



Wonderware
ArchestrA™ Trend
Client User's Guide

No part of this documentation shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Schneider Electric Software, LLC. No liability is assumed with respect to the use of the information contained herein.

Although precaution has been taken in the preparation of this documentation, Schneider Electric Software, LLC assumes no responsibility for errors or omissions. The information in this documentation is subject to change without notice and does not represent a commitment on the part of Schneider Electric Software, LLC. The software described in this documentation is furnished under a license agreement. This software may be used or copied only in accordance with the terms of such license agreement.

© 2015 Schneider Electric Software, LLC. All rights reserved.

Schneider Electric Software, LLC
26561 Rancho Parkway South
Lake Forest, CA 92630 U.S.A.
(949) 727-3200

<http://software.schneider-electric.com/>

ArchestrA, Avantis, DYNsIM, EYESIM, Foxboro, Foxboro Evo, I/A Series, InBatch, InduSoft, IntelaTrac, InTouch, PIPEPHASE, PRO/II, PROVISION, ROMeo, Schneider Electric, SIM4ME, SimCentral, SimSci, Skelta, SmartGlance, Spiral Software, VISUAL FLARE, WindowMaker, WindowViewer, and Wonderware are trademarks of Schneider Electric SE, its subsidiaries, and affiliated companies. An extensive listing of Schneider Electric Software, LLC trademarks can be found at: <http://software.schneider-electric.com/legal/trademarks/>. All other brands may be trademarks of their respective owners.

Contents

	Welcome	9
	Documentation Conventions	9
	Technical Support	9
Chapter 1	About the Trend Client	11
	Understanding the Trend Client	11
	Differences between the Trend Client and the ActiveFactory Trend Control	12
	Data Sources for the Trend Client	12
Chapter 2	Configuring the Trend Client	15
	Placing the Trend Client into an ArcestrA Symbol	15
	Removing the Trend Client from an ArcestrA Symbol	16
	Validating the Configuration of the Trend Client	16
	Clearing the Configuration from the Trend Client	16
	Adding a Pen	17
	Connecting a Pen to an InTouch Tag	18
	Configuring Pen Details and Options	20
	Configuring Pens for Historical Sources and Tags	22
	Configuring Historical Sources	23
	Creating a Wonderware Historian Connection	24
	Creating an InTouch LGH Connection	26

Editing a Server Connection	27
Removing a Server Connection	27
Using the Tag Picker	28
Using the Servers Pane	29
Showing/Hiding the Servers Pane	29
Adding a Group	30
Adding a Tag to a Group	30
Deleting a Group or Tag Reference	30
Renaming a Group	31
Viewing Server Details	31
Using the Filter Pane	32
Using the Tags Pane	33
Configuring the Trend Appearance	34
Setting Chart Options	36
Setting Data Bindings	38
Handling Trend Events	40
Chapter 3 Scripting the Trend Client	41
Trend Client Properties	41
Chart.AddMultiplePens	44
Chart.BackgroundColor	44
Chart.Freeze	44
Chart.FreezeDurationMS	45
Chart.HidePenList	45
Chart.Labels	45
Chart.PenPrecision	46
Chart.RefreshEntireChartIntervals	46
Chart.RetrievalMode	46
Chart.UpdateRateMS	46
HistorySources	47
Pen.Color	47
Pen.Count	47
Pen.Description	48
Pen.Expression	48
Pen.Format	48
Pen.HistorySource	49
Pen.HistoryTagName	49
Pen.Index	49
Pen.Name	49
Pen.Precision	50
Pen.RetrievalMode	50
Pen.Style	50

