tab to activate the **Message** property sheet:



Date/Time
Date Format: DD MMM
Time Format: HH:MM
Displayed Time: LCT - Last Changed Time
Sort Order: [✓ LCT] OAT

Date Time State Class Type Priority Name Group Provider Value Limit

Tip Each option you select will appear as a separate field in the generated printout. Fields that are set to specific character lengths will be truncated at those limits.

If you right-click a text box in any alarm configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

Note Most printers print either 80 or 132 characters per line.

4. To print the alarm date, select the **Date** option and then, click the arrow to select the format for the date. The available formats are:

Selection	Format	Selection	Format
DD MMM	28 Feb	**MM/DD**	02/28
DD MM YYYY	28 Feb 1997	**MM/DD/YY**	02/28/97
DD/MM	28/02	**MMM DD**	Feb 28
DD/MM/YY	28/02/97	**MMM DD YYYY**	Feb 28 1997

5. To print the alarm time, select the **Time** option then click the arrow to select the format for the time. The available formats are:

Selection	Description
AP	Selects the AM/PM format. For example, three o'clock in the afternoon is displayed as 3:00 PM. A time without this designation defaults to 24 hour military time format. For example, three o'clock in the afternoon is displayed as 15:00.
HH	Prints the hour the alarm/event occurred.
MM	Prints the minute the alarm/event occurred.
SS	Prints the second the alarm/event occurred.
SSS	Prints the millisecond the alarm/event occurred.

6. In the sort order box below **Time**, select the order that the alarms will be sorted in the alarm record. There are three choices:

Selection	Description
OAT	Original Alarm Time - that is, the date/time stamp of the onset of the alarm.
LCT	Last Changed Time - that is, the date/time stamp of the most recent change of status for the instance of the alarm: onset of the alarm, change of sub-state, return to normal, or acknowledgment.
LCT But OAT on ACK	Last Changed Time, but Original Alarm Time on ACK - that is, LCT will be used while the alarm is UNACKed, then OAT will be used after the alarm has been ACKed.

7. To print the alarm state, select the **Alarm State (UnAck, Ack)** option.

8. To print the alarm class, select the **Alarm Class (VALUE, DEV, ROC..)** option.

9. To print the alarm type, select the **Alarm Type (HIHI, LO, MAJDEV,...)** option.

   For more information on available alarm types, see "Alarm Types."

10. To print the alarm priority, select the **Priority** option.

11. To print the alarm name (tagname), select the **Alarm Name** option. In the **Length** box, type the number of characters (64 characters maximum) allowed for the alarm name.

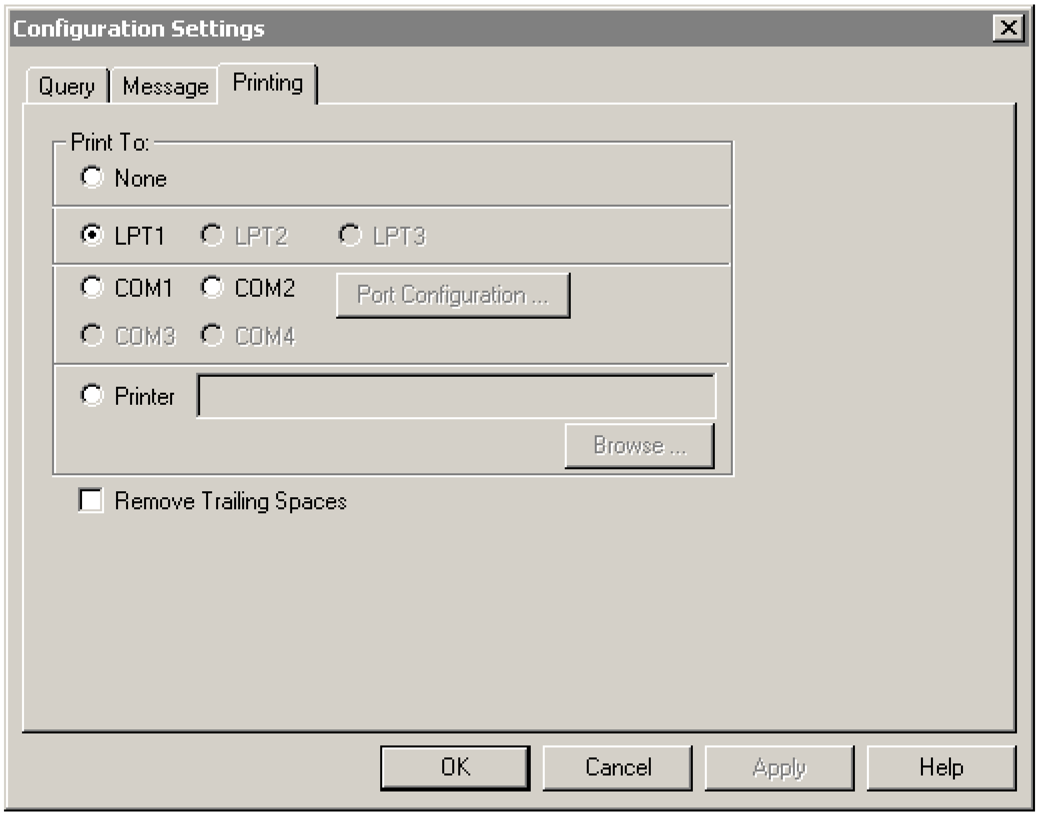
12. To print the alarm group name, select the **Group Name** option. In the **Length** box, type the number of characters (64 characters maximum) allowed for the alarm group name.

13. To print the name of the alarm provider, select the **Alarm Provider** option. In the **Length** box, type the number of characters (64 characters maximum) allowed for the alarm provider name.

14. To print the value of the tagname, select the **Value at Alarm** option. In the **Length** box, type the number of characters (32 characters maximum) allowed for the value. The number should be large enough to provide the desired level of precision.
15. To print the tagname's alarm limit, select the **Limit** option. In the **Length** box, type the number of characters (32 characters maximum) allowed for alarm limit. The number should be large enough to provide the desired level of precision.
16. To print the operator ID associated with the alarm condition, select the **Operator** option. In the **Length** box, type the number of characters (16 characters maximum) allowed for the operator's ID.
17. To print the Alarm Comment associated with the tagname, select the **Comment** option. In the **Length** box, type the number of characters (131 characters maximum) allowed for the comment.
18. Select the **Remove Trailing Spaces** option to remove the extra trailing spaces from a printed field when the length of the actual field value is less than what was configured for that field.

To configure alarm printing

1. Click the **Configure** menu in the **Alarm Printer** dialog box or click the **Configure Alarm** query tool on the Alarm Printer toolbar.
2. The **Configuration Settings** dialog box appears.
3. Select the **Printing** tab.


Configuration Settings

Query | Message | Printing

Print To:
None
[✓ LPT1] LPT2 LPT3
COM1 COM2 Port Configuration ...
COM3 COM4
Printer
Browse ...

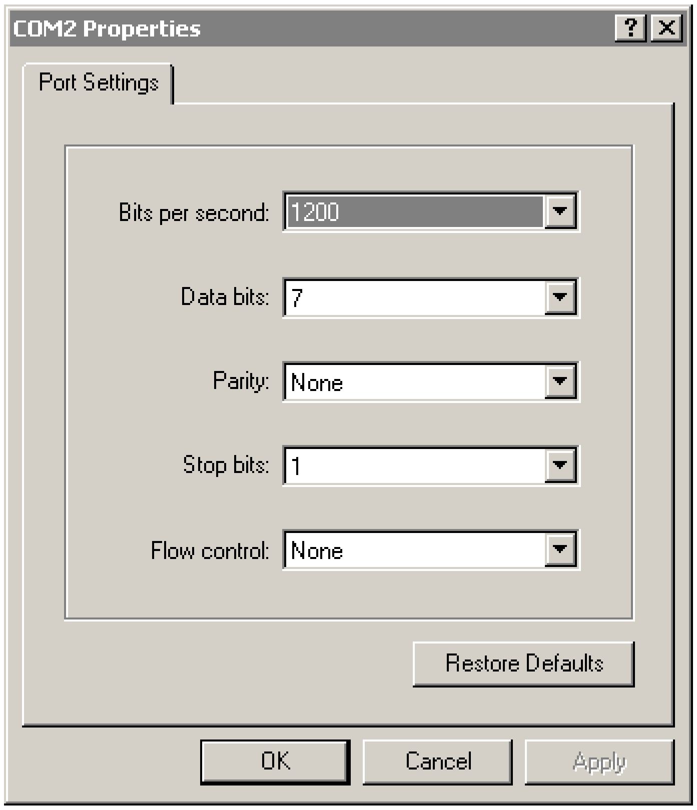
Remove Trailing Spaces

4. Select the option for the printer you want Alarm printer to use to print reports - the serial port (COM1, COM2, etc.), the parallel port (LPT1, LPT2, etc.), or a Windows printer.

Note The printer that you used to print your alarm status changes should not be used for other printing, as this will interfere with the Alarm Printer printing.

---

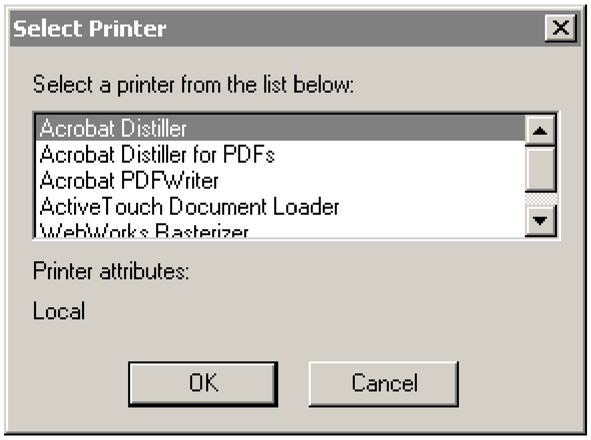
5. If a COM port has been selected, click **Port Configuration** to configure the serial port. The **COM# Properties** dialog box appears.



Bits per second: 1200
Data bits: 7
Parity: None
Stop bits: 1
Flow control: None

6. Configure your COM port by clicking each field’s arrow and selecting the appropriate setting for the COM port that you are using, then click **OK**.

7. If a Windows printer has been selected, type the name of the printer or click **Browse** to find an available printer. If you clicked **Browse**, the **Select Printer** dialog appears.



8. Scroll the list to find the desired printer, click the printer name, then click **OK**. The printer can be connected locally or to a network. Any Windows printer can be used as the output for the Alarm Printer.

Note If the desired printer does not appear in the Select Printer list, you can add the printer to your Windows configuration. For more information, see your Windows operating system documentation.

Running the Alarm Printer's Alarm Query

Each query logs all of the alarms specified in the Alarm Printer Configuration File (*.ALC) that is currently open or by the settings currently selected during Alarm Printer configuration if no file has been specified.

For more information on configuration options, see "Configuring the Alarm Printer."

You may run multiple queries with Alarm Printer, each query specifying different parameters. Each query will be conducted by a separate instance of Alarm Printer. If two instances of Alarm Printer are running the same query, the entries will be duplicated.

While Alarm Printer is running, you can manually start or stop queries as follows:

To start or stop an alarm query

On the **Query** menu, select **Start/Stop**.

Tip To quickly start or stop a query, click the start/stop query tool on the toolbar.

To automatically start the Alarm Printer with a specific file open

You can automatically launch the Alarm Printer and open a specific file when your system starts up by using the following command in a .BAT file:

```
ALMPRT.EXE MYQUERY.ALC
```

Where, **`MYQUERY.ALC`** is the name of the Alarm Printer Configuration file that opens. (Specifying the **`.EXE`** extension is optional.)

To automatically start the Alarm Printer and run a query

To prevent the loss of any query data due to a system inadvertently being shutdown and restarted, you can automatically launch the Alarm Printer and automatically run a specific query by using the following command in a .BAT file:

```
ALMPRT.EXE -q MYQUERY.ALC
```

By using the `-q` in the command, your query runs automatically when the system starts up. (Specifying the **`.EXE`** extension is optional).

Alarm DB Logger Utility

The InTouch Distributed Alarm system includes the Alarm DB Logger utility that logs alarms and events to a Microsoft SQL Server or Microsoft Data Engine (MSDE) database.

MSDE is a light version of SQL Server that has its own special attractions. High on the list is the ease with which you can attach databases initially built for MSDE to a full SQL Server service. There is no need to upsize the database or copy individual tables in a database from MSDE to a full SQL Server. This makes it more suitable for environments where it isn't cost effective to deploy vast computer resources. The maximum size of an MSDE database is 2 GB.

Alarm DB Logger is an Alarm Consumer. You configure it with an **alarm query** that defines which alarms are to be logged. You use the Alarm DB Logger to specify alarm queries and to log the resultant alarm records. These alarm queries are sent via the Alarm Consumer interface of the Distributed Alarm System.

The Alarm DB Logger also has the ability to auto-reconnect. When the connection to the database is lost, the logger continually checks for the database connection at regular intervals. When the connection is re-established, logging resumes.

The Alarm DB Logger reports all errors whether running as a service or a normal application to the Logger.

The Alarm DB Logger consists of the following two components:

1. **Alarm DB Logger Manager Utility** - This utility is a separate executable that solely takes care of starting and stopping the logging operations. It is launched and starts working either as a service or a normal application (depending upon the running mode you select in the Alarm DB Logger Manager). The logging utility retrieves the setting information from the registry and performs the logging.

2. **Alarm DB Logger Configuration Utility** - This utility takes care of user input and database configuration. The Alarm DB Logger Manager allows you to select the mode in which the Alarm DB Logger will run (either as a windows service or a normal application).

Note The Alarm DB Logger Manager only saves the setting values into the registry. The utility is responsible for starting and stopping the Alarm DB Logger. It is also responsible for displaying the status of Smart Cache. When Alarm DB Logger Manager (almlogwiz.exe) is closed while wwalmlogger.exe is running (either by pressing the Esc key or by clicking the "X" button on the upper right of the dialog box), the logging operation does not stop.

The progress control status indicates the percentage fill of the in-memory buffer with alarm records. The alarms are buffered when the SQL Server connection is down, and/or when the alarms are coming at a rate faster than the logging rate of Alarm DB Logger.

The Alarm DB Logger Configuration Utility provides you with the ability to:

- Run the application as a Windows Service or as a normal application
- Select the database connection type - SQL Server or MSDE
- Create the necessary SQL tables in the database
- Specify the alarm query that will be a part of the logging instance
- Select the logging mode - Detailed or Consolidated
- Enable/Disable logging of events
- Set performance tuning parameter - The auto reconnect rate is not the same as the performance tuning parameter. It depends on the time out for a connection attempt associated with SQL ServerServer.
- Store the setting in the registry

For more information, see "Alarm DB Logger Configuration."

Logging to SQL Database

The Alarm DB Logger logs alarm data into the database. If the OLEDB Provider is SQL Server, you will need to specify the SQL Server machine in the Alarm DB Logger Manager. Alarm DB Logger automatically creates the necessary data structures, if they do not already exist in the database.

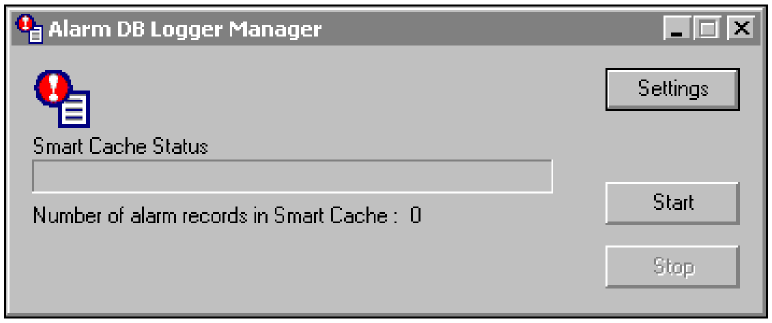
For more information, see "Distributed Alarm Database Views."

Using the Alarm DB Logger Utility

The Alarm DB Logger utility is a separate executable that takes care of the logging work. It is launched and starts working either as a service or a normal application (depending upon the running mode you select when you configure the Alarm DB Logger). The logging utility retrieves the setting information from the registry and performs the logging. The Alarm DB Logger is an Alarm Consumer.

To use the Alarm DB Logger utility

1. Start up the Alarm DB Logger Manager. The **Alarm DB Logger Manager** dialog box appears.



Smart Cache Status

Number of alarm records in Smart Cache : 0

Tip When minimized, the Alarm DB Logger Manager appears as an icon in the system tray. When you right-click the icon, a popup menu appears displaying the following commands:

Start - Begins the alarm logging process.
Stop - Ends the alarm logging process.
Settings - Opens the **Alarm DB Logger Manager - Configuration** dialog box.
Hide Window -Minimizes the Alarm DB Logger Manager dialog box to an icon in the system tray.
Show Window - Opens and maximizes the Alarm DB Logger Manager dialog box.
Close - Exits the Alarm DB Logger Manager Utility.

The **Smart Cache Status** indicates the percentage fill of the in-memory buffer with alarm records.

2. Click **Settings** to configure the Alarm DB Logger. The **Alarm DB Logger Manager - Configuration** dialog box appears.

   For more information on configuring the Alarm DB Logger, see "Alarm DB Logger Configuration."

3. Click **Start** to begin the alarm logging process.
4. Click **Stop** to end the alarm logging process.

Alarm DB Logger Configuration

Before you begin using the Alarm DB Logger, you must configure a few items, such as your database connection, your query list, the logging mode and so on. This information is configured via the Alarm DB Logger Manager.

The first Alarm DB Logger Manager Utility dialog box deals with your database connection. You can either select SQL Server or Microsoft Data Engine (MSDE) for storage of data. The Alarm DB Logger Configuration Utility also allows you to create necessary data structures (tables, views, and stored procedures) and to test your database connection.

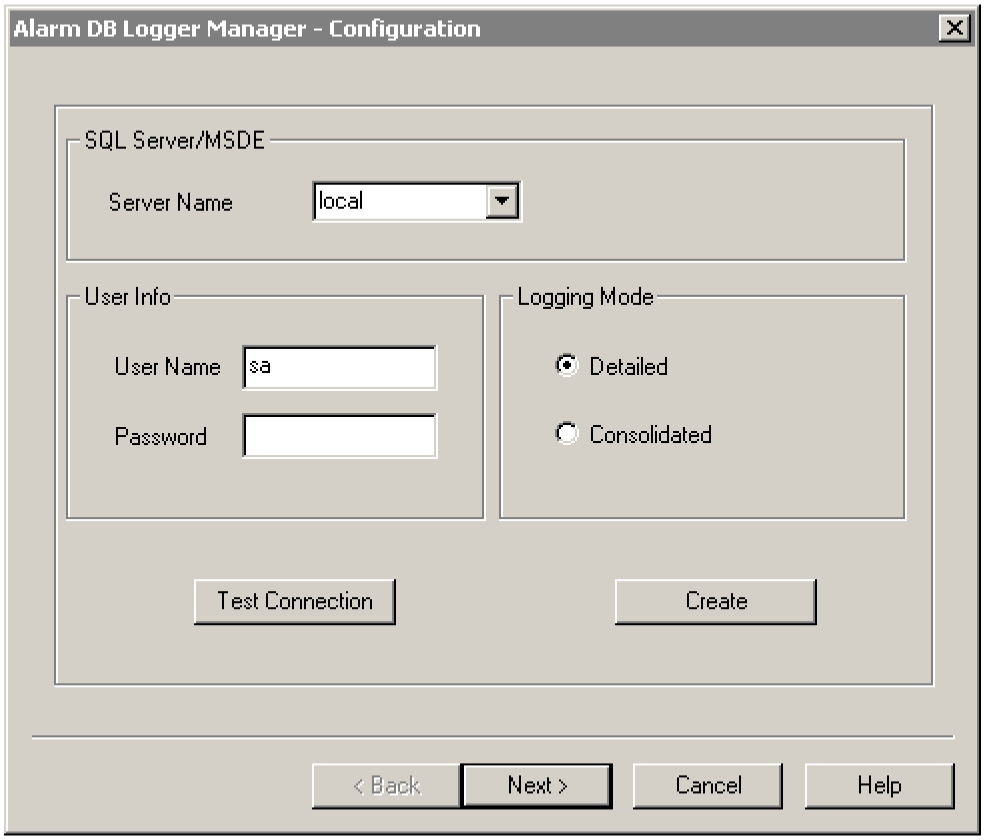
When the Alarm DB Logger Manager is used to create databases, tables and stored procedures, there are three user accounts with different authentication/security levels created at the same time. The Manager has the same authentication/security levels as IndustrialSQL Server. These levels are:

Account	**Password**
wwAdmin	wwadmin
wwPower	wwpower
wwUser	wwuser

To configure Alarm DB Logger

1. Start up the Alarm DB Logger Manager. The **Alarm DB Logger Manager** dialog box appears.

2. Click **Settings**. The **Alarm DB Logger Manager - Configuration** dialog box appears.



Server Name: local

User Name: sa

Password:

Logging Mode: [✓ Detailed] Consolidated

3. In the **SQL Server/MSDE** group, click the **Server Name** arrow to open the listing of available SQL/MSDE Servers and then select the name of the database server that you want to use.
4. In the **User Info** group, type in your **User Name** and **Password** in the respective entry boxes. (All **Password** characters are displayed as ** for security reasons).

5. In the **Logging Mode** group, click the **Detailed** or **Consolidated** radio button.

   The **Consolidated** mode reduces the size of the database. When alarms are logged to the database, the alarm and event records in the consolidated mode continue to display the original data, since there is no new record generated with each change. In Detail mode, the record shows the data at the time of the event.

   In Consolidated Mode, the first UNACK_ALM represents the origination of the alarm and the next UNACK_ALM represents the most recent sub-state change (if any). If the alarm goes through only one substate change, only one UNACK_ALM record will be seen. Otherwise, two UNACK_ALM records will be seen. However, if the alarm goes through several sub-state changes (e.g. Hi->HiHi->Hi), the second UNACK_ALM record will represent only the final sub-state.

   When **RTN Implies ACK** is selected and an alarm returns to normal without being ACKed, the ACK_RTN and ACK records are created in both Consoldated and details modes of logging.

Important! If you want to change between **Detailed** and **Consolidated** mode, you must recreate the database and you will lose any existing data. Creating a database will overwrite any pre-existing alarm database information.

6. Click **Test Connection** to test your connection to the target database.
7. Click **Create** to create new database/tables on the selected Server.
8. Click **Next** to access the **Alarm DB Logger Manager - Query Selection** dialog box.

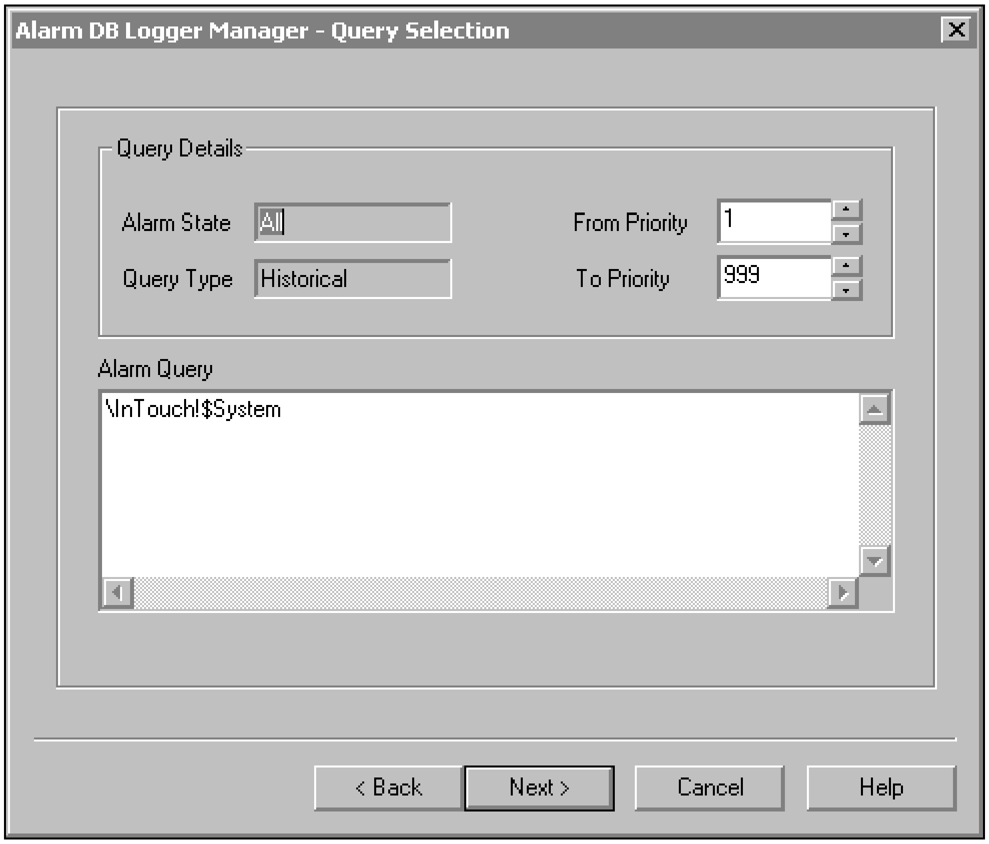
Alarm DB Logger Query Configuration

The second Alarm DB Logger Manager dialog box refers to the query details that need to be selected for alarm logging. It provides you with a multi-line text entry box to key in the query. It also allows you to select the **from** and **to** alarm priority values.

To configure query details

1. If it is not already running, start up the Alarm DB Logger Manager Utility. The **Alarm DB Logger Manager Configuration** dialog box appears.

2. Click **Next**. The **Alarm DB Logger Manager - Query Selection** dialog box appears.



Alarm DB Logger Manager - Query Selection

Query Details

Alarm State: All

Query Type: Historical

From Priority: 1

To Priority: 999

Alarm Query

```
\InTouch!$System
```

Note The **Alarm State** field displays the alarm state for logging. The **Query Type** field displays the type of query. (These are both read-only fields.)

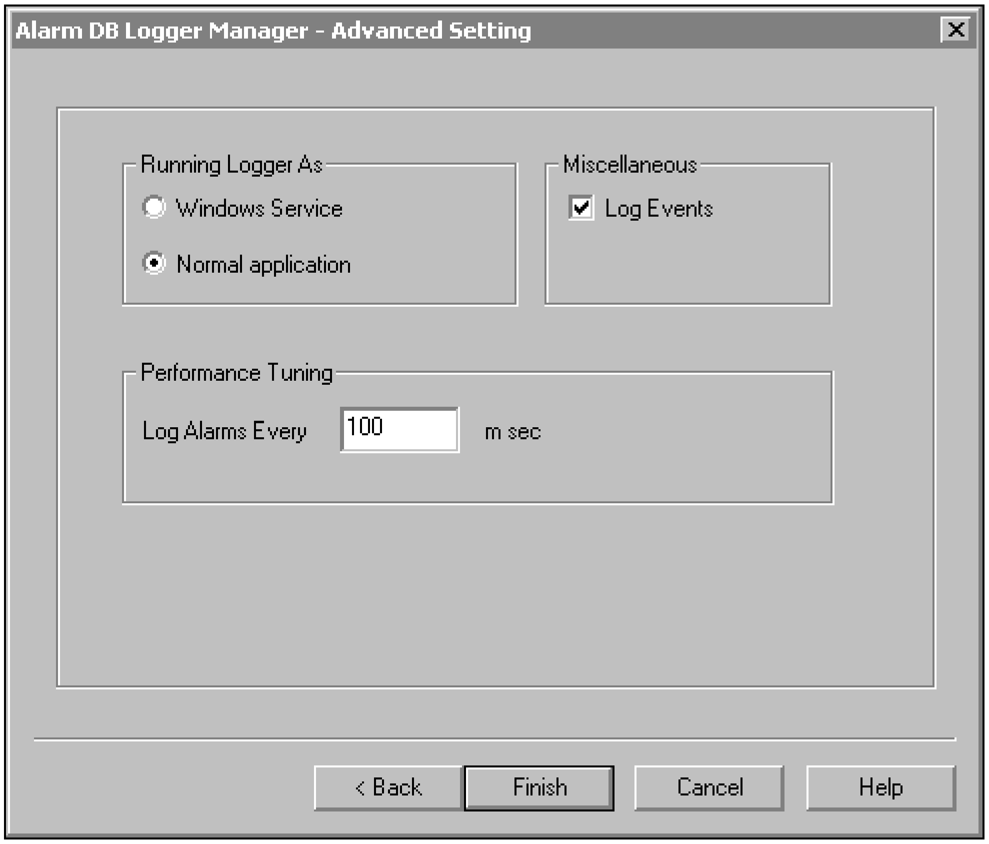
3. Click the **From Priority** arrows to select the starting of the alarm priority range.
4. Click the **To Priority** arrows to select the ending of the alarm priority range.
5. In the multi-line **Alarm Query** entry box, type the sets of InTouch alarm queries that you want to perform.
6. Click **Next**. The **Alarm DB Logger Manager - Advanced Setting** dialog box appears.

Alarm DB Logger Advanced Setting Configuration

The third Alarm DB Logger Manager dialog box refers to the advanced query settings, such as the running mode (either as a Windows Service or a normal application). It also provides you with the option for logging of events. You can tune the performance of the alarm logging by setting the frequency at which the alarms are to be flushed from the memory buffer to the database.

To configure advanced setting details

1. If it is not already running, start up the Alarm DB Logger Manager. The **Alarm DB Logger Manager Configuration** dialog box appears.
2. Click **Next**. The **Alarm DB Logger Manager - Query Selection** dialog box appears.
3. Click **Next**. The **Alarm DB Logger Manager - Advanced Setting** dialog box appears.


Alarm DB Logger Manager - Advanced Setting

Running Logger As
Windows Service [✓ Normal application]

Miscellaneous
Log Events ✓

Performance Tuning
Log Alarms Every 100 m sec

4. In the **Running Logger As** group, select the option that you want to use for the logger:
   - **Windows Service** - Configures the logger to function as a Windows service
   - **Normal Application** - Configures the logger to function as a normal application

5. Select the **Log Events** option to log all Events to the events table.
6. In the **Log Alarms Every # msec** entry box, type the interval at which the alarms are to be logged in milliseconds.

Note The reconnect rate is not the same as the msec rate set. The msec rate is the interval at which alarms should be read. The reconnect rate depends on the time out for a connection attempt associated with the SQL Server.

Setting the Performance Tuning parameter too low will impact system performance.

7. Click **Finish** to close the Alarm DB Logger Manager Utility and save all logger configuration settings into the registry.

New Alarm DB Logger Features

InTouch has three security modes. The security modes are Operating System-Based Security, InTouch Security and ArchestrA Security. When the security mode is set to Windows Security, the alarm provider sends the operator's full name and domain name as a part of the alarm record. Alarm DB Logger then stores these additional fields in the alarms database.

The InTouch Alarm DB Logger also stores the UNACK duration value in the alarms database. The UNACK Duration is the ACK time minus the time of the most recent alarm transition, where all dates and times are represented in GMT. Whenever an alarm record is ACKed, the Alarm DB Logger calculates the UNACK Duration as follows:

If the alarm has been through a sub-state transition (eg. Hi to HiHi), the UNACK duration is the ACK time minus the sub-state time. Otherwise, the UNACK duration is the ACK time minus the alarm origination time.

If the alarm record transition type is any type other than ACK, the Alarm DB Logger logs the default value for the UNACK Duration column (NULL value).

The Alarm DB View Control also displays the return duration. This is calculated as the return time minus the alarm origination time. The alarm duration is not stored in the database, but it is calculated on demand and presented by the database views.

InTouch and its related applications will NOT launch the Alarm DB Logger in the event that it is not already running.

Alarm DB Purge/Archive Utility

The Alarm DB Purge/Archive utility is a separate application installed along with InTouch. The Alarm DB Purge/Archive utility takes care of configuring your alarm database for Purge/Archive operations and showing the status of any purging activity. The utility also provides you with the ability to specify timer-based purging and archiving.

You will use the Alarm DB Purge/Archive Utility to:

- Select the type of records that need to be purged and/or archived (consolidated alarm records or alarm detail records).
- Specify automatic timer-based purging/archiving based on your configuration.
- Purge the whole database, if necessary.
- Set the time of day that the purging/archiving activity will be performed.
- Specify the time interval at which the purging/archiving will be performed. It can be daily, weekly or monthly.
- Display the status of the purging/archiving.
- Store the status in the log file. (You can use this log file in the future to check the state of your purging/archiving activity.)

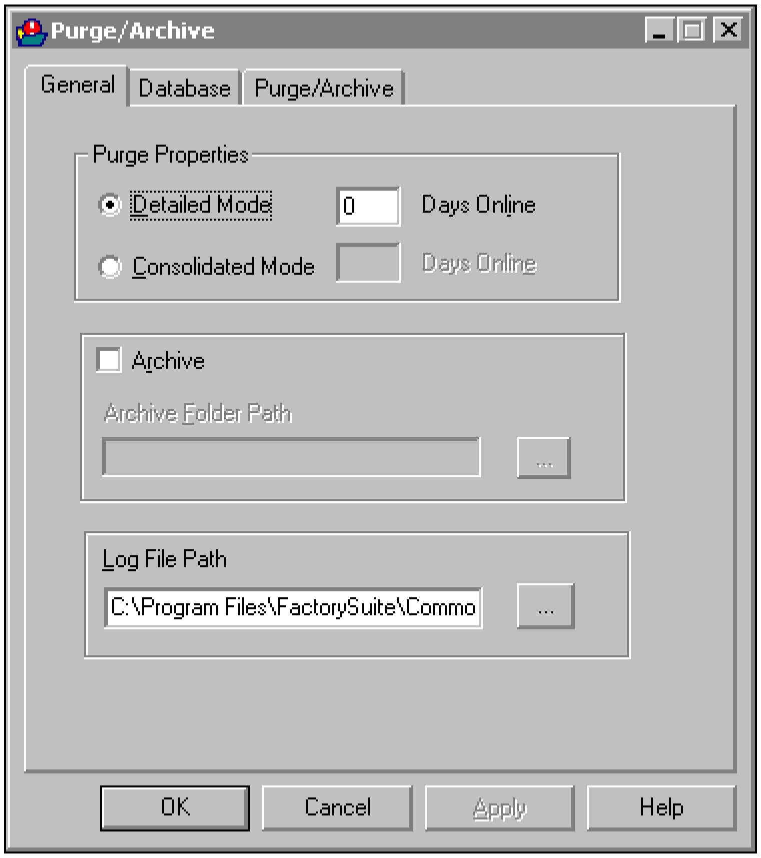
Purge/Archive General Properties Configuration

The Alarm DB Purge/Archive Utility **General** property sheet refers to the Purge/Archive properties. You can select the type of table that needs to be purged - either **AlarmDetail** or **AlarmConsolidated** tables. You are given the option to archive the data that you want to purge. You can choose the path to the folder in which you want to archive the data. Archive files are created for each table that is purged/archived. The archived file names are automatically created by a unique combination of the table name, date and time. For example, the name of the archive file for **AlarmMaster** that was archived on November 10, 2000 at 5:30 p.m. would be **AlarmMaster_11102000_1730.txt**.

For more information, see "Distributed Alarm Database Views."

To purge/archive your alarm database

1. Start up the Alarm DB Purge/Archive utility. The **Purge/Archive** dialog box appears.



(● Detailed Mode) 0 Days Online

( ) Consolidated Mode Days Online

[ ] Archive

C:\Program Files\FactorySuite\Commo

When minimized, the Alarm DB Purge/Archive Utility appears as an icon in the system tray. When you right-click the icon, a popup menu appears displaying the following commands: -

Command	Desciption
Purge All Now	Purges the entire database.
Purge Now	Starts purging now with the present settings.
Cancel Purge	Stops the current purging process. (This command is only active when purging is going on.) When you stop purging, the database is rolled back to its original state.
Activate	Activates timer-based purging.
DeActivate	Deactivates the timer for purging.
Test Now	Perform a test purge to verify your connection to the database and target locations
Clear Status	Clears the status window.

Command	Desciption
Hide Window	Minimizes the Alarm DB Purge/Archive Utility to an icon in the system tray.
Show Window	Opens and maximizes the Alarm DB Purge/Archive Utility.
Exit	Closes the Alarm DB Purge/Archive Utility.

If you right-click any Alarm DB Purge/Archive configuration property sheet the same popup menu appears.

2. In the **Purge Properties** group, select the option that you want to use for purging:
   - **Detailed Mode** to purge alarms that have been logged in the Detailed mode.
   - **Consolidated Mode** to purge alarms that have been logged in Consolidated mode.
3. The respective **Days Online** entry box becomes active after you select a **Purge Properties** mode. Type in the number of days you would like to retain the data. All data from the day previous to the number specified will be purged. (Valid entries are 0-9999.) If the number is 0, all records are purged except the records logged on the same day.
4. Select the **Archive** option and then specify the full path (up to 255 alphanumeric characters) to the database file that you want to archive either by typing the full path in the **Archive Folder Path** entry box, or by clicking the respective Browse (...) button to locate and select the file. (Once selected, the path will automatically appear in the entry box.)
5. In the **Log File Path** entry box, specify the full path (up to 255 alphanumeric characters) to the folder where you want to store your status file, or click the respective Browse (...) button to locate and select the folder. (Once selected, the path will automatically appear in the entry box.)

Note By default, the log file (wwalmpurge.log) is stored in the FactorySuite\Common folder, which can be overridden by the user. This file is appended every time a purge/archive activity is performed.

6. Click **Apply** to save the settings to the registry, or click **OK** to save the settings and close the Alarm DB Purge/Archive utility.

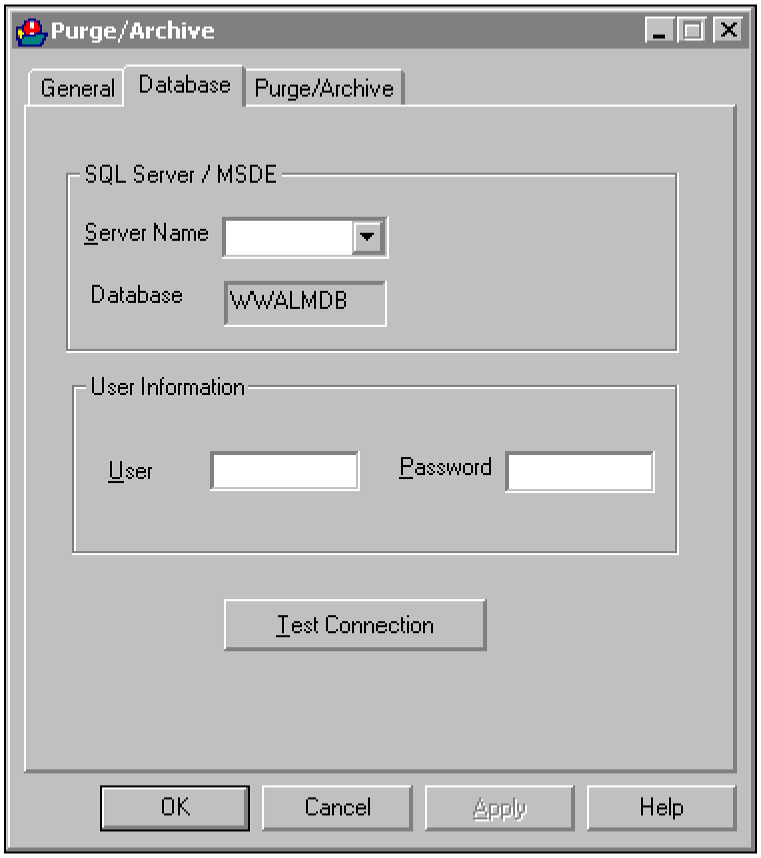
Purge/Archive Database Configuration

The Alarm DB Purge/Archive **Database** property sheet allows you to choose which server the purge/archive will access.

To configure the Alarm DB Purge/Archive Utility database

1. Start up the Alarm DB Purge/Archive utility. The **Purge/Archive** dialog box appears.

2. Click the **Database** tab to activate the **Database** property.



Database: WWALMDB

If you right-click the property sheet, a popup menu appears displaying the following commands:

Command	Desciption
Purge All Now	Purges the entire database.
Purge Now	Starts purging now with the present settings.
Cancel Purge	Stops the current purging process. (This command is only active when purging is going on.) When you stop purging, the database is rolled back to its original state.
Activate	Activates timer-based purging.
DeActivate	Deactivates the timer for purging.
Test Now	Perform a test purge to verify your connection to the database and target locations
Clear Status	Clears the status window.
Hide Window	Minimizes the Alarm DB Purge/Archive Utility to an icon in the system tray.

Command	Desciption
Show Window	Opens and maximizes the Alarm DB Purge/Archive Utility.
Exit	Closes the Alarm DB Purge/Archive Utility.

3. In the **SQL Server** / **MSDE** group, click the **Server Name** arrow to open the listing of available SQL/MSDE Servers, then select the name of the database server that you want to use.
4. The **Database** box displays the database name. (This is a read-only field that defaults to **WWALMDB**.)
5. In the **User Information** group, type in your **User** name and **Password** in the respective entry boxes. (All **Password** characters are displayed as ** for security reasons.)
6. Click **Test Connection** to test your connection to the database.
7. Click **Apply** to save the settings to the registry, or click **OK** to save the settings and close the Alarm DB Purge/Archive utility.

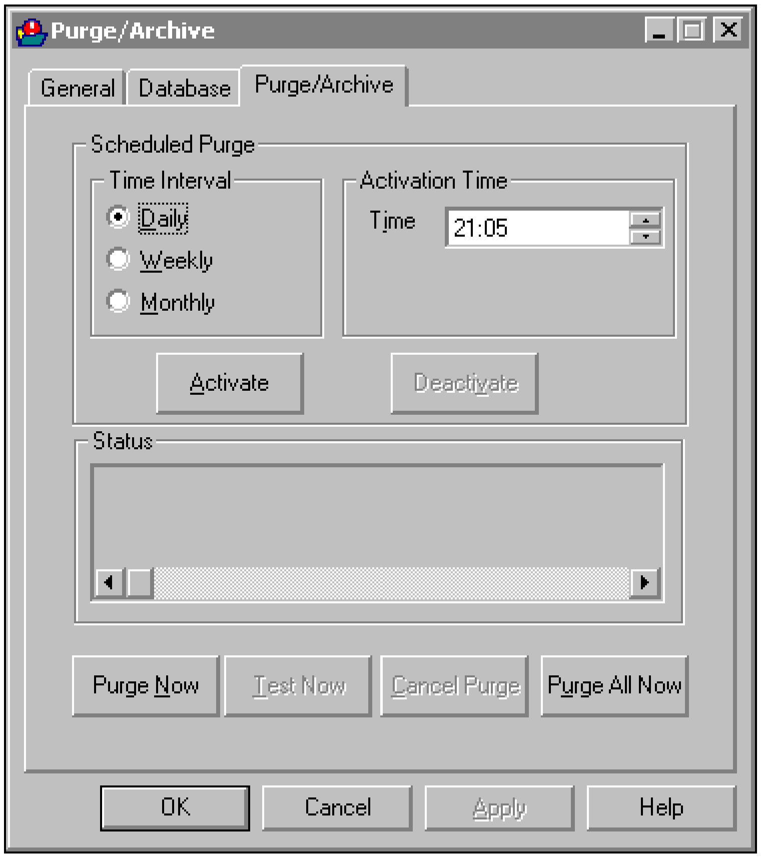
Purge/Archive Configuration

The Alarm DB Purge/Archive Utility **Purge/Archive** property sheet allows you to set the time at which automatic purging and archiving should trigger. It also allows you to perform a test purge to verify your connection to the database and target locations and to start and stop purging.

To configure the Alarm DB Purge/Archive Utility time interval

1. Start up the Alarm DB Purge/Archive utility. The **Purge/Archive** dialog box appears.

2. Click the **Purge/Archive** tab to activate the **Purge/Archive** property sheet.



If you right-click the property sheet, a popup menu appears displaying the following commands:

Command	Desciption
Purge All Now	Purges the entire database.
Purge Now	Starts purging now with the present settings.
Cancel Purge	Stops the current purging process. (This command is only active when purging is going on.) When you stop purging, the database is rolled back to its original state.
Activate	Activates timer-based purging.
DeActivate	Deactivates the timer for purging.
Test Now	Perform a test purge to verify your connection to the database and target locations
Clear Status	Clears the status window.
Hide Window	Minimizes the Alarm DB Purge/Archive Utility to an icon in the system tray.

Command	Desciption
Show Window	Opens and maximizes the Alarm DB Purge/Archive Utility.
Exit	Closes the Alarm DB Purge/Archive Utility.

3. Select the **Time Interval** option that you want to use:
   - **Daily** - Starts purging daily at the time selected in **Time** field.
   - **Weekly** - Starts purging weekly on the day selected in the **Day** field and at the time selected in Time field.
   - **Monthly** - Starts purging every month on the day selected in the **Day** field and at the time selected in the **Time** field.

Note The **Day** default is blank. You must select a day if you select either **Weekly** or **Monthly**.