Pen.TrendHi	51
Pen.TrendLo	51
Pen.TrendType	51
Pen.Units	52
Pen.Visible	52
Pen.Width	52
PenSelectorHeight	53
PenUQRelativeOpacity	53
PenUQRelativeThickness	53
PlotArea.BackgroundColor	54
PlotArea.BorderColor	54
PlotArea.GradientEndColor	54
PlotArea.GradientType	55
PlotArea.GridColor	55
PlotArea.GridHorizontal	56
PlotArea.GridStyle	56
PlotArea.GridVertical	56
PlotArea.GridWidth	57
PlotArea.HighlightCurrentPen	57
PlotArea.PenHighlightColor	57
PlotArea.PenHighlightWidth	58
PlotArea.SingleTagMode	58
PlotImage	58
ShowContextMenu	58
SuppressErrors	59
TimeAxis.Cursor1.Color	59
TimeAxis.Cursor1.Pos	59
TimeAxis.Cursor1.Style	60
TimeAxis.Cursor1.Width	60
TimeAxis.Cursor2.Color	60
TimeAxis.Cursor2.Pos	61
TimeAxis.Cursor2.Style	61
TimeAxis.Cursor2.Width	61
TimeAxis.LabelColor	62
TimeAxis.NumGridPerValue	62
TimeAxis.NumValues	62
TimeAxis.ShowCursors	63
TimeSelector	63
TimeSelector.DurationMS	63
TimeSelector.EndDate	64
TimeSelector.StartDate	64
TimeSelector.TimeDuration	64
ToolTipText	67

TrendVersion	67
ValueAxis.Label	67
ValueAxis.NumGridPerValue	67
ValueAxis.NumValues	68
Visible	68
Trend Client History Sources	68
HistorySource.Authentication	69
HistorySource.Domain	69
HistorySource.Password	69
HistorySource.RetainPassword	69
HistorySource.ServerName	70
HistorySource.Type	70
HistorySource.UNCPATH	70
HistorySource.UserID	71
HistorySources.Count	71
HistorySources.Items	71
Trend Client Methods	72
AddHistorianSource	73
AddPen	73
ClearPens	74
DeleteCurrentPen	75
GetHistorianSource	75
GetPenValAtX1	75
GetPenValAtX2	76
GetStartAndEndTimes	76
MoveNextPen	77
MovePrevPen	77
RefreshData	78
RefreshTimes	78
RemoveHistorianSource	79
RemovePen	79
ScaleAllPens	79
ScaleAutoAllPens	80
ScaleAutoPen	80
ScaleDownAllPens	81
ScaleDownPen	81
ScaleMoveAllPensDown	82
ScaleMoveAllPensUp	82
ScaleMovePenDown	82
ScaleMovePenUp	83
ScalePen	83
ScaleUpAllPens	84
ScaleUpPen	84

SetCurrentPen	84
SetDuration	85
SetStartAndEndTimes	85
TimeSelector.GetStartAndEndTimes	86
TimeSelector.RefreshTimes	87
TimeSelector.SetStartAndEndTimes	87
UpdateHistorianSource	88
Trend Client History Sources	88
HistorySource.Connect	89
HistorySource.Disconnect	89
HistorySources.Add	90
HistorySources.GetSource	90
HistorySources.Remove	91
HistorySources.Update	91
Trend Client Events	92
CurrentPenChanged	92
DatesChanged	92
MouseClicked	92
PenDisplayChanged	92
PenlistChanged	93
ShutDown	93
SizeChanged	93
Startup	93
StateChanged	93
Colors in Trend Client	94
.NET Colors	94
Scripting Differences between Trend Client and ActiveFactory Trend Control	98
 Appendix A Understanding Data Retrieval	 103
Understanding Retrieval Modes	103
Cyclic Retrieval	103
Full Retrieval	105
 Index	 107

Welcome

You use the ArchestrA Trend Client in ArchestrA symbols to show real-time values on a chart. The ArchestrA Trend Client is delivered as part of the ArchestrA Symbol Editor.

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

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

Documentation Conventions

This documentation uses the following conventions:

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

Technical Support

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

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

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

Chapter 1

About the Trend Client

To use the Trend Client, you should already have a basic understanding of the Wonderware Integrated Development Environment (IDE), the ArcestrA Symbol Editor, and the InTouch HMI before continuing. For more information, see the documentation for each of these products.