4. After you have selected the **Time Interval** option, click the **Time** up/down arrows to select the time of day that you want the purge/archive activity to begin.
5. After you have selected the **Time Interval** option (for **Weekly** and **Monthly** only), click the **Day** arrow to select the day on which you want to perform the purge-archive activity. (Day of the week when **Time Interval** is **Weekly**, or day of the month when **Time Interval** is **Monthly**.)
6. The status of the purging activity is displayed in the **Status** area.
7. Click **Purge Now** to start purging now with the present settings.

Note Clicking **Purge Now** overrides the activation time and starts the purging and archiving immediately. **Purge Now** checks for the presence of an archive file and appends to the same. In case the archive file is not present, the file is created as per the naming convention and then used for archiving. **Purge Now** does not delete entries in tables like **ProviderSession**, **Query**, and **Cause** that are linked to the main tables like **AlarmMaster** through foreign key constraints. The related records in these tables are written to the files to maintain the data consistency and also retained on the database.

8. Click **Test Now** to perform a test purge to verify your connection to the database and target locations. (This button is only active if the **Archive** option in the General tab has been selected. It creates blank archive files in the specified archive path.)
9. Click **Cancel Purge** to stop purging. (This button is active only when purging is going on.) When you stop purging, the database is rolled back to its original state.
10. Click **Purge All Now** to purge the entire database.

Caution! Select **Purge All Now** only when Alarm DB Logger is not running. When Alarm DB Logger is logging into the database, if **Purge All Now** is committed successfully, the Alarm DB Logger stops logging data into the database and starts buffering the records.

11. Click **Apply** to save the settings to the registry, or click **OK** to save the settings and close the Alarm DB Purge/Archive utility.

Alarm DB Restore Utility

The Alarm DB Restore Utility is used to restore your archived alarm records from a backup file to your database.

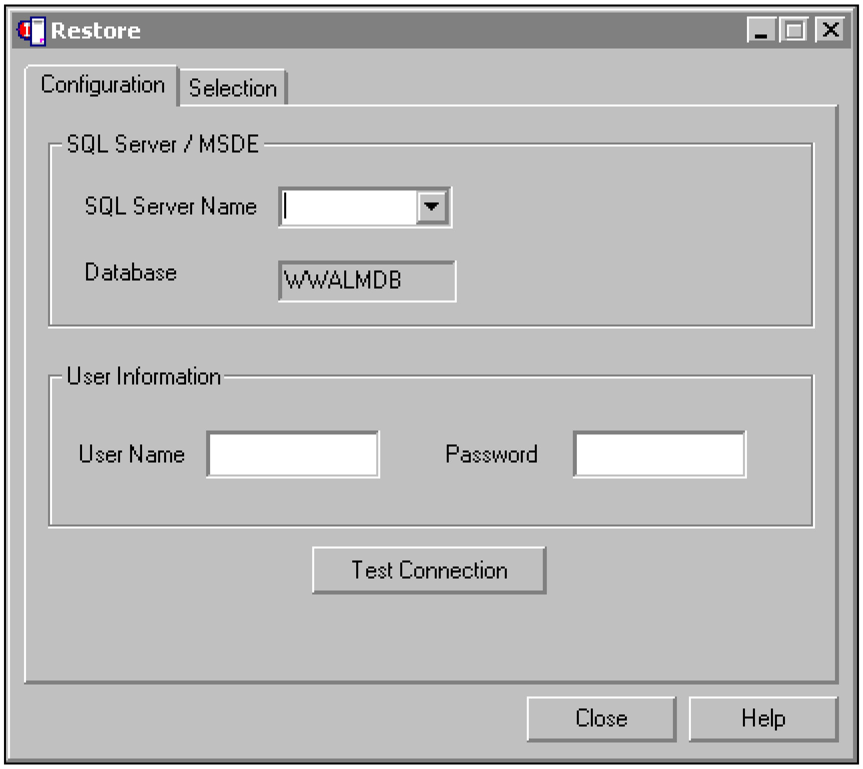
The Alarm DB Restore Utility assists you in restoring the archived data to a database for performing further operations.

Restore Database Configuration

The **Configuration** property sheet is used to specify the database into which the data needs to be restored. If the specified database is not present on the server, you will be prompted to create a new database on the server. Upon confirmation, the new database is created with the default parameters on the server indicated.

To configure a database for restoring:

1. Start up the **Alarm DB Restore** utility. The **Restore** dialog box appears with the **Configuration** tab active.



When minimized, the Alarm DB Restore Utility appears as an icon in the system tray. When you right-click the icon, a popup menu appears displaying the following commands:

Command	Description
Restore	Begins the restoring process.
Cancel Restore	Ends the restoring process.
Clear Status	Clears the status window.
Hide Window	Minimizes the Alarm DB Restore Utility to an icon in the system tray.
Show Window	Opens and maximizes the Alarm DB Restore Utility.
Exit	Closes the Alarm DB Restore Utility.

If you right-click any Alarm DB Restore configuration property sheet the same popup menu appears.

2. In the **SQL Server / MSDE** group, click the **Server Name** arrow to open the listing of available SQL/MSDE Servers, then select the name of the database server that you want to use.

3. The **Database Name** box displays the database name. (This is a read-only field that defaults to WWALMDB.)

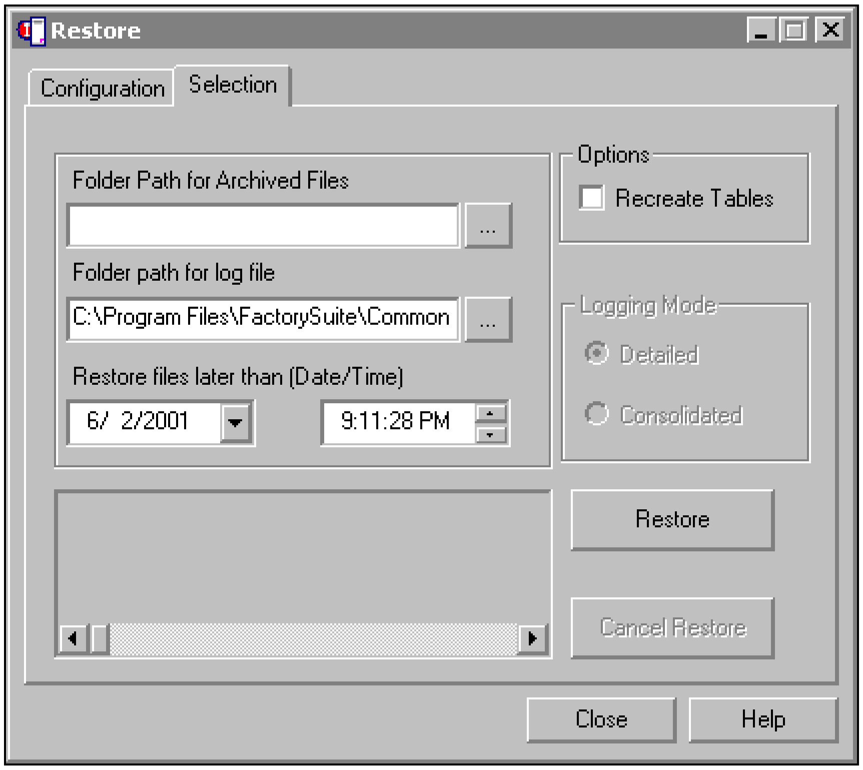
4. In the **User Information** group, type in your **User** name and **Password** in the respective entry boxes. (All **Password** characters are displayed as ** for security reasons.)
5. Click **Test Connection** to test your connection to the database.

Restore Filename Configuration

The Alarm DB Restore Utility **Selection** property sheet refers to the filename from which restoration needs to be done. It also displays a status list box that shows the progress status of the restoration of data into the database. You can cancel the restore in progress from this dialog box. If you cancel the restore, the database is rolled back to its original state.

To select a database for restoring:

1. Start up the Alarm DB Restore utility. The **Restore** dialog box appears with the **Configuration** tab active.
2. Click the **Selection** tab to activate the **Selection** property sheet.



Folder Path for Archived Files:
Folder path for log file: C:\Program Files\FactorySuite\Common
Restore files later than (Date/Time): 6/ 2/2001 9:11:28 PM
[ ] Recreate Tables
Logging Mode: (● Detailed) Consolidated

If you right-click the property sheet, a popup menu appears displaying the following commands:

Command	Description
Restore	Begins the restoring process.
Cancel Restore	Ends the restoring process.

Command	Description
Clear Status	Clears the status window.
Hide Window	Minimizes the Alarm DB Restore Utility to an icon in the system tray.
Show Window	Opens and maximizes the Alarm DB Restore Utility.
Exit	Closes the Alarm DB Restore Utility.

3. In the **Folder Path** for **Archived Files** entry box, type the full path (up to 255 alphanumeric characters) to the location of the archived files, or click the respective Browse (...) button to locate and select the folder where archived files are to be created and stored. (Once selected, the path will automatically appear in the entry box.)
4. In the **Folder path for log file** entry box, type the full path (up to 255 alphanumeric characters) where the log files will be created and stored, or click the respective Browse (...) button to locate and select the folder. (Once selected, the full path for the log file will automatically appear in the entry box.)
5. Select **Recreate Tables** if you want to recreate the tables in the target database. If you do not select this option, the data will be appended to the existing tables.

Caution! Recreating tables results in the loss of existing data.

6. When you select the **Recreate Tables** option, the **Logging Mode** group becomes active. Select the mode that you want to use:
   - **Detailed** - Restores the detailed tables only
   - **Consolidated** - Restores the consolidated tables only
7. In the **Restore files later than (Date/Time)** fields, select the date and time from which the restoration is to be done.

Note The date specifies the date from which the restoration is to be done. The time specifies the time that the restoration is to begin on the indicated day. Both of these fields default to the current date and time.

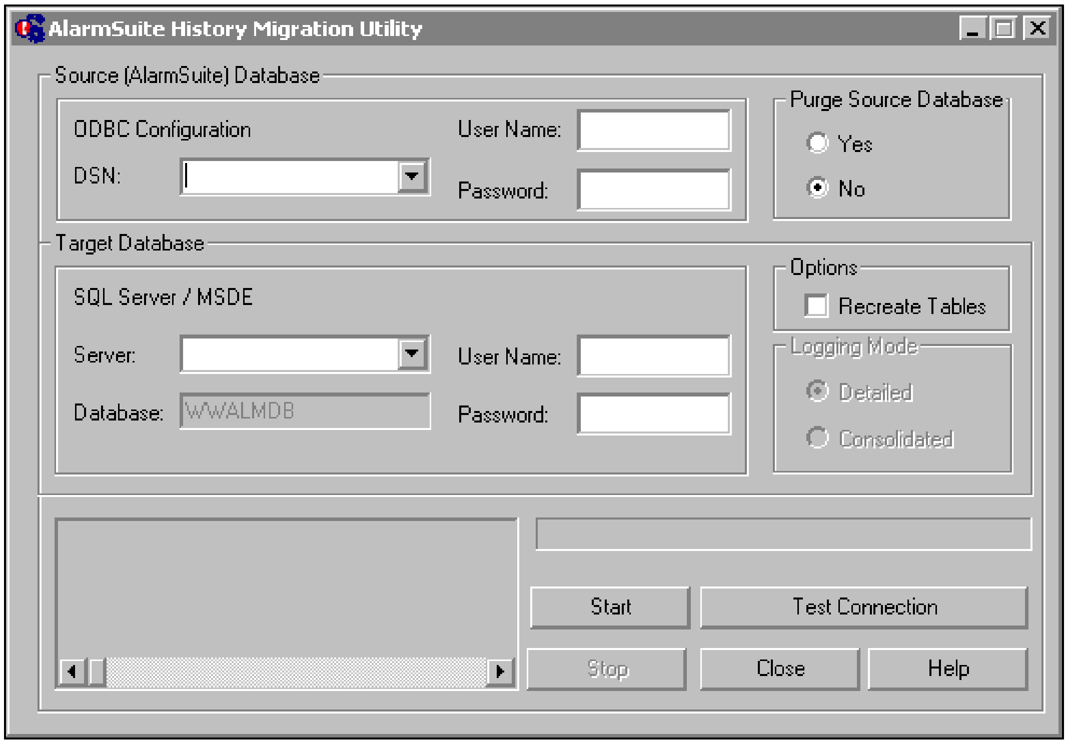
AlarmSuite History Migration Utility

The AlarmSuite History Migration Utility is installed along with InTouch. The AlarmSuite History Migration Utility takes care of migrating your AlarmSuite data from an AlarmSuite Database to the new InTouch Distributed Alarm System’s database schema.

For more information, see "Distributed Alarm Database Views."

To migrate AlarmSuite history:

1. Start up the AlarmSuite History Migration utility. The **AlarmSuite History Migration Utility** dialog box appears.



Database: WWALMDB

Purge Source Database: Yes [✓ No]

Logging Mode: [✓ Detailed] Consolidated

2. In the **Source (AlarmSuite) Database** group, click the **DSN** arrow to open the list of ODBC DSN names configured in the local machine.
3. Select the ODBC DSN name that you want to use.
4. In the **User Name** entry box, type your user name for the DSN.
5. In the **Password** entry box, type your user password for the DSN. (All entries will be shown as ** for security reasons.)
6. Select the **Purge Source Database** option that you want to use:
   - **Yes** - Purge the source database at the end of data migration. (Table creation is part of the transaction, but creation of the database is not. If you select **Yes**, the transaction with purging is committed after migration the data from the source database.)
   - **No** - Do not purge the source database at the end of data migration
7. In the **Target Database** group, click the **Server Name** arrow (in the **SQL Server** / **MSDE** group) to open the listing of available SQL/MSDE Servers. Select the name of the database server that you want to use.
8. The **Database Name** box displays the database name. (This is a read-only field that defaults to WWALMDB.)
9. In the **User Name** entry box, type your SQL Server user name.
10. In the **Password** entry box, type your SQL Server password. (All **Password** characters are displayed as ** for security reasons.)

11. Select **Recreate Tables** if you want to recreate the tables in the target database. If you do not select this option, the data will be appended to the existing tables.

Note Recreating tables results in the loss of existing data.

12. When you select the **Recreate Tables** option, the **Logging Mode** group becomes active. Select the mode that you want to use:
    - **Detailed** - Restores the detailed tables only
    - **Consolidated** - Restores the consolidated tables only
13. Click **Test Connection** to test your connection to the source and target databases.
14. Click **Start** to begin migrating the data from the source database to the target database.
15. Click **Stop** to abort the migration process that is in progress. This rolls back the transaction. (Clicking **Close** is equivalent to clicking **Stop**.)

Note Table creation is part of the transaction, but creation of the database is not. If you have set the **Purge Source Database** option to **Yes**, the transaction with purging will be committed after migrating the data from the source database.

16. Once the migration process is complete, click **Close** to exit the utility.

Note While data migration is taking place, the source or the destination database should not be used for logging new data. This may lead to some loss of data integrity leading to false search results.

C H A P T E R 1 2

Real-time and Historical Trending

InTouch provides you with two types of trend display objects: "Real-time" and "Historical." You can configure both trend objects to display graphical representations of multiple tagnames over time. Real-time trends allow you to chart up to four pens (data values), while Historical trends allow you to chart up to eight pens. Both types of trends are created using special tools in WindowMaker. InTouch also provides you with complete control over the configuration of your trends. For example, you can specify the time span, value range, grid resolution, location of time stamps, location of value stamps, number of pens, and color attributes.

The FactorySuite Productivity Pack includes a Pen Configuration Grid that allows you to chart 16 pens. For more information, see your *Productivity Pack User's Guide.*

InTouch also supports a distributed history system that allows you to retrieve historical data from any InTouch historical log file, even across a network.

In addition to its trending capabilities, InTouch includes the HistData utility, which is designed to work with InTouch historical log files. The HistData utility converts encrypted historical log files (.lgh) to comma separated variable (.csv) files for use in spreadsheet or text editing environments such as Microsoft Excel.

Contents

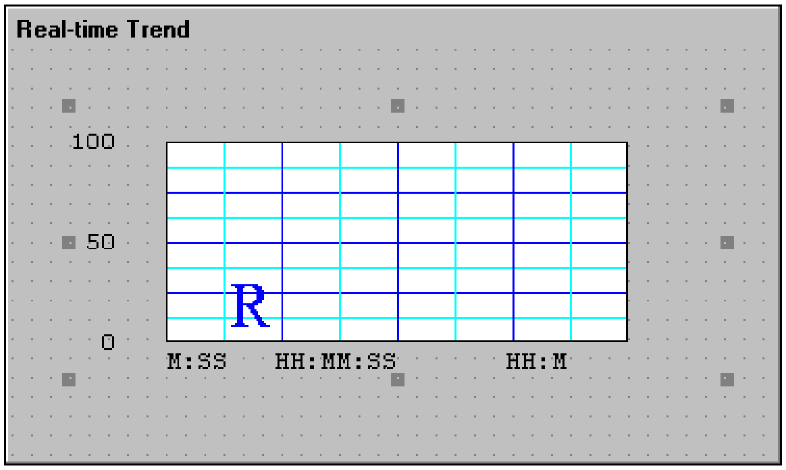
Real-time Trends

Real-time trends are dynamic. They are updated continuously during runtime. They plot the changes of up to four local tagnames or expressions as they occur.

Creating a Real-time Trend

To create a real-time trend

1. Select the real-time trend tool in the **Drawing Toolbar**.
2. Click in the window, then drag the mouse diagonally to draw a rectangle the size that you want your trend to be. (You can draw the trend chart any size you choose, and there is no limit to the number of charts you can place on a screen.)
3. Release the mouse. The real-time trend object appears in the window:



In runtime, the data is written in the trend from the right to the left.

4. Double-click the trend to open its configuration dialog box.

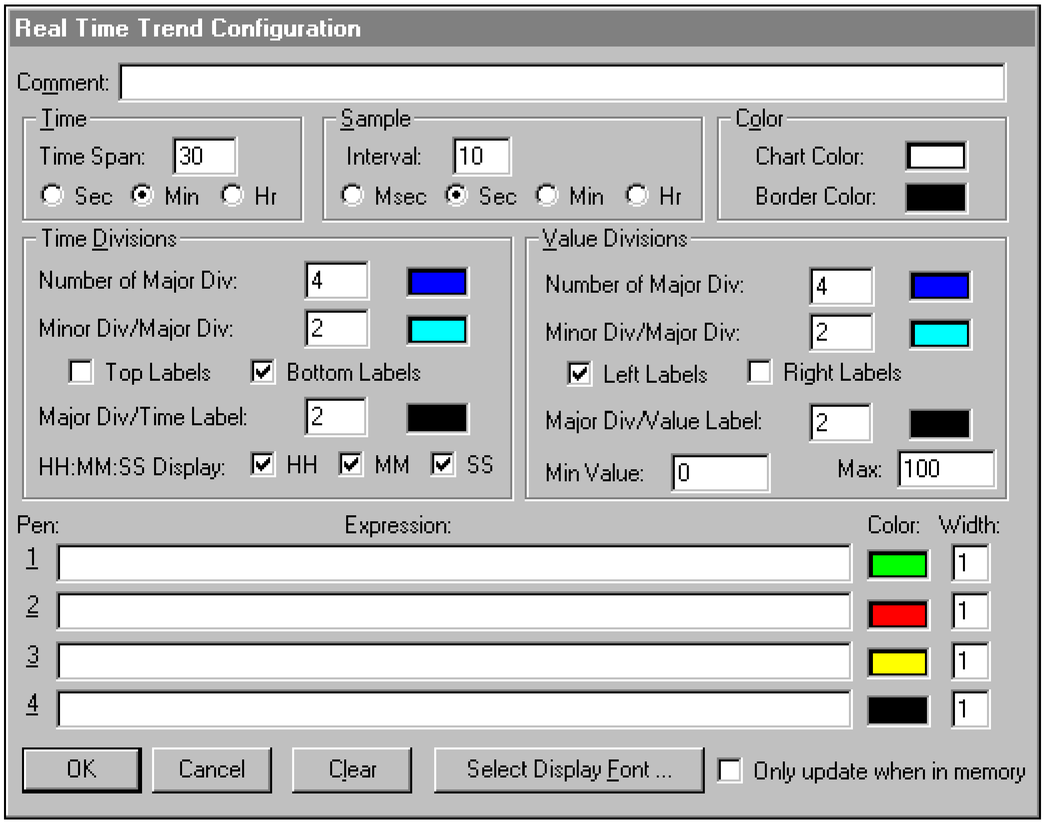
   A trend object is like any other object drawn in WindowMaker. It can be moved by grabbing it with the mouse, or it can be resized by grabbing one of the object "handles." You can place multiple trends in a window.

Configuring a Real-time Trend

The first time you paste a real-time trend object, the system default configuration settings are used. Once you have configured a real-time trend, the next one you create will, by default, be configured with the same settings.

To configure a real-time trend

1. Double-click the trend or, with the trend selected, on the **Special** menu, click **Animation Links**. The **Real Time Trend Configuration** dialog box appears.



Real Time Trend Configuration

Comment:

Time — Time Span: 30 — Sec [✓ Min] Hr

Sample — Interval: 10 — Msec [✓ Sec] Min Hr

Color — Chart Color: ⬜ — Border Color: ⬛

Time Divisions
- Number of Major Div: 4 🟦
- Minor Div/Major Div: 2 🟦
- ☐ Top Labels ☑ Bottom Labels
- Major Div/Time Label: 2 ⬛
- HH:MM:SS Display: ☑ HH ☑ MM ☑ SS

Value Divisions
- Number of Major Div: 4 🟦
- Minor Div/Major Div: 2 🟦
- ☑ Left Labels ☐ Right Labels
- Major Div/Value Label: 2 ⬛
- Min Value: 0 Max: 100

Pen:	Expression:	Color:	Width:
1		🟩	1
2		🟥	1
3		🟨	1
4		⬛	1

Tip If you right-click a text box in the real-time trend configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

Note All entries made in the Real Time Trend Configuration dialog box are independent of the size of the trend and are not modifiable at runtime.

2. In the **Time Span** box, type length of time you want to display horizontally (x-axis) on the trend then select time increment option for the length of time.

   For example, it you enter 30 for the **Time Span** then select **Min**, the horizontal time span of the chart will be 30 minutes long.

3. In the **Sample Interval** box, type the frequency at which the trend expression will be evaluated and the chart updated, then select the option for the time increment to which the number will relate.

   For example, if you enter 10 for the **Interval** and select **Sec** for the time increment, the expression will be evaluated every 10 seconds.

4. In the **Color** group, click the **Chart Color** box to open the InTouch color palette. Click the color in the palette that you want to use for the trend's background.
5. In the **Color** group, click the **Border Color** box to open the InTouch color palette. Click the color in the palette that you want to use for the trend's border.

Tip Repeat this process for all color selections.

6. In the **Time Divisions** group, in the **Number of Major Div** box, type the number of major time divisions you want in the trend, and then select the color you want to use for the division lines.

   The maximum time between major time divisions is (65536 sec) or 18 hours, 12 minutes, 16 seconds.

   The number of major time divisions must be an even multiple of the number of **Minor Div/Major Div**.
7. In the **Time Divisions** group, in the **Minor Div/Major Div** box, type the number of minor time divisions that you want to be visible within each major time division, and then select the color you want to use for the division lines.
8. In the **Time Divisions** group, select **Top Labels** if you want time labels displayed at the top of the trend.
9. In the **Time Divisions** group, select **Bottom Labels** if you want time labels displayed at the bottom of the trend.

   Your trend can have both top and bottom labels or no labels at all.
10. If you are using time labels, in the **Time Divisions** group, in the **Major Div/Time Label** box, type the number of time labels per major time division line that you want for the trend.
11. In the **Time Divisions** group, select the color you want to use for the major time division lines.
12. The settings in **Value Divisions** group are configured the same way as the settings in the **Time Divisions** group, except the minor and major value divisions set the vertical value (y-axis) range for the trend. This range uses Engineering Units and is the same for all trended tagnames.

Tip To display decimal points for the minor and major value divisions at runtime, they must be formatted here to do so. For example, 0.00 to 100.00.

13. In the **Expression** box, type the local tagname or expression that you want each **Pen** to trend.

    Up to four pens can be visible in a trend. The pens can be used to display any local tagname or an expression that contains one or more local tagnames. (Message type tags cannot be logged or trended.) The ability to trend expressions is useful in creating custom displays to show tagnames with widely different ranges.
14. Click the color box to select the color that you want each pen to use to plot each tagname in the trend.

15. In the **Width** box, type the number of pixels wide you want each pen to be. Selecting a pen width greater than 1 significantly reduces performance in trend updating and printing of the trend.
16. Click **Select Display Font** to access the **Font** dialog box to select the font, style and size that you want to use when you print the trend.
17. Select **Only update when in memory** if you want your trend to update only when it is displayed in the active window.

    If you do not select this option, the trend will always be updated, even if it is not in an open window. This may result in slightly slower system performance of the overall system.
18. Click **OK**.

To Increase real-time trending performance

1. Set the pen width to '1'.
2. Be sure no other objects are placed on top of the Real-time trend.
3. Lower the number of "samples" being taken.

For example, if you set the Time Span to 30 minutes and the Sample Interval to 2 seconds, the number of samples taken during the 30 minutes will be calculated as:

```
30* 60/2 = 900
```

If you set the Time Span to 30 minutes and the Sample Interval to 5 seconds, the number of samples taken during the 30 minutes will be calculated as:

```
30* 60/5 = 360
```

Historical Trends

Historical trends provide you with a "snapshot" of data from a time and date in the past. They are not dynamic. Unlike real-time trends, historical trends are only updated when they are instructed to do so either through the execution of a QuickScript or an action by the operator, for example, clicking a button.

Up to eight tagnames (pens) can be trended at one time with no limit to the number of trends displayed. You have complete flexibility in designing the interface to your trend. You can create "scooters" that the operator "slides" over the trend to access a variety of data based on the scooter's current location. For example, when the operator positions the scooter over an area on the trend that has visible data, the time and values at that location for all database values being trended is returned to you.

You can also create buttons to zoom in and out between the scooters or to data, such as the maximum to minimum value. Average and standard deviation can be displayed for the complete chart or for the area between the scooters. Historical trends can also be scrolled by any amount of time. You can create custom scales and link them to the **.MinEU** and **.MaxEU** Tagname Dotfield**s** to display the minimum and maximum Engineering Units.

The distributed history system extends the retrieval capabilities of historical trends to include remote log databases. This system allows information from multiple historical log databases to be displayed in a single trend.

Note You must select the Log Data option for each tagname in the Tagname Dictionary in order for it to be trended.

For more information on logging tagnames, see "Logging Tagnames."

Also see, "Configuring Historical Logging Properties."

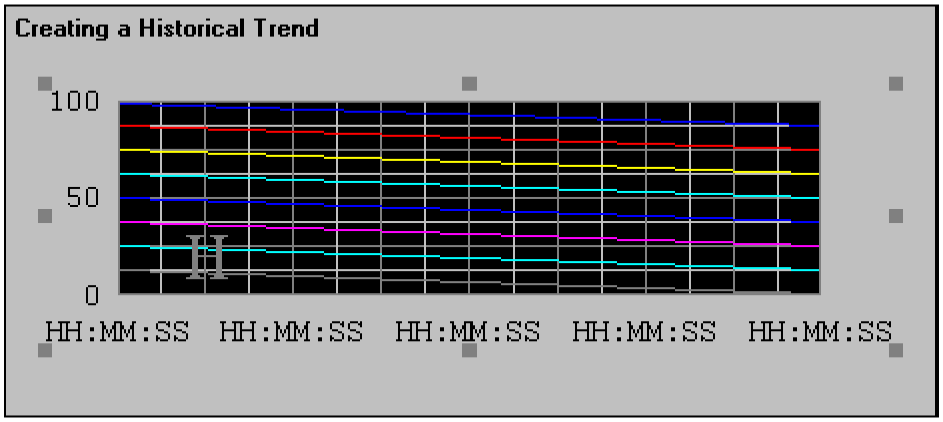
Creating a Historical Trend

To create a historical trend

1. Select the historical trend tool in the **Drawing Toolbar**. Historical trend tool used to draw historical trend objects.
2. Click in the window and then, drag the mouse diagonally to draw a rectangle the size that you want your trend to be.

   Tip You can draw the trend chart any size you choose. You can also place multiple trends in your window.

3. Release the mouse. The historical trend appears in the window:



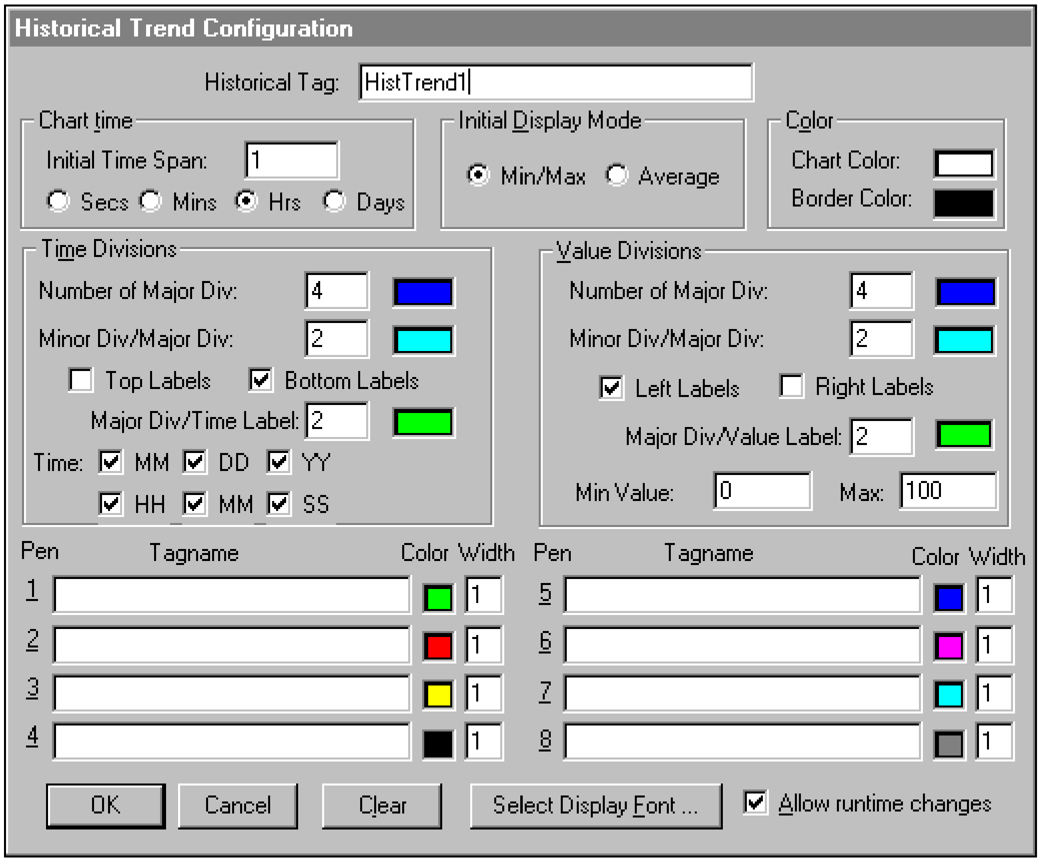
Tip A trend object is like any other object drawn in WindowMaker. It can be moved by grabbing it with the mouse or it can be sized by grabbing any its "handles."

Configuring a Historical Trend

The first time you paste a historical trend object, the system default configuration settings are used. Once you have configured a trend, the next one you create will, by default, be configured with the same settings.

To configure a Historical trend

1. Double-click the trend or, with the trend selected, on the **Special** menu, click **Animation Links**. The **Historical Trend Configuration** dialog box appears.



Historical Trend Configuration

Historical Tag: HistTrend1

Chart time
Initial Time Span: 1
Secs / Mins / [✓ Hrs] / Days

Initial Display Mode
[✓ Min/Max] / Average

Color
Chart Color: ⬜
Border Color: ⬛

Time Divisions
Number of Major Div: 4 🟦
Minor Div/Major Div: 2 🟦
☐ Top Labels ☑ Bottom Labels
Major Div/Time Label: 2 🟩
Time: ☑ MM ☑ DD ☑ YY
☑ HH ☑ MM ☑ SS

Value Divisions
Number of Major Div: 4 🟦
Minor Div/Major Div: 2 🟦
☑ Left Labels ☐ Right Labels
Major Div/Value Label: 2 🟩
Min Value: 0 Max: 100

Pen	Tagname	Color	Width	Pen	Tagname	Color	Width
1		🟩	1	5		🟦	1
2		🟥	1	6		🟪	1
3		🟨	1	7		🟦	1
4		⬛	1	8		⬛	1

☑ Allow runtime changes

Tip If you right-click a text box in any historical trend configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

2. In the **Historical Tag** box, type the tagname that you want to use for the trend.

   If the tagname you type is not currently defined in the Tagname Dictionary, you will be asked if you want to define it now. If you select **Yes** to define the tagname now, InTouch will automatically display the **Tagname Dictionary** dialog box and default the tagname type to **Hist Trend**. (The tagname must be defined as a **Hist Trend** type.) You must use a different tagname for each historical trend.

3. In the **Initial Time Span** box, type length of time you want to display horizontally (x-axis) on the trend then select time increment option for the length of time.

   Example: If you enter 30 for the **Initial Time Span** then select **Min**, the horizontal time span of the chart will be 30 minutes long.

4. Select **Initial Display Mode** that you want to use for the trend as follows:

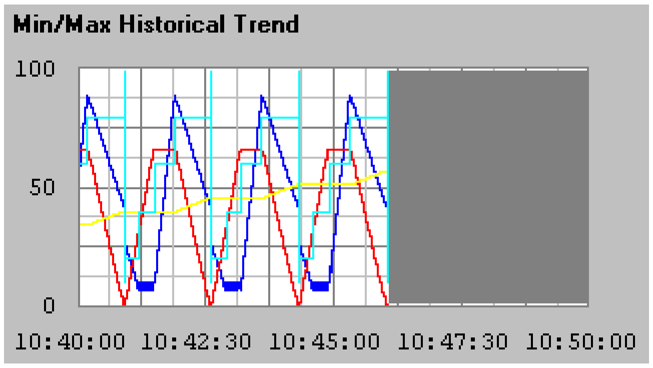
Initial Display Mode	Description
Min/Max	Each pixel on the chart will display the minimum to maximum range the point covered in the time represented by that pixel.
Average	Displays the average value for each pixel, for example, time segment.

5. In the **Color** group, click the **Chart Color** box to open the InTouch color palette. Click the color in the palette that you want to use for the trend's background.

6. In the **Color** group, click the **Border Color** box to open the InTouch color palette. Click the color in the palette that you want to use for the trend's border.

   Repeat this process for all color selections.

   The blank area on the right side indicates that no data was collected during that time period either because WindowViewer was not running or, historical logging was turned off.



7. In the **Time Divisions** group, in the **Number of Major Div** box, type the number of major time divisions you want in the trend, and then select the color you want to use for the division lines.

   The number of major time divisions must be an even multiple of the number of **Minor Div/Major Div**.

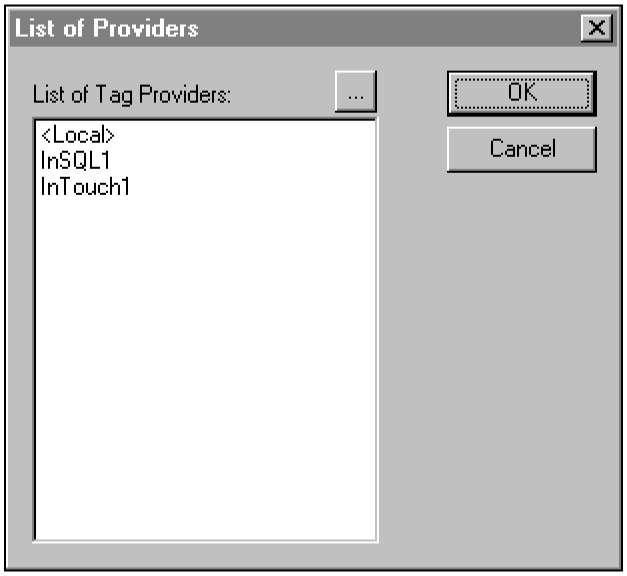
8. In the **Time Divisions** group, in the **Minor Div/Major Div** box, type the number of minor time divisions that you want to be visible within each major time division, and then select the color you want to use for the division lines.

9. In the **Time Divisions** group, select **Top Labels** if you want time labels displayed at the top of the trend.

10. In the **Time Divisions** group, select **Bottom Labels** if you want time labels displayed at the bottom of the trend. Your trend can have both top and bottom labels, or no labels at all.

11. If you are using time labels, in the **Time Divisions** group, in the **Major Div/Time Label** box, type the number of time labels per major time division line that you want for the trend.
12. In the **Time Divisions** group, select the color you want to use for the major time division lines.
13. The settings in **Value Divisions** group are configured the same way as the settings in the **Time Divisions** group. The minor and major value divisions set the vertical value (y-axis) range for the trend. This range uses Engineering Units and is the same for all trended tagnames.

Tip To display decimal points for the minor and major value divisions at runtime, they must be formatted here to do so. For example, 0.00 to 100.00.



<Local>
InSQL1
InTouch1

14. In the **Tagname** box, double-click to open the **List of Providers** dialog box. Select the tagname provider you want to use for this **Pen**. Click OK. A browser will display showing the local or remote tagnames for that provider. Double-click a tagname to select it. Repeat for the tagname or expression that you want each **Pen** to trend.

    Up to eight pens can be visible in a trend. (Message type tags cannot be logged or trended.)

Note The FactorySuite Productivity Pack includes a Pen Configuration Grid that allows you to chart 16 pens. For more information, see your *Productivity Pack User's Guide.*

15. Click the color box to select the color that you want each pen to use to plot the tagname in the trend.
16. In the **Width** box, type the number of pixels wide you want each pen to be.

Note Selecting a pen width greater than 1 significantly reduces performance in screen updating and printing.

17. Select **Allow runtime changes** if you want the operator to be able to make changes to the trend's configuration in runtime. These changes include changing pen assignments, start date, time and so on.

    If you select this option, when the operator clicks on the trend (or touches it when using a touch screen) in runtime, the **Runtime Setup** dialog box will appear and he will be able to make changes to the trend.

    For more information, see "Updating a Historical Trend in Runtime."

18. Click **Select Display Font** to access the **Font** dialog box to select the font, style and size that you want to use for the trend display.

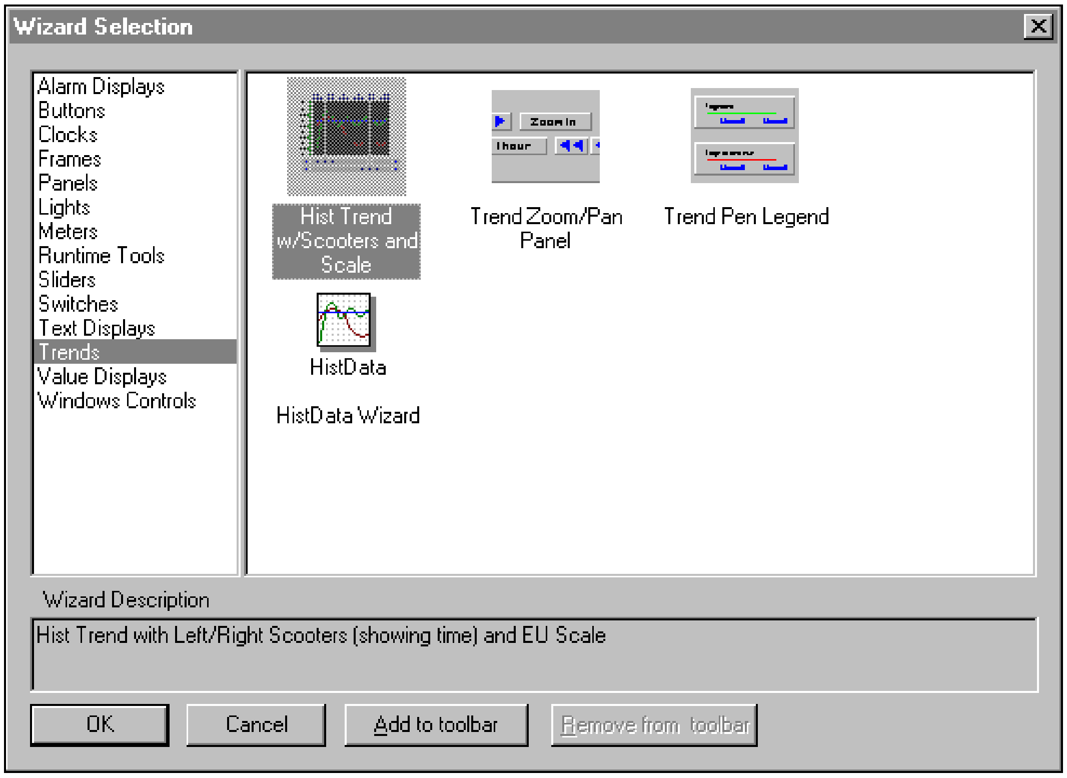
19. Click **OK**.

Using Historical Trend Wizards

InTouch provides you with a quick and easy method to create a historical trend: the trend wizard. The trend wizard allows you to configure a full-featured historical trend with scooters, zooming, and so on., with just a few mouse clicks.

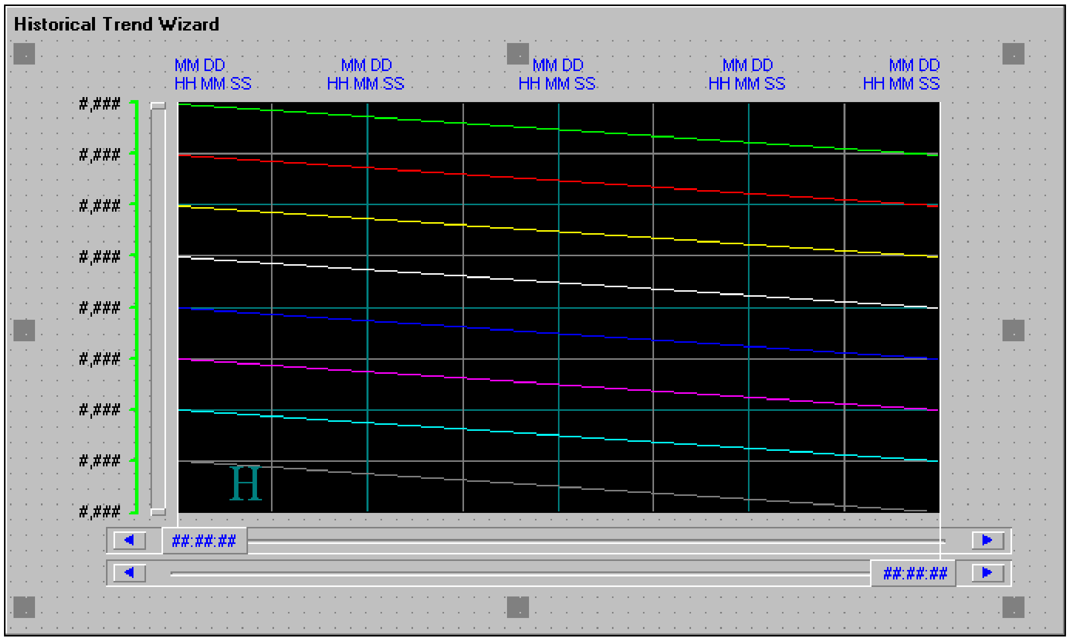
To use the historical trend wizards

1. Click the wizard tool in the **Wizard Toolbar**. The **Wizard Selection** dialog box appears.



Hist Trend with Left/Right Scooters (showing time) and EU Scale

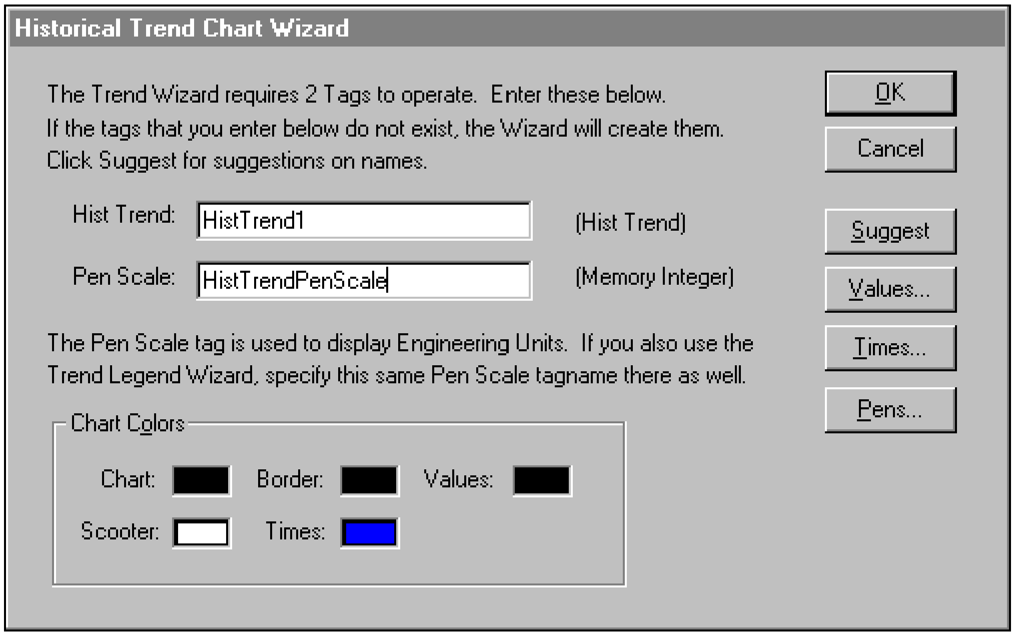
2. Select **Trends** in the list of wizards to display the available trend wizards.
3. Select the **Hist Trend with Scooters** wizard, then click **OK**. The dialog box closes and your window reappears with the cursor in the "paste" mode.
4. Click in the window to paste the trend wizard.



Tip A trend object is like any other object drawn in WindowMaker. It can be moved by grabbing it with the mouse or it can be resized by grabbing one of the object "handles." You can place multiple trends in a window.

5. You are now ready to configure the trend wizard.

6. Double-click on the trend wizard to open the **Historical Trend Chart Wizard** configuration dialog box:


Historical Trend Chart Wizard

The Trend Wizard requires 2 Tags to operate. Enter these below.
If the tags that you enter below do not exist, the Wizard will create them.
Click Suggest for suggestions on names.

Hist Trend: HistTrend1 (Hist Trend)

Pen Scale: HistTrendPenScale (Memory Integer)

The Pen Scale tag is used to display Engineering Units. If you also use the Trend Legend Wizard, specify this same Pen Scale tagname there as well.

Tip If you right-click a text box in the wizard configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

7. Enter the required information to configure the trend, then click **OK**.

   Click **Suggest** if you want the wizard to automatically fill in the configuration settings. The settings you configure for a historical trend wizard are the same as those you configure when you create the historical trend object drawn using WindowMaker's trend tool in the **Draw Object Toolbar**.

8. To add zoom and movement functions or pen controls to your trend, use the trend Zoom/Pan Panel and Trend Pen Legend wizards, respectively. For these components to all work together, they must use the **Hist Trend** tagname.

   Like all InTouch wizards, this wizard can be broken into its individual components.

To break the wizard

1. Select the historical trend wizard.
2. On the **Arrange** menu, click **Break Cell** or, click the break cell tool in the **Arrange Toolbar**.
3. You can then customize it to your needs.

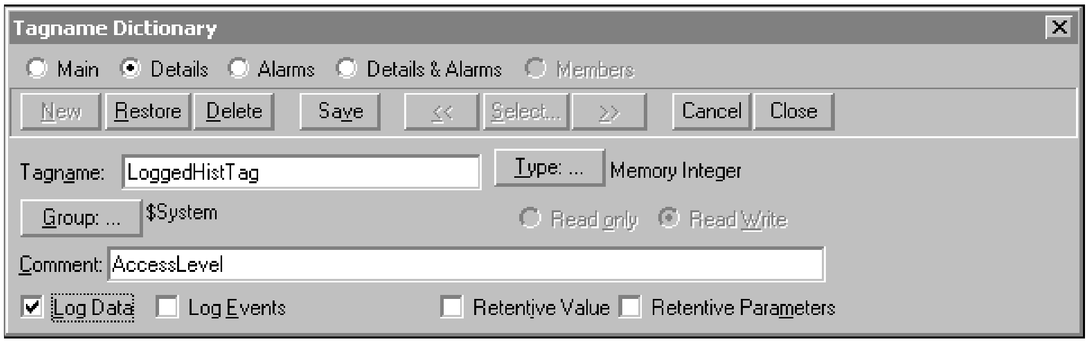
Logging Tagnames

In WindowViewer, the value of logged tagnames are written to the historical log file each time they change more than the specified **Log Deadband** and, by default, once an hour regardless of change. For a tagname's value to be written to the historical log file, it must be configured to be logged in the Tagname Dictionary.

For integer and real (floating point) tagname types, you can set the **Log Deadband** in their respective details dialog boxes. The **Log Deadband** controls how many Engineering Units a tagname's value must change before it is logged to disk.

To configure a tagname for logging

1. On the **Special** menu, click **Tagname Dictionary** or, in the Application Explorer, double-click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears.



Tagname: LoggedHistTag
Type: Memory Integer
Group: $System
Comment: AccessLevel
[✓ Log Data]

2. Open the desired tagname's definition, then select **Log Data**.

Note In order for your tagnames to actually be logged, you must enable logging as described in the next section.

If you change a tagname from logged to not logged, the data already logged for the tagname will not be accessible.

Any changes made in WindowMaker to logging while WindowViewer is running are ignored until WindowViewer is restarted.

The Min/Max Engineering Units are very important for displaying historical trend data. The historical trend displays from 0-100% of EU scale.

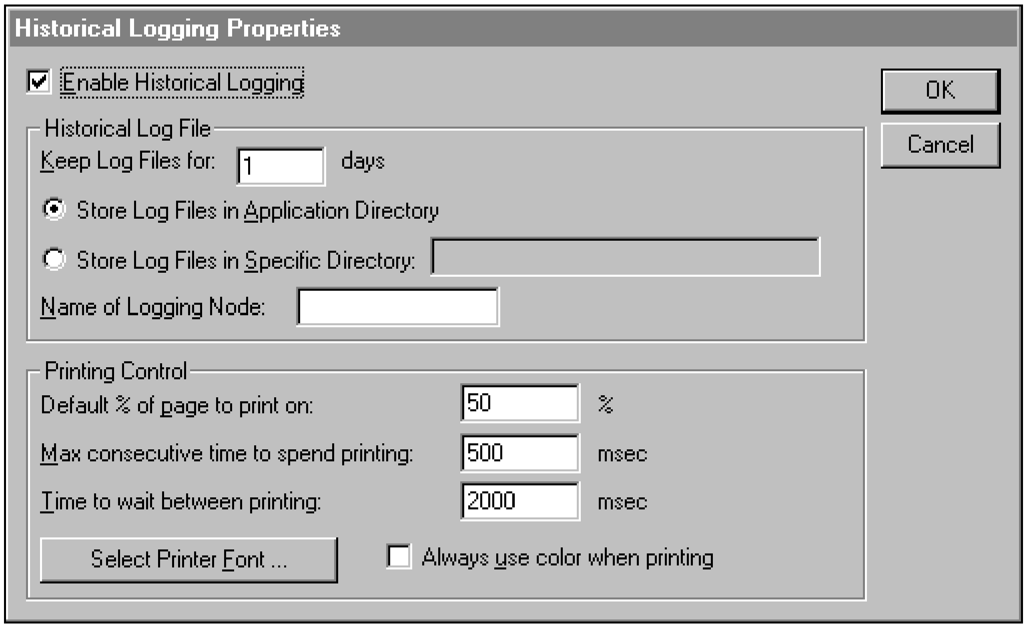
Configuring Historical Logging Properties

In order for the tagnames that you have configured with the **Log Data** option to be written to the historical log file, the global logging function must be selected.

To configure historical logging

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

2. To quickly access the dialog box, in the Application Explorer under **Configure**, double-click **Historical Logging**.



Historical Logging Properties

[✓] Enable Historical Logging

Historical Log File

Keep Log Files for: 1 days

(●) Store Log Files in Application Directory

( ) Store Log Files in Specific Directory:

Name of Logging Node:

Printing Control

Default % of page to print on: 50 %

Max consecutive time to spend printing: 500 msec

Time to wait between printing: 2000 msec

[ ] Always use color when printing

Tip If you right-click a text box in any historical trend configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

3. Select **Enable Historical Logging** to turn on global tagname logging.

4. In the **Keep Log Files for** box, type the number of days' (prior to today) log files that you want to keep saved to disk.

   InTouch will create and save two historical log files each day (24 hours). Therefore, disk space must be considered when you set this value. If your hard disk does not have enough free space to save a historical log file, logging will stop and you must free disk space then restart logging. You can start and stop historical logging in runtime by linking the **$HistoricalLogging** internal tagname to a button or QuickScript or by using the **Restart Historical Logging** Command in WindowViewer.

   For example, if you type 10 and today is the 12th day of the month, the log files for the 2nd through the 12th (10 days plus today) will be save on disk. The file for the 1st will automatically be deleted. If you type a zero, the log files are kept indefinitely.

5. Select **Store Log Files in Application Directory** if you want the historical log file to be saved in your application directory. Or, select **Store Log Files in specific Directory** and type the complete path to the directory that you want to use.

Note This entry must be either a DOS path such as c:\histlog or, if you are performing distributed history, it must be a Universal Naming Convention (UNC) path such as \\node\share\directory.

If configured to write historical data to the master application node's "Application Directory", all NAD nodes will attempt to write their historical data to the master application. To avoid this, on each NAD node, configure historical data to write to a local directory, **not** the master application node.

For more information, on distributed history, see "The Distributed History System."

By default, historical log files are named as follows:

YYMMDD00.LGH and YYMMDD00.IDX

where:

YY	Displays the year the file was created (99,01)
MM	Displays the month the file was created (01-12)
DD	Displays the day the file was created (01-31)
00	Always displays zeros

For example, if the files were created on October 31, 1997, they would be named as follows:

97103100.LGH

and,

97103100.IDX

Note This version of InTouch supports the newer log files with extensions of .LGH and .IDX. Earlier versions of InTouch used the extension .LOG for log files.

6. In the **Name of Logging Node** box, type the NetDDE node name (not the computer name) for the node that will be logging to the history log file.
7. In the **Default % of page to print on** box, type the percentage ratio for the page size to trend size.

   Example: If you use 50 for the percentage, when you print a historical trend, it will fill half of the page (vertically and horizontally). A printout this size would take roughly one quarter of the time to prepare as a full page printout.

   Tip There are many factors that affect the performance of printing historical trends. The primary performance factor is the size of the chart on the printed page. You can improve performance by reducing the percentage of the page that is used.

8. In the **Max consecutive time to spend printing** type the number of milliseconds (processor time slice) the historical trend print module will spend consecutively printing.

9. In the **Time to wait between printing** box, type the number of milliseconds the historical trend print module will wait between printouts.
10. Select **Always use colors when printing** if you are using a color printer or plotter.
11. Click **Select Printer Font**, to access the Windows **Font** dialog box.
12. Click **OK** to save your settings and close the dialog box.

Controlling Historical Logging Frequency

When logging is enabled via the **Historical Logging** command, the values of all tagnames designated to be logged will automatically be written to the Historical Log file once an hour if the value of the tagname does *not* change. If the tagname *does* change by an E.U. value greater than its LogDeadband (which defaults to 0), it will be logged automatically at the time of change. You can override the default by adding the parameter **ForceLogging=#** to the **[INTOUCH]** section of your **INTOUCH.INI** file (located in your application directory). The value for this parameter represents minutes and can be set between 5 and 120.

To log the *current* value of the tagname (even if the change is less than or equal to the log deadband value), add the parameter **ForceLogCurrentValue=1**. For example:

By adding the parameters **ForceLogging=15** and **ForceLogCurrentValue=1**, current tagname values will be written to the Historical Log file at least every 15 minutes or when the value of the tagname changes.

Note The **ForceLogCurrentValue** parameter forces the current value of the tagname to be logged even if the change is less than or equal to the Log Deadband.

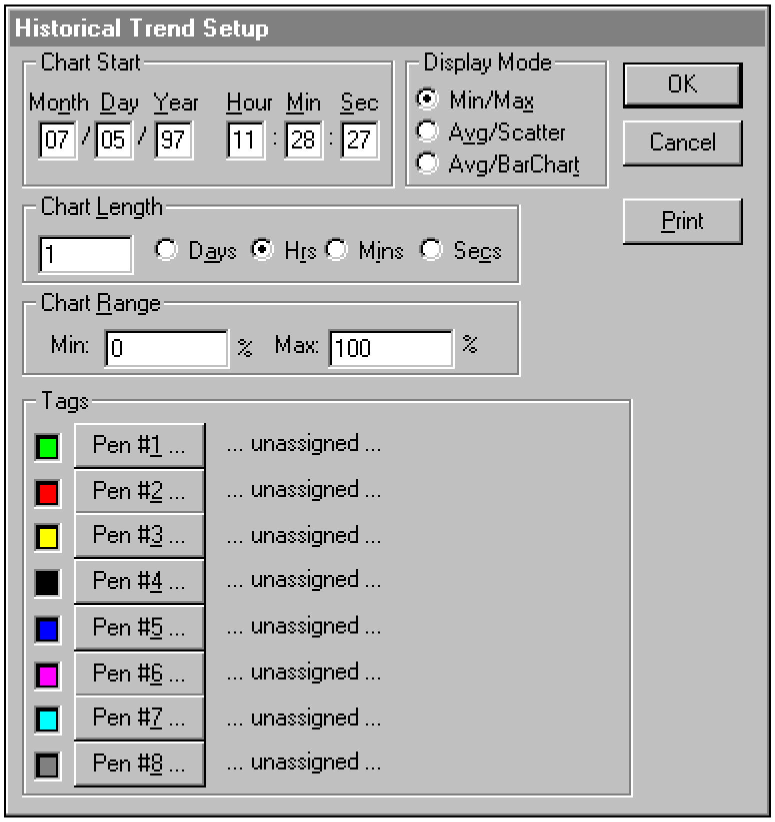
For more information on Log Deadbands, see "Logging Tagnames."

Configuring a Historical Trend in Runtime

If you selected the **Allow runtime changes** option when you configured your historical trend, the trend will be "touch-sensitive" in WindowViewer and the operator will be able to change the pen assignments, change the start date and time, and so on.

To configure a historical trend in runtime

1. Click the trend in WindowViewer, the **Historical Trend Setup** dialog box appears.



Historical Trend Setup

Chart Start
Month / Day / Year: 07 / 05 / 97
Hour : Min : Sec: 11 : 28 : 27

Display Mode: [✓ Min/Max] Avg/Scatter Avg/BarChart

Chart Length: 1 Days [✓ Hrs] Mins Secs

Chart Range
Min: 0 % Max: 100 %

Tags

🟩	Pen #1 ...	... unassigned ...
🟥	Pen #2 ...	... unassigned ...
🟨	Pen #3 ...	... unassigned ...
⬛	Pen #4 ...	... unassigned ...
🟦	Pen #5 ...	... unassigned ...
🟪	Pen #6 ...	... unassigned ...
▨	Pen #7 ...	... unassigned ...
▨	Pen #8 ...	... unassigned ...

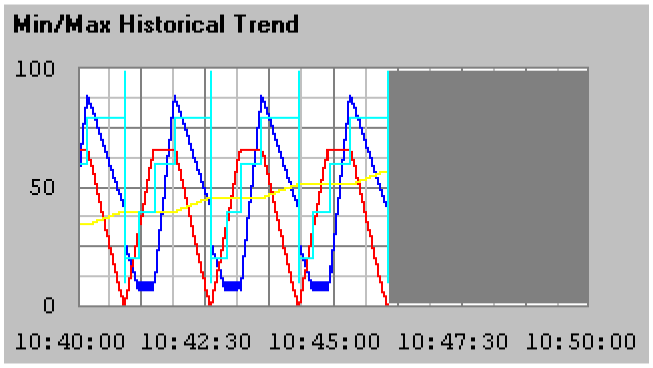
2. In the **Chart Start** group, type the starting date and time for the chart.
3. Select the **Display Mode** for your chart.

Note The display mode of the trend affects performance. The primary is the length of the lines being drawn to generate the trend. The longer the lines, the longer it takes to generate the trend. Line widths are also a performance factor; wide lines take significantly longer to draw. **Min/Max** or **Average/Scatter** trends are generally much faster to generate than **Average/Bar Chart**.

There are three modes as illustrated and described in the examples below:

Min/Max Historical Trend

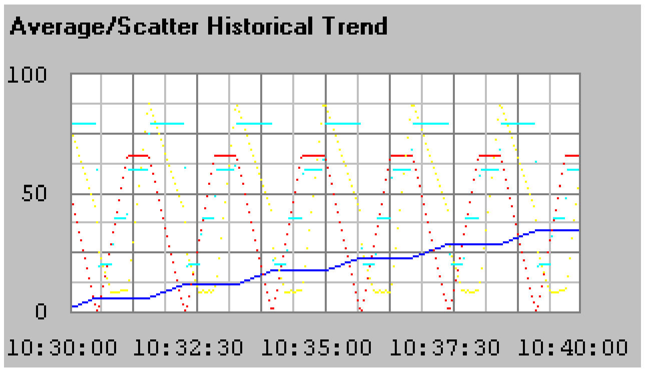
This mode displays the trends or changes in the percentage of Engineering Units scale as a vertical line over the time span with emphasis on time flow and rate-of-change, rather than amount of change.



Note The blank area on the right side indicates that no data was collected during that time period either because WindowViewer was not running or historical logging was turned off.

Average/Scatter Historical Trend

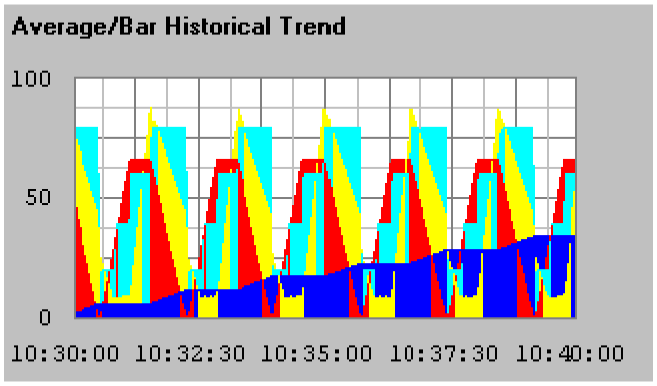
This mode shows the average value of the point during the time intervals.



For more information on the Average/Scatter Historical Trend, see "Average/Scatter Historical Trend Calculation."

Average/Bar Chart Historical Trend

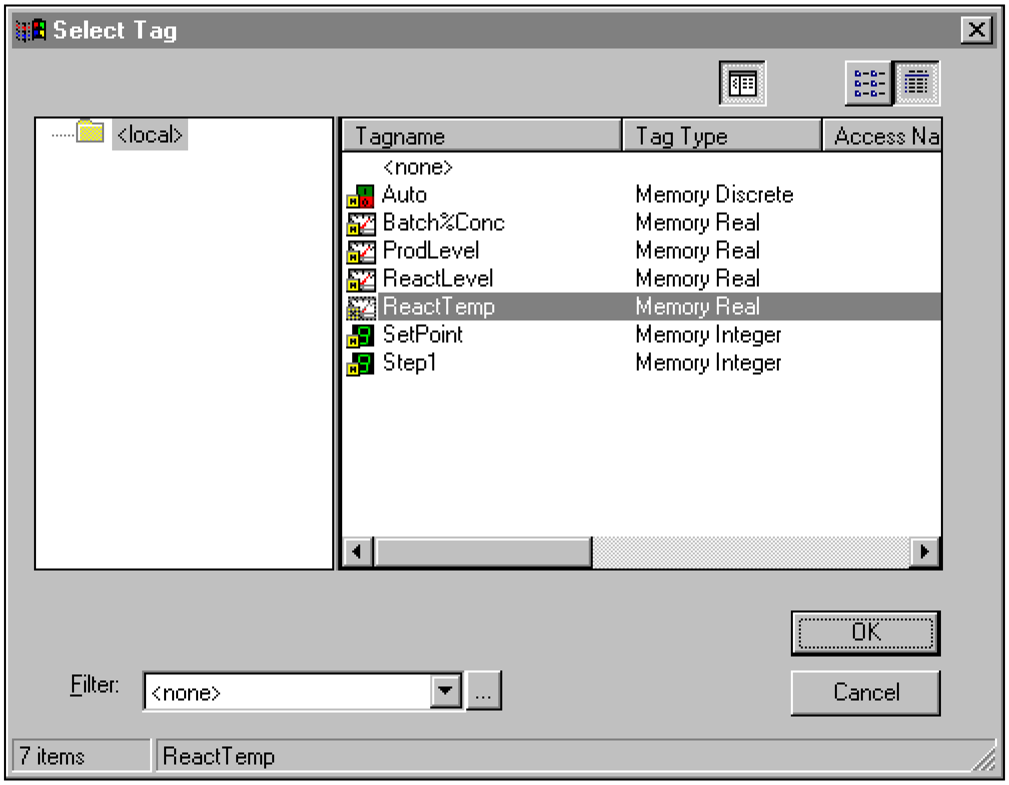
This mode shows the average value of the point during the time intervals in bar form.



4. In the **Chart Length** box, type the horizontal (x-axis) length of time to be displayed on the trend, and then select the time increment for the length. For example, if you type a 1 and select **Hrs**, your trend will be 1 hour long.
5. In the **Chart Range** boxes, type the percentage of Engineering Units scale that the trend is to zoom in/out (vertical (y-axis) range to be displayed on the trend).

Note The units for the range are a "percentage" of Engineering Units scale. These values should be from 0 to 100. For example, if you want to trend the variance of the selected tags from 40 to 45 percent of scale, enter 40 and 45 in the **Min** and **Max %** range boxes respectively.

6. Click each **Pen#** to select the tagname that you want the pen to trend. The Tag Browser appears in the filtered selection mode:



Tagname	Tag Type	Access Na
<none>		
Auto	Memory Discrete	
Batch%Conc	Memory Real	
ProdLevel	Memory Real	
ReactLevel	Memory Real	
ReactTemp	Memory Real	
SetPoint	Memory Integer	
Step1	Memory Integer	

7 items ReactTemp

Note Only the tagnames that are defined with the **Log Data** option selected will be displayed for the selected tag source.

7. Double-click the tagname that you want the selected pen to plot on the trend, or select the tagname, and then click **OK**. The **Historical Trend Setup** dialog box will reappear showing the selected tagname next to the **Pen#** button you originally clicked.

Tip You can click the **Filter** arrow to open the list of defined filters that you can use to populate the Tag Browser. The first entry of this list is **<none>**, which means that no filter is being used. Only the tagnames that are defined with the **Log Data** option selected will be displayed for the selected tag source.

When you use a filter or, click the **Filter** ... button and create a new filter, the Tag Browser will be repopulated with all tagnames defined with the **Log Data** option that meeting the criteria specified in the filter for the selected tag source.

For more information on the Tag Browser and filters, see Chapter 6, "Tagname Dictionary."

8. Click **Print** to print the historical trend.

The printing operation takes place "in the background" while WindowViewer continues to process all other inputs. WindowViewer will add two items to its menu during printing: **CancelPrint** and ***X* % Done**. Clicking on **CancelPrint** will cancel the current print job.

After selecting **Print**, do not change the trend until the **CancelPrint** and **X % Done** items disappear in the WindowViewer menu bar. During this time, WindowViewer is saving the trend information in memory for printing. Once these two items disappear in the menu bar, the trend can be changed without affecting the print that is in progress.

You can create a button to print the historical trend by linking it to an action QuickScript that executes **PrintHT** QuickScript function.

```
PrintHT(HistTrendTagname);
```

The printing operation uses the current historical trend as a basis for printing. Therefore, if any field in the **Historical Trend Setup** dialog box is changed, the **Print** button will not be active. Changes made in the setup cannot be printed until you click **OK** in the **Historical Trend Setup** dialog box, and then access it again and click **Print**.

Note Printing the Historical Trend using the **Print** option or the **PrintHT()** function will not print the y & x values. Use **PrintWindow()** or **PrintScreen()** to print the x & y values.

For more information on printing historical trends, see "Configuring Historical Trend Printing."

Average/Scatter Historical Trend Calculation

Average/Scatter is one of the three display modes for a Historical Trend object. When a tagname is defined with its **Log Data** option turned on, the value of the tagname is logged to the Historical Log file. The logged data is read from the Historical Log file and displayed on the Historical Trend object. In the case of Average/Scatter, the number of dots displayed on the object depends on the speed at which the tagname's value is changing. The Historical Trend object is divided into number of divisions. The number of divisions is restricted to 2000 therefore, the maximum number of points that can be displayed is 2000.

For example, consider an application where the tagname to be trended is logged every second with a time interval of 1 hour. The number of points in the display buffer is restricted to 2000, which in effect means that the time range of 3600 seconds would be split into intervals of 3600/2000 or 1.8 seconds each. Therefore, the value of the tagname plotted would actually be the average value over this time interval of 1.8 sec.

If there is change in the tagname's value during this 1.8-second interval, the average value is computed and displayed as a dot on the Historical Trend object. If the value stays constant then it is displayed as a line.

Consider another scenario where the tagname chart length is 1 second. Since the number of points in the display buffer is restricted to 2000, the time range of 1 second would be split into intervals of 1/2000 or 0.0005 seconds each (that is, 0.5 Msec which is rounded to 1 Msec). Therefore, the value of the tagname plotted would actually be the average value over this time interval of 1 Msec.

When a slider is associated with the chart length of the Historical Trend and it is moved fast, a lesser number of dots is displayed compared to the number of dots displayed when the slider is moved slowly.

Updating a Historical Trend in Runtime

In WindowViewer when a historical trend is first shown, it will display data for the specified configurations. Unlike real-time trends, historical trends do not update themselves continuously. A **change must be made to the trend** in order for it to update itself after the initial data display. Any of the following methods can be used to update the trend:

To update a Historical Trend in runtime

1. Select **Allow runtime changes** in the **Historical Trend Configuration** dialog box (in WindowMaker) so the operator can manually change the trend's time and/or date to force the update.
2. Use the following in a QuickScript or on a button to allow the operator to update the chart:
   `Hist_TrendTag.UpdateTrend = 1`
3. Use any of the following in a QuickScript or on a button:

   ```
   HTUpdateToCurrentTime(Hist_Tag);
   HTScrollLeft(Hist_Tag,Percent);
   HTScrollRight(Hist_Tag,Percent);
   HTZoomIn(Hist_Tag,LockString);
   HTZoomOut(Hist_Tag,LockString);
   HTSetPenName(Hist_Tag,PenNum,Tagname);
   ```
4. Change any of the following trend tagname dotfields:

   .**ChartStart**

   .ChartLength

   .MaxRange

   .MinRange

   .Pen1-.Pen8

For more information on using QuickScript functions and .fields, see your online *InTouch Reference Guide.*

Configuring Historical Trend Printing

There are many factors that affect the performance of printing historical trends. The primary performance factor is the size of the trend on the printed page. The display mode of the trend also affects printing performance. **Min/Max** or **Average/Scatter** printouts are usually generated much faster than **Average/Bar Chart** trends. The longer and wider the lines on the trend are, the longer it takes to print.

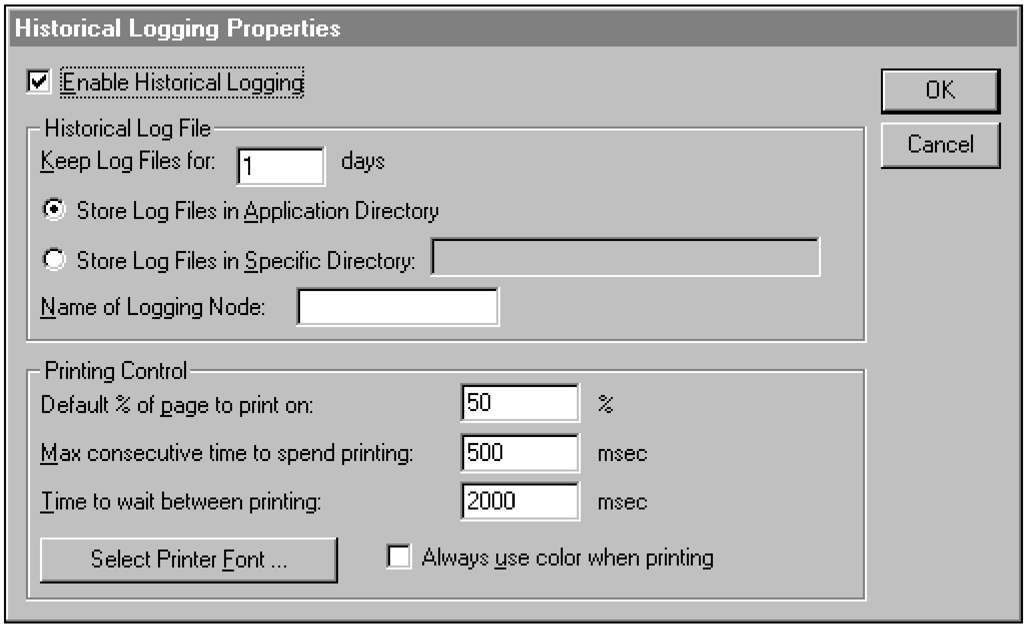
Since the printing operation takes place "in the background", InTouch devotes a certain amount of time to print processing and a certain amount of time to other processing. The times in this equation are controlled by the values set in the **Max consecutive time to spend printing** and **Time to wait between printing** boxes when you configure historical logging.

In other words, InTouch spends the number of milliseconds that you specified in the **Max consecutive time to spend printing** box processing printing and then spends the number of milliseconds that you specified in the **Time to wait between printing** field processing other requests. To raise the priority of printing, increase the value for **Max consecutive time to spend printing** and decrease the value for **Time to wait between printing**. To lower the priority of printing, do the opposite.

To configure historical trend printing

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

Tip To quickly access the dialog box, in the Application Explorer under **Configure**, double-click **Historical Logging**.


Historical Logging Properties

[✓] Enable Historical Logging

Historical Log File

Keep Log Files for: 1 days

(●) Store Log Files in Application Directory

( ) Store Log Files in Specific Directory:

Name of Logging Node:

Printing Control

Default % of page to print on: 50 %

Max consecutive time to spend printing: 500 msec

Time to wait between printing: 2000 msec

[ ] Always use color when printing

Tip If you right-click a text box in any historical trend configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

2. Specify the percentage of the page to be used for printing the trend in the **Default % of page to print on** box.

   If you type 50 in this box, WindowViewer will use half of the page (vertically and horizontally). A printout this size takes one quarter of the time to prepare versus a full-page printout.

   As a printing alternative, you may want to investigate using the **PrintWindow** QuickScript function.

3. In the **Max consecutive time to spend printing** box, type the process time slice (milliseconds) the print module will consecutively spend printing.

4. In the **Time to wait between printing** box, type the time (milliseconds) the print module will wait before taking another processor time slice.

   Another factor that affects printing performance is the background color that you select for the trend. In most cases, a white background will print much faster. The best test is to experiment with white versus a color and see if there is a significant difference.

5. Click **Select Printer Font** to access the **Font** dialog box to select the font, style and size that you want to be used for the printout.

6. Click **OK**.

Historical Trend Dotfields

For a given historical trend tagname there are many **.field**s that only apply to historical trend tagnames. Each historical trend **.field** is briefly described below.

Field	Description
.ChartLength	Read/write integer tagname dotfield used to control the length of time displayed in a Historical trend graph. **.ChartLength** displays the length of the chart in seconds.
.ChartStart	Read/write integer tagname dotfield used to control the starting time and/or to scroll the corresponding historical trend. **.ChartStart** displays the number of elapsed seconds since 12:00 a.m., 1/1/70.
.DisplayMode	Read/write analog tagname dotfield used to determine the method to be used in displaying values on the trend.

Field	Description
.MaxRange, .MinRange	Read/write real tagname dotfields used to represent the percentage of the tagname's Engineering Unit range that should be displayed for each tagname being trended. The limits for **.MaxRange** and **.MinRange** are from 0 to 100 and **.MinRange** should always be less than **.MaxRange**. If a value less than 0 or greater than 100 is assigned to either of these fields, the value will be clamped at 0 or 100. If **.MinRange** is greater than or equal to **.MaxRange**, the trend will not display any data.
.Pen1 - .Pen8	Read/write TagID type tagname dotfields used to control the tagname being historically trended by each pen. A TagID type tagname can only be equated to another TagID tagname. It cannot be mixed with any other tagname type unless the **.TagID** extension is added to the other tagname. **.TagID** cannot be used for remote history provider tagnames.
.ScooterLockLeft	Read/write discrete field. When the value of this field is TRUE, the RIGHT scooter cannot move to the left of the left scooter's position. (0=FALSE, 1=TRUE).
.ScooterLockRight	Read/write discrete field. When the value of this field is TRUE, the LEFT scooter cannot move to the right of the right scooter's position. (0=FALSE, 1=TRUE).
.ScooterPosLeft	Read/write real field which represents the position of the left scooter (range 0.0 to 1.0).
.ScooterPosRight	Read/write real field which represents the position of the right scooter (range 0.0 to 1.0).
.TagID	Read/write TagID tagname dotfield used in conjunction with the Historical Trend .Pen1 - .Pen8 TagID tagnames to monitor and/or control the tagname being trended by a pen.
.UpdateCount	Read-only integer field that is incremented when a retrieval is complete for the trend
.UpdateInProgress	Read-only discrete field that shows historical data retrieval status (0=no retrieval in progress, 1=retrieval in progress).
.UpdateTrend	Read/write discrete tagname dotfield that can be set to 1 to cause a Historical trend to update using all current values.

For more information on using **dotfields**, see your online *InTouch Reference Guide.*

Historical QuickScript Functions

There are several internal functions that you can use to specify the tagname to be trended by each pen, display the value at a scooter location, scroll the trend by a percentage, etc.

For complete examples of how to use these functions and their valid arguments, see your *InTouch Reference Guide*.

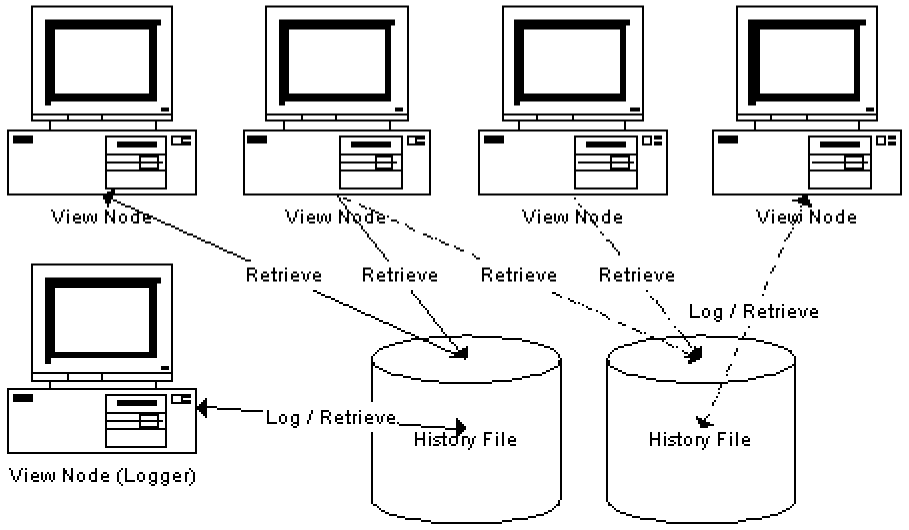
Function	Description
HTGetLastError	Determines if there was an error during the last retrieval of a specified pen.
HTGetPenName	Returns the tagname of the tagname currently used for the specified pen # of the specified trend.
HTGetTimeAtScooter	Returns the time in seconds since 00:00:00 hours GMT, January 1, 1970 for the sample at the scooter location specified by *ScootNum* and *ScootLoc*. *UpdateCount*, *ScootNum*, and *ScootLoc* cause the expression to be evaluated when any of these parameters change. This ensures that the expression is evaluated after new retrievals or after a scooter moves.
HTGetTimeStringAtScooter	Returns the string containing the time/date for the sample at the scooter location specified by *ScootNum* and *ScootLoc*. *UpdateCount*, *ScootNum*, and *ScootLoc* cause the expression to be evaluated when any of these parameters change. This ensures that values are updated after new retrievals or after a scooter moves. The format of the string determines the contents of the return value.
HTGetValue	Returns a value of the requested type for the entire trend's specified pen.
HTGetValueAtScooter	Returns a value of the requested type for the sample at the specified scooter position, trend and pen #. The *UpdateCount* parameter will cause the expression to be evaluated after a retrieval is complete.
HTGetValueAtZone	Returns a value of the requested time for the data contained between the right and left scooter positions for a trend's specified pen.
HTScrollLeft	Sets the start time of the trend to a value older than the current start time by a percentage of the trend's width. The effect is to scroll the date/time of the chart to the left by a given percent.

Function	Description
HTScrollRight	Sets the start time of the trend to a value newer than the current start time by a percentage of the trend's width. The effect is to scroll the date/time of chart to the right by a given percent.
HTSetPenName	Assigns a different tagname to a trend's pen.
HTUpdateToCurrentTime	Causes the data to be retrieved and displayed with an end time equal to the current time. The start time will be equal to EndTime minus the Width of the chart.
HTZoomIn	Calculates a new chart width and start time. If the trend's .ScooterPosLeft is 0.0 and the .ScooterPosRight is 1.0, then the new chart width equals the old chart width divided by two. The new start time will be calculated based on the value of *LockString*.
HTZoomOut	Calculates a new chart width and start time. The new chart width is the old chart width multiplied by two. The new start time will be calculated based on the value of *LockString*.

The Distributed History System

InTouch provides a distributed history system that allows retrieval of historical data from any InTouch application, even those across a network. This system extends the capabilities of the standard InTouch history by allowing remote retrieval of data from multiple historical databases simultaneously. These databases are referred to as history providers. Up to eight history providers can be displayed simultaneously, one for each historical trend chart pen.

Using the capabilities of the distributed history system, you can easily configure a networked system that provides access to multiple history providers:



Each distributed history file is limited to one node writing (logging) to the file. However, there is no restriction on the number or type of InTouch nodes that can view that file.

Only applications developed in InTouch Version 5.6 or later can be history providers. To remotely view history files from an earlier version, you must first convert that application to Version 5.6 or later.

A remote node retrieving data from a history file may not see data for the last hour of data (based on the logger node's time). Remote trends can only view data that has been written to the logging node's disk.

Data for each tagname checked for 'Log Data' is automatically written to disk after 22 samples for that tagname have been collected. If **HTUpdateToCurrentTime()** function is executed, data is written to disk regardless of the number of samples collected. By default, data is written to disk once an hour. You can change this interval by adding the following line to the intouch.ini file:

```
ForceLogging=X;
```

Where **X** is minutes and can be set to any interval between 5 and 120.

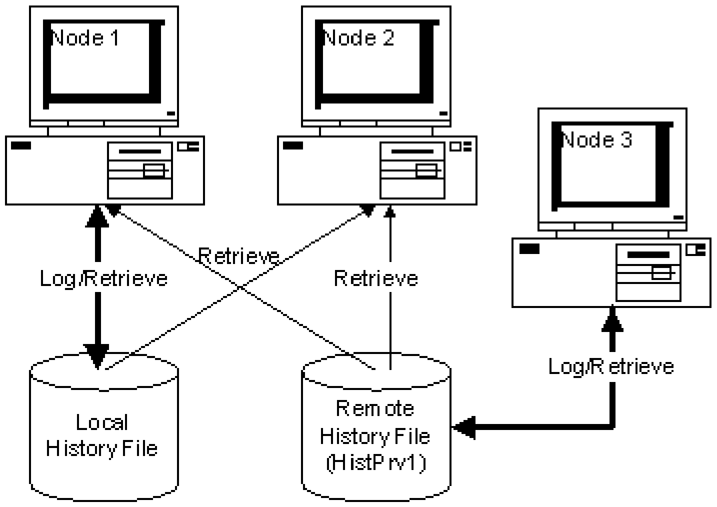
Note The Wonderware NetDDE Helper service must be running when you use Distributed History.

For more information on Windows NT services, see Appendix A, "Overview of the InTouch Windows NT Services."

Using the Distributed History System

The following diagram illustrates how you should setup your distributed history system. This system is a typical distributed application using Network Application Development (NAD) to distribute the application.

For more information on NAD, see Chapter 5, "Building a Distributed Application."



Nodes 1 and 2 contain copies of the same InTouch application; however, the application is configured to allow only Node 1 to log **to** a local history file, whereas either node can retrieve **from** the local history file or the remote history file. Node 3 is also logging to and retrieving from the remote history file location. This provider (Node 3) is assigned the name **HistPrv1**. Node 1 is both a development and runtime station, while Node 2 is just a runtime station.

The major steps you need to perform to create this application include:

1. Create a history provider list.
2. Create and configure a historical trend object.
3. Configure the application for distributed logging.
4. Distribute the application.

All of these steps are described in this chapter.

Distributing Your Application

You can distribute your application manually or by using the NAD system. When you distribute your application, the historical provider list file is distributed as part of the application.

For more information on using NAD, see Chapter 5, "Building a Distributed Application."

After you have distributed your application, you can run the View nodes and retrieve both local tagnames and tagnames from a remote history provider. While the application will run on all the View nodes, only the logging node will log to the historical log file; other nodes will only be able to read from it.

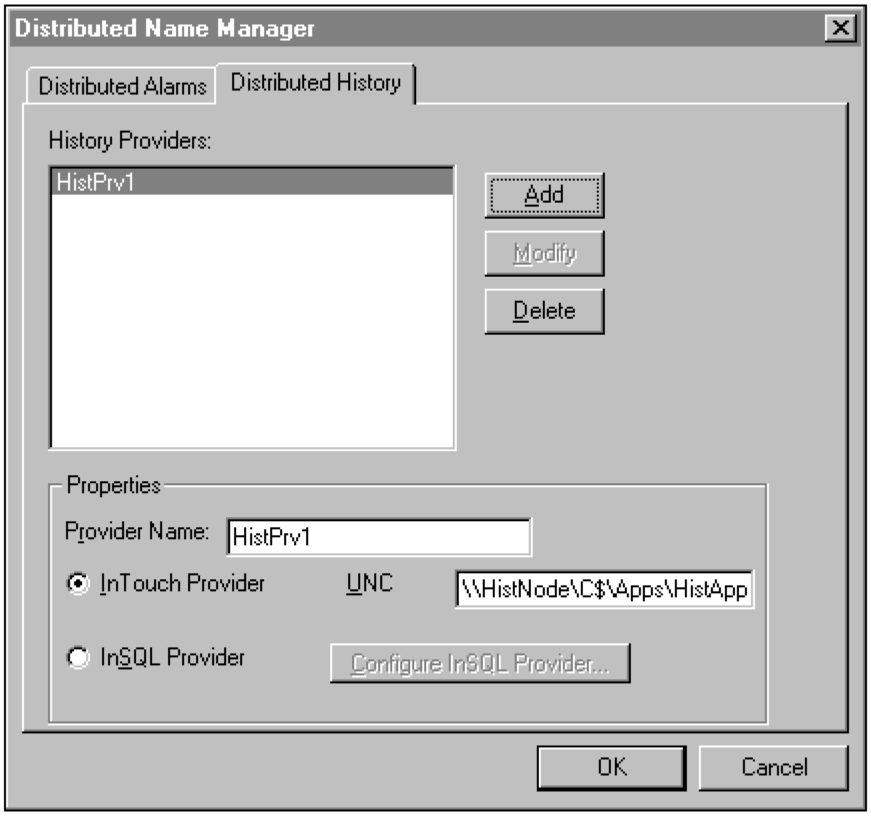
Configuring the Distributed History Provider List

Each remote history provider you intend to retrieve historical data from must be registered in the InTouch history provider list. This list allows you to specify a name and network location for each history provider. These names will be used whenever you refer to a history provider in InTouch.

To configure the historical provider list

1. On the **Special** menu, point to **Configure**, and then click **Name Manager**. The **Distributed Name Manager** dialog box appears.

Tip To quickly access the dialog box, in the Application Explorer, under **Configure**, double-click **Distributed Name Manager**.



History Providers: HistPrv1

Properties

Provider Name: HistPrv1

(●) InTouch Provider UNC \\HistNode\C$\Apps\HistApp

( ) InSQL Provider

Tip If you right-click a text box in any historical trend configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

2. Click the **Distributed History** tab to activate the distributed history providers property sheet.

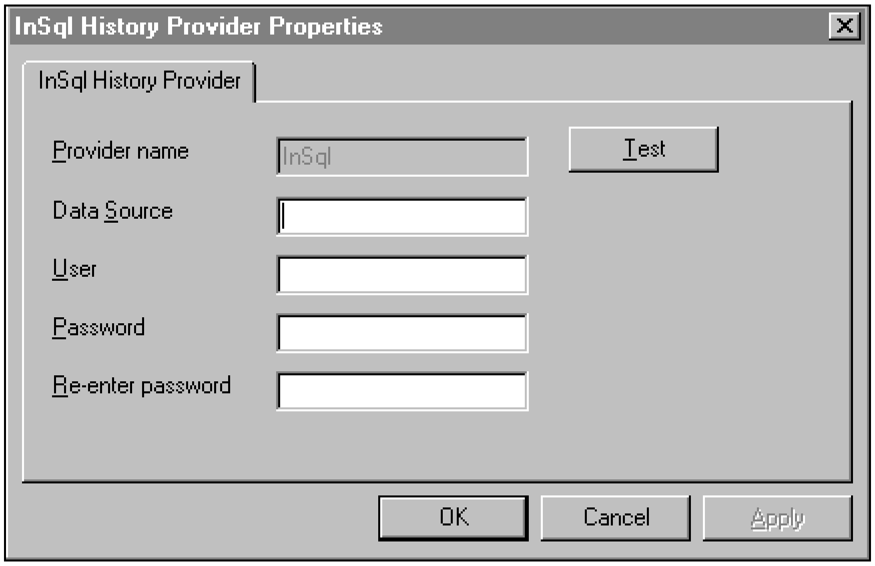
3. In the **Provider Name** box, type the name you want to use for the new historical provider.
4. If you want to access an InTouch application's log file, select **InTouch Provider** and type the UNC (Universal Naming Convention) path for the InTouch application directory in the **UNC** box. The valid format is:

   ```
   \\Node\Share\ApplicationPath
   ```

   In the above example, the path to **HistPrv1** includes the node name "HistNode," the Share "C$," and the application path of "\Apps\HistApp."

   If the UNC location is password-protected, you must first establish a connection using the Windows Explorer.

   For more information of UNC paths, see Chapter 5, "Building a Distributed Application.".
5. Select **InSQL Provider** to access data in an IndustrialSQL Server runtime database, and then click **Configure InSQL Provider**. The **InSql History Provider Properties** dialog box appears.



Provider name InSql
Data Source
User
Password
Re-enter password

   By default, the logon parameters from the last successful logon will be displayed. If necessary, modify the logon parameters to connect to the selected IndustrialSQL Server.
6. In the **Provider name** box, type the name (user defined) you want to use for the InSQL provider.
7. In the **Data Source** box, type the name (35 characters or fewer) of the node where the IndustrialSQL Server database resides.
8. In the **User** box, type the name for your log on account.
9. In the **Password** box, type the password for the log on account.
10. In the **Re-enter password** box, type the password again to verify it.
11. Click **Test** to validate the connection to the InSQL Server. A message box will appear informing you of the success or failure of the connection. Click **OK** to close the message box.

12. Click **OK**.

Note A user account is comprised of the user name and password. A user account must be associated with the right to retrieve data, or else the log on will fail. For more information on user accounts, contact your system administrator.

When data is queried in the InSQL database for the InTouch trend object, 1000 evenly-spaced rows are retrieved for the given time period (a row count) and plotted on the historical trend. The minimum and maximum values shown for a tagname in the historical trend may not be the actual minimum and maximum values for the tagname.

HistData **cannot** retrieve historical information from an InSQL Provider.

For more information on history providers, see "Configuring the Distributed History Provider List."

13. Click **Add**.

When WindowViewer is presented with a history provider name, the history system will look up that name in the provider list. If the name exists in the list, it reads the history log file from that provider. If the name does not exist, the reference is ignored and an error message is written to the Logger. While the local InTouch application is considered a history provider, it does not need to be configured in this file.

Configuring Remote History Providers

The historical trend supports the display of tagnames from both local and remote history providers.

To display a tagname from a remote history provider

1. Double-click the historical trend to access the **Historical Trend Configuration** dialog box.
2. In each pen's **Tagname** box, type the tagname in the format:

```
HistPrv1.tagname
```

Each pen can reference a different remote history provider. For example, when you configure the historical trend, if you want Pen1 to plot the tagname Boiler1 in the remote history provider defined as HistPrv1, you would type HistPrv1.Boiler1 in Pen1's Tagname box.

Note The **.TagID** tagname dotfield cannot be used in remote history provider tagnames references.

Using the Tag Browser to Access Remote History Providers

The following procedure demonstrates how the application developer can use the Tag Browser to select a remote tagname reference.

To define a remote history provider as a tag source

1. Create an Access Name that specifies the **Node Name** where the history provider resides.

   The **Node Name** you specify in the Access Name does not have to be the actual name of the node where the tagname resides. But, you must create an Access Name or you won't be able to define the remote history provider as a tag source.
2. Double-click the historical trend. The **Historical Trend Configuration** dialog box will appear.
3. Double-click a pen's **Tagname** input box. The Tag Browser will appear.
4. Click the Define Tag Sources button to define the remote history provider as a tag source.
5. Click the **Tag Source** arrow and select the new remote history provider tag source in the list, or click the Tree View button and select the tag source in the tree view pane. The Tag Browser will then be repopulated with the selected remote history provider's tagnames.
6. Double-click the tagname that you want to assign to the historical pen, or, select it, and then click **OK**.
7. The **Historical Trend Configuration** dialog box reappears with the selected tagname displayed in the pen's **Tagname** box in the format: **AccessName:Item**.
8. Replace the **AccessName:** portion with the history provider name that you defined in the Distributed Name Manager. For example, **HistPrv1.Tagname**.

   This process may seem cumbersome, but once you have defined the history provider as a tag source in the Tag Browser, each time you double-click another tagname input box, you simply double-click the tagname in the Tag Browser, and then replace the **AccessName:** portion with the history provider name. By using this process you will reduce your chance of error when you specify a remote history provider tagname.

Note In WindowViewer, if runtime changes are allowed for the historical trend, when the user clicks a pen button to change the tagname, the Tag Browser appears but, only the local application's tagnames will be accessible.

For more information on using the Tag Browser, see Chapter 6, "Tagname Dictionary."

Dynamically Configuring Remote History Providers

In runtime, you can also dynamically configure a historical trend's remote history provider by creating a QuickScript that specifies the remote history provider tagname reference in the **HTSetPenName** function. For example:

```
HTSetPenName("HistTrendTag", 1, "HistPrv1.Boiler1");
```

Where, 1 specifies the pen that will plot the specified remote history provider tagname.

Note The runtime Historical Trend Setup dialog box and .Pen are not supported for remote history providers.

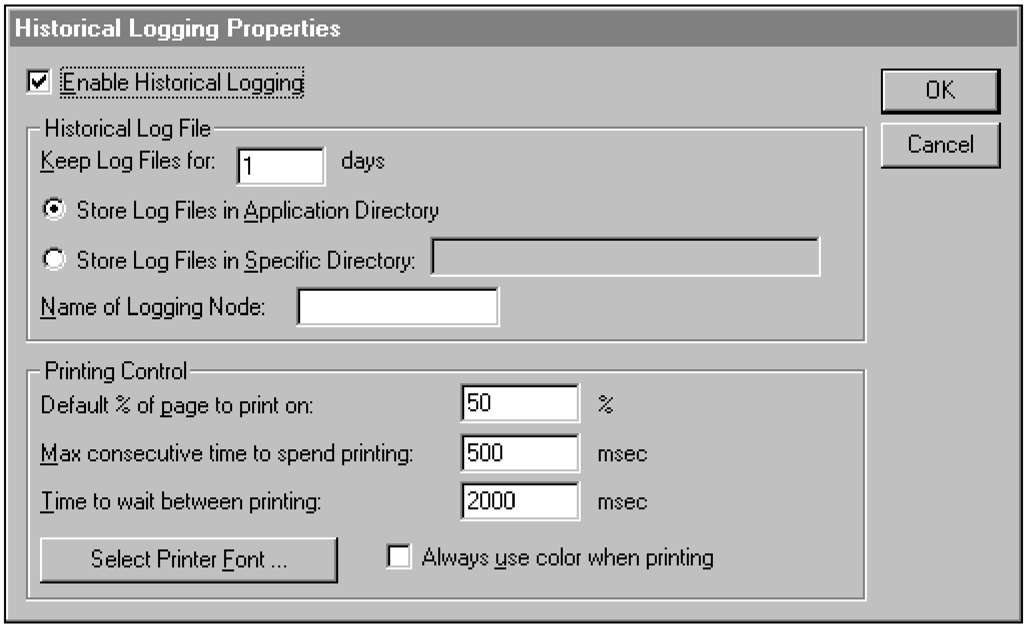
For more information on the InTouch QuickScript functions, see your online *InTouch Reference Manual*.

Configuring Distributed Historical Logging

To configure distributed historical logging

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

   Tip To quickly access the dialog box, in the Application Explorer under **Configure**, double-click **Historical Logging**.


Historical Logging Properties

[✓] Enable Historical Logging

Historical Log File

Keep Log Files for: 1 days

(●) Store Log Files in Application Directory

( ) Store Log Files in Specific Directory:

Name of Logging Node:

Printing Control

Default % of page to print on: 50 %

Max consecutive time to spend printing: 500 msec

Time to wait between printing: 2000 msec

[ ] Always use color when printing

Tip If you right-click a text box in any historical trend configuration dialog box, a menu will appear displaying the commands that you can apply to the selected text.

2. Select **Enable Historical Logging** to turn on global tagname logging.
3. Select **Store Log Files in Specific Directory**, and then in the input box, type the path of the location where the log files will be stored. You must type a valid Universal Naming Convention (UNC) path. For example, "\\Node\Share\Path."

   If NAD is being used, make sure that the path points to a directory other than the application directory.

4. In the **Name of Logging Node** box, type the name of the node that will be logging to the history log file.

   This setting will only allow the node named here to log to the file.

5. Click **OK**.

Note When an application with the **Enable Historical Logging** option selected is distributed to a WindowViewer node, that node checks this option to determine if it should log or not. If **Enable Historical Logging** is selected, the possible settings are:
Field equals name of Node - Logging enabled
Field does not equal name of Node - Logging disabled

Creating Historical Trend Scooters

Scooters are positional indicators along a time scale that can be changed in order to retrieve specific pieces of data for precise points of time. By tying slider objects to a scooter **.field**, you can slide over a historical trend display and access a desired section of data. QuickScript functions are provided to access the average, minimum, and maximum values at a specified scooter position. A left and right scooter can be created and using additional InTouch QuickScript functions, can return values based on an analysis performed on the data between the scooters or at the scooter location. Analysis types include Average, Min, Max, Min/Max Value, Min/Max EU and Standard Deviation. Zooming in can also be performed between the two scooters.

For more information on the scooter .fields, see "Historical Trend Dotfields."

You can also add the ability to display data based on a known location on the chart. Zooming in and out of a trend is also beneficial. The following pages describe the links and expressions that can be used to incorporate this functionality into historical trends.

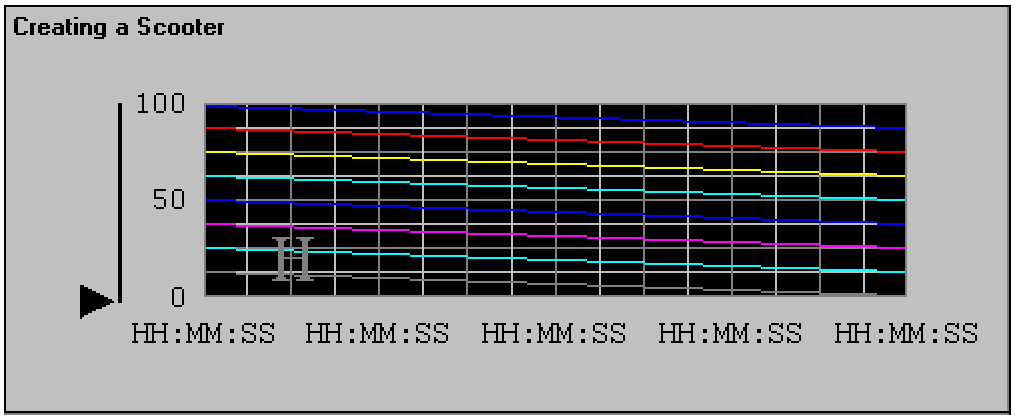
Tip These features are already configured in the Historical Trend Wizards.

For more information, see "Using Historical Trend Wizards."

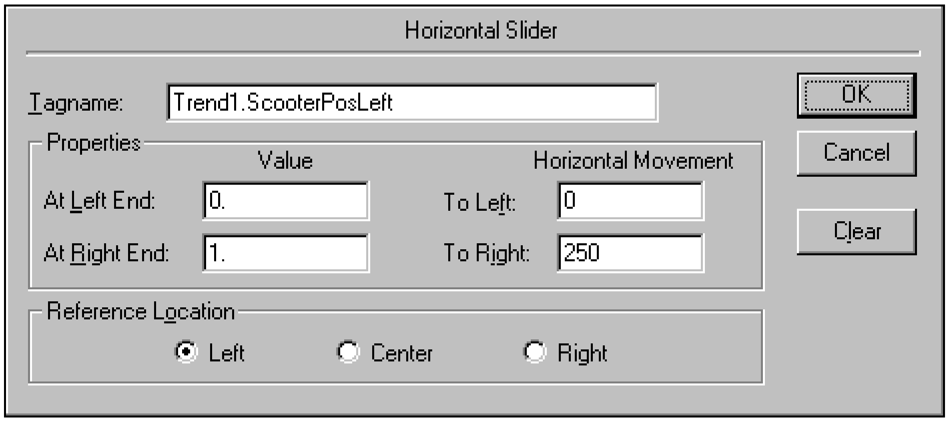
You can use the **Horizontal Slider** link to create a scooter that moves over the top of your historical trend. (Each trend supports two scooters, left and right.)

To create a scooter

1. Create the object to be used as your scooter. In the example below, we will use a polygon and vertical line symbol:



2. To properly define the scooter's horizontal slider link, you will need to know how wide your chart is. To determine the chart's width, draw a horizontal line from one end of it to the other and note the values displayed in WindowMaker's status bar. Look at the third value in from the left -- this is the width of your chart. Write this number down. Delete the "measuring" line.
3. Double-click on the scooter object. The animation link selection dialog box will appear.
4. In the **Touch Links - Sliders** section, click **Horizontal**. The **Horizontal Slider** dialog box appears.



Horizontal Slider

Tagname: Trend1.ScooterPosLeft

Properties

	Value		Horizontal Movement
At Left End:	0.	To Left:	0
At Right End:	1.	To Right:	250

Reference Location: [✓ Left] Center Right

5. In the **Tagname** box, type the name of the historical trend plus the **.field**, **.ScooterPosLeft**.

   For example, if the trend has the tagname **Trend1**, then the tagname is **Trend1.ScooterPosLeft**. The **At Left End** value is 0.0 and the **At Right End** value is 1.0. The **Horizontal Movement To Left** is 0 and **To Right** is the pixel value you found when the horizontal line was drawn from one end of the trend to the other. (In the example above, the chart was 250 pixels wide.)

6. Click **OK**.

Position the left scooter at the very left edge of your trend. The left scooter is now complete. Duplicate the process for the right scooter using **.ScooterPosRight**. The value fields remain the same.

Displaying Values at Scooter Positions

To display values based on the current location of the scooter, create a numeric text object, for example, #.00 and assign it to a **Value Display - Analog** link with an expression. For example:

```
HTGetValueAtScooter( "Trend1", Trend1.UpdateCount, 1,
   Trend1.ScooterPosLeft, 1, "PenValue" )
```

This example will retrieve the value for Pen1 at scooter position left on Trend1. If the slider is moved in Runtime, the **Value Display - Analog** link will automatically be updated with the new value at the new scooter location.

Retrieving Values Between Zones

To receive data between the scooters current location in the form of either, max value, min value, average value or standard deviation, create a numeric text object, for example, #.00, and assign an **Value Display - Analog** link with an expression. For example:

```
HTGetValueAtZone( "Trend1", Trend1.UpdateCount,
   Trend1.ScooterPosLeft, Trend1.ScooterPosRight, 1,
   "PenMaxValue" )
```

This example will retrieve the Max value for Pen1 on Trend1 between scooter position left and scooter position right.

Zooming In and Out

To zoom in on the trend, create a button and assign it to a **Touch Pushbutton - Action** link. Choose the **On Button Down** condition and enter the following QuickScript:

```
HTZoomIn( "Trend1", "Center" );
```

This example will "anchor" the center time of the trend and the new chart width will equal the old chart width divided by two if the scooters are at the far left and right positions. If the scooters are moved, the new chart width is the time between scooter left and scooter right and the ***LockString*** ("Center") is not used.

For more information using historical QuickScript functions, see "Historical QuickScript Functions."

Historical Trending and Daylight Savings Time

InTouch's Historical Logging creates a new Historical Log file each day at midnight based on the computer clock. When the value of a tagname checked for historical logging changes more than the Log Deadband, the value is written to the Historical Log File. Each record in the Historical Log file contains the updated value of the tagname as well as the time and date the value was updated. The InTouch historical engine uses Universal Coordinated Time (UTC) which, is also known as Greenwich Mean Time (GMT), to reference when recording and retrieving data. The following examples show how the Time Zone and Daylight Savings Time are taken into account by the Historical engine:

Let's assume we have a tagname (**tag1**) that changes its value at 10:00:00 am (computer time). West of GMT:

Logged_TimeStamp=Computer_Time+(Offset -DST)

Where **Logged_TimeStamp** is the Time Stamp of the record entered in the log file.

Example 1

The Time Zone is set for Pacific Standard Time (GMT - 8) and Daylight Savings Time is enabled:

Logged_TimeStamp=10:00:00 +(8-1)=17:00:00

Example 2

The Time Zone is set for Pacific Standard Time (GMT -8) and Daylight Savings Time is not enabled.

Logged_TimeStamp=10:00:00 +(8-0)=18:00:00

Example 3

The Time Zone is set for Amsterdam, Berlin, Bern, Rome (GMT +1) and Daylight Savings Time is enabled.

Logged_TimeStamp=10:00:00 +(-1+1)=10:00:00

Example 4

The Time Zone is set for Monrovia, Casablanca (GMT0) and Daylight Savings Time is not observed:

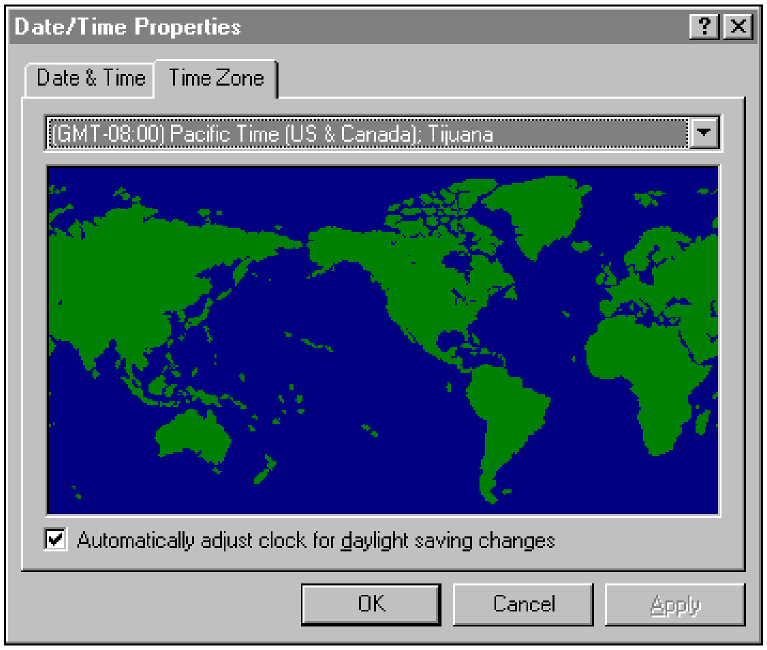
Logged_TimeStamp=10:00:00 +(-0 +0)=10:00:00

In example 4, the **TimeStamp** in the log file is the same as the computer time (which is the ideal setting).

Historical Trending also takes into account the Time Zone and Daylight Savings Time settings of the computer on which the historical data retrieval occurs. If the Time Zone settings are different between the logging computer and the retrieval computer, the record logged at 10:00:00 (logging node time) may show a different time on the retrieval node. In most cases, if the logging and retrieval are performed on the same computer (or on computers with identical Time Zone settings) the retrieved records will be correct.

To set your time zone in the Windows Control Panel

1. Open the Windows Control Panel.
2. Double-click the **Date/Time** icon or, double-click the clock in the Windows **Taskbar**. The **Date/Time Properties** dialog box appears.



(GMT-08:00) Pacific Time (US & Canada); Tijuana

☑ Automatically adjust clock for daylight saving changes

3. Click the **Time Zone** tab, and then click the arrow to open the list of time zones.
4. Select your time zone in the list.
5. Uncheck **Automatically adjust clock for daylight saving changes**.
6. Click **OK**.

Note The Windows 2000 and Windows NT operating systems may be set to automatically adjust the clock for daylight savings time. It is recommended that this feature be disabled since it will cause incorrect values to be reported when the change occurs for daylight savings time. To disable this feature, use the Date/Time utility in the Windows Control Panel or, double-click the clock on the Windows Taskbar.

For more information on system time, see Chapter 5, "Building a Distributed Application."

Automatically Changing the System Time

Windows 2000 and Windows NT operating systems will attempt to automatically adjust the clock for daylight savings time. You can turn this feature off through the Date/Time utility in the Windows Control Panel and then, adjust the time manually or, use InTouch QuickScripts to automatically set the clock at these times:

To set the clock forward in the spring for Windows 2000 and Windows NT

Create the following Condition QuickScript:

```
$Year == yyyy and $Month == 04 and $Day == dd and
$Hour == 02 and DaylightSavingsTime == 0 ;
```

where:

yyyy = the year (i.e., 1993, 1994, or 1995 ...)

dd = the date of the time change

DaylightSavingsTime = a user-defined memory discrete tagname to indicate daylight savings time. Its initial value is 0.

ON TRUE:

```
DaylightSavingsTime = 1;
```

StartApp "c:\Winnt\System32\control.exe Date/Time";

{you may need to adjust the path to reflect where the file resides on your system} **SendKeys "%(t)" ;**

```
SendKeys "%(03)" ;
SendKeys "%{TAB}" ;
SendKeys "%{TAB}" ;
SendKeys "%{TAB}" ;
SendKeys " {ENTER}"
Flag=0;
```

Note Depending on the speed of your computer system, the delay introduced by the second condition script (while true) can be increased or decreased. The delay is introduced to allow the CONTROL.EXE to start before sending the keys.

To set the clock forward in the spring for Windows 95 (or later) using a command line

Create the following Condition QuickScript:

```
$Year==yyyy and $Month == 04 and $Day == dd and $Hour ==02
    and DaylightSavingsTime == 0;
```

where:

yyyy = the year (i.e., 1993, 1994, or 1995 ...)

dd = the day of the time change for that year

DaylightSavingsTime = a user-defined memory discrete tagname to indicate daylight savings time Its initial value is 0.

ON TRUE:

```
DaylightSavingsTime = 1;
StartApp "c:\Command.com /c time 03:00";
```

To set the clock forward in the spring for Windows NT using the WWDosCommand() Function

Create the following Condition QuickScript:

```
$Year==yyyy and $Month == 04 and $Day == dd and $Hour ==02
   and DaylightSavingsTime == 0;
```

where:

yyyy = the year (i.e., 1993, 1994, or 1995 ...)

dd = the day of the time change for that year

DaylightSavingsTime = a user-defined memory discrete tagname to indicate daylight savings time. Its initial value is 0.

ON TRUE:

```
DaylightSavingsTime = 1;
WWDosCommand ("time 03:00", "Invisible";
```

To set the clock back in the fall for the Windows 2000 or Windows NT operating systems

Create the following Condition QuickScript:

```
$Year == yyyy and $Month == 10 and $Day == dd and
$Hour == 02 and DaylightSavingsTime == 0 ;
```

where:

yyyy = the year (i.e., 1993, 1994, or 1995 ...)

dd = the day of the time change for that year

DaylightSavingsTime = a user-defined memory discrete tagname to indicate daylight savings time. Its initial value is 0.

Note Whenever a systems clock is set back, the historical logging engine may overwrite existing data in the historical log file. To prevent this loss of data, back up the log files before setting the clock back.

ON TRUE:

```
DaylightSavingsTime = 1;
StartApp "c:\windows\control.exe Date/Time" ;
```

Note You may need to change the path to reflect where CONTROL.EXE resides on your computer system.

Flag is a user defined memory integer tagname. Its initial value is 0.

```
Flag==1;
```

WHILE TRUE EVERY 1000 MSEC:

```
SendKeys "%(t)" ;
SendKeys "%(01)" ;
SendKeys "01" ;
SendKeys "%{TAB}" ;
SendKeys "%{TAB}" ;
SendKeys "%{TAB}" ;
SendKeys "% {ENTER}"
Flag=1;
Flag==1;
```

Note Depending on the speed of your computer system, the delay introduced by the second condition script (while true) can be increased or decreased. The delay is introduced to allow the CONTROL.EXE to start before sending the keys.

To set the clock backward in the fall for Windows 95 using a command line

Create the following Condition QuickScript:

```
$Year==yyyy and $Month == 04 and $Day == dd and $Hour ==02
   and DaylightSavingsTime == 0;
```

where:

yyyy = the year (i.e., 1993, 1994, or 1995 ...)

dd = the day of the time change for that year

DaylightSavingsTime = a user-defined memory discrete tagname to indicate daylight savings time. Its initial value is 0.

ON TRUE:

```
DaylightSavingsTime = 1;
StartApp "c:\Command.com /c time 01:00";
```

To set the clock backward in the fall for Windows NT using the WWDosCommand() Function

Create the following Condition QuickScript:

```
$Year==yyyy and $Month == 04 and $Day == dd and $Hour ==02
   and DaylightSavingsTime == 0;
```

where:

yyyy = the year (i.e., 1993, 1994, or 1995 ...)

dd = the day of the time change for that year

DaylightSavingsTime = a user-defined memory discrete tagname to indicate daylight savings time. Its initial value is 0.

ON TRUE:

```
DaylightSavingsTime = 1;
WWDosCommand ("time 01:00","Invisible";
```

Note Whenever a system's clock is set back, the historical logging engine may overwrite existing data in the historical log file. To prevent this loss of data, back up your log files before setting the clock back.

To set the clock backward in the fall and prevent loss of data

Create the following Condition QuickScripts:

```
$Year==yyyy and $Month == 04 and $Day == dd and $Hour ==02
   and DaylightSavingsTime == 0;
```

where:

yyyy = the year (i.e., 1993, 1994, or 1995 ...)

dd = the day of the time change for that year

DaylightSavingsTime = a user-defined memory discrete tagname to indicate daylight savings time. Its initial value is 0.

ON TRUE:

```
DaylightSavingsTime = 1;
```

```
$Historicallogging=0;
```

This script will stop historical logging.

```
StartApp "C:\Winnt\system32\control.exe date/time
```

Note You may need to change the path to reflect where CONTROL.EXE resides on your computer system.

Flag is a user defined memory integer tagname. Its initial value is 0.

```
Flag==1;
```

WHILE TRUE EVERY 1000 MSEC:

```
SendKeys "(t)";

SendKeys "01";

SendKeys "%{TAB}" ;

SendKeys "%{TAB}" ;

SendKeys "%{TAB}" ;

SendKeys "{ENTER}"

FileMove("c:\history\9910dd00.igh",
   "c:\backup","monitor");

Flag=2;  Flag==2
```

WHILE TRUE EVERY 5000 MSEC:

```
FileMove("c:\history\9910dd00.idx", "c:\backup",
   "monitor");

Flag=3;

Flag==3
```

WHILE TRUE EVERY 5000 MSEC:

```
$Historicallogging=1;

Flag=0;
```

Where:

history is the directory where the log files reside

backup is the directory the log files will move to.

9910DD.IGH and 9910DD00.IDX are the log files being moved

'DD' is replaced by the current day.

The delay of 5000 Msec in the Condition QuickScripts can be increased or decreased depending upon the size of the log files and the speed of your computer system.

HistData Utility Program

The HistData utility program provides DDE (Dynamic Data Exchange) access to the historical data files created by InTouch. It is used to move selected historical data into a requesting program such as Microsoft Excel. HistData provides you with the ability to immediately view historical data or create a file for later access. Access to the historical data may be accomplished via macro functions in a requesting program or from within InTouch.

Note The HistData program should be started (then reduced to an icon) prior to starting any program that will be using it.

The HistData program cannot be used with remote tagname references.

The HistData Database

The HistData program contains its own internal database. The items in the internal database are used to specify start period, duration and sampling interval, and so on, for the historical data to be accessed. The following lists the items defined in the HistData program:

DATADIR

Message type: Pathname of the directory containing the historical data files, for example, C:\InTouch\App.

DBDIR

Message type: Pathname of the directory containing the InTouch Tagname Dictionary, for example, C:\InTouch\App.

STARTDATE

Message type: Data sample start date in the format MM/DD/YY

STARTTIME

Message type: Data sample start time in the format HH:MM:SS using the 24-hour clock.

DURATION

Message type: The length of time for which data is to be returned. **DURATION** can be expressed in weeks, days, hours, minutes and seconds.

The following are the valid characters:
w (week), **d** (day), **h** (hour), **m** (minute), **s** (second).

Fractional values are also permitted.
For example, .5s for 500milliseconds

To request a single sample, set **DURATION** to 0 (zero).

INTERVAL

Message type: The length of time in between samples. **INTERVAL** can be expressed in weeks, days, hours, minutes and seconds, for example, 1w represents 1 week. Fractional values are also allowed, for example, .25d represents 6 hours.

(The valid characters are the same as those for **DURATION**.)

Note The maximum length of time allowed for **DURATION** and **INTERVAL** is 6 weeks. This applies to all request types, days, seconds, etc. For example, if using days, 42 is the maximum (7 days x 6 weeks = 42).

TAGS

Message type: The list of tagnames to return historical data for. **TAGS** is entered in the form "TagA,TagB,TagZ". In addition, the date and/or time for a sample can be requested by using the internal system tagnames **$Date** and **$Time**. For example:

"$Date,TagA,TagB" or,
"$Time,TagA,TagB" or,
"$Date,$Time,TagA,TagB"

TAGS1, TAGS2,....

Message type: The **TAGS** string can be 131 characters in WindowViewer and 255 characters in Excel. The string can be appended for longer requests by adding tagname items named "Tags1," "Tags2" and so on. If a tagname needs additional tagname text appended to it, a plus (+) is entered at the end of the string. For example:

TAGS="$Date,ProdLevel,ProdTemp,+"

TAGS1="ReactLevel,Temp,GasLevel,+"

TAGS2="MotorStatus"

Note Duplicate tagnames are not allowed and the maximum length of each tags string is 512 bytes.

PRINTTAGNAMES

Discrete type: This item defaults to 1 and causes HistData to print the tagnames on the first line of the output file above the associated column of values. If the tagnames are not to be printed, this item's value must be changed to 0 (zero).

DATA

Message type: This item is used to hold the requested data in the HistData program in comma separated variable format. It is used by other applications which want to **ADVISE** or **REQUEST** data via DDE.

SENDDATA

Integer type: When set to 1, HistData will update the **DATA** item with the requested data. When the update is complete, **SENDDATA** is automatically reset to 0 (zero).

Note To you receive an error message telling you that you have requested too much data when you use **SENDDATA**, shorten the **DURATION** or reduce the number of tagnames that you are requesting.

FILENAME

Message type: Complete pathname of the file to write the requested data, for example, C:\INTOUCH\HDFILE.CSV.

WRITEFILE

Integer type: When set to 1, HistData will write the requested data to the file specified by the **FILENAME** item name. When the file update is complete, **WRITEFILE** is automatically reset to 0 (zero).

STATUS

Discrete type: Displays the status of the last operation. A 1 indicates success and a 0 (zero) indicates an error occurred.

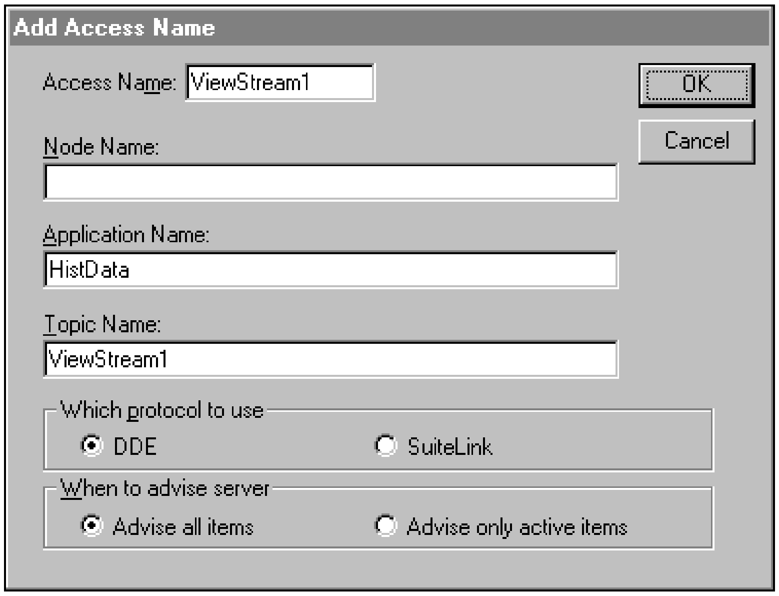
ERROR

Message type: A string containing a description of the last error. It will be "None" when **STATUS** is 1, and will contain an error message string when **STATUS** is 0 (zero).

Using HistData with InTouch

In this section, we have created a sample window to show you one method that you can use to request and display historical data in InTouch.

In order for InTouch to request data from the HistData program, the following Access Name was defined:


Add Access Name

Access Name: ViewStream1

Node Name:

Application Name: HistData

Topic Name: ViewStream1

Which protocol to use: [✓ DDE] SuiteLink

When to advise server: [✓ Advise all items] Advise only active items

The **Access Name** can be any arbitrary name up to 32 characters. The **Topic Name** can also be any arbitrary name up to 32 characters. It is recommended that the same name be used for both of these items. The **Application Name** must be the program name, **HistData** (less the .exe).

It is also recommended that the **Advise all items** option be selected whenever HistData is being used.

For more information on Access Names, see Chapter 13, "I/O Communications."

After the Access Name was defined, the following I/O type tagnames were created for each HistData internal database item:

Tagname	I/O Tag Type	Tagname	I/O Tag Type
DATA	Message	**SENDDATA**	Integer
DATADIR	Message	**STARTDATE**	Message
DBDIR	Message	**STARTTIME**	Message
DURATION	Message	**STATUS**	Discrete
ERROR	Message	**TAGS**	Message
FILENAME	Message	**WRITEFILE**	Integer
INTERVAL	Message	**PRINTTAGNAMES**	Discrete

If you want to send the data not only to the .CSV file, but also to the item called **Data** so that it can be accessed from other applications, create the following two additional tagnames:

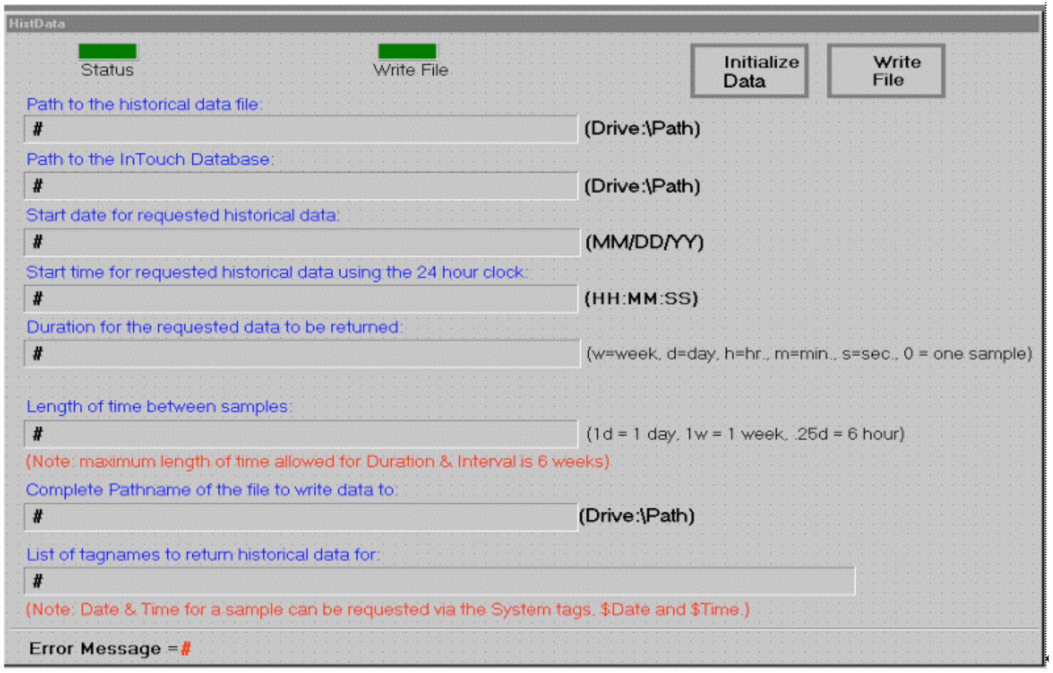
Tagname	I/O Tag Type	Access Name	Item
HDWSendData	Discrete	Viewstream1	**SendData**
HDWData	Message	Viewstream1	**Data**

Note If you create and use these tagnames and then request a lot of information you will get the error: Too much data requested – shorten the duration or reduce the number of tags.

The HistData tagnames and the Access Name are automatically created when you use the HistData wizard. To use the HistData wizard:

1. Click the Wizard tool.
2. Open the Trends group of wizards.
3. Double-click the HistData wizard and paste it into your window.
4. Double-click the HistData wizard and assign it a Hist Tend type tagname.
5. Click **OK** to create the HistData tagnames.

After you have created the I/O type tagnames, create a new window called **HistData** that looks like the following example:


HistData

Status

Write File

Initialize Data

Write File

Path to the historical data file:
(Drive:\Path)

Path to the InTouch Database:
(Drive:\Path)

Start date for requested historical data:
(MM/DD/YY)

Start time for requested historical data using the 24 hour clock:
(HH:MM:SS)

Duration for the requested data to be returned:
(w=week, d=day, h=hr., m=min., s=sec., 0 = one sample)

Length of time between samples:
(1d = 1 day, 1w = 1 week, .25d = 6 hour)
(Note: maximum length of time allowed for Duration & Interval is 6 weeks)

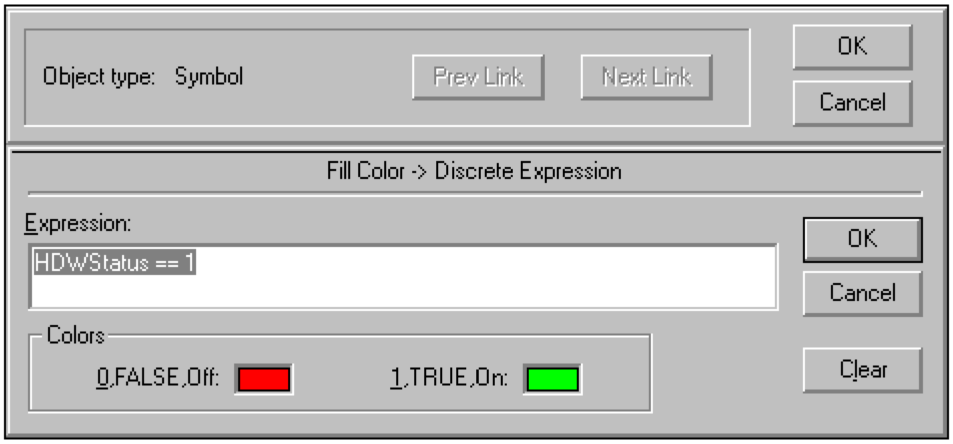
Complete Pathname of the file to write data to:
(Drive:\Path)

List of tagnames to return historical data for:
#
(Note: Date & Time for a sample can be requested via the System tags, $Date and $Time.)

Error Message = #

The # symbols are linked to an input link. For example, the # symbol has a User Inputs/String link to the tagname **HDWDataDir**. The user input link allows you to change the value of the tagnames during runtime.

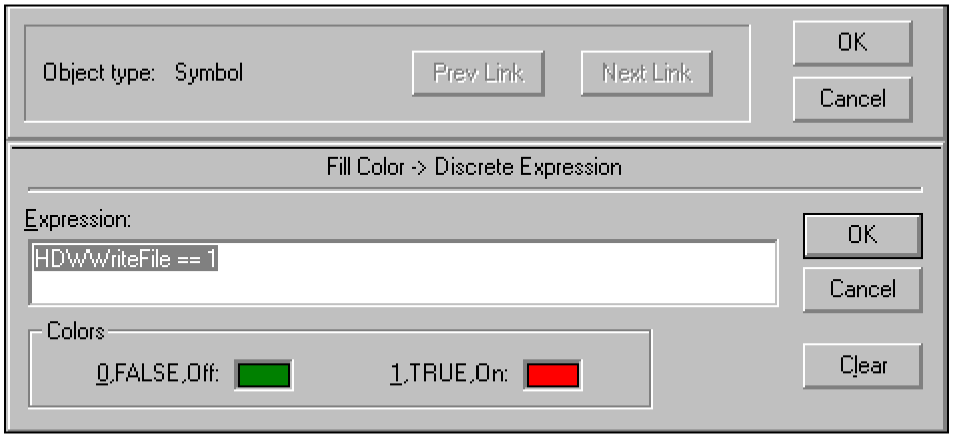
The **Status** button is linked to a fill color—discrete expression, based on the tagname **HDWStatus**:



HDWStatus == 1

0,FALSE,Off: 🟥 1,TRUE,On: 🟩

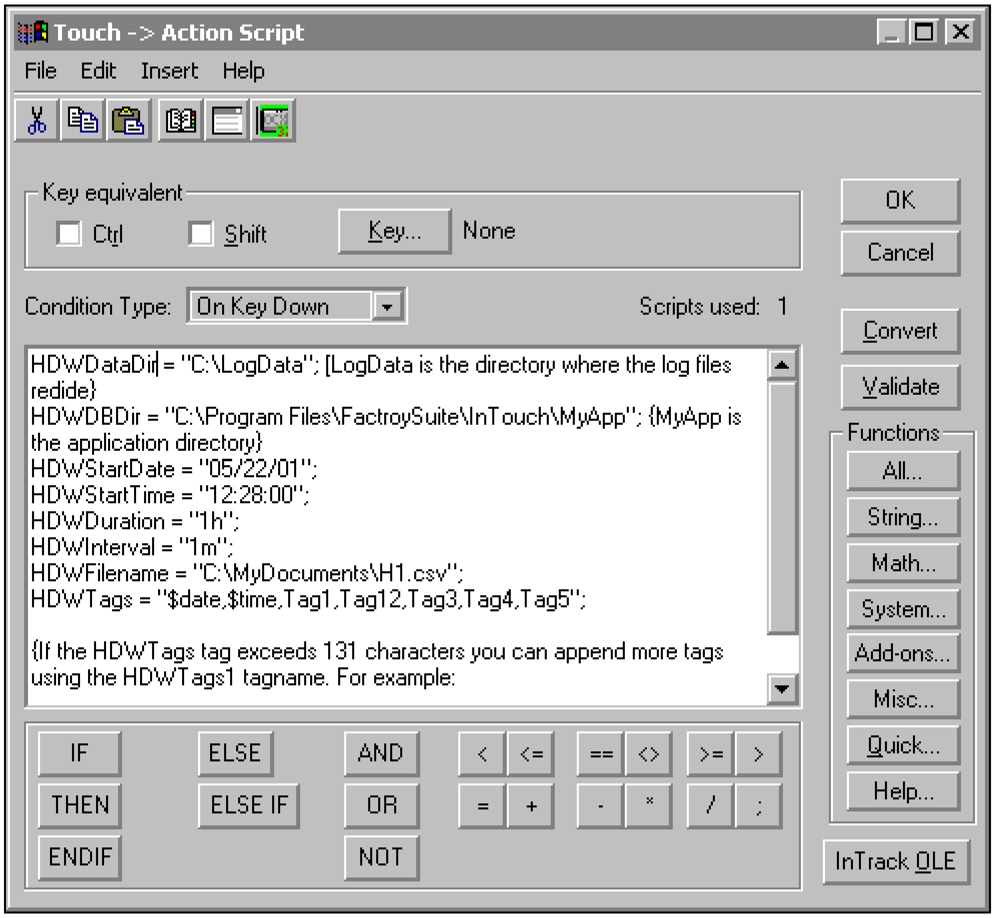
The **Write File** button is linked to a fill color—discrete expression, based on the tagname **HDWWriteFile**:



HDWWriteFile == 1

0,FALSE,Off: 🟩 1,TRUE,On: 🟥

The **Initialize Data** button is linked to a Touch Pushbutton—Action script.



Key equivalent: None

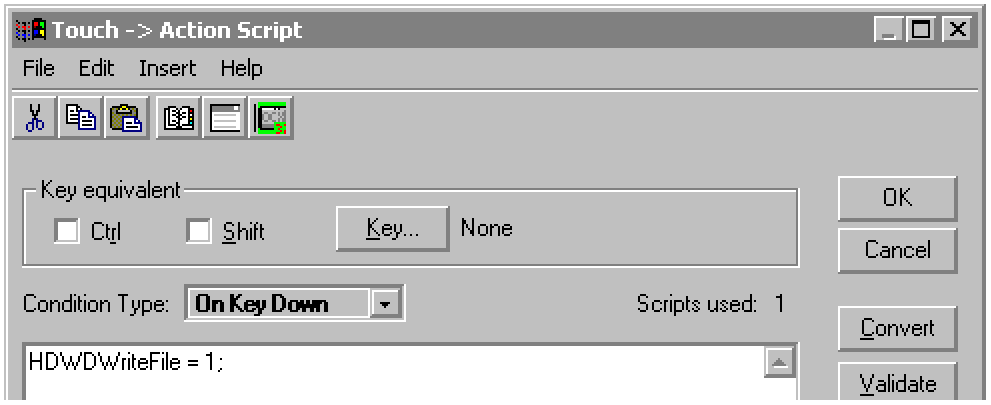
Condition Type: On Key Down

Scripts used: 1

```
HDWDataDir = "C:\LogData"; [LogData is the directory where the log files
redide}
HDWDBDir = "C:\Program Files\FactroySuite\InTouch\MyApp"; {MyApp is
the application directory}
HDWStartDate = "05/22/01";
HDWStartTime = "12:28:00";
HDWDuration = "1h";
HDWInterval = "1m";
HDWFilename = "C:\MyDocuments\H1.csv";
HDWTags = "$date,$time,Tag1,Tag12,Tag3,Tag4,Tag5";

{If the HDWTags tag exceeds 131 characters you can append more tags
using the HDWTags1 tagname. For example:
```

When the **Initialize Data** button is pushed, the HistData items are initialized with the desired values. If necessary, these values can also be changed during runtime by using the User Inputs Links.

The **Write File** button is linked to a Touch Pushbutton—Action script:



Key equivalent: None

Condition Type: On Key Down

Scripts used: 1

```
HDWDWriteFile = 1;
```

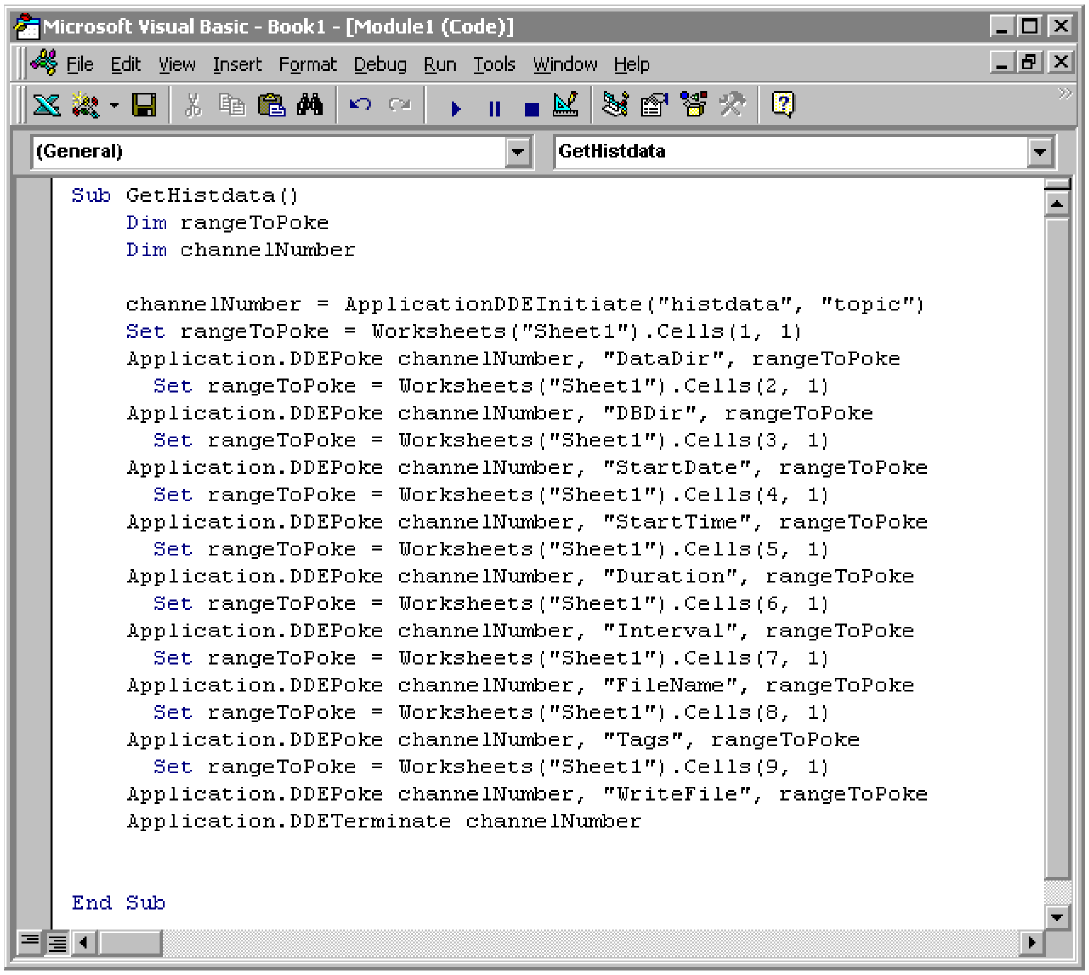
When clicked, the **WriteFile** button generates the .CSV file.

After all the above items are configured and saved, start **HistData** and minimize it. Next, start WindowViewer and open the HistData window. Click the **Initialize** button and make changes to the HistData items if needed, and then click the **WriteFile** button. If the operation was successful, the value of **Status** will be **ON** and the color associated with the **ON** status will display. If the operation was not successful, the value of **Status** will be **OFF** and **Error Message** will display the cause of the failure.

Using HistData with Excel

The HistData program responds to the **INITIATE**, **POKE**, and **TERMINATE** functions of products such as Microsoft Excel. The **POKE** function with a **keyword** (an internal database item) is used to set up the parameters that define a query. Once the query is properly set up, the macro is run to request the selected historical data.

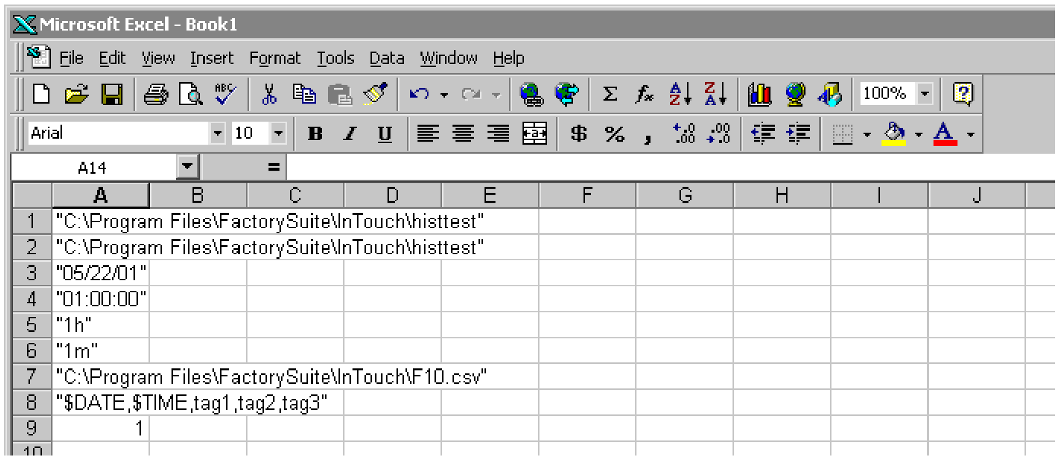
Excel macros can be written to interface with HistData. Here is an example of a macro written with Excel 5.0VBA (same for Excel 7.0 or Excel 97):


Microsoft Visual Basic - Book1 - [Module1 (Code)]
File Edit View Insert Format Debug Run Tools Window Help
(General)
GetHistdata

```
Sub GetHistdata()
    Dim rangeToPoke
    Dim channelNumber

    channelNumber = ApplicationDDEInitiate("histdata", "topic")
    Set rangeToPoke = Worksheets("Sheet1").Cells(1, 1)
    Application.DDEPoke channelNumber, "DataDir", rangeToPoke
      Set rangeToPoke = Worksheets("Sheet1").Cells(2, 1)
    Application.DDEPoke channelNumber, "DBDir", rangeToPoke
      Set rangeToPoke = Worksheets("Sheet1").Cells(3, 1)
    Application.DDEPoke channelNumber, "StartDate", rangeToPoke
      Set rangeToPoke = Worksheets("Sheet1").Cells(4, 1)
    Application.DDEPoke channelNumber, "StartTime", rangeToPoke
      Set rangeToPoke = Worksheets("Sheet1").Cells(5, 1)
    Application.DDEPoke channelNumber, "Duration", rangeToPoke
      Set rangeToPoke = Worksheets("Sheet1").Cells(6, 1)
    Application.DDEPoke channelNumber, "Interval", rangeToPoke
      Set rangeToPoke = Worksheets("Sheet1").Cells(7, 1)
    Application.DDEPoke channelNumber, "FileName", rangeToPoke
      Set rangeToPoke = Worksheets("Sheet1").Cells(8, 1)
    Application.DDEPoke channelNumber, "Tags", rangeToPoke
      Set rangeToPoke = Worksheets("Sheet1").Cells(9, 1)
    Application.DDEPoke channelNumber, "WriteFile", rangeToPoke
    Application.DDETerminate channelNumber


End Sub
```

The data that is poked contained in a cell in the Excel spreadsheet1, called Sheet1.

Here is an example of Sheet1 containing the data to be poked:



	A	B	C	D	E	F	G	H	I	J
1	"C:\Program Files\FactorySuite\InTouch\histtest"									
2	"C:\Program Files\FactorySuite\InTouch\histtest"									
3	"05/22/01"									
4	"01:00:00"									
5	"1h"									
6	"1m"									
7	"C:\Program Files\FactorySuite\InTouch\F10.csv"									
8	"$DATE,$TIME,tag1,tag2,tag3"									
9	1									

Common HistData Error Messages

Error Message	Cause/Solution
Too much data requested – shorten the duration or reduce the number of tagnames.	This error occurs when you are using the SendData item and too much data is requested. If the only purpose is to create a .CSV file with the data from the encrypted log files, then ***do not use SendData***.
Could not open file C:\FILES1\HISTDATA.CSV	The directory **Files1** does not exist. Also, check the spelling of the path.
Could not open file C:\FILES\	No CSV file was defined.
DATADIR item invalid	The directory specified for the DataDir item does not exist. Check the spelling.
STARTDATE item invalid	Invalid format sent to the StartDate item.
No log files found	There are no log files for the requested date in the path that is specified for the DataDir item.
Could not find tagname TAGX in database	The requested tagname (in this case, **TAGX**) does not exist in the tagname dictionary. Check the spelling of the tagname.
Could not find tagname .x in: C:\IT6.0B\HISTEST	The file **tagname.x** does not exist or it has been corrupted.

C H A P T E R 1 3

I/O Communications

InTouch uses the Microsoft Dynamic Data Exchange (DDE), FastDDE, NetDDE and Wonderware SuiteLink protocols to communicate with other Windows programs, Wonderware I/O Servers and third-party I/O Server programs that are communicating with the real world.

Contents

Supported Communication Protocols

Dynamic Data Exchange (DDE) is a communication protocol developed by Microsoft to allow applications in the Windows environment to send/receive data and instructions to/from each other. It implements a client-server relationship between two concurrently running applications. The *server* application provides the data and accepts requests from any other application interested in its data. Requesting applications are called *clients*. Some applications such as InTouch and Microsoft Excel can simultaneously be both a *client* and a *server*.

FastDDE provides a means of packing many proprietary Wonderware DDE messages into a single Microsoft DDE message. This packing improves efficiency and performance by reducing the total number of DDE transactions required between *client* and *server*. Although Wonderware's FastDDE has extended the usefulness of DDE for our industry, this extension is being pushed to its performance constraints in distributed environments.

NetDDE extends the standard Windows DDE functionality to include communication over local area networks and through serial ports. Network extensions are available to allow DDE links between applications running on different computers connected via networks or modems. For example, NetDDE supports DDE between applications running on IBM PCs connected via LAN or modem and DDE-aware applications running on non-PC based platforms under operating environments such as VMS and UNIX.

Wonderware SuiteLink uses a TCP/IP based protocol. SuiteLink is designed specifically to meet industrial needs, such as data integrity, high-throughput, and easier diagnostics. This protocol standard is supported for both Microsoft Windows NT and Windows 2000.

SuiteLink is not a replacement for DDE, FastDDE, or NetDDE. Each connection between a client and a server depends on your network situation. SuiteLink was designed specifically for high speed industrial applications and provides the following features:

Value Time Quality (VTQ) places a time stamp and quality indicator on all data values delivered to VTQ-aware clients.

Extensive diagnostics of the data throughput, the server loading, computer resource consumption, and network transport are made accessible through the Microsoft Windows NT operating system performance monitor. This feature is critical for the scheme and maintenance of distributed industrial networks.

Consistent high data volumes can be maintained between applications regardless if the applications are on a single node or distributed over a large node count.

The network transport protocol is TCP/IP using Microsoft's standard Winsock interface.

Wonderware SuiteLink

Wonderware SuiteLink uses a TCP/IP based protocol. SuiteLink is designed specifically to meet industrial needs, such as data integrity, high-throughput, and easier diagnostics. This protocol standard is supported for both Microsoft Windows NT and Windows 2000.

To use the SuiteLink communication protocol

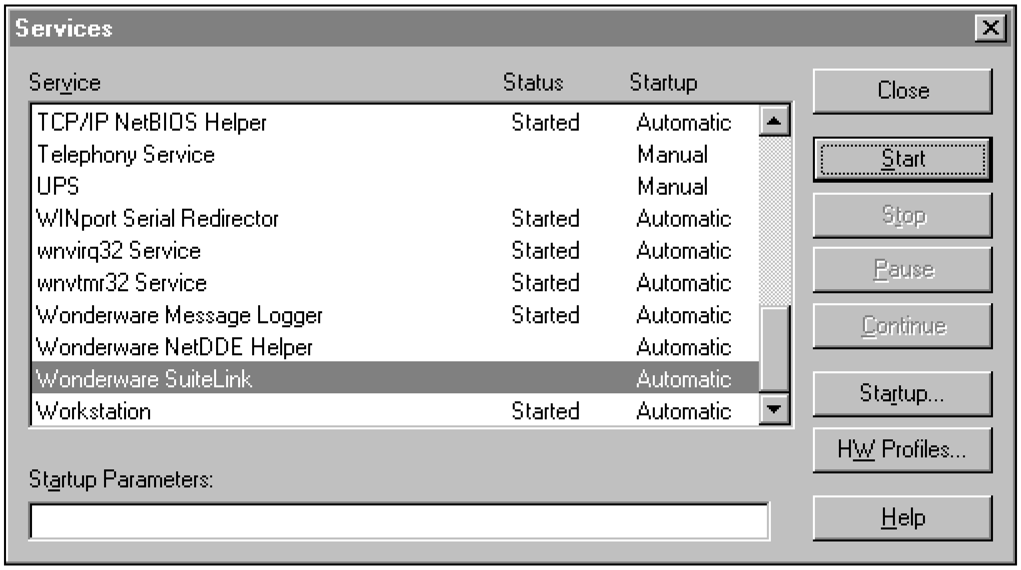
1. You must have Microsoft TCP/IP configured and working properly.
2. You must use computer names (**Node Names**) of no more than 15 characters.

   For more information on installing and configuring Microsoft TCP/IP, see your Microsoft Windows operating system's documentation.
3. Wonderware SuiteLink must be running as a service. If for some reason SuiteLink has been stopped, you will need to start it again. (SuiteLink is automatically installed when you install InTouch and by default, it is configured to startup automatically as an NT Service.)

   For more information on Windows NT services, see Appendix A, "Overview of the InTouch Windows NT Services."

To start SuiteLink as an NT Service

1. Open the Windows Control Panel.
2. Double-click **Services**. The **Services** dialog box appears.



Service	Status	Startup
TCP/IP NetBIOS Helper	Started	Automatic
Telephony Service		Manual
UPS		Manual
WINport Serial Redirector	Started	Automatic
wnvirq32 Service	Started	Automatic
wnvtmr32 Service	Started	Automatic
Wonderware Message Logger	Started	Automatic
Wonderware NetDDE Helper		Automatic
Wonderware SuiteLink		Automatic
Workstation	Started	Automatic

Startup Parameters:

3. Select **Wonderware SuiteLink**, and then click **Start**.
4. Click **Close**.

The InTouch I/O Address Convention

InTouch identifies an element of data in an I/O Server program by using a three-part naming convention that includes the *application name, topic name* and *item name*. To obtain data from another application the *client* program (InTouch) opens a channel to the *server* program by specifying these three items.

In order for InTouch to acquire a data value from another application, it must also know the name of the *application* providing the data value, the name of the *topic* within the application that contains the data value, and the name of the specific *item* within the *topic*. In addition, InTouch needs to know the data's type; discrete, integer, real (floating point), or message (string). This information determines the I/O type for the tagname when it is defined in the InTouch database. Now, when WindowViewer is running, it will automatically perform all of the actions required to acquire and maintain the value of this *item*.

For example, in the case of Excel, the *application name* is "Excel," the *topic name* is the name of the specific spreadsheet that contains the data and the *item name* is the identification of the cell on the spreadsheet to/from which the data is to be read/written.

The InTouch I/O Address

When another Windows application requests a data value from InTouch, it also must know the three I/O address items. The following describes the I/O address convention for InTouch:

VIEW *(application name)* identifies the InTouch runtime program that contains the data element.

TAGNAME *(topic name)* is the word always used when reading/writing to a tagname in the InTouch database.

ActualTagname *(item name)* is the actual tagname defined for the item in the InTouch Tagname Dictionary.

For example, to access a data value in InTouch from Excel, a DDE Remote Reference formula would be entered in the cell into which the data value is to be written:

```
=VIEW|TAGNAME!'ActualTagname'
```

Note If you are networking using Wonderware NetDDE, the *application name* portion of the I/O address must be prefixed with the remote node's name preceded by two backslashes and followed by one backslash. For example:

```
\\NodeName\VIEW|TAGNAME!'ActualTagname'
```

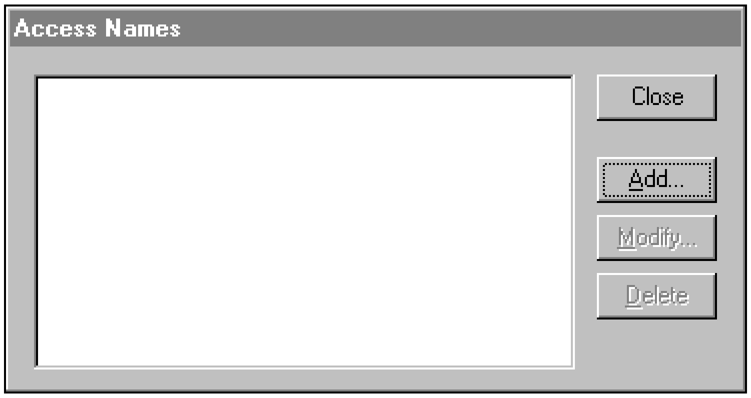
InTouch Access Names

When you create I/O type tagnames or remote tagname references, they must be associated with an Access Name. Access Names contain the information that is used to communicate with other I/O data sources including the node name, application name and topic name.

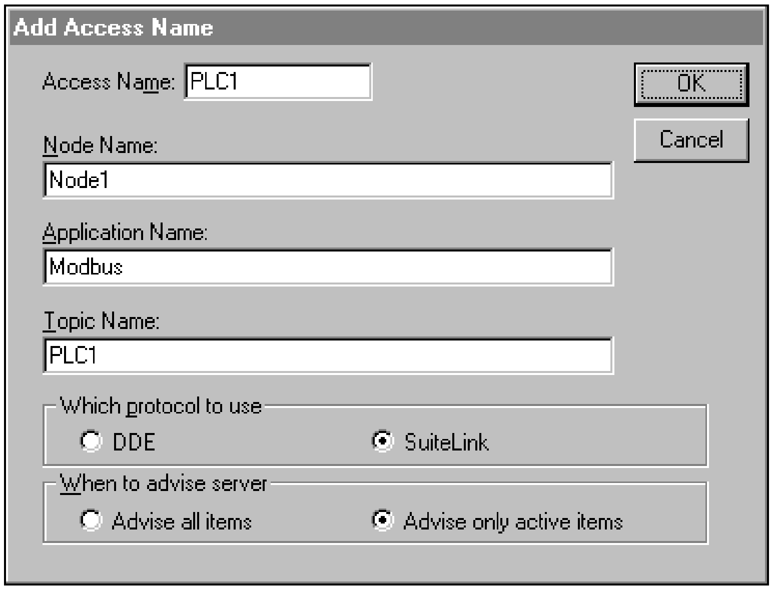
To create an Access Name

1. On the **Special** menu, click **Access Names**, or in the Application Explorer under **Configure**, double-click **Access Names**. The **Access Names** dialog box appears.

2. In the Application Explorer, you can right-click **Access Names**, and then click **Open**. You can also create Access Names while you are defining an I/O type tagname in the Tagname dictionary.



3. Click **Add**. The **Add Access Name** dialog box appears.



Access Name: PLC1

Node Name: Node1

Application Name: Modbus

Topic Name: PLC1

Which protocol to use: DDE [✓ SuiteLink]

When to advise server: Advise all items [✓ Advise only active items]

4. In the **Access Name** box, type the name you want InTouch to use to this Access Name. (For simplicity, use the same name that you will use for the *topic name* here.)

   InTouch uses Access Names to reference real-time I/O data. Each Access Name equates to an I/O address, which can contain a Node, Application, and Topic. In a distributed application, I/O references can be set up as global addresses to a network I/O Server or local addresses to a local I/O Server.

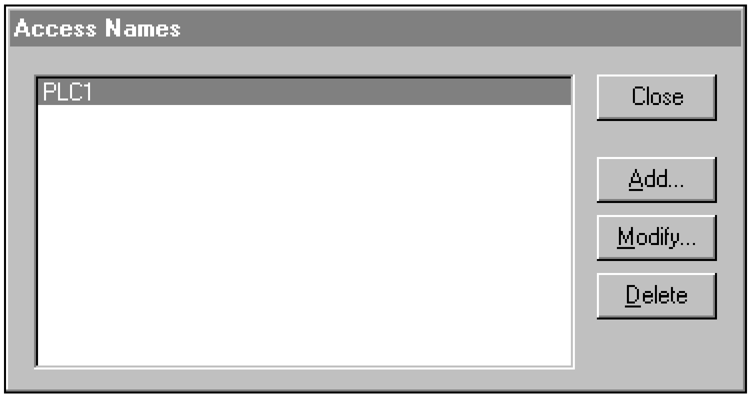
5. If the data resides in a network I/O Server, in the **Node Name** box, type the remote node's name.

6. In the **Application Name** box, type the actual program name for the I/O Server program from which the data value will be acquired. In this case the value is coming from the Wonderware Modbus I/O Server, therefore **MODBUS** is used. **Do** **not** enter the **.exe** extension portion of the program name.
7. In the **Topic Name** box, type the *topic name* you want to access. The **Topic Name** is an application-specific sub-group of data elements. In the case of data coming from a Wonderware I/O Server program, the *topic name* is the **exact** same name configured for the *topic* in the I/O Server program. When communicating with Microsoft Excel, the *topic name* must be the name given to the spreadsheet when it was saved. For example, Book1.xls.
8. Select the protocol that you are using.

   For more information, see "Supported Communication Protocols:"
9. Select the option you want to use to advise the server:

Option	Definition
Advise all items	Polls for all data whether or not it is in visible windows, alarmed, logged, trended or used in a script. Selecting this option will impact performance, therefore its use is not recommended.
Advise only active items	Polls only points in visible windows and points that are alarmed, logged, trended or used in any script. **Note** A touch pushbutton action script will not be polled unless it appears in a visible window.

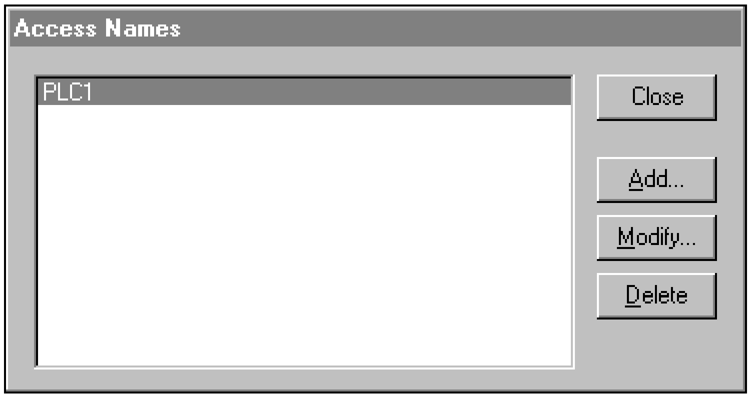
10. Click **OK** to accept the new Access Name and close the dialog box. The **Access Names** dialog box reappears displaying the new Access Name selected in the list.



11. Click **Close** to close the dialog box and return to your tagname definition.

To modify or delete an Access Name

1. On the **Special** menu, click **Access Names**, or in the Application Explorer under **Configure**, double-click **Access Names**. The **Access Names** dialog box appears.
2. In the Application Explorer, you can right-click **Access Names**, and then click **Open**.



3. To change an Access Name's definition, select it in the list, and then click **Modify**. The **Modify Access Name** dialog box appear. Make your required changes, and then click **OK**. The **Access Names** dialog box reappears. Click **Close** or repeat this procedure if you need to modify other defined Access Names.
4. To delete an Access Name, select it in the list, and then click **Delete**. A message box will appear asking you to confirm the deletion of the selected Access Name. Click **Yes** to delete it or click **No** to cancel the deletion. Click **Close** or repeat this procedure if you need to delete other defined Access Names.

Note Access Names that are associated with tagnames cannot be deleted.

Defining an I/O Item in InTouch

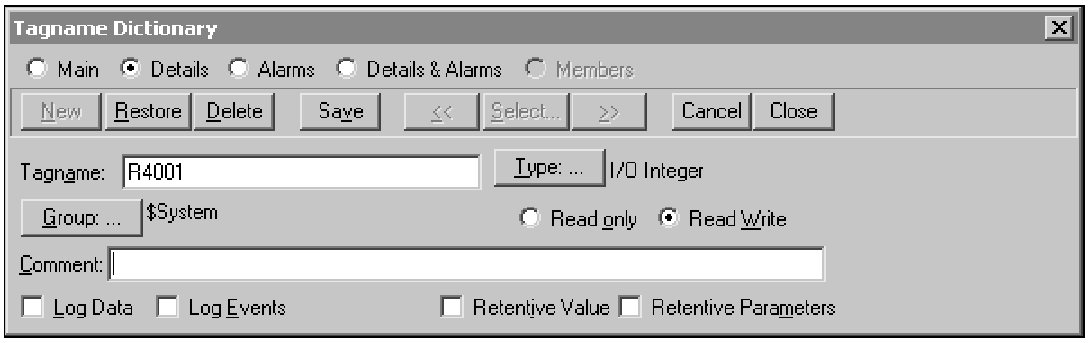
InTouch can receive data from other local or remote Windows applications when you define I/O type tagnames in the Tagname Dictionary. Each I/O type tagname references a valid *item* in the I/O Server program.

For more information on distributed applications see, Chapter 5, "Building a Distributed Application."

To define an I/O type tagname

1. On the **Special** menu, click **Tagname Dictionary**, or in the Application Explorer double-click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears.
2. Click **New**. The **Tagname** box clears.

Tip If you right-click any of the text entry boxes in any of the Tagname Dictionary dialog boxes, a menu will appear displaying the commands that you can apply to the selected text.



Main [✓ Details] Alarms Details & Alarms Members

Tagname: R4001

Type: I/O Integer

Group: $System

Read only [✓ Read Write]

Comment:

☐ Log Data ☐ Log Events ☐ Retentive Value ☐ Retentive Parameters

The first time you access the Tagname Dictionary, the definition for the internal system tagname **$AccessLevel** is displayed. Once you define tagnames in the Tagname Dictionary, when you access it again, the last edited tagname's definition is displayed.

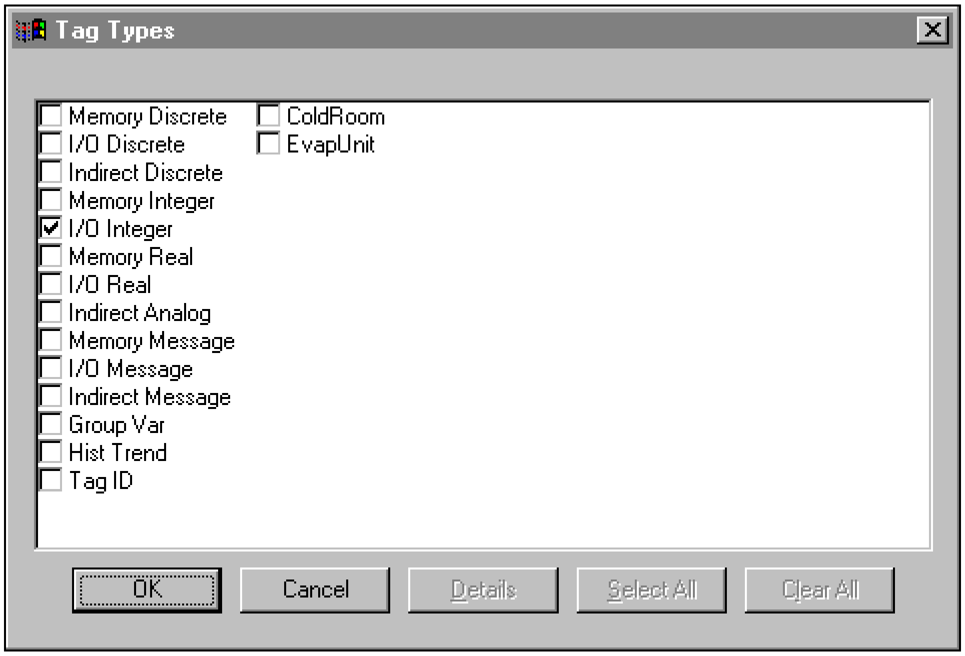
3. In the **Tagname** box, type the name you want to use for the new tagname.

   Tagnames can be up to 32 characters long and must begin with an alpha character (A-Z or a-z). The remaining characters can be A-Z, a-z, 0-9, !, @, -, ?, #, $, %, _, \ and &.

   Tagnames are also auto-indexed. For example, if you enter and save tagname R4001, and then click **New**, the tagname will automatically be indexed to R4002. If an tagname contains a character separating numbers, it is auto-indexed by the first whole number InTouch finds. For example, N7-0 would be indexed as N7-1. Positive changes only are permitted. For example, R4002 to R4003, R4003 to R4004 and so on.

   You cannot use the word **RetVal** for a tagname. This is a reserved word. If you attempt to use this word, and then try to edit a QuickFunction you will receive the error message "A variable cannot have that name. Tag exists."

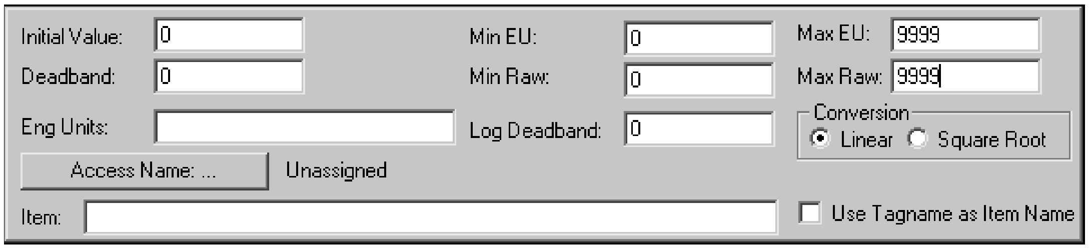
4. Click **Type**. The **Choose tagname type** dialog box appears.



5. Select the I/O type for the tagname as follows:

Tag Type	Input/Output Value
I/O Discrete	True, On, Yes (1) or False, Off, No (0)
I/O Integer	Whole number
I/O Real	Floating decimal point
I/O Message	String

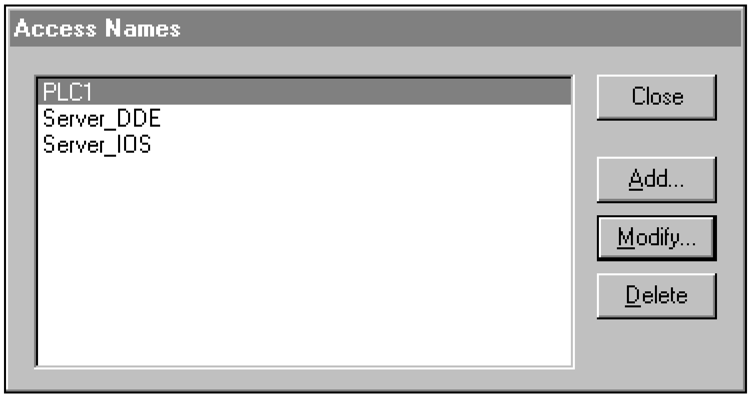
6. Once you select the I/O type, click **OK**. The respective "details" dialog box for the selected I/O type appear. For example, if you select I/O Integer, the following dialog box appears.


Initial Value: 0
Deadband: 0
Eng Units:
Min EU: 0
Min Raw: 0
Log Deadband: 0
Max EU: 9999
Max Raw: 9999
Conversion: [✓ Linear] Square Root
Access Name: ... Unassigned
Item:
Use Tagname as Item Name

Tip If the "Details" dialog box does not appear, click **Details** at the top of the screen.

7. Specify all the required details for defining the *item*.

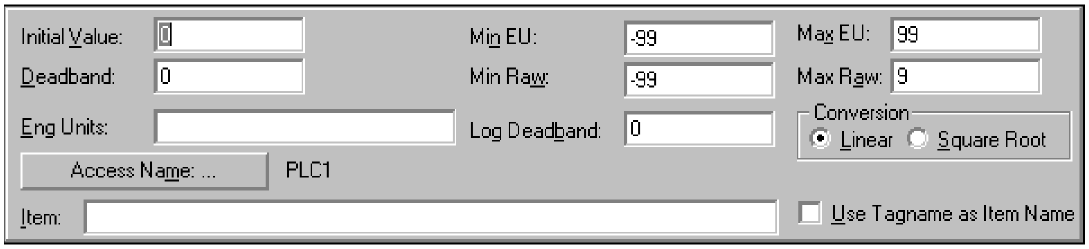
8. Click **Access Name**. The **Access Names** dialog box appears.



PLC1
Server_DDE
Server_IOS

9. Double-click the Access Name that you want to use in the list or select it, and then click **Close**.

   The Access Name that you selected (now associated with this tagname definition) appears adjacent to the **Access Name** button in the details dialog box. For example:



Initial Value: 0
Deadband: 0
Eng Units:
Min EU: -99
Min Raw: -99
Log Deadband: 0
Max EU: 99
Max Raw: 9
(● Linear ○ Square Root)
Access Name: ... PLC1
Item:
☐ Use Tagname as Item Name

10. In the **Item** box, type the *item name* for the data value in the I/O Server program.

   Note It is important to understand that the "tagname" is the name used within InTouch to refer to a data value. The **Item** is the name used by a remote Windows application to refer to the same value. These names do not have to be the same but, it is recommended when applicable to use the same names. Also, if the **Item** is a cell in Excel, it must be specified either by its defined name in Excel, or by its row/column identification. For example, R1C1.

11. Click **Close**.

   For more information on defining I/O tagnames, see Chapter 6, "Tagname Dictionary."

Monitoring the Status of an I/O Conversation

WindowViewer supports a built-in *topic name* called **IOStatus** (**DDEStatus** in versions prior to InTouch 7.0) that can be used to monitor the status of specific I/O conversations.

Using IOStatus Topic Name

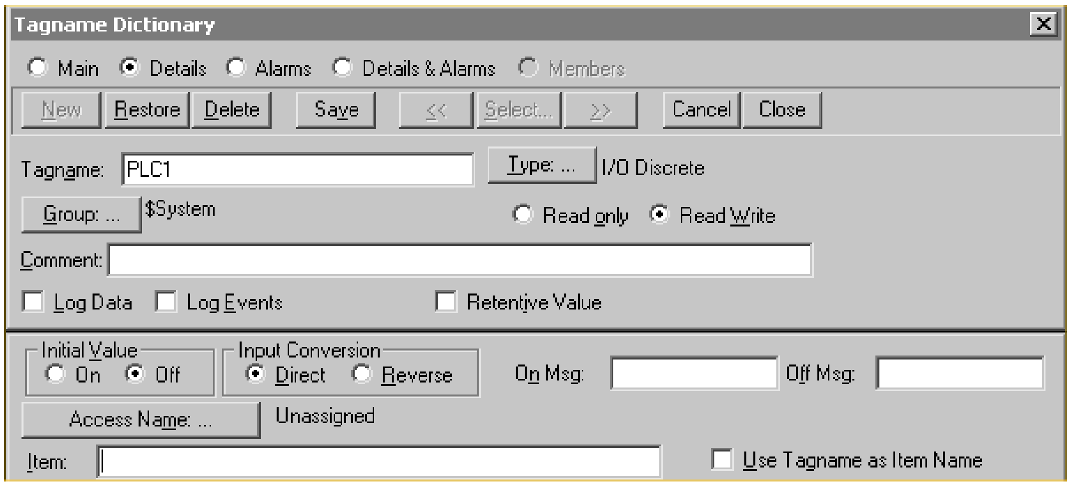
Let's assume that WindowViewer (View) is communicating with the Wonderware Simulate I/O Server to a PLC that has been defined in the I/O Server with **PLC1** for its *topic name*.

(Simulate is a generic Wonderware I/O Server that is intended to be used as a training tool. Simulate is included with FactorySuite.)

To monitor the status of I/O communications

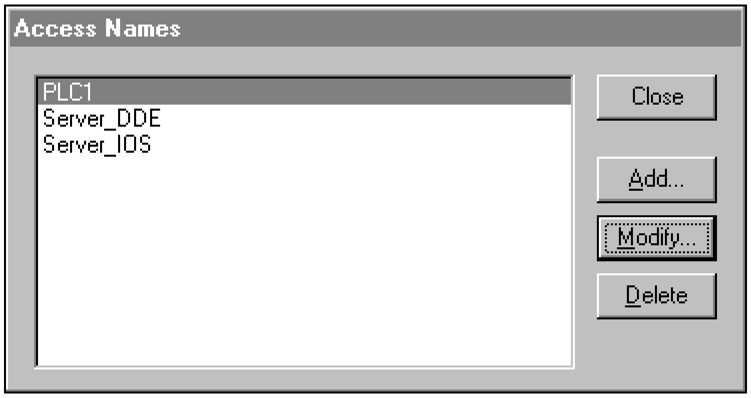
1. On the **Special** menu, click **Tagname Dictionary**, or in the Application Explorer, double-click **Tagname Dictionary**. The **Tagname Dictionary** dialog box appears.
2. Create an **I/O Discrete** type tagname. (In this example, for simplicity, we will make our tagname the same as the *topic name* that we want to monitor.)

   When you are monitoring a I/O conversation using **IOStatus**, you must define at least one I/O type tagname to the Access Name being monitored.



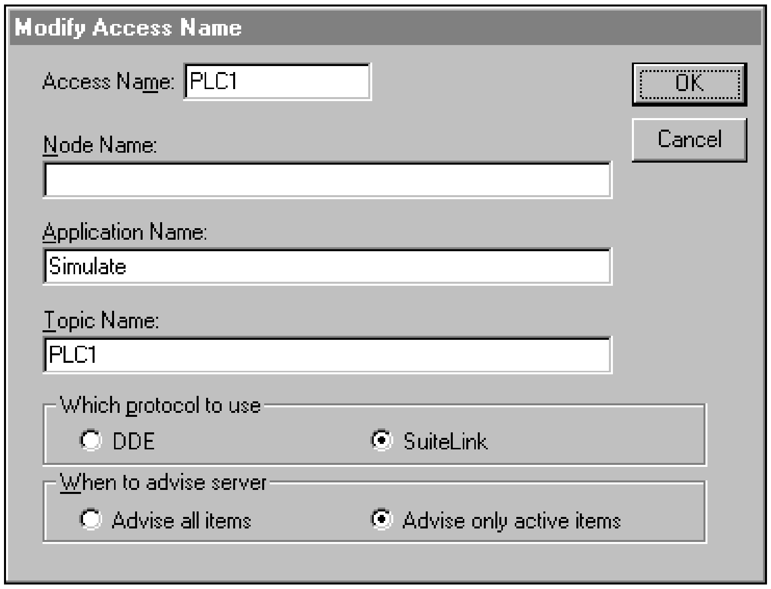
Tagname: PLC1
Type: I/O Discrete
Group: $System
( ) Read only [✓ Read Write]
Comment:
Log Data [ ] Log Events [ ] Retentive Value [ ]
Initial Value: On [✓ Off]
Input Conversion: [✓ Direct] Reverse
On Msg:
Off Msg:
Access Name: Unassigned
Item:
Use Tagname as Item Name [ ]

3. Click **Access Name** to assign the tagname to an Access Name definition that defines **IOStatus** for its *topic name*. The **Access Name** dialog box appears.



PLC1
Server_DDE
Server_IOS

Notice that an Access Name definition called **PLC1** (the topic we want to monitor) currently exists. To be sure that this is the correct Access Name (its **Topic Name** is **PLC1**), click **Modify** to view the definition.



Access Name: PLC1

Node Name:

Application Name: Simulate

Topic Name: PLC1

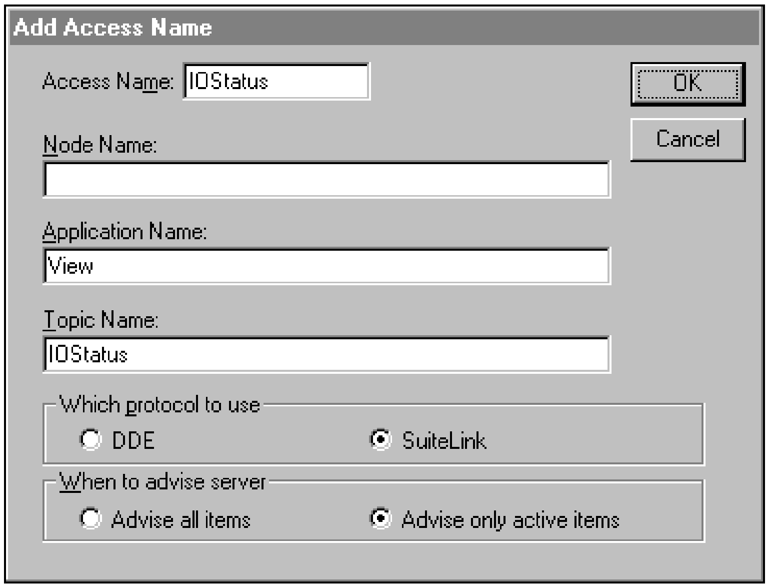
Which protocol to use: DDE [✓ SuiteLink]

When to advise server: Advise all items [✓ Advise only active items]

Finding the Access Name containing the right *topic name* in this example was easy because we kept the tagname and the **Topic Name** the same.

4. Click **Cancel** to close the dialog box and return to the initial **Access Name Definition** dialog box.

5. Click **Add**. The **Add Access Name** dialog box appears.


Add Access Name

Access Name: IOStatus

Node Name:

Application Name: View

Topic Name: IOStatus

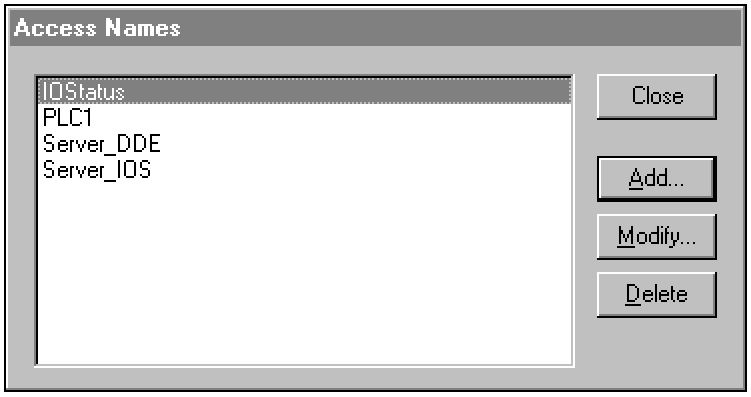
Which protocol to use: DDE [✓ SuiteLink]

When to advise server: Advise all items [✓ Advise only active items]

6. In the **Access Name** box, type **IOStatus**.

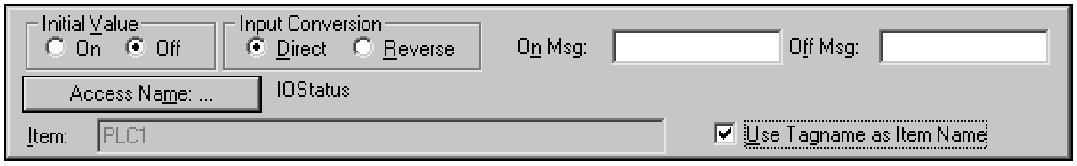
   Since you are going to monitor the status in WindowViewer, in the **Application Name** box, type "View."

7. In the **Topic Name** box, type the InTouch internal *topic*, **IOStatus**.
8. Select **Advise only active items**.
9. Click **OK** to close the dialog box. The initial **Access Name Definition** dialog box reappears displaying your new **Access Name**, **IOStatus**, in the list:


Access Names

IOStatus
PLC1
Server_DDE
Server_IOS

10. Click **Close** to close the dialog box and associate the new **Access Name** with your **I/O Discrete** tagname:



Initial Value: On [✓ Off]
Input Conversion: [✓ Direct] Reverse
Access Name: IOStatus
Item: PLC1
[✓ Use Tagname as Item Name]

11. In the **Item** box, type the **Access Name** for the actual **Topic Name** that you want to monitor. In this case, **PLC1**.

    Since your tagname is the same as the **Topic Name**, you can select **Use Tagname as Item Name** and automatically enter it for the **Item**.

    Note When using the built-in topic **IOStatus** (**DDEStatus** prior to InTouch Version 7.0) to monitor an I/O conversation, the name you type in the **Access Name** box is always also used for the **Item**.

Using IOStatus Topic Name in Excel

Excel can also be used to perform this same type of monitoring by entering the same information in a formula in a spreadsheet cell. For example, to monitor the same topic as above, the following would be entered:

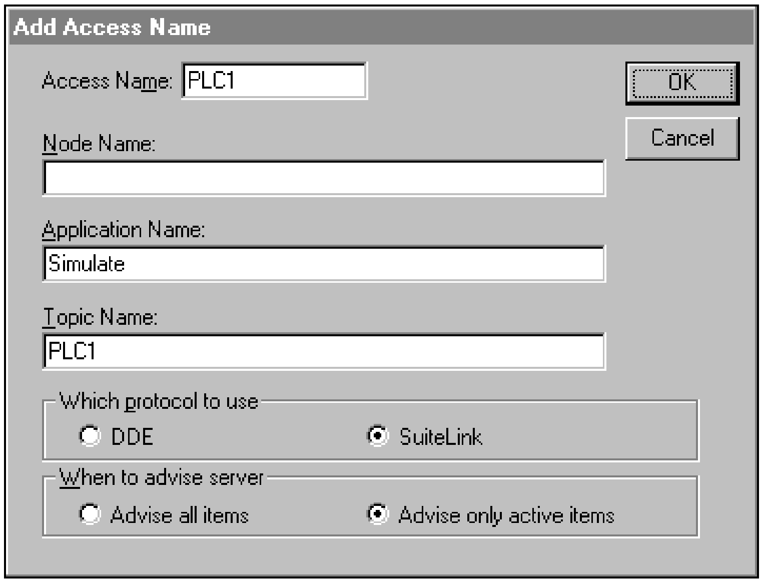
```
=view|IOStatus!'PLC1'
```

Monitoring I/O Server Communications Status

For each *topic name* being used, there is a built-in discrete *item*, **Status**, that you can use to monitor the state of your communications with the I/O Servers program. **Status** is set to "0" when communications with the device fails (cable disconnected, PLC is powered down, and so on.) and set to "1" when communications is successful.

Note When you monitor the status of a topic using the Status item, there must at least one I/O point active to the topic being monitored.

From InTouch, you can read the state of the server communications by defining a tagname and associating it with the *topic* configured for the device by using the word **Status** as the *item name*. For example, if WindowViewer is communicating with a PLC using the Wonderware Simulate I/O Server, the Access Name definition would be:



Access Name: PLC1

Node Name:

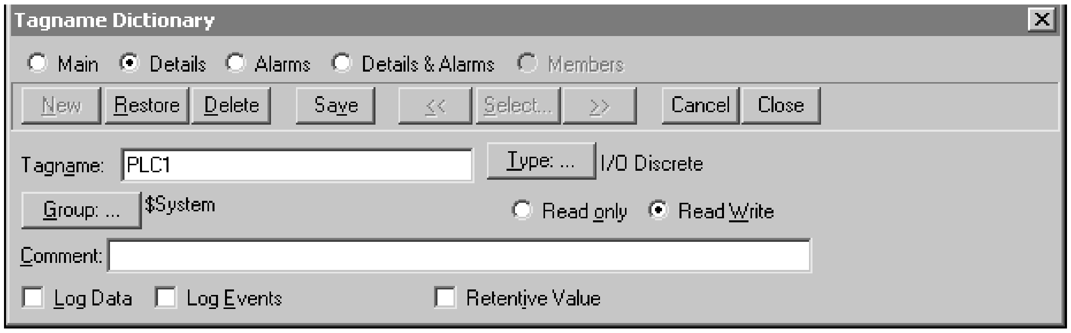
Application Name: Simulate

Topic Name: PLC1

Which protocol to use: DDE [✓ SuiteLink]

When to advise server: Advise all items [✓ Advise only active items]

To monitor the status of all communication to the *topic*, PLC1, you would create the following tagname definition:



Main [✓ Details] Alarms Details & Alarms Members

Tagname: PLC1

Type: I/O Discrete

Group: $System

Read only [✓ Read Write]

Comment:

Log Data Log Events Retentive Value

For more information on troubleshooting I/O communications see your Wonderware I/O Server's User Guide.

Tip From Excel, you can read the status of the PLC communications by entering the following formula in a cell:

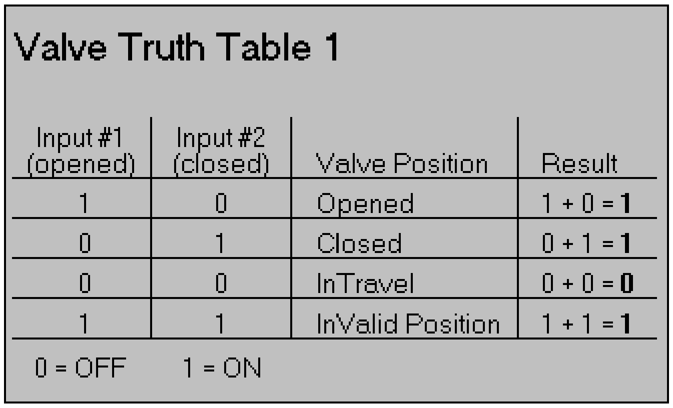
=SIMULATE|PLC1!'STATUS'

Monitoring Multiple Input Device Status

This section describes how you can display the status of an object using multiple inputs.

Example 1

In this example, the status of a spring-return motorized valve using two inputs is being viewed. The two inputs represent a pair of limit switches installed on the valve. One input is only on when the valve is in the open position and off when the valve is in travel or closed. The other input is only on when the valve is in the closed position and off when the valve is in the travel or open position. A truth table of the inputs would appear as follows:

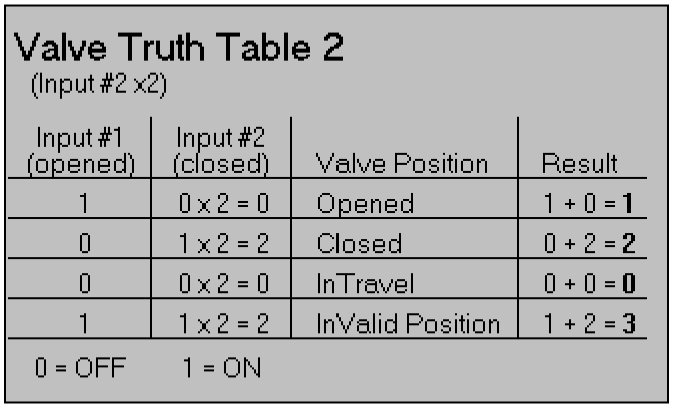


Valve Truth Table 1

Input #1 (opened)	Input #2 (closed)	Valve Position	Result
1	0	Opened	1 + 0 = **1**
0	1	Closed	0 + 1 = **1**
0	0	InTravel	0 + 0 = **0**
1	1	InValid Position	1 + 1 = **1**

0 = OFF 1 = ON

The inputs can be weighed by multiplying the closed input by 2. The results of the valve positions then change to the following:



Valve Truth Table 2
(Input #2 x2)

Input #1 (opened)	Input #2 (closed)	Valve Position	Result
1	0 x 2 = 0	Opened	1 + 0 = **1**
0	1 x 2 = 2	Closed	0 + 2 = **2**
0	0 x 2 = 0	InTravel	0 + 0 = **0**
1	1 x 2 = 2	InValid Position	1 + 2 = **3**

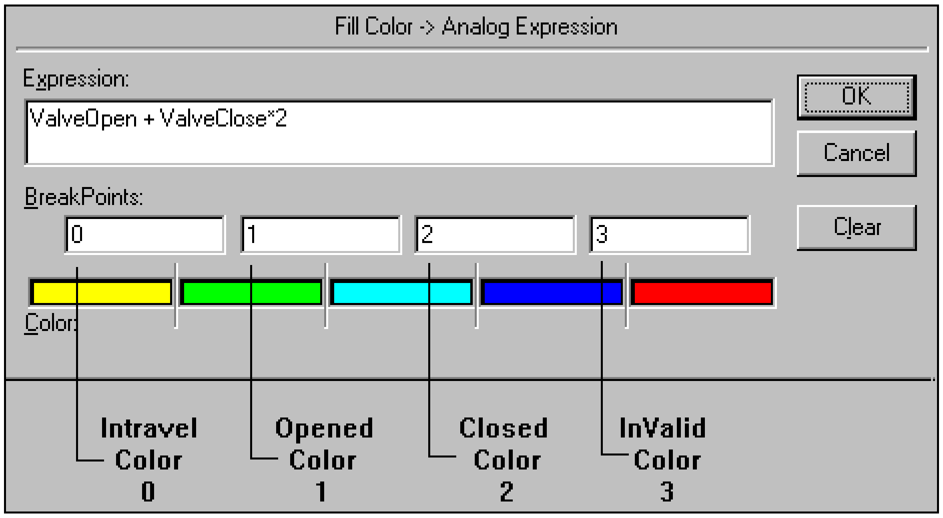
0 = OFF 1 = ON

Note The invalid position can be used to show a defective limit switch.

Now that there is a significant numerical difference between the valve positions, a **Fill Color - Analog** animation link can be used to display the valve status.

For more information on creating animation links, see Chapter 7, "Creating Animation Links."

Two I/O Discrete tagnames are created. One for the valve open input and one for the valve closed input. For example, **ValveOpen** and **ValveClosed**. An object is created to display the valve status. This object is assigned to a **Fill Color - Analog** animation link with the following properties:



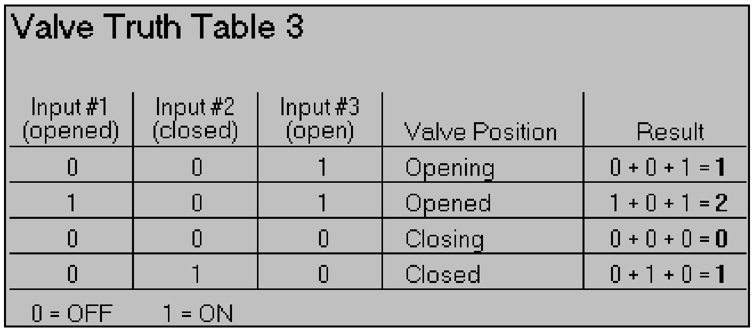
Fill Color -> Analog Expression

Expression: ValveOpen + ValveClose*2

BreakPoints: 0, 1, 2, 3

Example 2

In this example, one more input has been added to the existing two. This new input is the actual output to the valve that causes it to open. The new input will be on when the valve is opening or open, and off when the valve is closing or closed. The new truth table appears as follows:

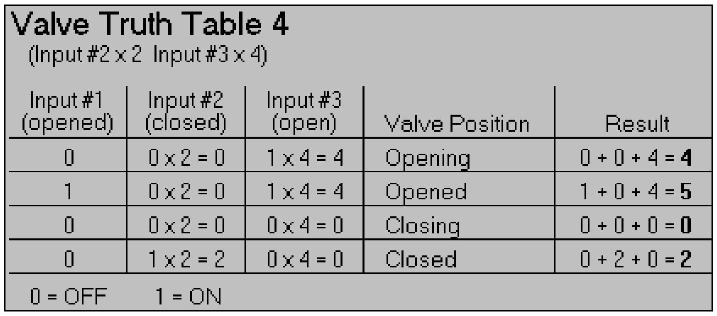


Valve Truth Table 3

Input #1 (opened)	Input #2 (closed)	Input #3 (open)	Valve Position	Result
0	0	1	Opening	0 + 0 + 1 = **1**
1	0	1	Opened	1 + 0 + 1 = **2**
0	0	0	Closing	0 + 0 + 0 = **0**
0	1	0	Closed	0 + 1 + 0 = **1**

0 = OFF 1 = ON

Once again the inputs are weighed. As previously explained, the closed input will be multiplied by 2 and the new input will be multiplied by 4 for the following results:

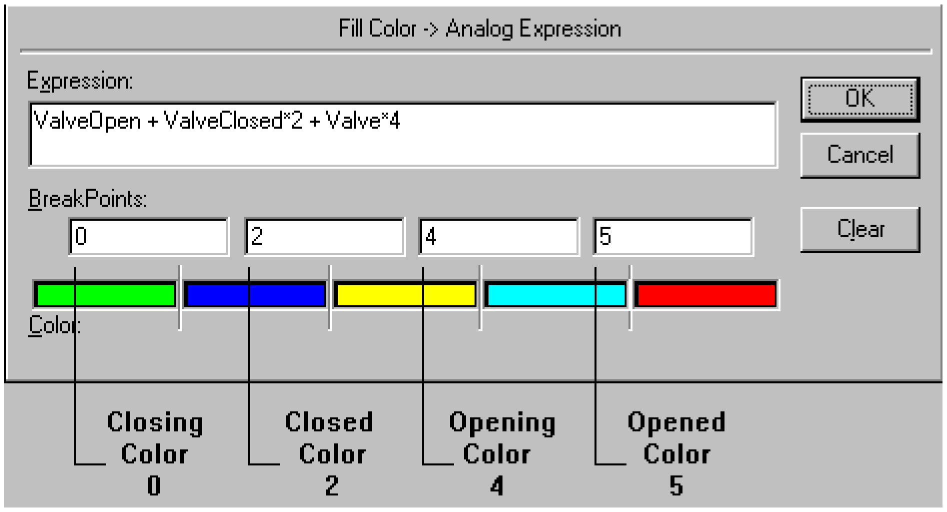


Valve Truth Table 4
(Input #2 x 2 Input #3 x 4)

Input #1 (opened)	Input #2 (closed)	Input #3 (open)	Valve Position	Result
0	0 x 2 = 0	1 x 4 = 4	Opening	0 + 0 + 4 = **4**
1	0 x 2 = 0	1 x 4 = 4	Opened	1 + 0 + 4 = **5**
0	0 x 2 = 0	0 x 4 = 0	Closing	0 + 0 + 0 = **0**
0	1 x 2 = 2	0 x 4 = 0	Closed	0 + 2 + 0 = **2**

0 = OFF 1 = ON

Another I/O Discrete tagname (**Valve**) is created for the new open input and assigned to a **Fill Color - Analog** animation link with the following properties:



Expression: ValveOpen + ValveClosed*2 + Valve*4

BreakPoints: 0, 2, 4, 5

Using this method, additional inputs can be used. The fourth input would be multiplied by 8, the fifth by 16, and so on.

C H A P T E R 1 4

Terminal Services for InTouch

This chapter provides you with an overview of the Terminal Services concept and its features and benefits. It is assumed that you have previous knowledge of how to install and use Terminal Services. For more information please refer to the *Terminal Services for InTouch Deployment Guide*.

Contents

Introduction

Terminal Services is the configurable service included in the Microsoft® Windows® 2000 Server operating system that gives it the capability to run 32-bit Windows-based applications centrally from a server. Terminal Services are fully integrated with the Windows 2000 Server kernel. Terminal Services emulator clients are available for many different desktop platforms (MS-DOS®, Windows, Macintosh, UNIX, Java and others). Non-Windows based desktops require third party add-on software such as Citrix® MetaFrame.

Unlike the traditional client/server environment, when Terminal Services are enabled in Windows 2000 Server, all the application processing occurs on the server. The Terminal Services client performs no local processing of applications, it just displays the application output. The Terminal Services technology transmits only the application presentation—the graphical user interface (GUI)—to the client. Each user logs on and perceives his or her session only, which is transparently managed by the server operating system and is independent from any other client session.

The major benefit to implementing a Terminal Services multi-user, server-centric computing environment is simplification of your system management and your desktop/system deployment. By implementing a server-centric type of computing environment you can significantly reduce your total cost of ownership as well as lower operation costs.

From an application development perspective, one of the biggest benefits of Terminal Services is that well-behaved 16- or 32-bit Windows-based applications run as is—no programming changes are required to run them under Terminal Services. However, this does not mean that all of your existing applications run equally well under Terminal Services. Understanding how to design applications that take advantage of the new capabilities of Windows 2000 Terminal Services is important.

FactorySuite applications that run the application logic on a back-end server while using the client device for data capture or display, are more appropriate than others for deployment in a multi-user server-centric environment.

Using Terminal Services requires a change in the way networks are laid out, and a change in the way users are thought of.

For more information, see *Terminal Services for InTouch Deployment Guide.*

Note Neither Microsoft nor Citrix recommends running any BackOffice product, such as SQL Server, on a Terminal Server except when the user load is very small. As the load increases, the background processes queue while waiting to run, giving the appearance of a lockup.

The adoption of a multi-user approach to computing is not an "either or" decision—multi-user computing can coexist extremely well within a distributed system or client/server oriented information infrastructure. In many application environments it will be necessary to use multiple environments to achieve the best end-user efficiency and productivity.

Windows Terminal Services can be installed on any machine that supports Windows 2000 Server. It requires approximately 14 MB of additional hard disk space to host the client installation files, but otherwise no additional space for the operating system. However, the real requirements are substantially higher for a machine that will be used with Windows Terminal Services in application server mode. Since each user will be executing his or her programs on the server itself, you need to determine exactly how your users work and what their real requirements are. Each installation will be different.

For more information, see the "System Requirements" section of the Preface.

This chapter is intended to provide you with a brief overview of the Windows® 2000 Terminal Services™ concept.

The Terminal Server Concept

The Terminal server is a return to the mainframe concept of computing. All application processing and logic takes place on the Terminal server. As terminals connect to the Terminal server, they open a session and utilize a portion of its resources. Since the Terminal server is providing all the resources, the maintenance and upgrading of the clients is minimized. The majority of the attention is shifted back to the centralized server, as it is in the mainframe computer model. This greatly simplifies maintenance.

Communication between the server and clients is made possible through server/client software such as RDP (Remote Desktop Protocol) and ICA (Independent Computing Architecture). Each terminal needs an installed RDP or ICA client application that passes the input data (keystrokes, mouse movements, and serial communication) to the communication software on the server. This information is processed by the server, and the graphic presentation (screen shot) is returned to the terminal for display

RDP (Remote Desktop Protocol) is the communication protocol whose server-side software is included with Windows 2000 Server Terminal Services. It includes a utility for creating RDP client disks for 16-bit and 32-bit Windows computers to communicate over TCP/IP.

ICA (Independent Computing Architecture) is a third party communication protocol from Citrix. It is used in conjunction with Citrix Device Services or Citrix MetaFrame server-side software. ICA provides greater flexibility than RDP, with client software available for Unix, Linux, OS/2, DOS, Windows CE, PDA, Java, and Macintosh, in addition to Windows 3.11/95/98/NT and Windows 2000. The ICA protocol can be used over IPX, SPX, NetBIOS, and RAS remote access in addition to TCP/IP. The ICA protocol is fully supported by Windows 2000.

Installing the RDP or ICA protocol on a traditional computer allows the traditional computer to become a client that does its processing on the Terminal server and displays the Terminal server desktop on its screen. This is a "fat client" because the computer has an operating system and a hard drive, but connects and runs applications from the Terminal server. The fat client system is often used to extend the life of aging computers. A Windows 3.11 computer can run Windows 2000 by loading a RDP or ICA client and running as a terminal. The disadvantage of fat clients is the possibility of disk failure and the need for continued maintenance of the hard drive.

"Thin Clients" are clients that lack a hard drive and are designed for use with Terminal servers. The machines either have the operating system and client software embedded on a chip or download the operating system and client from the server. These connect to the Terminal server and run their applications there. Thin clients are advantageous because they have a minimal amount of required maintenance and they increase reliability by eliminating the chance of hard drive failure. Thin clients that use the network download of the operating system can have its system updated without replacing chips or components. Upgrading the downloading software on the server will upgrade all those thin clients.

The idea of Total Cost of Ownership (TCO) is simple: Maximize the company's return on investment (ROI) in technology while minimizing the cost involved in doing so. The costs you are attempting to minimize can be broken down into two categories: *hard costs* and *soft costs*. Hard costs are the costs of purchasing and deploying the hardware. Soft costs are associated with end-user support, and training, as well as maintenance of your production environment. The use of thin clients (defined as terminals without a hard drive), terminals, new appliance-like devices, and scaled-down PCs promises to reduce the acquisition *hard cost* of computer hardware, especially at the desktop level, while reducing administrative *soft costs* related to systems management. In other words, by implementing thin client/server-centric computing models, IS managers can significantly lower their Total Cost of Ownership for computer equipment while, at the same time, improve their level of service provided to users.

User Environment

You access Terminal Services from a client by running the Terminal Services Client command on the Windows **Program** menu. When you connect to the Terminal server, the environment on your client machine looks the same as the Windows 2000 server and Windows 2000 Professional environment. The fact that the application is not running locally is completely transparent to the user.

The application opens in the Terminal Services window on the client desktop. All application processing takes place on the server running Terminal Services and the server sends the display to all clients.

The multi-user system environment of Terminal Services consists of three parts:

1. **Terminal Services Server**. The server manages the computing resources for each client session and provides all users who are logged on with their own unique environment. The server receives and processes all keystrokes and mouse actions that the remote client performs and directs all display output for both the operating system and applications to the appropriate client.
2. **Remote Desktop Protocol (RDP)**. RDP supports communication between the client and the server. RDP is optimized to move graphic interface elements to the client. RDP is an application-layer protocol that relies on Transmission Control Protocol/Internet Protocol (TCP/IP) to carry it across the network. RDP is based on the International Telecommunication Union (ITU) T.120 standard for multi-channel conferencing.
3. **Client**. The Terminal Session opens as a window within the local desktop environment. Running within that window is the remote desktop of the Terminal server. The computer or device needs only the minimum amount of software necessary to establish a connection to the server and present the user interface.

Terminal Services Benefits

This section describes the features and benefits of using Terminal Services.

- **Bringing Windows 2000 to desktops faster**. Terminal Services helps bridge the gap while older desktops are migrated to Windows 2000 Professional, providing the Windows 2000 desktop experience "virtually" to non-computer desktops and computers that require hardware upgrades to run a full Windows 2000 operating system locally. Terminal Services clients are available for many different desktop platforms including MS-DOS, Windows-based Terminals, Macintosh, and UNIX. (Connectivity to MS-DOS, Macintosh, and UNIX-based computers requires additional software).
- **Takes full advantage of existing hardware**. Terminal Services extends the model of distributed computing by allowing computers to operate as both thin clients and full-featured personal computers simultaneously. Computers can continue to be used as they have been within existing networks while also functioning as thin clients capable of emulating the Windows 2000 Professional desktop.
- **Remote administration and support**. Terminal Services provides remote administration for the Windows 2000 Server, giving system administrators a method of remotely managing their server from any client over a WAN or dial-up connection. Administrators can remotely manage Windows 2000 servers from a single desktop. Administrators have access to system management tools and can perform all administrative tasks, including software installation, as if they were performing them locally at the server.
- **Replace text-based terminals**. Because many Windows-based terminals also natively support terminal emulation on the same device, you can replace text-based terminals with Windows-based terminals. Windows-based terminals enable you to work with data from mainframe systems to have access to newer Windows-based applications, such as e-mail.
- **Increased Security and Reliability**. Multiple encryption levels enable administrators to encrypt all or some of the data transmitted between the Windows 2000 Server and Terminal Services clients at three different levels (low, medium, or high), depending upon security needs. In addition, the Terminal Services logon process includes change password, unlock desktop, and unlock screen saver features. The logon process is encrypted, ensuring secure transfer of user name and password. Terminal Services supports both 40-bit and 128-bit encryption (available only in the U.S. and Canada) between server and client.

  Because no application or user data ever resides on the client, Terminal Services provides you with more control for security. Terminal Services also provides multi-level encryption support, which you can enable whenever there is a risk of unauthorized transmission interception on the link between the server and the client. There are three levels of encryption available: low, medium and high. All levels of encryption use the standard RSA RC4 encryption.

  The use of thin clients can help prevent the loss of data from computer failure. Since the data is processed and stored on the Terminal server, damage to the client does not lead to destruction of data. This decreases the number of nodes that need to be hardened for data protection.

  For more information, refer to the Microsoft Windows® 2000 Server Administrator's Companion.

- **Remote Control**. The Remote Control feature allows an Administrator to temporarily control another user's session and see the user's actions. The administrator can also interact with the user and execute commands on behalf of the user. To take advantage of Remote Control using RDP, both clients must be connected to a Windows 2000 Terminal Server.

Note Before you enable remote control of a session, you can inform the user by displaying a message on the client machine and requiring client permission to view or take part in the session. The settings of the remote control can be set so that control can take place without user acceptance. The mouse and keyboard inputs can be enabled or turned off.

- **Centralized deployment of programs**. With Terminal Services running on a Windows 2000 Server, all program execution, data processing, and data storage occur on the server, centralizing the deployment of programs. Terminal Services ensures that all clients can access current versions of a program. Software is installed only once on the server, rather than every desktop throughout the organization, reducing the costs associated with updating individual computers.
- **Centralized Management**. Windows Terminal Services provides you with the ability to manage centrally while still allowing the individual user the flexibility of using the Windows desktop environment. Since all applications in a Windows Terminal Services session reside only in one place—on the server—all upgrades of processing power and systems software versions are conducted at the server level rather than at the desktop level, thus eliminating time-consuming field upgrades. There is no client application software to develop, install and update. Therefore, installation and related soft costs are significantly reduced. User profiles can be stored on the Terminal Server so client desktops are administered centrally, thus improving applications management efficiency (centralized backup and restore, revision level upgrades, and so on).

In addition, Windows Terminal Services allows an administrator to view what is happening in a user's session, or even to directly control it. Support personnel can actually see exactly what the user is seeing without leaving their desks. If the user is configured accordingly, the support person can share control of the session, walking the user through a difficult problem. Resulting in lowered maintenance costs because of less time spent trouble-shooting; decreased threat of local viruses, software upgrades are easier and failed platforms can be easily swapped with no loss of data

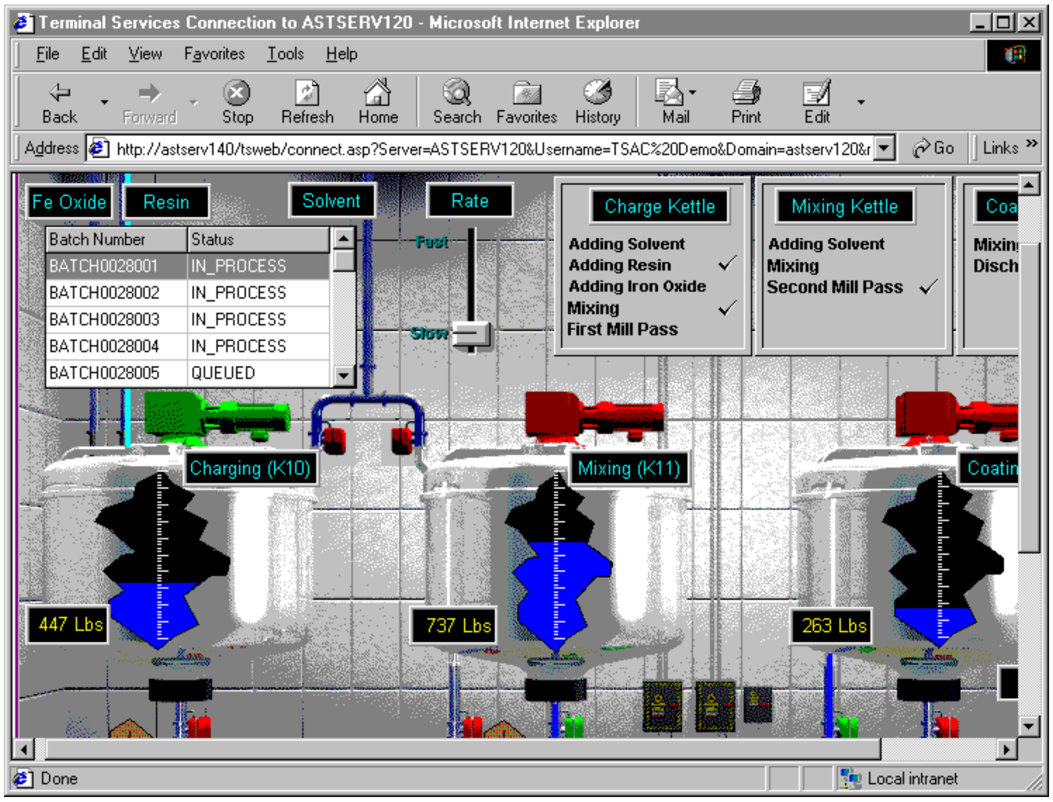
When configured in remote administration mode, Windows Terminal Services can also be used as a management tool. Administrators can log on directly to the server from their desktop and perform normal system maintenance without having to sit at the server console.

By implementing Windows Terminal Services you will centralize three components that traditionally reside at the user's desktop:

1. Application support and maintenance
2. Maintenance of the operating system
3. Storage management

Terminal Services Advanced Client

Microsoft's Terminal Services Advanced Client (TSAC) is a Win32®-based ActiveX® control that can be used to run Terminal Services sessions within Microsoft® Internet Explorer. By using TSAC, you can now run full-featured InTouch applications across the Internet, with the same performance and speed as if you were on the local area network.


Terminal Services Connection to ASTSERV120 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address
Go
Links
Fe Oxide
Resin
Solvent
Rate
Fast
Slow
Charge Kettle
Mixing Kettle
Coa
Charging (K10)
Mixing (K11)
Coatin
447 Lbs
737 Lbs
263 Lbs
Done
Local intranet

http://astserv140/tsweb/connect.asp?Server=ASTSERV120&Username=TSAC%20Demo&Domain=astserv120&r

Batch Number	Status
BATCH0028001	IN_PROCESS
BATCH0028002	IN_PROCESS
BATCH0028003	IN_PROCESS
BATCH0028004	IN_PROCESS
BATCH0028005	QUEUED

Charge Kettle: Adding Solvent; Adding Resin ✓; Adding Iron Oxide; Mixing ✓; First Mill Pass

Mixing Kettle: Adding Solvent; Mixing; Second Mill Pass ✓

Mixin; Disch

Benefits

The downloadable ActiveX control provides almost the same functionality as the full Terminal Services Client, but is designed to deliver this functionality over the Web. The TSAC provides the following benefits:

- **Run sessions within Internet Explorer**. Terminal emulation software does not need to be installed on the client machine. Only Internet Explorer 4 or later and an URL address pointing to the **terminal server** is necessary.
- **Quick and easy access to terminal servers**. The TSAC is especially useful for fact, on-demand access to **terminal server**s. URL addresses can contain optional fields, such as username and screen size, to make accessing different **terminal server**s as simple as clicking on a "Favorites" link.

- **Common interface**. The common look and feel of Internet Explorer make it a preferred GUI for viewing MS Office™ applications, browsing plant information, or doing trend analysis using ActiveFactory™ or SuiteVoyager™.

Installation

The TSAC is a free ActiveX control available from the Microsoft website. It must be installed on a computer running Internet Information Services (IIS) version 4.0 or later. This dependency applies to the Web server only. Users can download the control and view a session from any supported web browser (Windows 32-bit versions of Internet Explorer 4.x or 5.x).

Note For the most recent information or to download the TSAC, visit http://www.microsoft.com/WINDOWS2000/downloads/recommended/TSAC/default.asp

How to Use

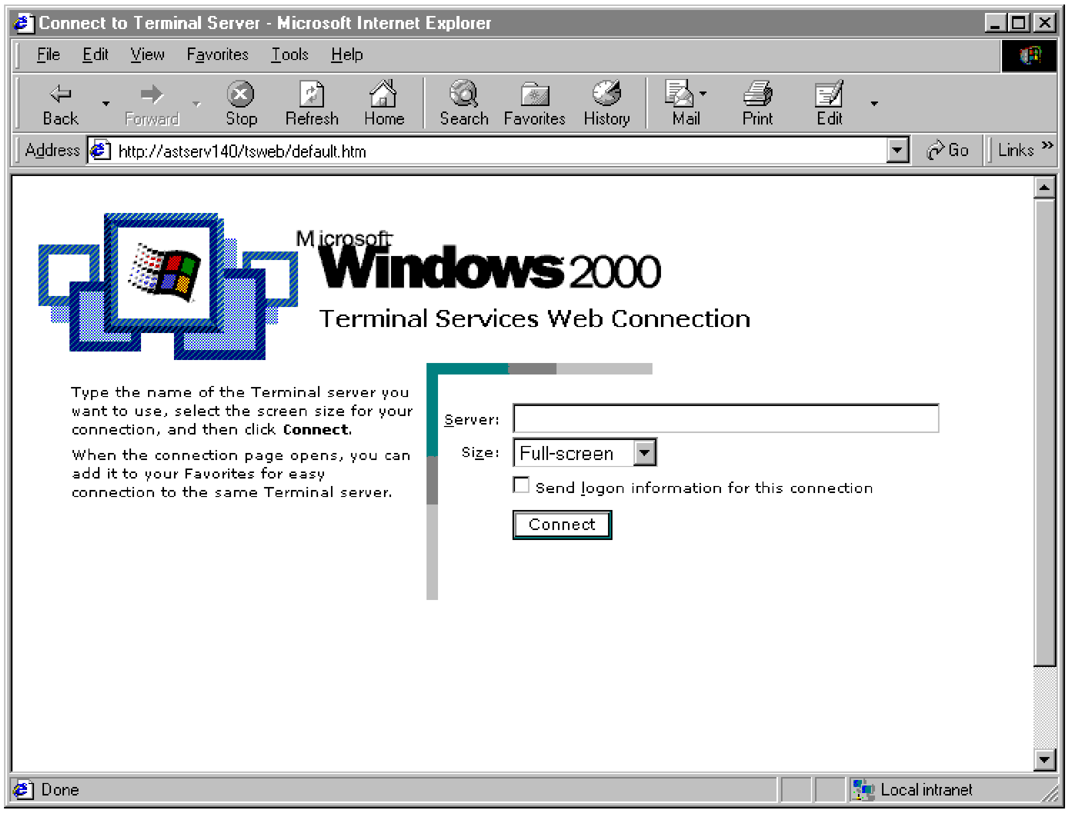
Once the TSAC is installed on the Web server, users can point to a default login page and/or pass specific user information to initiate a **terminal server** session. Three sample Web pages are installed in the TSWeb directory. These pages can be run as they are, or they can be modified.

Note For information on how to configure and use the sample pages, please refer to the Microsoft® Terminal Services ActiveX® Client Control Deployment Guide.

- **Default.htm**. Default.htm is a sample logon page that is designed to collect **terminal server** connection information from the user. You access the default page by the following URL:

  http://MyWebServer/TSWeb/default.htm

  Where `MyWebServer` is the computer name or IP address of the Web server.


Connect to Terminal Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit
Address http://astserv140/tsweb/default.htm
Go Links
Microsoft Windows 2000
Terminal Services Web Connection

Type the name of the Terminal server you want to use, select the screen size for your connection, and then click **Connect**.

When the connection page opens, you can add it to your Favorites for easy connection to the same Terminal server.

Server:
Size: Full-screen
☐ Send logon information for this connection
Connect

- **Connect.asp**. Connect.asp is a sample page that contains the actual ActiveX client control, which hosts the terminal server session. By design, Connect.asp does not run alone, but must be called with the following parameters:

  http://MyWebServer/TSWeb/connect.asp?Server=MyTSServer&Usernam e=MyUser&Domain=MyDomain&rW=80&rH=600

  Where `MyWebServer` is the computer name or IP address of the Web server, `MyTSServer` is the computer name of the **terminal server**, `MyUser` is a valid logon name, and `MyDomain` is the name of the computer that has the logon name defined.

Note To use the sample page, Active Server Pages (ASP) must be enabled on the Web sever.

If your Internet access goes through a *Firewall*, make sure to open TCP port 3389.

Securing Web-based Applications

Beyond the safety and liability issues of remotely controlling a process, the Internet has an increased risk of unauthorized access. The Internet is a public medium, and any connection may inadvertently expose sensitive information and/or damage systems by malicious acts. To adequately protect your **terminal server** and the process it controls, you should develop a sound information security (INFOSEC) policy. Your INFOSEC policy should include Physical Security, Network Security, Application Security, and Security Auditing.

Physical Security

Physical security addresses the operating environment of your servers and connected client systems.

- Place your terminal server in a protected room that is free from physical threat and adverse conditions. Make the room available only to authorized (trusted) personnel.
- Develop a schedule to back-up data and publish procedures on how to restore it.
- Evaluate your risk if the terminal server goes down. Hardware protection such as surge suppressors, uninterruptible power supplies, and redundant servers will help keep your system running. Network Load Balancing or systems with Assured Availability will mitigate the chance that a component failure will stop production.

Network Security

Network security addresses the data transfer between the **terminal server** and client computers.

- Provide adequate computer log-on security.

  For more information, see *Terminal Services for In Touch Deployment Guide*.

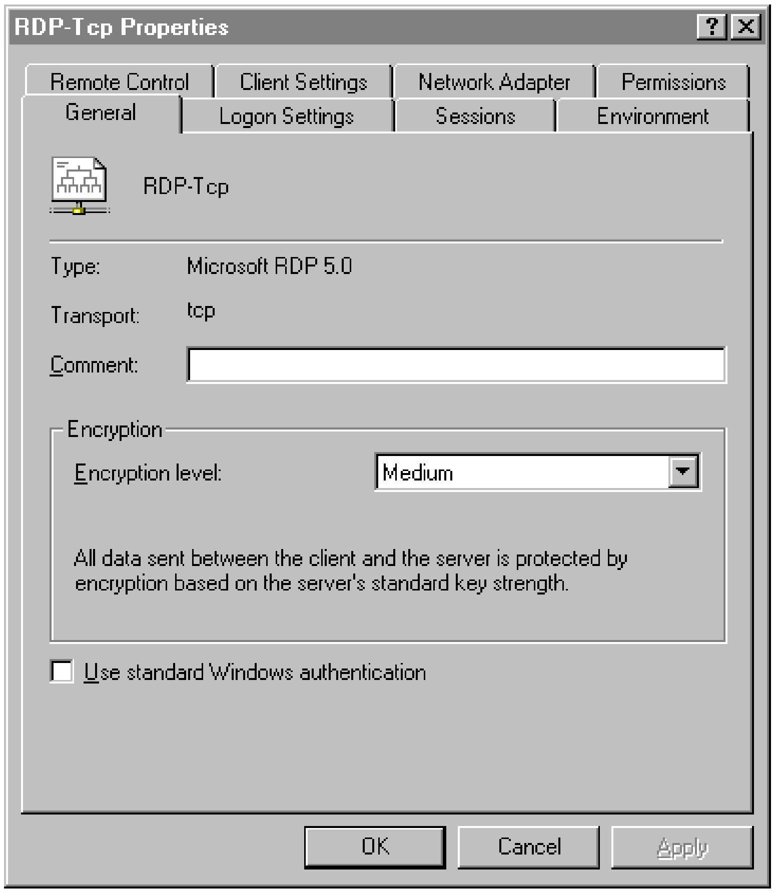
- Enable medium (or higher) encryption. Encryption prevents spoofing, which refers to any unauthorized attempts to intercept an address, user identification, partial or even total transmission of data. Terminal Services provides multilevel encryption. All levels use the standard RSA RC4 encryption model.

Level	Description
Low	This level secures all data sent from the client to the server by using either a 56-bit or 40-bit key. A Windows 2000 **terminal server** uses a 56-bit key when Windows 2000 clients connect to it, and a 40-bit key when earlier versions of the client connect. This input-only encryption is used to protect sensitive data, such as a user password.
Medium (default)	This level secures data sent in both directions (from the client to the server and from the server to the client) by using either a 56-bit or a 40-bit key. A Windows 2000 **terminal server** uses a 56-bit key when Windows 2000 clients connect to it, and a 40-bit key when earlier versions of the client connect. Use medium encryption to secure sensitive data as it travels over the network to display on remote clients.
High	High encryption affects all data sent in both directions, but encrypts using the 128-bit key.

To enable encryption

1. Click **Start** on the Windows Taskbar, point to **Programs**, point to **Administrative Tools**, and then click **Terminal Services Configuration**.

2. Double-click **RDP-Tcp**. The **RDP-Tcp Properties** dialog box appears.



RDP-Tcp

Type: Microsoft RDP 5.0

Transport: tcp

Comment:

Encryption

Encryption level: Medium

All data sent between the client and the server is protected by encryption based on the server's standard key strength.

☐ Use standard Windows authentication

3. Click the **General** tab to activate the **General** property sheet.
4. Select the appropriate **Encryption level**.

Application Security

Application Security addresses the security embedded in your InTouch application, Industrial *SQL Server*, and other sensitive information systems.

- Use the $Operator tagname to provide security within the InTouch application. By applying security to your application, you can control specific functions that an operator is allowed to perform by linking those functions to internal tagnames.

  For more information on the $Operator tagname, see the "Using InTouch Security" section in Chapter 2 of your online *InTouch User's Guide*.

- Replace the GetNodeName() QuickScript with the new TseGetClientId() QuickScript to identify the client computer. When using Terminal Services, GetNodeName() returns the name of the terminal server, not the name of the client computer.

- Add a password to the SQL Server system administrator (SA) account. When you install Industrial SQL Server, an all-powerful "sa" login ID is created with an empty password. Do not use this account to access data. Use the default login IDs (for example, wwUser), instead.

  For more information on database security, refer to the "Managing Security" chapter in the *Industrial SQL Server Administrator's Guide*.

Security Auditing

Security Auditing addresses the ability for you to monitor intrusion attempts. If you suspect that your system is under any sort of attack, then you can enable logging for an array of auditable events. By default, security logging/auditing is disabled because it usually requires excessive processing resources. We, therefore, recommend that you initially select only a few events to monitor.

Caution! Security Auditing requires significant resources. Make sure to enable auditing when you evaluate your pilot server, or you may undersize the hardware.

To configure auditing, refer to the Audit Policy, which is part of the Windows 2000 Local Security Policy.

Additional Information

For further exploration of these and related security considerations, please refer to the following resources:

National Computer Security Center (NCSC) "Rainbow Books" – http://www.radium.ncsc.mil/tpep/library/rainbow/index.html

Common Criteria (CC) for Information Technology Security Evaluation – http://www.commoncriteria.org/cc.html

Microsoft Privacy & Security Fundamentals: Security – http://www.microsoft.com/privacy/safeinternet/security/best_practices/default.htm

Windows 2000 Security Technical Overview – http://www.microsoft.com/technet/win2000/win2ksrv/sectech.asp

Default Access Control Settings in Windows 2000 – http://www.microsoft.com/technet/win2000/win2ksrv/technote/secdefs.asp

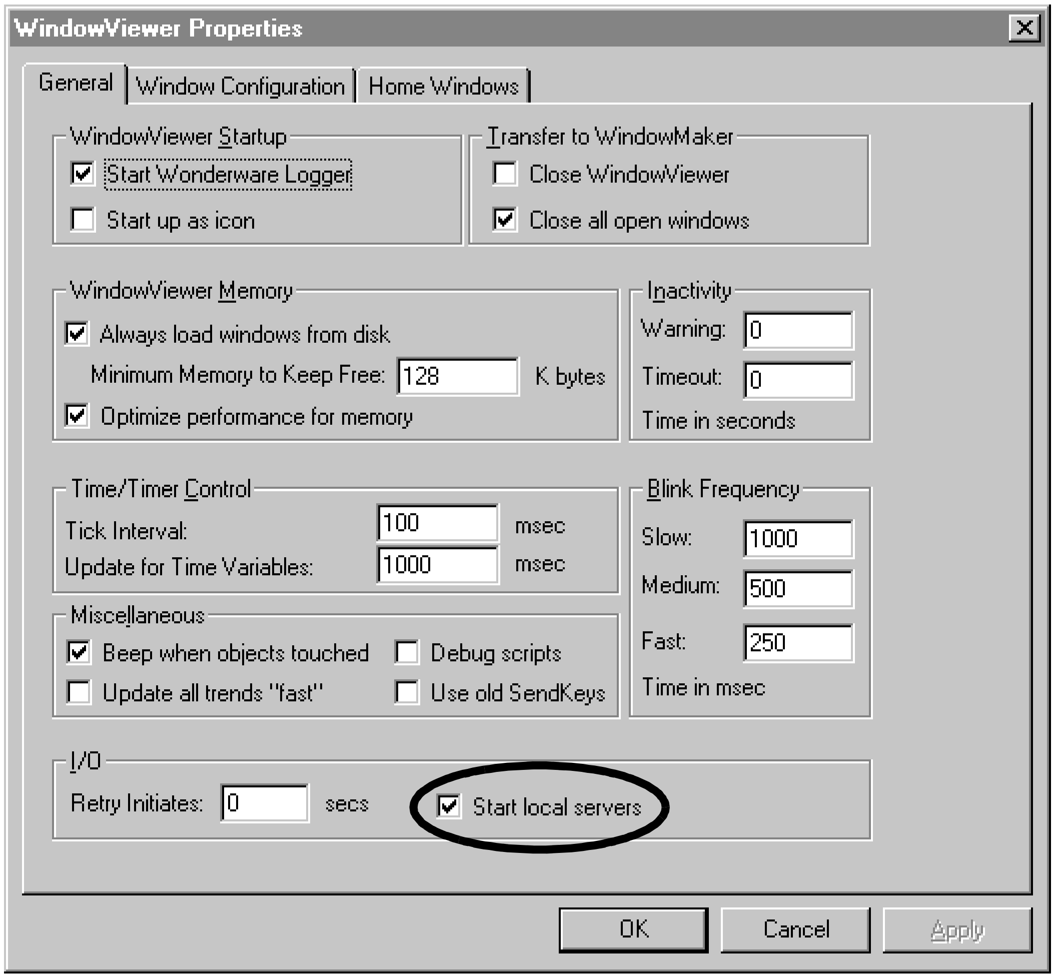
Known Issues and Limitations

The following table describes the limitations and suggested work-around you may need to implement when running applications on a **terminal server**. Wonderware is aggressively resolving many of these issues in the next release.

Feature	Supported?	Comment
AlarmSuite Logger	No	Use a tagname server (separate computer only)
DDE to an I/O Device or MS Office 2000 (for example, Excel)	No	Use a tagname server (console or separate computer). This includes DDE QuickScripts: **WWExecute()**, **WWpoke()** and **WWRequest()**
DDE from MS Office 2000 (for example, Hot-link configured in Excel)	Yes	Excel and InTouch must be running in the same session
Historical Trending	Yes	Use a tagname server or NAD to log values. Multiple sessions may read the same historical files
InBatch	No	In work - to be supported in a future release
InControl	Yes	InControl must run on the console, and requires v7.1 Patch 03 or later.
InSQL ActiveX Controls (OLE DB)	Yes	InSQL should be on a separate computer
InTouch Alarm Logger	No	In work - to be supported in a future release
InTrack OLE Automation	No	In work - to be supported in a future release
"Playsound" QuickScript	Yes	Requires Citrix MetaFrame
Printing Alarms	No	In work to be supported. Parallel printers may be configured to print from the client using the net use command. Refer to Tech Note 149 for more information
Retentive tags	Yes	Must use NAD
SPC Pro	No	Not supported
SQL Access (ODBC)	Yes	Database should be on a separate computer
SuiteLink to an I/O Device or another InTouch application.	Yes	When communicating to another view session, include the **terminal server** Node name and append the IP address of the desired session to the Application name. For example, view10.103.25.6.

Starting Local I/O

InTouch cannot start I/O servers in a **terminal server** environment. To avoid receiving an "Initializing I/O" error message when **WindowViewer** starts, turn off the **Start Local Servers** option on the **WindowViewer General** property page.



WindowViewer Properties

General | Window Configuration | Home Windows

WindowViewer Startup
- [✓] Start Wonderware Logger
- [ ] Start up as icon

Transfer to WindowMaker
- [ ] Close WindowViewer
- [✓] Close all open windows

WindowViewer Memory
- [✓] Always load windows from disk
- Minimum Memory to Keep Free: 128 K bytes
- [✓] Optimize performance for memory

Inactivity
- Warning: 0
- Timeout: 0
- Time in seconds

Time/Timer Control
- Tick Interval: 100 msec
- Update for Time Variables: 1000 msec

Blink Frequency
- Slow: 1000
- Medium: 500
- Fast: 250
- Time in msec

Miscellaneous
- [✓] Beep when objects touched
- [ ] Debug scripts
- [ ] Update all trends "fast"
- [ ] Use old SendKeys

I/O
- Retry Initiates: 0 secs
- [✓] Start local servers

OK | Cancel | Apply

Note Depending on the sequence that view sessions start, you may need to execute the **IOReinitialize** QuickScript. Remember ALL servers (I/O devices or view applications) must be running before starting an application that reads values from these servers.

Script Execution

Because all applications running on a **terminal server** use a single timing reference (server clock), there is a chance that scripts may not execute during abnormal CPU loading. Abnormal CPU loading can be caused by excessive video processing or when several applications have the same script triggers defined (such as an End-of-Shift event). It is possible, therefore, that if the server is busy processing scripts from many clients, it may not start a script on another client during the interval when the timer would normally start the script. This may cause the script on the client to not execute.

To ensure proper script execution, combine scripts with common triggers and move them to a single application, such as a tagname server. This is one of the primary reasons for pilot deployment. Pilot deployment gives you an opportunity to perform "what-if" scenarios and determine if your hardware selection is adequate.

Server Hardware Requirements

Terminal Services for InTouch must be installed on a new **Windows 2000 Server** or **Advanced Server**. Do not upgrade from a Windows NT system. The following table provides recommended hardware platforms. They should give you good performance with a representative InTouch application.

Recommended Hardware

CPU [1]	PhysicalMemory [2, 3]	Virtual Memory [4]	Number of Clients
Pentium III 450 MHz	384 MB	960 MB	5
Pentium III 500 MHz	1024 MB	2560 MB	15
Pentium III 700 MHz	2048 MB	5120 MB	25

1. Multi-processors can improve performance.
2. Add 128 MB RAM for Windows 2000 Advanced Server.
3. Memory requirements depend on application load and the number of users connected. RDP will need 40-60 MB per user running InTouch, while ICA will require slightly more.
4. Virtual memory (page file size) should be 250% of the physical memory.

Note A good way to estimate how many users a server can support is to measure system performance with two to five users on the system, and then scale the results.

For more information on analyzing system performance, see *Terminal Services for In Touch Deployment Guide*.

Hard Disk Space

One or more hard disks with a minimum of 2 GB on the partition that will contain the system files.

The use of RAID, Redundant Array of Inexpensive Disks, will help prevent loss of data and server downtime.

Networking

10/100Mbps network adapter card. Network that uses the TCP/IP protocol.

Other Drives

CD-ROM drive

A high-density 3.5 inch disk drive as drive A, unless:

- The computer supports starting the Setup program from a compact disc.

- You are installing Windows 2000 over a network.

Accessories

Keyboard, mouse (or other pointing device) and a monitor (VGA or better).

A UPS (Uninterrupted Power Supply).

Note Before you install Windows 2000, verify that your hardware is on the Windows 2000 Hardware Compatibility List (HCL). Because Microsoft provides tested drivers for only those devices that are listed on the Windows 2000 HCL, using hardware that is not listed on the HCL may cause problems during and after installation. You can find the most recent version of the HCL on the Internet at http://www.microsoft.com/hwtest/hcl.

Examining Peripheral Devices that Affect Performance

Peripheral devices can also affect the performance of a server running Terminal Services:

- Hard disks. Disk speed is critical for terminal server performance. Small computer system interface (SCSI) disk drives, especially devices compatible with Fast SCSI and SCSI-2, have significantly better throughput than other types of drives.
- Network adapter. A high-performance network adapter is recommended, especially if users require access to data that is stored on network servers or client/server applications such as Wonderware InTouch. Using multiple adapters can significantly increase network throughput.

RDP Client Hardware Requirements

Clients that run Terminal Services are not required to have much processing power. For example, a Pentium with 32 MB of RAM and a VGA video card is sufficient. Therefore, it is very easy to integrate Terminal Services into a network that has older computers and equipment.

Note Using a standard VGA card may limit your display size and color depth.

There will be some performance considerations depending on the model of the client you are using. For example, a device that uses Microsoft Windows CE as its operating system will not operate as quickly as the same device would if it used Linux.

Planning Your Terminal Services Installation

The key to successful Terminal Services installation is proper planning. Perform the following tasks before you install Terminal Services:

- Identify the client applications (for example, Wonderware InTouch) that you need to install on the server.

- Identify the hardware requirements for your clients.
- Determine the server configuration that is required to support clients.
- Identify the licenses that are required for Terminal Services as well as all other applications that you will be running.

Before you install Terminal Services, identify the applications that you intend to deploy on your client desktops. Most applications that run properly on Windows 2000 run on a Terminal server. Some applications, however, may require minor modification to run successfully in a Terminal Services environment. Install applications on a test server before you deploy these applications in your production environment to ensure compatibility with your existing applications.

For more information, see the *Terminal Services for InTouch Deployment Guide*.

Installing Terminal Services

You can install Terminal Services on the server during Windows 2000 Server Setup, or you can install Terminal Services after Setup through **Add/Remove Programs** in the Control Panel.

Important! Terminal Services must be installed before you install any software product. Software products must be added in the "Install Mode" to a terminal Server. Using **Add/Remove Programs** in the Windows Control Panel automatically activates the Install Mode. This will allow you to configure the software for all users on the machine instead of limiting it to the user who installed the program.

Terminal Services Components

The Terminal Services installation includes the following components and options:

- **Terminal Services**. Provides the multiple session environment for clients to access Windows-based applications on the server. Terminal Services includes two sub-components:
- **Client Creator Files**. Contains a wizard for creating installation disks for Terminal Services clients.
- **Enable Terminal Services**. Enables the Terminal Services software on the server. You can use this option to start and stop Terminal Services.
- **Terminal Services Licensing**. Configures the computer as a Terminal Services license server that provides client licenses.

To install Terminal Services

1. Click **Start** on the Windows Taskbar, point to **Settings**, and then click **Control Panel**.
2. Double-click the **Add/Remove Programs** icon. The **Add/Remove Programs** dialog box appears.

3. Click **Add/Remove Windows Components**. The **Windows Components Wizard** appears.
4. Select the **Terminal Services** and **Terminal Services Licensing** options, and then click **Next**. (Terminal Services licensing may be installed on a separate machine.)
5. Select the **Application server mode** option, and then click **Next**.
6. Select the **Permissions compatible with Windows 2000 Users** option, and then click **Next**.
7. Select the **Your domain or workgroup** option and provide the directory location for the licensing server database

   Note This option only appears if you selected the **Terminal Services Licensing** option.

8. Click **Next** to begin the installation.

Once installation is complete, several items are added to your **Administrative Tools** menu. The following table describes these additions:

Item	Description
Terminal Services Client Creator	Creates floppy disks for installing Terminal Services client software.
Terminal Services Configuration	Manages Terminal Services protocol configuration and server settings. Only one RDP connection can be configured per network adapter.
Terminal Services Licensing	Manages client access licenses for Terminal Services.
Terminal Services Manager	Manages and monitors sessions and processes on the server running Terminal Services.

Installing Terminal Services for InTouch

To install the Terminal Services for InTouch, you must log on to the server running Terminal Services by using the built-in Administrator account.

Note If you have already installed a Non Terminal Services version of InTouch, It is not possible to upgrade to a Terminal Services version of InTouch. It is, however, possible to upgrade from a previously installed Terminal Services version of InTouch to the latest version.

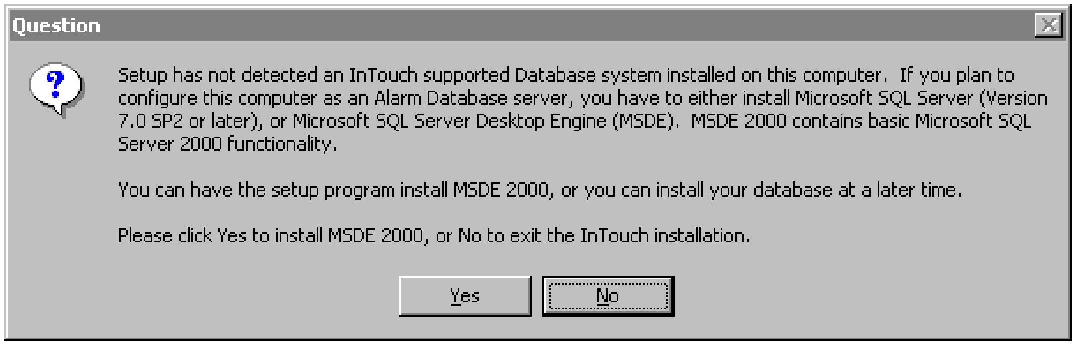
To install Wonderware Terminal Services for InTouch

1. Insert the Wonderware Terminal Services for InTouch compact disc into your CD-ROM drive, and wait for the auto-run to activate the InTouch **SETUP.EXE** program automatically.
2. The **Welcome** dialog box appears. Click **Next**.

3. Follow the on-screen instructions to install InTouch.
4. Once the System Files and Common components are installed you will be prompted to restart your system. Click **OK** to reboot the computer. Once your system is restarted, InTouch installation will resume automatically.

Note If there is an existing Terminal Services for InTouch installed on this system, you will not be prompted to restart your system as mentioned in step 4 above.

5. After InTouch is successfully installed, a dialog box appears telling you that installation is complete.
6. Also, after InTouch is successfully installed, the following **Question** dialog box may appear:



Setup has not detected an InTouch supported Database system installed on this computer. If you plan to configure this computer as an Alarm Database server, you have to either install Microsoft SQL Server (Version 7.0 SP2 or later), or Microsoft SQL Server Desktop Engine (MSDE). MSDE 2000 contains basic Microsoft SQL Server 2000 functionality.

You can have the setup program install MSDE 2000, or you can install your database at a later time.

Please click Yes to install MSDE 2000, or No to exit the InTouch installation.

Click **Yes** to proceed with the MSDE 2000 installation, or click **No** to exit the InTouch installation.

Note Microsoft recommends installing SQL Server/MSDE on a node other than the Terminal Server node. Please see the InTouch Terminal Services Deployment Guide or the InTouch Alarm Deployment Guide for more information on how to setup the SQL Server/MSDE node.

When installing InTouch, you have three options: **Full Development System**, **Runtime Only**, or **Factory Focus**. This option determines how the **Terminal Server** and the corresponding clients will run InTouch. You cannot mix and match InTouch configurations. For example, if you want some clients to run Runtime and other clients to run Factory Focus, then you will need two **Terminal Servers**.

7. Install the appropriate Wonderware License.

   You are now ready to use Terminal Services for InTouch.

Testing Your Applications in a Terminal Environment

To ensure that your application installs and works properly in a Terminal Services environment, it is critical that you test it in that environment. It is important to construct a typical usage scenario with the appropriate number of sessions running the application for a time period that simulates actual usage.

Terminal Services for InTouch QuickScript Functions

This section describes the three InTouch QuickScript functions available for Terminal Services.

TseGetClientId()

Returns a string version of the client ID (the TCP/IP address of the client) if the View application is running on a Terminal server client. Otherwise it returns an empty string.

Syntax *MessageResult*=`TseGetClientId();`

Parameter	Description
MessageResult	A Message tagname that displays the client ID.

Remarks This ID is used internally to generate SuiteLink server names and logger file names.

Example `MsgTag=TseGetClientID();`
Client IP Address, for example, **10.103.202.1** would be returned to **MsgTag**.

TseQueryRunningOnConsole()

Returns a non-zero integer value if the View application is running on a Terminal server console. Otherwise it returns a value of zero (0).

Syntax *Result*=`TseQueryRunningOnConsole();`

Parameter	Description
Result	Returns 0 if View is not running on a TS console.

Remarks None

Example `IntTag=TseQueryRunningOnConsole();`
Returns IntTag = 1 if WindowViewer is running on a Terminal Server console.

TseQueryRunningOnClient()

Returns a non-zero integer value if the View application is running on a Terminal server client. Otherwise it returns a value of zero (0).

Syntax *Result*=`TseQueryRunningOnClient();`

Parameter	Description
Result	Returns 0 if View is not running on a TS client.

Remarks None

Example `IntTag=TseQueryRunningOnClient();`
Returns IntTag = 1 if WindowViewer is running on a Terminal Server client.

Establishing a Terminal Session

After you have completed the server and client configuration, you can establish a Terminal session. You can access the network and local resources, including the hard disks and printers, from the client. When ending a session, you can either disconnect to rejoin the session later or log off to close the session completely.

Connecting to a Terminal Server

When you connect to a Terminal server, you can select options to accommodate slow networks and improve the performance of your session.

To connect to a Terminal server

1. Start Terminal Services Client. The **Terminal Services Client** dialog box appears, which contains the following options:

For this option	Do this
Server	Enter the name of a Terminal server or a TCP/IP address
Screen area	Select a screen resolution. This setting is not dependent on the screen resolution of the server.
Available servers	Select a server from a list of available servers.
Low-speed connection	Click Low-speed connection if using a modem or for a slow network.
Cache bitmaps to disk	Click Cache bitmaps to disk to save desktop display elements to the local cache. This option will cause the screen to refresh from the local cache and improve performance

2. Click **Connect**.
3. Log on to the Terminal server.

Ending a Terminal Session

Terminal Services provides two options for you to end a Terminal session:

- Disconnecting from a session. Disconnecting leaves the session running on the server. You can reconnect to the server and resume the session. For example, if you are performing a time-consuming task on the server, such as running a query on a database, you start the task and disconnect from the session. Later, you can log on to the server again, resume the session and either resume the task or check results.
- Logging off from a session. Logging off from a session ends the session running on the server. Any applications running within the session will be closed and unsaved data is lost. It is important for you to log off from a session to make server resources available for new sessions.

Other Application Issues

Some applications have features that may prevent them from working with Terminal Services or cause them to perform poorly. For example, applications that require special hardware such as bar code scanners or smart card readers can be used with a Terminal Services client only if:

- The devices are connected to the computer or terminal in such a way that the peripheral device is recognized as a keyboard-type device.

The connecting software and hardware support the connection to the client.

Configuring Client Settings

To ensure that system resources are available for active Terminal sessions, you can set time limits for disconnected and idle sessions.

You specify time limits for sessions on the **Sessions** property sheet in the **RDP-Tcp Properties** dialog box in Terminal Services Configuration on the server. The following table describes the settings for limiting the length of a session.

Setting	Description
End a disconnected session	Specifies the maximum duration that a disconnected session is retained. The session will be reset and can no longer be resumed after the time limit has expired.
Active session limit	Specifies the maximum connection duration. When the time limit is reached, the session will be disconnected, leaving the session active on the server or reset.
Idle session limit	Specifies the maximum idle time (time without connection activity) allowed before the session is disconnected or reset.

C H A P T E R 1 5

InTouch Application Publisher

This chapter provides you with an overview of the InTouch Application Publisher that you will use to create a self-extracting file that contains all relevant files and setup procedures that you need to minimize download time over the Internet.

Contents

Publishing an InTouch Application

The Application Publisher creates a self-extracting file that contains all relevant files and setup procedures that you need to install the application onto another InTouch node. The Application Publisher compresses the Application to minimize download time over the Internet.

Note You can copy these compressed applications to floppy disks, zip disks, and so on, for installation on other nodes.

The Application Publisher's compression rate is approximately 19 to 1. This rate allows a 20 MB, full development application to fit on a standard floppy disk. If the file is runtime only, a 30 to 40 MB application may fit on a single floppy disk.

Once you publish an application, you re-constitute as an InTouch application on one or more other InTouch 7.1 (or later) node(s).

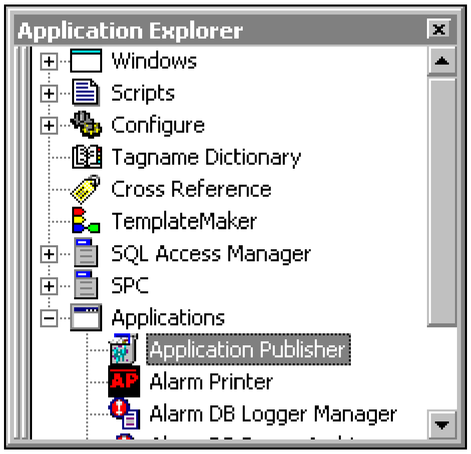
You can use several methods to load the published application file onto another InTouch node:

- Copy to floppy disk
- Copy the file over your company Intranet
- Copy the file over the Internet

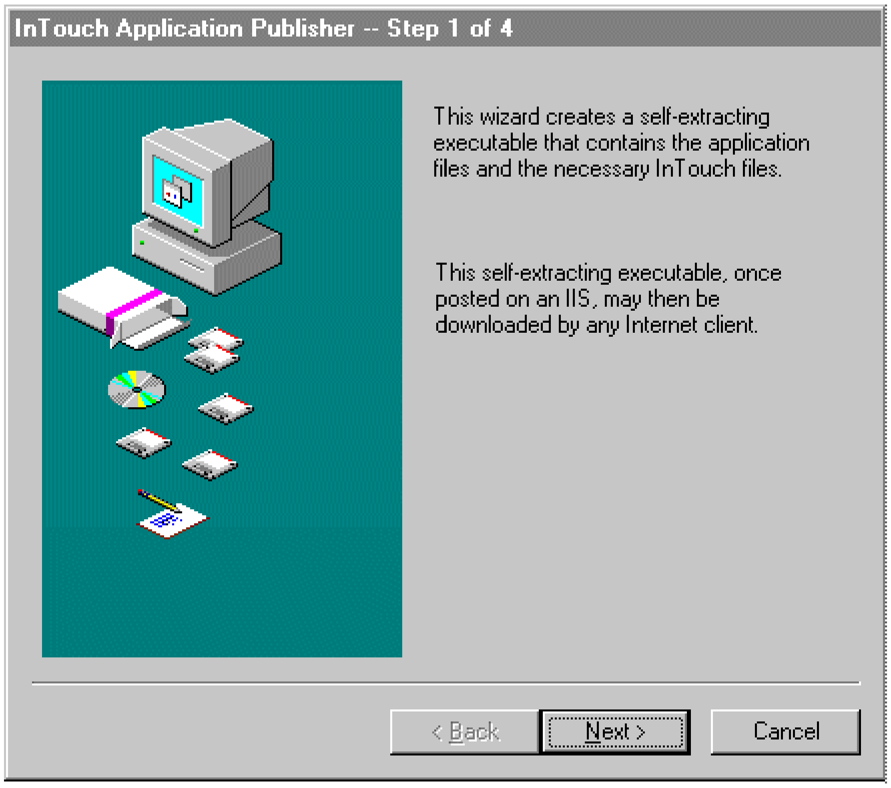
- A re-constituted InTouch application can be used by running the self-extracting file as a program on the target node.

To publish an InTouch application

1. In WindowMaker, from the Application Explorer, open **Applications**.



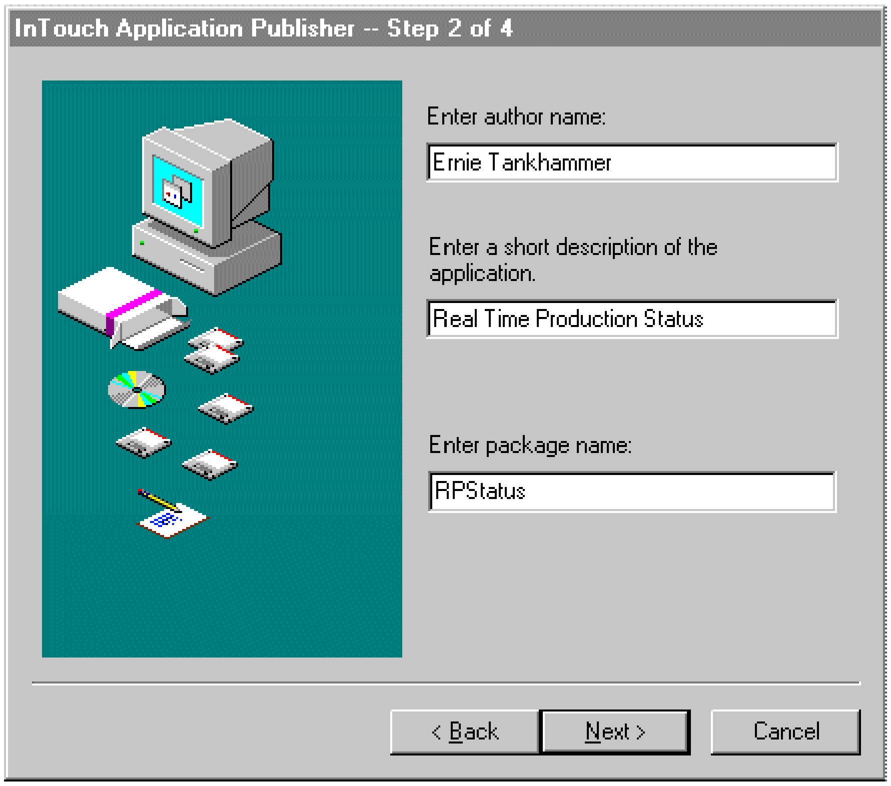
2. Select **Application Publisher**. The **InTouch Application Publisher – Step 1 of 4** dialog box appears.



This wizard creates a self-extracting executable that contains the application files and the necessary InTouch files.

This self-extracting executable, once posted on an IIS, may then be downloaded by any Internet client.

3. Click **Next**. The **InTouch Application Publisher – Step 2 of 4** dialog box appears.



InTouch Application Publisher -- Step 2 of 4

Enter author name: Ernie Tankhammer

Enter a short description of the application. Real Time Production Status

Enter package name: RPStatus

4. In the **Author Name** box, type the name of the person(s) (up to 256 characters) to contact regarding the application. (This is an optional entry.) For example:

   Ernie Tankhammer

   Ernie Tankhammer Senior Engineer Phone 1-800-555-1234

5. In the **Description** box, type a description for the application (up to 256 characters). (This is an optional entry.) For example:

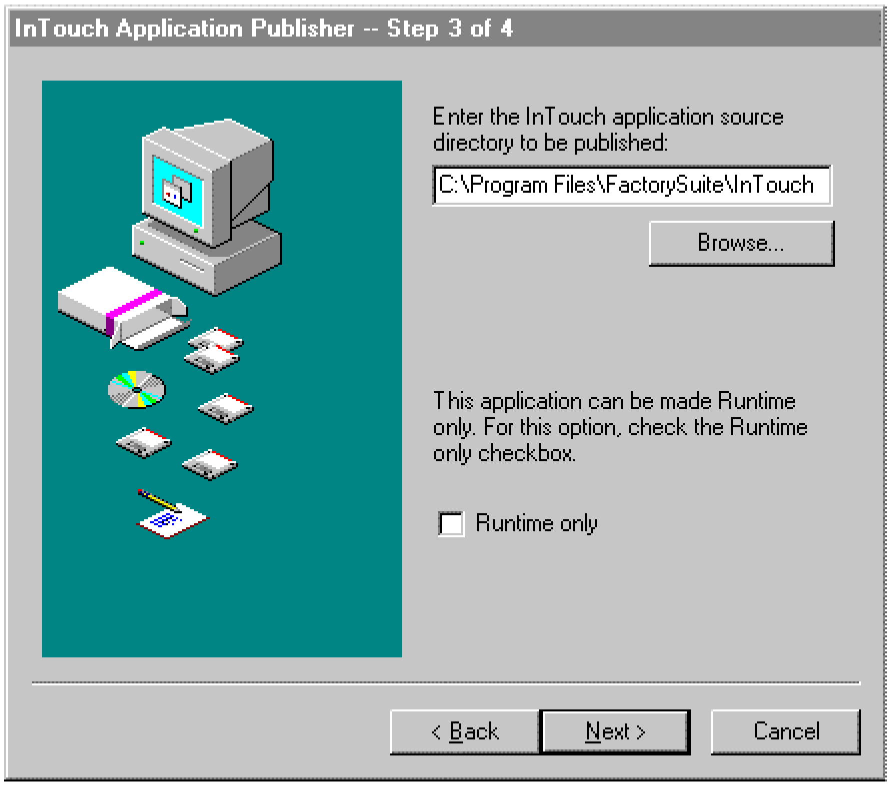
   Real Time Production Status

   Plant Number 5-Area 3 that requires access to zone 3 FactorySuite File Server for data point access.

6. In the **Package Name** box type a name for the package (up to 32 standard Windows file naming characters.). You **must** type a **Package Name**.

Note The **Package Name** is a unique name that identifies a published application. It is used to maintain simultaneous downloadable versions of the same published application.

7. Click **Next**. The **InTouch Application Publisher – Step 3 of 4** dialog box appears.


InTouch Application Publisher -- Step 3 of 4

Enter the InTouch application source directory to be published:

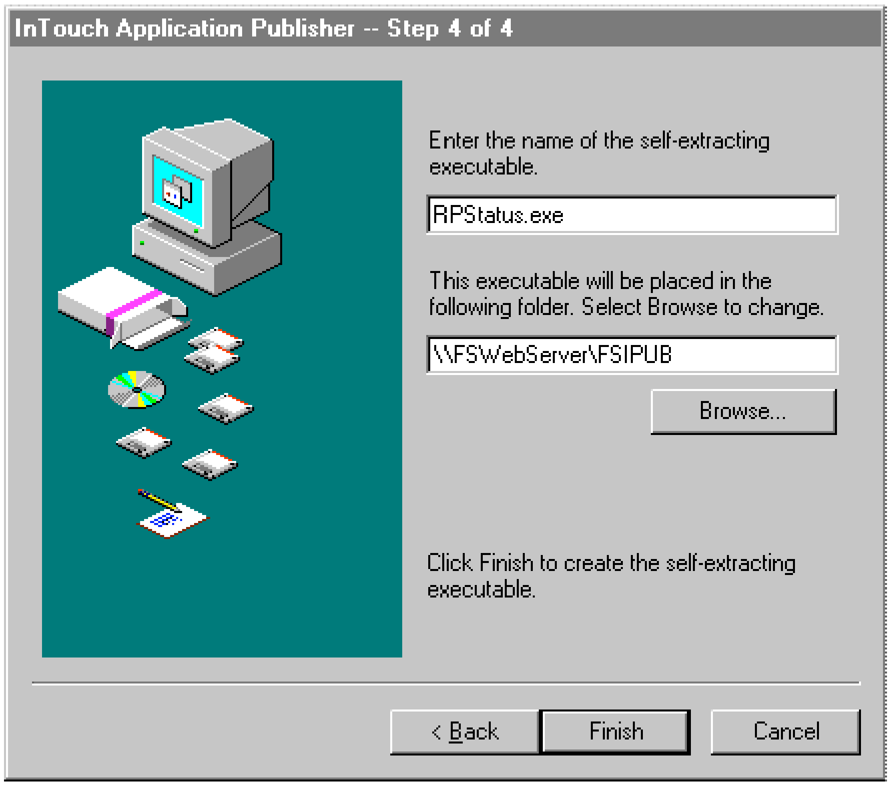
C:\Program Files\FactorySuite\InTouch

This application can be made Runtime only. For this option, check the Runtime only checkbox.

☐ Runtime only

8. In the entry box, type the path to the InTouch application source. The default is the WindowMaker application directory.
9. Select the **Runtime only** option to prevent the inclusion of development WindowMaker files in the published file.

10. Click **Next**. The **InTouch Application Publisher – Step 4 of 4** dialog box appears.



Enter the name of the self-extracting executable.

RPStatus.exe

This executable will be placed in the following folder. Select Browse to change.

\\FSWebServer\FSIPUB

Click Finish to create the self-extracting executable.

11. Verify that the executable name in the first box is correct. (By default, this will be the same as the **Package Name** you specified.)
12. In the second box, type the path to the directory where you want to save the published file. (By default, this will be saved in your current Temp directory.) Or, click **Browse** to locate a different directory.

Note In the example above, the file will be posted to the FactorySuite Web Server on the company intranet.

13. Click **Finish**.

Publishing Applications with Multiple Resolutions

You can publish your applications in multiple resolutions. This will eliminate a client's inability to view your application at the proper resolution.

To publish an application in a different resolution

1. While running WindowMaker, change the screen resolution on your PC.
2. WindowMaker will notify you that your resolution has changed and prompt you to accept the change, click **OK**.

3. From the Application Explorer, open **Applications** and then, run the Application Publisher.
4. Change the **Package Name** for the application. For example, change **Dairy** to **Dairy_2**.

Note If you want to keep the same **Package Name** and just change the file name (by default, these are the same), change the file name in the **InTouch Application Publisher – Step 4 of 4** dialog box.

The file name **must** be changed to post multiple resolutions of the same application. If the file name is the same, it will only overwrite the initial application.

When posted to the web server it appears as:

Description	Dairy Processing Application
Publisher	Navin Johnson
File Name…………	Dairy.exe / Video Resolution…(1024x768)

Description	Dairy Processing Application
Publisher	Navin Johnson
File Name…………	Dairy_2.exe / Video Resolution…(800x600)

A P P E N D I X A

Overview of the InTouch Windows NT Services

A service is a process in Windows NT that performs a specific unattended background system function without any user interface. You can configure services to automatically startup when the computer on which the application is installed starts up. Services are also started without compromising Windows NT's security system. This Appendix discusses the Windows NT services for InTouch (7.1 or later).

InTouch has four Windows NT services:

InTouch Service	Windows NT Service Default Location
Wonderware Logger	C:\Program files\Factorysuite\Common\Wwlogsvc.exe
Wonderware SuiteLink	C:\Program files\Factorysuite\Common\Slssvc.exe
Wonderware NetDDE Helper	C:\Program files\Factorysuite\Common\Wwnetdde.exe
Wonderware WindowViewer	C:\Program files\Factorysuite\Intouch\View.exe

Note For more information on the Windows NT services for other FactorySuite components, see the *Wonderware FactorySuite Administrator's Guide*.

Why Use Windows NT Services?

The InTouch Windows NT services provide you with the following:

Operator Log on and Log off.

During normal plant operations, the operators usually log on and log off the Windows NT operating system between work shift changes. When WindowViewer runs as a Windows NT service, it continues to log historical data, gather alarm information, process scripts, act as an I/O server, and write values as an I/O client even as different operators log on and off.

Power loss.

If the electrical power is disrupted, it can take the plant operators some time before normal plant operations resume. Most disaster recovery plans require that essential computer systems start immediately after the electrical power is restored. Microsoft Windows NT Workstations and Windows NT Servers can restart automatically after the power is restored. Since the InTouch component software is run as a set of Windows NT services, your vital systems can begin running immediately without compromising Windows NT security.

Continuous operations.

The modern plant views resource efficiency as a primary concern. To achieve this level of throughput many plants use continuous operations. When the InTouch components run as Windows NT services, a plant can operate continuously 24 hours a day, seven days a week.

Windows NT Services Running in Desktop Context versus System Context

When you start up the Windows NT operating system, the services that are configured to automatically start will start in the "background" with no visible graphical user interface (GUI) appearing on the desktop. The services in this situation are running in the *system context*. When an operator or other user logs on the system, any services that are running in the system context that have an associated GUI will automatically appear on the desktop when the user logs on. In this situation, the services are now running in the *desktop context*.

Therefore, if you configure WindowViewer service to automatically start, the service will run in the system context as soon as the Windows NT system starts up. Then, when a user logs onto the system, the WindowViewer service continues to run but in the desktop context (WindowViewer's GUI automatically appears.)

Note If you have InTouch Access Names defined with the **Advise only active items** option turned on, along with I/O tagnames that are active only in certain InTouch application windows (tagnames that are not used anywhere else in the application), it is possible to deactivate those tagnames. Once you log off the system, WindowViewer, though running, has no visible GUI for you to navigate the application with the keyboard, mouse or other pointing device. However, if you close an application window from a QuickScript, the window automatically is unloaded from memory, thus terminating the link to those tagnames.

Creating a Master User Account

When the Windows NT operating system starts up a service, it runs in a user context called the *local system account*. Every Windows NT service that is configured to automatically start runs under this local system account. When you log onto the Windows NT operating system with your own account and you launch an application, it will run under your user account.

The local system account has very few privileges within the Windows NT security system and there's no guarantee that the users who log onto the system will have the rights to run the InTouch services. This includes actions such as accessing files, exchanging data between other programs and services and modifying the registry. InTouch requires that the user account have a higher access level in the Windows NT security system than the local system account so that the services can do these types of actions.

To accomplish this, you must create a **"master" user account** for InTouch. (A recommended name for the user account is "wwservices".) This master account should have Administrative privileges on the local computer so that it has the rights to start up the InTouch services. Then, during the InTouch installation when the FactorySuite **Common Components** dialog box appears, you can enter the node name of the local computer (not the domain name), along with your user name and password of the master account.

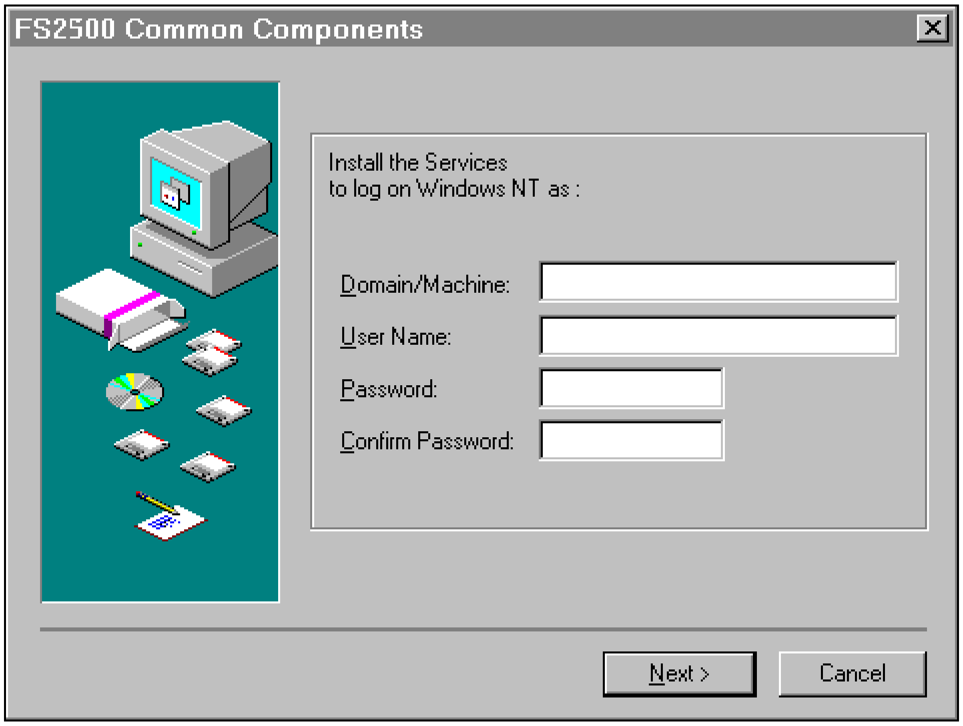
Note Any changes to the master account must be done during installation of InTouch or by running the Wonderware Service User program, WWUSER.EXE.

Once you setup a master account and log onto the computer and start up an InTouch component (such as WindowViewer), the component's services run under the master account instead of the user account. This ensures that you have the necessary privileges to start up the InTouch services.

Note When discussing Windows NT services, you may hear the term *impersonation account* used in place of the master account. An impersonation account is the User/Group account that provides access to the restricted resource "area" of your site or server. In Windows NT, this is called the NT User/Group Account, which must be set up by the System Administrator.

Configuring System Privileges

During InTouch installation, you will be prompted to provide your user name and password for your administrative account. This information is used to set up your NT user impersonation account. The Wonderware services such as, Wonderware NetDDE Helper and Wonderware WindowViewer will use this information for automatic logon, and for automatically starting the appropriate services during unattended start up.



Install the Services
to log on Windows NT as :

Domain/Machine:	
User Name:	
Password:	
Confirm Password:	

To configure system privileges

1. In the **Domain/Machine** box, type the system domain name or the node name.
2. In the **User Name** box, type your user identification.
3. In the **Password** box, type your system password.

Note The **User Name** and **Password** that you provide must be a valid Administrator level logon configured through the NT User Manager.

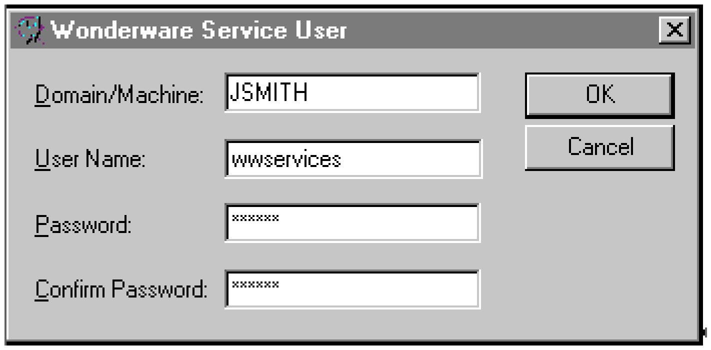
4. In the **Confirm Password** box, retype you system password to verify it.

Tip After installation, if you need to alter this information, run the Wonderware Service User application (wwuser.exe) located in your installed directory. For example, \Program Files\FactorySuite\Common.

The following procedure describes how you setup a new Master Account if you have already installed InTouch but need to change the installation to use a new "master" user account.

To setup FactorySuite Services to use a new Master Account

1. Run the Wonderware Service User program. (By default, wwuser.exe is located in C:\Program Files\Factorysuite\Common). The **Wonderware Service User** dialog box appears.



Domain/Machine: JSMITH
User Name: wwservices
Password: ******
Confirm Password: ******

Note The information you enter here must correspond to the current master account.

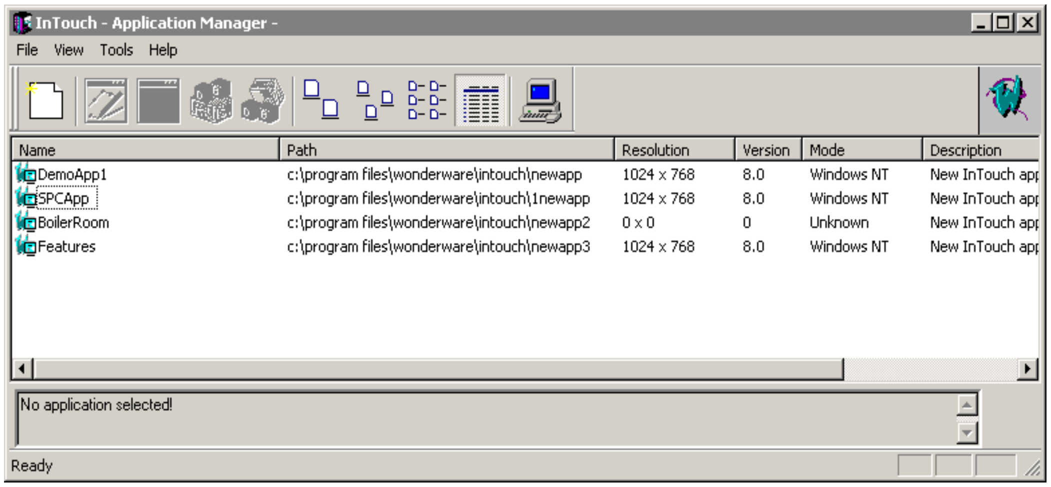
2. In the **Domain/Machine** box, type the machine name.
3. In the **User Name** box, type the name for the new Master Account.
4. In the **Password** box(s) type the password.
5. Click **OK**.

Configuring the InTouch Services

The InTouch services, **Wonderware Logger**, **Wonderware SuiteLink** and **Wonderware NetDDE Helper**, are installed and configured during the FactorySuite installation to automatically start up. However, you must configure **WindowViewer** to automatically start up as a Windows NT service.

To configure WindowViewer to start as an NT service

1. Start the InTouch program (intouch.exe). The **InTouch Application Manager** dialog box appears.



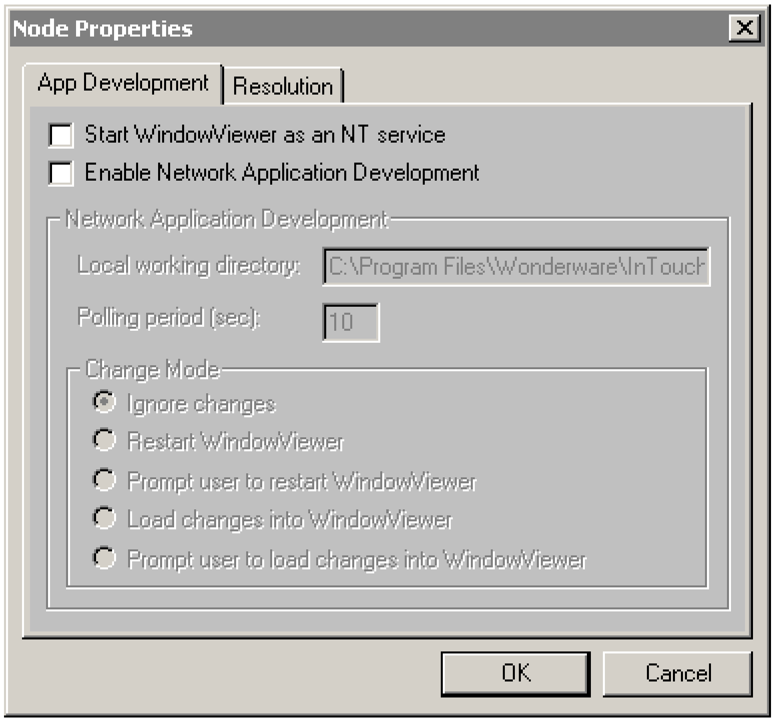
Name	Path	Resolution	Version	Mode	Description
DemoApp1	c:\program files\wonderware\intouch\newapp	1024 x 768	8.0	Windows NT	New InTouch apr
SPCApp	c:\program files\wonderware\intouch\1newapp	1024 x 768	8.0	Windows NT	New InTouch apr
BoilerRoom	c:\program files\wonderware\intouch\newapp2	0 x 0	0	Unknown	New InTouch apr
Features	c:\program files\wonderware\intouch\newapp3	1024 x 768	8.0	Windows NT	New InTouch apr

No application selected!

Ready

2. Click the **Node Properties** tool or on the **Tools** menu, click **Node Properties**. The **Node Properties** dialog box appears with the **App Development** property page active.

Tip To quickly access the **Node Properties** dialog box, right-click a blank area of the display window and then, click **Node Properties**.



Start WindowViewer as an NT service

Enable Network Application Development

Network Application Development

Local working directory: C:\Program Files\Wonderware\InTouch

Polling period (sec): 10

Change Mode

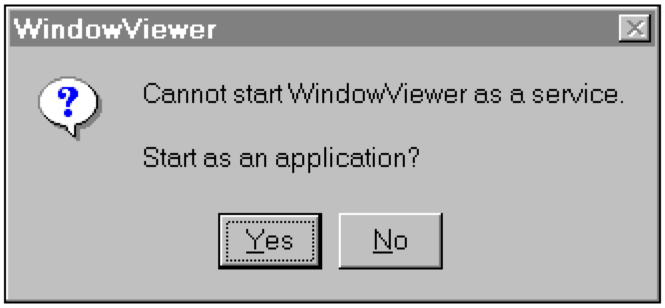
- [✓ Ignore changes]
- Restart WindowViewer
- Prompt user to restart WindowViewer
- Load changes into WindowViewer
- Prompt user to load changes into WindowViewer

3. Select the **Start WindowViewer as an NT service** option.

Note Network Application Development is disabled when WindowViewer is enabled as a Windows NT service.

4. Click **OK**.

Note If WindowViewer is configured as an NT service and subsequently started directly (from its icon, the Windows startup menu and so on), there will be approximately a 15 second delay before WindowViewer will display a window. This delay is due to WindowViewer attempting to connect to the NT Service Control Manager. Upon failing to connect to the Service Control Manager, WindowViewer will display the following message box:


WindowViewer

Cannot start WindowViewer as a service.

Start as an application?

If you click **Yes**, WindowViewer is started as an application not an NT service. If click **No** the command to start WindowViewer is canceled.

Tip Here are some other methods you can use to start WindowViewer as an NT service:

1. In WindowMaker, click the Runtime! FastSwitch in the upper right hand corner on the menu bar.

2. Start WindowViewer from the InTouch Application Manager.

3. To start the WindowViewer service from the command prompt use the following command:

Net Start View

4. To stop the WindowViewer service from the command prompt, use the following command:

Net Stop View

Caution! If WindowViewer is configured to automatically start up as an NT service, when the system restarts, the last InTouch application referenced by the InTouch Application Manager automatically starts up. However, if you are editing more than one InTouch application, and you should lose power or, restart the system, the wrong application may start up.

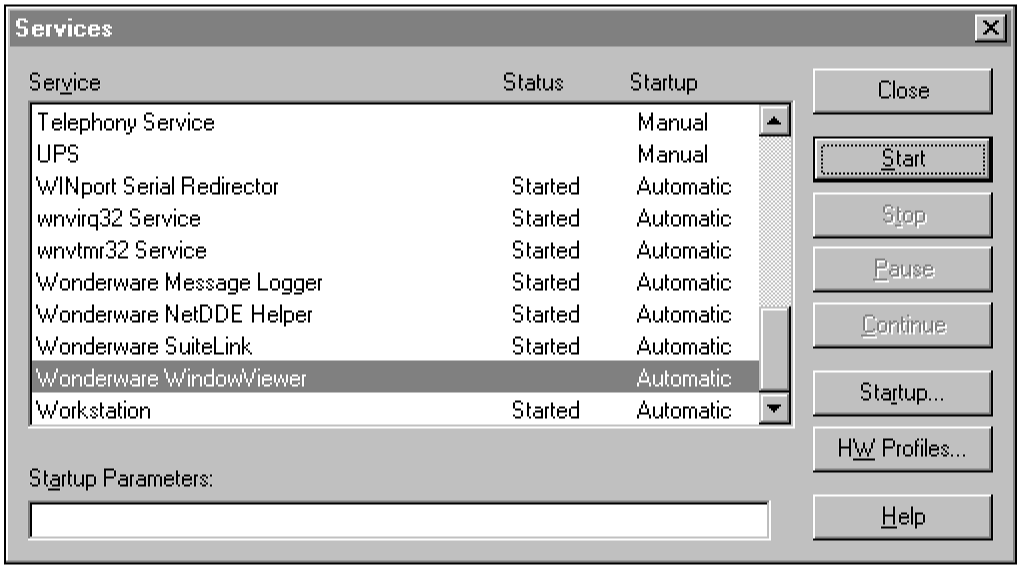
To stop the WindowViewer service from running

You can stop WindowViewer from running as an NT service by turning off the **Start WindowViewer as an NT service** option in the **Node Properties** dialog box.

You can stop WindowViewer or any other service from running by shutting down the computer. You can also stop (or start) any service through the Windows Control Panel.

To stop/start an NT service through the Control Panel

1. Open the Control Panel and double-click the **Services** icon.
2. The **Services** dialog box appears.



Service	Status	Startup
Telephony Service		Manual
UPS		Manual
WINport Serial Redirector	Started	Automatic
wnvirq32 Service	Started	Automatic
wnvtmr32 Service	Started	Automatic
Wonderware Message Logger	Started	Automatic
Wonderware NetDDE Helper	Started	Automatic
Wonderware SuiteLink	Started	Automatic
Wonderware WindowViewer		Automatic
Workstation	Started	Automatic

3. Highlight the service that you want to stop (or start) in the Service list and then, click either **Stop** or **Start** and then, click **Close**.

Manually Installing/Removing an NT Service

Normally, InTouch's Windows NT services are automatically installed during the FactorySuite installation procedure and they are removed by double-clicking the Control Panel's **Add/Remove Programs** icon. However, if you need to manually install or remove an InTouch Windows NT service, (for example, Wonderware WindowViewer), perform the following steps.

To install a Windows NT service

1. Select **Run** from the Start menu or open an MS-DOS Command Prompt window.
2. Enter this command:

```
"C:\Program Files\FactorySuite\InTouch\view.exe" -
   install
```

To remove a Windows NT service

1. Select **Run** from the Start menu or open an MS-DOS Command Prompt window.
2. Enter this command:

```
"C:\Program Files\FactorySuite\InTouch\view.exe" -
   remove
```

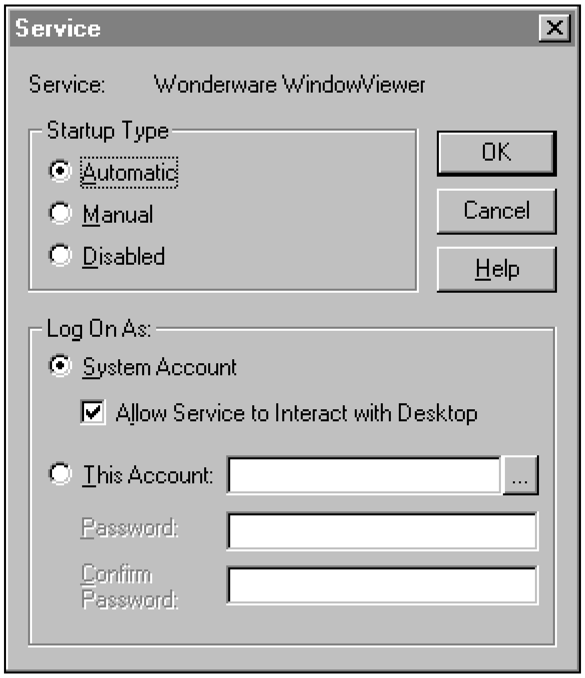
Note Both of these commands are examples that install and remove the Wonderware WindowViewer service. The pathname will include the **\Common** directory instead of the \InTouch directory for the other InTouch services.

If you enter the command in the **Run** dialog box, be sure to enclose the command with double quotes (" "). For example, "Program Files". If you remove a service, Windows NT will stop the service first and then remove it. Removing a service will not delete the application's .exe file.

Configuring NT Services Startup Options

To configure an InTouch service's startup option

1. Open the Windows Control Panel.
2. Double-click the **Services** icon. The **Services** dialog box appears.
3. Select the desired InTouch service in the Service list and then, click **Startup**.
4. A second **Service** dialog box appears.



Service: Wonderware WindowViewer

Startup Type: [✓ Automatic] Manual Disabled

Log On As: [✓ System Account]

[✓] Allow Service to Interact with Desktop

This Account:

Password:

Confirm Password:

5. In the **Startup Type** group, select the option that you want to use and then, click **OK**.

Option	Description
Automatic	When Windows NT restarts, the service automatically starts without any user intervention.
Manual	A user or an application process must explicitly start the service.
Disable	The service is prevented from starting. (usually reserved for troubleshooting purposes).

Note Wonderware recommends that you use the default startup options for all InTouch services.

Dependencies Between InTouch Services

When a service is installed, a list of dependencies is provided to the Windows NT operating system. If a service depends on other services starting before it can start, Windows NT checks to make sure that the other services are running *before* allowing the service to start. Depending on your requirements for running WindowViewer, you should be aware of the following dependencies.

The Wonderware NetDDE Helper service must be running if you plan to use Distributed Alarming or Distributed History or, if you intend to access Network DDE data. The Wonderware NetDDE Helper service also depends on both the Network DDE and Network DDE DSDM services being installed and configured for either Manual or Automatic startup. During installation, the Wonderware NetDDE Helper service is configured for Manual startup, meaning WindowViewer will automatically start this service on boot.

If you need WindowViewer to act as a SuiteLink server or client, then the Wonderware SuiteLink service must be running. The Wonderware SuiteLink service also requires that Microsoft TCP/IP be installed.

If you want to store any messages or errors while WindowViewer is running, you must make sure that the Wonderware Logger service is installed. Both the Wonderware SuiteLink and Wonderware Logger services should be installed and configured to run in **Automatic** startup as part of the FactorySuite common installation.

For more information, see "Configuring NT Services Startup Options."

Logging Off Windows NT

When you log off the Windows NT operating system, it sends the control log off messages, **wm_close** and **wm_terminate**, to close all applications that are running under the desktop context. By default, Microsoft Network DDE is installed and configured to "listen" for these log off messages. If any messages are received, Microsoft Network DDE will automatically terminate its open network connections and, while you are logged off the system, Microsoft Network DDE will not permit any new connections to be made.

However, the **Wonderware NetDDE Helper** service also listens and captures the log off messages for Microsoft Network DDE to preserve the Microsoft Network DDE connections. Therefore, while you are logged off, Microsoft Network DDE allows the new connections to be made.

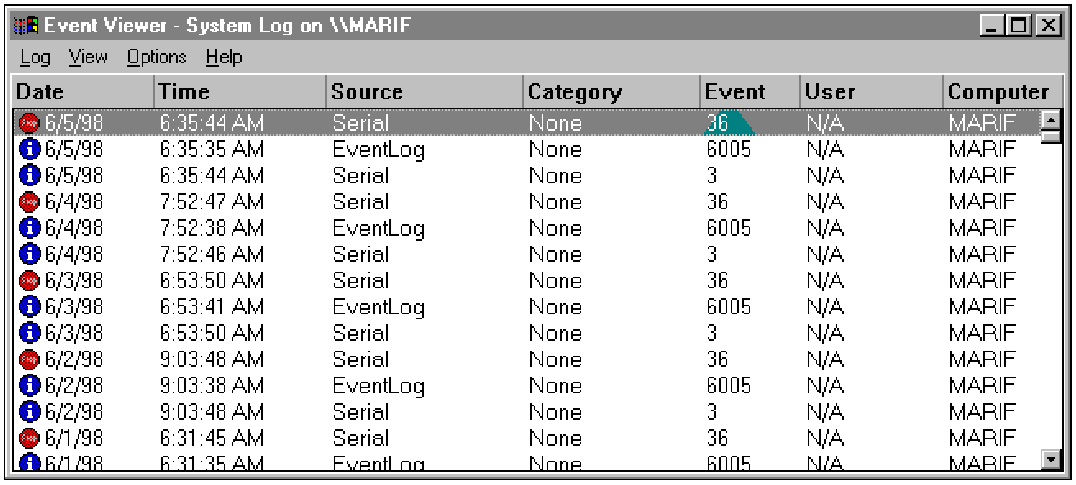
For more information, see "Windows NT Services Running in Desktop Context versus System Context."

Troubleshooting InTouch Services

Here are some common problems and solutions that Wonderware Technical Support has found when installing or running the InTouch services:

Error "One or more services failed to start…"

If you run into this error, from the Start menu, select **Programs**/**Administrative Tools (Common)**/**Event Viewer**. The **Event Viewer – System Lo***g* dialog box appears displaying a list of informational messages, warnings or errors that may have occurred during the start up of any Windows NT services:



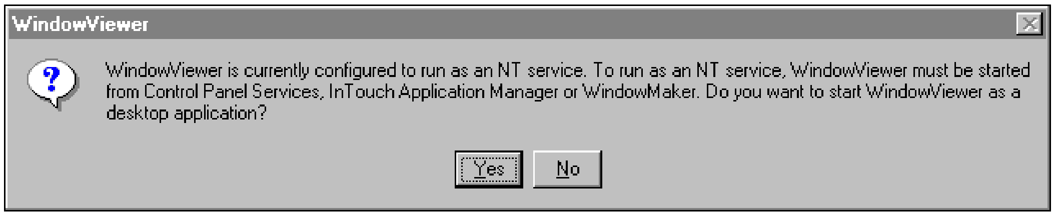
Date	Time	Source	Category	Event	User	Computer
6/5/98	6:35:44 AM	Serial	None	36	N/A	MARIF
6/5/98	6:35:35 AM	EventLog	None	6005	N/A	MARIF
6/5/98	6:35:44 AM	Serial	None	3	N/A	MARIF
6/4/98	7:52:47 AM	Serial	None	36	N/A	MARIF
6/4/98	7:52:38 AM	EventLog	None	6005	N/A	MARIF
6/4/98	7:52:46 AM	Serial	None	3	N/A	MARIF
6/3/98	6:53:50 AM	Serial	None	36	N/A	MARIF
6/3/98	6:53:41 AM	EventLog	None	6005	N/A	MARIF
6/3/98	6:53:50 AM	Serial	None	3	N/A	MARIF
6/2/98	9:03:48 AM	Serial	None	36	N/A	MARIF
6/2/98	9:03:38 AM	EventLog	None	6005	N/A	MARIF
6/2/98	9:03:48 AM	Serial	None	3	N/A	MARIF
6/1/98	6:31:45 AM	Serial	None	36	N/A	MARIF
6/1/98	6:31:35 AM	EventLog	None	6005	N/A	MARIF

You can view any warnings or errors that resulted from an InTouch service failing to start. If the **Event Viewer** indicates that the Wonderware WindowViewer service failed to start, more than likely it is because it depended on other InTouch services that are not running.

For more information, see "Dependencies Between InTouch Services."

After you've configured the Wonderware WindowViewer service to run, it must be started and stopped as a service.

If you start up WindowViewer from a shortcut icon or by selecting **Programs/Wonderware FactorySuite/InTouch WindowViewer** from the Start menu, WindowViewer will pause for about 15 seconds and then display this message:


WindowViewer

WindowViewer is currently configured to run as an NT service. To run as an NT service, WindowViewer must be started from Control Panel Services, InTouch Application Manager or WindowMaker. Do you want to start WindowViewer as a desktop application?

Click **Yes** to cause the Wonderware WindowViewer service to start as a desktop application under your user account with its GUI displayed.

Caution! If you choose **Yes** to run WindowViewer as a desktop application, when you log off the system, WindowViewer will close. It will not continue to run as a service since it was never originally started as a service. Meaning that after you log off, your InTouch application will not continue to run and collect historical data, display alarms or process scripts.

To allow your InTouch application to continue running as a service, you will need to log back on and configure WindowViewer to start as an NT Service.

Click **No** to cancel the operation. The WindowViewer application will not be started.

For more information, see the "Running WindowViewer as an NT Service." section of Chapter 2.

InTouch Services Fail to Install or Start Up

If InTouch services fail to install or start up after you install InTouch, perform the following steps:

1. Bring up the Windows NT User Manager window and create a new master user account.

   Note This user account must have Administrative privileges on the local computer in order to start up a InTouch component as a service. If you do not see your computer's node name in the domain list, then manually type in the node name.

   For more information, see, "Creating a Master User Account."

2. Verify that your computer's node name is no longer than 14 characters. If there are underscore characters (_) or dashes (-) in the node name, modify the node name to something equal to or less than 14 characters without underscores or dashes.

3. During installation when you are prompted to enter the domain name, type in the node name of your computer, not the domain name. Then type in the username that was created in step 1and your password.

4. If you have already installed InTouch, you can still specify the domain name, user name and password by running the Wonderware Service User program, wwuser.exe, from the directory C:\Program Files\Factorysuite\Common.
5. Reboot your computer.
6. Log onto your network domain with any valid user account. Even if your domain goes down, it will not affect your InTouch application that is running on the local computer.

Registry Keys for the InTouch Services

Each of the four InTouch services are listed as keys in the Windows NT registry, which are the following:

InTouch ServiceRegistry Keys

Wonderware SuiteLink
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SLS

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\slssvc

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SuiteLink

Wonderware Logger
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WWLOGSVC

Wonderware NetDDE Helper
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WWNetDDE

Wonderware WindowViewer
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\VIEW

Glossary of Terms

Accelerators

Accelerators are used by the application in creating a keyboard interface. They are normally offered as alternatives to using the menu for indicating choices. An accelerator is a keystroke that has special meaning to the application and that can be used to generate a command message.

Access

The obtaining of data. Locating desired data.

Active Application

The application that created the window that currently has the keyboard focus. Applications do not need to be the active application in order to receive and process messages. Applications are notified by message whenever they are gaining or losing the status of "the active application." The user normally determines the active application, but applications can override this decision.

ActiveX

A loosely defined set of technologies that allows software components to interact with each other.

ActiveX Control/Container

ActiveX controls, originally known as OLE controls or OCXs, are standalone software components that perform specific functions in a standard way. They define standard interfaces for reusable components. ActiveX controls are not separate applications. Instead, they are servers that are placed into a control container. To use ActiveX controls, they must to be placed in an ActiveX container. InTouch is an ActiveX container. VisualBasic and internet browsers are also ActiveX containers.

Alarm

A warning signal that is displayed or activated whenever a critical deviation from normal conditions occur.

Algorithm

A sequence of instructions which are mechanically carried out to perform a procedure.

Analog

Referring to the representation of numerical quantities by the measurement of continuous physical variables.

Application

A program or group of programs used for a particular kind of work, such as InTouch.

Application Publisher

The InTouch Application Publisher creates a self-extracting file that contains all relevant files and setup procedures that you need to install the application onto another InTouch node. The Application Publisher compresses the Application to minimize download time over the Internet.

Argument

A variable to which either a logical or a numerical value may be assigned. Up to 16 arguments can be specified for an InTouch Quick Function. See **QuickFunctions**.

Assignment Operator

An operator used in an assignment statement that causes the value on the right to be placed into the variable on the left of the operator.

Assignment Statement

A programming language statement that gives a value to a variable, such as in *x = x + 1 or y = 6*.

Asterisk

A symbol (*) used to represent a multiplication operator in many programming languages.

Asynchronous

Pertaining to a mode of data communications that provides a variable time interval between characters during transmission. See **Synchronous transmission**.

B

An abbreviation for byte or baud. Use bytes when referring to storage, or baud rate when referring to communications. Kb = 1000 bytes or baud (technically 1K = 1024 bytes). See **Baud** or **Byte**.

Background

In multiprogramming, the environment in which low priority programs are executed. Also, the part of a display screen not occupied with displayed characters or graphics (foreground).

Backing Up

The creation of a backup copy of a specified file or files, transferring them from either a floppy disk or a hard disk to another removable or fixed disk.

Bandwidth

On a network, the transmission capacity of a communications channel stated in megabits per second (Mbps). For example, Ethernet has a bandwidth of 10 Mbps. Fast Ethernet has a bandwidth of 100 Mbps.

Baud Rate

A unit for measuring data transmission speed. One baud is 1 bit per second. Since a single character requires approximately 8 bits to represent it, divide the baud rate by 8 to calculate the characters per second (cps) to be transmitted. For example, 300 baud equals 37.5 cps, 1200 baud equals 150 cps, 2400 baud equals 300 cps.

Beta Testing

Pretesting of hardware and software products with selected "typical" users, to discover bugs before the product is released to the general public.

Binary

Pertaining to the number system with a radix of 2, or to a characteristic or property involving a choice or condition in which there are exactly two possibilities.

Binary Code

A coding system in which the encoding of any data is done through the use of bits--that is, 0 or 1.

Binary Coded Decimal (BCD)

A computer coding system in which each decimal digit is represented by a group of four binary **1**s and **0**s.

BIOS

An acronym for **B**asic **I**nput/**O**utput **S**ystem. In some operating systems, the part of the program that customizes it to a specific computer.

Bit

A binary digit; a digit (1 or 0) in the representation of a number in binary notation. The smallest unit of information recognized by a computer and its associated equipment. Several bits make up a byte, or a computed word.

Bitmap

A memory image of a portion of a display device surface. In Windows, a bitmap is actually a data structure containing a pointer to this memory image, plus information about the display device. The amount of memory required for a bitmap is device-specific, being dependent upon the color capabilities and pixel resolution of the device in question.

Boot

To start or restart a computer system by reading instructions from a storage device into the computer's memory. It involves loading part of the operating system into the computer's main memory. If the computer is already turned on, it's a "warm boot;" if not, it's a "cold boot."

Border

The line surrounding the current active window. A window can be resized by dragging on the border when the two-header arrow is present.

Buffer

An area of storage used to temporarily hold data being transferred from one device to another. Used to compensate for the different rates at which hardware devices process data; for example, a buffer would be used to hold data waiting to print, in order to free the CPU for other tasks, since it processes data at a much faster rate.

Bus

A channel or path for transferring data.

Button

Large rounded-rectangular or small round buttons which appear in dialog boxes. Click with the cursor arrow to select the button's option or command.

Byte

A grouping of adjacent binary digits operated on by the computer as a unit. The most common size byte contains 8 binary digits.

CD-ROM

Compact Disc, Read-Only Memory A compact disc format that holds over 650 MB of data.

Check Box

A small square box which appears in a dialog box that can be turned on or off. Check boxes are generally associated with multiple options which can be set. To set a check box option, move to it and click the mouse button. When an **X** appears it is selected. When it is blank it is not.

Client

A machine that exchanges data with a server. A software program that shares data with the server.

Client/Server

A network that has client machines that rely on a server.

Clipboard

A storage area for holding data (text, bitmap, graphic object, etc.) which is being copied or moved to another application or window.

Close

To remove an application's window and icon from the screen and free the memory used by the application. To close an application, choose the *Control/Close* command. Once an application is closed, it must be run to use it again.

Command

A word or phrase, usually found in a menu, that carries out an action.

Command Button

A round-cornered rectangle with a label on it that describes an action, such as **OK**, **Cancel** or **Close**. When chosen, the command button carries out the action.

Command Key

Any keyboard key used to perform separate functions.

COM Port

A serial communication port on a PC.

Command Line

The string of arguments that follow any MS-DOS command, including the command to initiate an application program. The arguments in the command line are passed to the MS-DOS function or the program at startup time.

Computer Graphics

A general term meaning the appearance of pictures or diagrams, as distinct from letters and numbers, on the display screen or hard-copy output device.

Concatenate

To link together or join two or more character strings into a single character string, or to join one line of a display with the succeeding link.

CONFIG.SYS

An ASCII text file that MS-DOS processes when the system is turned on or restarted. It allows the user to configure certain aspects of the operating system, such as the number of internal disk buffers allocated, the number of files that can be open at one time, etc.

Console

The administrative session that is run on the server.

Control Name

See **Windows Controls**

Crop

In computer graphics, to cut off some part of an image.

CSV

Comma Separated Variable is the format used by the Clipboard for transfer of columns of text and numerical data between applications. A CSV data item is like text with each variable separated by commas. Although Microsoft Excel is probably the principle creator of CSV clipboard data, many DOS applications support this format.

Current File

The file that is running in the application.

Database

A collection of logically related records or files. A database consolidates many records into a common pool of data records which serves as a single central file.

Default

An option, command, or device that is automatically selected or chosen by the system. For example, one of the command buttons in a dialog box is already selected when the dialog box is opened. This indicates that it is the default value and will be chosen automatically if the **<Enter>** key is pressed. Default values are overridden by selecting another appropriate option, command, or device.

Device Driver

A program that controls how the computer interacts with a devices such as a printer, monitor, or mouse. A device driver enables the use of devices with the computer.

DHCP

Dynamic Host Configuration Protocol. A TCP/IP protocol used to automatically assign IP addresses and configure TCP/IP for network clients.

Dialog Box

A window that appears when Windows needs further information before it can carry out a command. For example, if the **Save** command on a **File** menu is selected a dialog box will appear asking for a name for the file to be saved under.

Directory

A structure for organizing files into convenient groups. A directory is like an address showing where files are located. A directory can contain files, or sub directories of files.

Discrete Value

A variable which only has two states: '1' (True, On) or '0' (False, Off).

Disk Operating System

(DOS) An operating system in which the operating system programs are stored on magnetic disks. Typically, it keeps track of files, saves and retrieves files, allocates storage space, and manages other control functions associated with disk storage.

Display

The physical representation of data on the screen.

Dithered

Intermingled dots of various colors which produce what appears to be a new color.

Document

A unit of printer output that must be printed contiguously; that is, no other output may be interspersed within a document. A document, then, is analogous to a report. The application must specify the start and end of each document.

Domain

In Windows 2000, a group of computers that share a security policy and a user account database. A Windows 2000 domain is not the same as an Internet domain.

Domain name

In Active Directory, the name given to a collection of networked computers that share a common directory. On Internet, the unique text name that identifies a specific host. A machine can have more than one domain name, but a given domain name points to only one machine. Domain names are resolved to IP addresses by DNS name servers.

Domain Name System (DNS)

A service on TCP/IP networks (the Internet included) that translates domain names into IP addresses. This allows users to employ friendly names like FinanceServer or Ourbusiness.com when querying a remote system, instead of using an IP address such as 123.45.67.89.

DRA

Dynamic **R**eference **A**ddressing allows dynamic runtime control of an I/O tagname's access name and item through dot fields. It also allows you to control access name properties through QuickScript functions.

DRC

Dynamic **R**esolution **C**onversion enables each View node to scale an application to a number of user-defined options, including a custom resolution. This scaling takes place while WindowViewer compiles the application, and does not require WindowMaker.

Drive

A letter in the range A-Z, followed by a colon (:), indicating a logical disk drive.

Dynamic Data Exchange

DDE is the passage of data between applications, accomplished without user involvement or monitoring. In the Windows environment, DDE is achieved through a set of message types, recommended procedures (protocols) for processing these message types, and some newly defined data types. By following the protocols, applications that were written independently of each other can pass data between themselves without involvement on the part of the user. For example, InTouch and Excel.

Dynamic Host Configuration Protocol

See **DHCP.**

ENTER Key

The key on the keyboard which executes a statement or command. Same as RETURN key on some keyboards.

Enterprise

Term used to encompass all of a business's operation including all remote offices and branches.

Ethernet

A local area network protocol developed by Xerox Corporation in 1976. Ethernet supports data transfer rates of 10 Mbps and uses a bus topology and thick or thin coaxial, fiber-optic, or twisted-pair cabling. A newer version of Ethernet called Fast Ethernet supports data transfer rates of 100 Mbps, and an even newer version, Gigabit Ethernet, supports data transfer rates of 1000 Mbps.

Events

Events are associated with ActiveX controls and occur through the ActiveX container. You can execute ActiveX control events in runtime (WindowViewer) by designing a particular action and associating it to the event by creating ActiveX Event Scripts. For example, **Control.click (shift)**. **FileViewer.DoubleClick (name)**. See **Properties** and **Methods**.

Expression

A general term for numerals, numerals with signs of operation, variables and combinations of these: 6, 3+6, n+10 are all expressions.

Extend

To select more than one item in a window. To extend a selection, hold down the SHIFT key until everything is selected.

Extension

The period and three letters at the end of a filename. An extension identifies what kind of information a file contains. For example, an extension may be .EXE, .BAT indicate that a file contains an application.

FactorySuite

Wonderware's software package that includes InTouch (and all its add-on programs and utilities), InBatch, InTrack, InControl and its I/O Server's, IndustrialSQL Server, FactorySuite Web Server, numerous other I/O Server programs, Productivity Pack, NetDDE for Windows and NetDDE Extensions for Windows NT.

Fast Ethernet

See **Ethernet**.

File

A mechanism for holding and storing information on a hard disk or diskette for later use. File also may refer to any document or database created by the user, such as a word processing document, spreadsheet, etc. Each file appears in its own window and in most cases, the name of the file will appear in the title bar at the top of the window.

Filename

Filenames consist of a base name containing no more than eight characters and a three-character extension. For example, INTOUCH.EXE.

Format

To prepare a disk so it can hold information. Formatting a disk erases any previously stored data. Format is the term used for an object rendition. In most Windows applications, available formats include Text, Bit map, etc.

Fully qualified domain name (FQDN)

A domain name that includes the names of all network domains leading back to the root so as to clearly indicate a location in the domain namespace tree. An example of an FQDM is Sales.Europe.Wonderware.com.

Gigabit Ethernet

See **Ethernet**.

Graphics Object

A visually oriented object, such as a scroll bar, bit map or icon that is used in the presentation of the visual interface. Graphic objects can be created by either the application or by Windows for use by the application.

Help

Online instructions that explain how to use a Windows application. The Help menu displays specific Help topics.

Highlight

Indicates that the object is selected and will be affected by the next action or command. A highlighted object appears in reverse video. A selected icon is outlined in white and displays the application's name.

HMI

Human-Machine Interface. A software program that allows an operator to control a manufacturing process. Also known as MMI, Man-Machine Interface. For example, Wonderware InTouch.

ICA

Independent Computing Architecture. A remote presentation services protocol from Citrix Systems that allows thin clients to access the server.

Inactive

A window or icon that is not selected. See **Select**.

Insertion Point

The place where text will be inserted when the user types. The insertion point usually appears as a flashing vertical line (the cursor) and can appear in the workspace or within a dialog box. The text typed appears to the left of the insertion point, which is pushed to the right as text is entered.

Integer

Any member of the set consisting of the positive and negative whole numbers and zero. Examples: -59, -3, 0.

Internet

The vast collection of inter-connected networks that all use TCP/IP and that evolved from ARPANET of the late 1960s and early 1970s. The Internet connects roughly 70,000 independent networks into a global network.

I/O

An abbreviation for Input/Output.

IP (Internet Protocol)

A common protocol that allows network communications.

IP number or address

A four-part number separated by periods (for example, 123.45.67.89) that uniquely identifies a machine on the Internet. Every machine on the Internet has a unique IP number; if a machine doesn't have an IP number, it isn't on the Internet. Most machines also have one or more domain names that are easier for people to remember.

Java

An advanced programming language similar to C and C++ used in Web pages to provide animation and other advanced features that make Web pages unique.

Key Accelerator

A special keyboard sequence that executes menu commands. For example, Ctrl + A. See **Accelerators**.

List Box

A box within a dialog box listing all available choices for a command. For example, a list of filenames on a disk. Usually an item is selected from the list box, then "OK" is chosen. If there are more choices than can fit in the list box, it will have vertical scroll bars. Selecting the down arrow next to the first item in the list will display the rest of the list box.

Load Balancing

Servers in a server farm that are configured so that the demands on the servers are spread equally.

Local Area Network (LAN)

A group of connected computers, usually located close to one another (such as in the same building or the same floor of the building) so that data can be passed among them.

Local Variable

You can declare local variables within a script to store temporary results and create complex calculations with intermediate scripting values without impacting or decreasing your licensed tagname count and increase performance. Local variables or tagnames can be used interchangeably within the same script.

Log on

The act of entering into a computer system; for example, "Log on to the network and read your email."

Logon

The account name used to gain access to a computer system. Unlike a password, the logon name isn't a secret.

Macro

A single, symbolic programming-language statement that when translated results in a series of machine-language statements.

Maximize

To make a window or icon fill the entire screen. To maximize a window, choose the *Control/Maximize* command, or click on the Maximize box in the upper right corner of the window. See **Minimize** and **Restore**.

MB

An abbreviation for megabyte. One million bytes. 1000KB.

Megabyte

1,048,576 bytes or 1024 kilobytes, actually; or roughly one million bytes or one thousand kilobytes.

Menu

Menus are group listings of available Windows and application commands. Menu titles appear in the menu bar at the top of the window. A command is chosen by displaying the menu, then choosing the desired command.

Menu Bar

The horizontal bar that lists the names of an application's menus. The menu bar appears below the title bar of a window Each Window's application has a menu bar that is distinct for that application, although some menus (and commands) are common to many of these applications.

Message Box

A special dialog box through which an application displays error messages or other important information. Message boxes alert the user when an error occurs or when the application needs information to complete an action or command.

Method

Methods are associated with ActiveX controls. They are similar to script function calls that you can call from the ActiveX container. For example, **Browser.Navigate("URLPageName")**, **Engine.start()**. See **Properties** and **Events**.

Millisecond

One thousandth of a second, abbreviated ms or msec.

Minimize

To turn a window into an icon. To minimize a window, choose the *Control/Minimize* command, or click on the Minimize box in the upper right corner of the window. See **Maximize** and **Restore**.

Mirroring

The display or creation of graphics that portray an image in exactly the reverse of its original orientation. For example, flipping the graphic on its x- or y-axis.

Mode

A method or condition of operation.

Modulo

A mathematical function that yields the remainder of division. A number *x* evaluated modulo *n* gives the integer remainder of *x/n*. For example, 200 modulo 47 equals the remainder of 200/47, or 12.

MS/DOS

An abbreviation for **M**ICRO**S**OFT **D**ISK **O**PERATING **S**YSTEM, the standard operating system used by the IBM Personal Computer and compatible computers. Developed by Microsoft, Inc.

Multitasking

The ability of a computer to perform two or more functions (tasks) simultaneously.

NAD

Network Application Development or NAD is an architecture that combines the best of the client-based and server-based architectures. NAD provides automatic notification of application changes and automatic distribution of the updated applications to View nodes.

Namespace

A name or group of names defined according to a naming convention; any bounded area in which a given name can be resolved. The Internet uses a hierarchical namespace that partitions names into categories known as top-level domain, such as .com, .edu, and .gov.

NetBIOS Enhanced User Interface (NetBEUI)

A small and fast protocol that requires little memory but can't be routed. Remote locations linked by routers can't use NetBEUI to communicate.

Node

A location on a tree structure with links to one or more items below it. On a LAN, a device that can communicate with other devices on the network. In clustering, a computer running Windows 2000 Advanced Server that is a member of a cluster.

NTFS file system

The native file system for Windows 2000 and Windows NT. Supports long file names, a variety of permissions for sharing files, and a transaction log that allows the completion of any incomplete file-related tasks if the operating system is interrupted.

Object

A set of data. Objects come in several formats; bitmap images, text, Real-time and Historical trend graphs, etc.

Off-line

Pertaining to equipment or devices not in direct communication with the central processing unit of a computer. Equipment not connected to the computer.

Operand

A quantity or data item that is operated upon.

Operating System

Software that controls the execution of computer programs and that may provide scheduling, debugging, input/output control, storage assignment, etc. Abbreviated OS.

Operator

In the description of a process, that which indicates the action to be performed on operands.

Option Button

A small round button appearing in a dialog box An option button is selected to set the option, but within a group of related option buttons, only one can be selected. An option button has a black dot when it is selected and is blank when it is not selected.

Option Button Group

A group of related options in a dialog box. Only one option button in a group can be selected at any one time.

Packet

The basic unit of information sent over a network. Each packet contains the destination address, the sender's address, error-control information, and data. The size and format of a packet depends on the protocol being used.

Page

A page is a block of information that is selected and stored in a file. For example, a paragraph of text from Microsoft Word may be a page and a chart from Microsoft Excel may be another. Pages may be held in a variety of formats in the same file. Pages are numbered as they are placed into the file.

Palette

The set of available colors in a computer graphics system.

Parity

An extra bit added to a byte, character, or word to ensure that there is always either an even number or an odd number of bits, according to the logic of the system. If, through a hardware failure, a bit should be lost in transmission, its loss can be detected by checking the parity. The same bit pattern remains as long as the contents of the byte, character or work remain unchanged.

Paste

To insert something into a document or file from the Clipboard. Some applications (including InTouch) may have a Paste command that performs this task. If using some other standard application that runs in a window, Windows adds the Paste command to the Control menu.

Path

The hierarchy of files through which control passes to find a particular file. Designates one or more disk drives and/or directory paths to be searched sequentially for a program or batch file if the file cannot be found in the current or specified drive and directory. The drives and/or directory paths are searched in the order they appear in the Path.

Pathname

A description of the location of a directory or file within the system. The pathname consists of the drive letter, a colon (:), followed by directory and subdirectory names, followed by a filename. Each name is separated from the previous one by a backslash (\). If not specified, a default drive and directory are used.

Pixel

A picture cell. Shortened version of "picture element." The visual display screen is divided into rows and columns of tiny dots, squares or cells, each of which is a pixel. The smallest unit on the display screen grid that can be stored or displayed. A computed picture is typically composed of a rectangular array of pixels. The resolution of a picture is expressed by the number of pixels in the display. For example, a picture with 560 x 720 pixels is much sharper than a picture with 275 x 400 pixels.

Poke

An instruction used to place a value (poke) into a specific location in the computer's storage.

Polling

A communications control method used by some computer/terminal systems whereby a computer asks many devices attached to a common transmission medium, in turn, whether they have information to send.

Port

That portion of a computer through which a peripheral device may communicate. A connection between the CPU and a peripheral device.

Precedence

Rules that state which operators should be executed first in an expression.

Process Control

The use of the computer to control industrial processes such as oil refining and steel production.

Process Control Computer

A computer used in a process control system, generally limited in instruction capacity, word length and accuracy. Designed for continuous operation in non-air-conditioned facilities.

Processing

The application that currently has control of the processor. An application is given control of the processor upon receipt of a message. It retains control of the processor until the message is processed.

Properties

Properties are associated with ActiveX controls and can be associated with InTouch tagnames. The properties that you can configure for a particular ActiveX control are determined by the ActiveX control designer. Some properties are one-directional, meaning either the property sets the tagname's value, or the tagname's value sets the property. While other properties are bi-directional, meaning the value can be set from either the tagname or the property.

Protocol

Set of rules or conventions governing the exchange of information between computer systems or applications.

Published Application

An application in a server farm that is shared equally among servers.

Queue

A group of items waiting to be acted upon by the computer. The arrangement of items determines the processing priority. For example, documents waiting to be printed.

QuickFunction

QuickFunctions are scripts that you can write and call from other scripts or expressions. You can use up to 16 arguments per QuickFunction. They are stored in the application in which they are created. QuickFunctions can be defined as asynchronous that, when executed, run in the background of the main WindowViewer (runtime) process.

QuickScript

Script created in InTouch. InTouch QuickScript capabilities allow you to execute commands and logical operations based on specified criteria being met. For example, a key being pressed, a window being opened, a value changing, and so on.

RAM

Random Access Memory. The computer's primary memory space.

Register

A high-speed device used in a central processing unit for temporary storage of small amounts of data or intermittent results during processing.

Remote Desktop Protocol (RDP)

The default connection protocol installed with Windows Terminal Services. RDP allows you to configure a wide variety of settings for each server. The client normally controls most of these settings.

Remote Tagname

A tagname that resides in a remote tag source but is referenced in a local InTouch application. Client applications can be designed without using any tagnames in the local Tagname Dictionary by using remote tagname references.

Restore

Icons can be restored to full-sized windows by double-clicking on them. To restore a window, choose the Restore command from the Control menu, or click on the Restore box in the upper right corner of the window. See **Maximize** and **Minimize**.

Retentive Tagname

A tagname that retains its current value and uses that value as its initial value upon restart whenever there is a system failure.

Run

To start an application. The Run command lets you specify parameters for the application. An application can also be run by double-clicking on its name or icon.

Running

An application that is "running" is an application that is in the system as a task, can receive messages, and is (normally) known to the user. From initiation to termination, an application is always running, but it is not always *processing*. See **Processing**.

Runtime

The time during which data is fetched by the control unit and actual processing is performed in the arithmetic-logic unit. Also, the time during which a program is executing.

Save

To store a file or changes to a file on a disk.

Scaling

The process of changing the size of an image.

Scroll

To move data or text up and down, or left and right to view parts of the file that cannot fit on the screen.

Scroll Bars

The bars that appear at the right side or bottom of a window. Use the scroll bars to move through a window that contains more information than can be shown on one screen. The scroll bar at the right side of a window scrolls vertically. The scroll bar at the bottom of a window scrolls horizontally.

Scroll Box

The small white box in the scroll bar. The scroll box reflects the position of the information within the window in relation to the total contents of the file. For example, if the scroll box is in the middle of the scroll bar, then the text or data in the window is in the middle of the file. The mouse can be used to scroll by dragging the scroll box in the scroll bar. See **Scroll Bars**.

Serial Port

An input/output port in a computer through which data is transmitted and received one bit at a time. In most cases, in personal computers, serial data is passed through an RS232C serial interface port.

Server

A computer that provides a service to other computers on a network. A file server, for example, provides files to client machines.

Server Farm

A group of connected servers that share responsibilities and are usually configured to allow processing to continue if one or more server crashes.

Service

Special kind of program that is "privileged" and operates at a very low level within the system. Services run automatically in the background, and they do not require a user to log on. Because the Windows NT operating system is a secure operating system, normal programs are not allowed to access hardware directly, such as a hard disk, or other system objects, such as the Event Lot. Service programs can access the hardware and system objects for other normal programs. For example, the Wonderware Logger, WindowViewer can both be run as NT services.

Spreadsheet

A program that arranges data and formulas in a matrix of cells. For example, Microsoft Excel.

Statement

An expression of instruction in a computer language.

Standalone

A single, self-contained computer system, as opposed to a computer that is connected to and dependent upon a remote computer system. A standalone computer will operate by itself, requiring no other equipment.

String

A connected sequence of characters or bits treated as a single data item.

Subdirectory

Subdirectories are located within Directories. They are a structure for organizing files into convenient groups. A subdirectory is like an address showing where files are located.

SuperTag

InTouch supports a template structure that allows you to define composite tagname types called SuperTags. SuperTag templates can contain up to 64 member tagnames and 2 nesting levels. See **TemplateMaker**.

Synchronous transmission

Data transmission in which the bits are transmitted at a fixed rate. The transmitter and the receiver both use the same clock signals for synchronization. See **Asynchronous**.

Syntax

The rules governing the structure of a language and its expressions.

Task

A task is an executing application. Task is a synonym for "process".

Tagname

The name assigned to a variable defined in the Tagname Dictionary (InTouch database).

TemplateMaker

InTouch utility that allows you to create SuperTag templates. See **SuperTag**.

Terminal

A device that allows you to send commands to another computer. At a minimum, this usually means a keyboard, a display screen, and some simple circuitry. You will usually use terminal software in a personal computer—the software pretends to be or emulates, a physical terminal and allows you to type commands to another computer.

Terminal Server

A server with a multi-user operating system the processes data for terminals.

Terminal Services Advanced Client

See **TSAC**

Text Box

A box where information needed to carry out a command is typed. A text box usually appears in a dialog box.

Thin Client

A terminal without a hard disk that is used to access a server.

Tiled Window

A tiled window is a window whose size, shape, and location on the screen is determined by Windows. Tiled windows are the only style of window that cannot overlap other windows, can be placed into the icon area and can have menus. Each application normally creates just one tiled window. All additional windows created by an application are normally cascading or popup windows.

Time slice

A unit of time.

Title Bar

The bar across the top of each window that contains the name of the application and the document or file being used by that application. (In **InTouch** an option exists to eliminate the Title bar.) Title bars are also used to move a window on the screen by grabbing it while dragging the mouse.

Toggle

Pertaining to any device having two stable states. Synonymous with flip-flop.

Touch-sensitive screen

A display screen on which the user can enter commands by pressing designated areas with a finger or other object.

Transmission Control Protocol/Internet Protocol (TCP/IP)

The protocol that networks use to communicate with each other on the Internet.

TSAC

Microsoft's Terminal Services Advanced Client is a Win32®-based ActiveX® control that can be used to run Terminal Services sessions within Microsoft® Internet Explorer. By using TSAC, you can now run full-featured InTouch applications across the Internet, with the same performance and speed as if you were on the local area network.

UNIX

A computer operating system designed to be used by many computer users at the same time (multi-user) with TCP/IP built in. The most common operating system for servers on the Internet.

Viewing Area

The viewing area (also called "Workspace") in Windows applications displays one page of a file. See **Workspace**.

Window

A rectangular area on the screen in which an application is viewed and worked. Multiple windows can be open on the screen at one time which can be sized and moved independently.

Windows

An operating environment developed by Microsoft.

Windows Application

An application that is designed especially for the Microsoft Windows operating environment and that uses all Windows features such as menus, scroll bars, and icons.

Workspace

The area of an application window that displays the application itself and all other open windows.

x-axis

On a coordinate plane, the horizontal axis.

y-axis

On a coordinate plane, the vertical axis.

Index