Understanding the Trend Client

Use the Trend Client to chart attribute current values from Wonderware Application Server and tag current values from the InTouch HMI software. This differs from the trend feature in the InTouch HMI because Trend Client fully supports ArcestrA and can initialize a trend pen with data from a historical source such as InTouch LogHistory/LGH files or the Wonderware Historian.

Note: The InTouch trends are still included as part of the InTouch HMI.

Differences between the Trend Client and the ActiveFactory Trend Control

If you use the ActiveFactory Trend control, you need to be aware that there are several differences between the Trend Client and the ActiveFactory Trend control. In contrast to the ActiveFactory Trend control, the Trend Client:

- includes the real-time as well as historical trending functionality.
- uses the data from a historical source such as InTouch LogHistory/LGH files or the Wonderware Historian.
- has fewer pens.
- uses real-time attribute or tag data as well as historized data stored in the Wonderware Historian or InTouch LogHistory/LGH files.
- does not require a separate license from the Wonderware Application Server.
- requires the InTouch HMI.
- only supports the full and cyclic retrieval modes of the Wonderware Historian when retrieving data.
- does not support configuring the default pen colors (although you can change them after you add a pen).
- does not provide scatter plots and target regions.

Data Sources for the Trend Client

The Trend Client can use real-time data from Wonderware Application Server and the InTouch HMI software and historical data from the Wonderware Historian or InTouch LGH files. You identify the historical data source when you configure a pen. For more information, see [Configuring Historical Sources](#) on page 23.

At run time, the Trend Client initializes the trend with historical data from the data source (if configured) and then appends real-time data for the pens. If the pen is not associated with a historical data source, the trend initially shows no data. It then starts charting information as real-time data becomes available. If you refresh a pen configured with historical data, the chart requeries the historical information, after which it appends the real-time data as it becomes available. Refreshing does not change a pen configured only for real-time data. If you refresh a trend that is charting both historical and real-time data, the historical information is recharted immediately, after which the trend charts real-time data as it becomes available.

The Trend Client supports the same syntax as the ArcestrA symbols. It also supports the built-in functionality of the `OwningObject` property for trends configured with relative naming. For information about ArcestrA symbols data configuration syntax and `OwningObject` functionality, see the *Creating and Managing ArcestrA Graphics User's Guide*. For more information on relative and absolute references, see the *Wonderware Application Server User's Guide*.

There are primary settings for specifying the time period you want to plot, such as a relative reference (the preceding two hours), or an absolute reference (July 15th at 10am for 2 hours) using scripting.

As an example, suppose you opened two trends created with the Trend Client. Both of them have a duration of 15 minutes, but the first one is only trending real-time data (meaning that the pen is associated with an expression but not with a historical tag) and the second is trending historical data (meaning that the pen is associated with both an expression and a historical tag). If you open both of these trends at the same time, what you see is described in the following table.

Time after starting both trends	What you see
0 minutes	The first trend shows no data. The second trend shows 15 minutes of historical data from the source.
5 minutes	The first trend shows 5 minutes of real-time data. The second trend shows 10 minutes of the historical data and 5 minutes of real-time data.
15 minutes	Both charts show 15 minutes of real-time data. All the historical data will have been pushed off the second trend by the new real-time data.

Chapter 2

Configuring the Trend Client

This section shows you how to place a Trend Client on the canvas and configure it. You can configure it either with the **Edit Animations** dialog box, or by changing individual properties in the Properties Editor

Placing the Trend Client into an ArcestrA Symbol

You can easily place the ArcestrA Trend Client into an ArcestrA Symbol by placing it onto the canvas.

To place the Trend Client into an ArcestrA Symbol



- 1 Open the ArcestrA symbol.
- 2 On the **Tools** panel, click the Trend Client icon. The cursor appears in insert mode.
- 3 Click on the canvas where you want to place the upper left corner of the Trend Client.

Removing the Trend Client from an ArcestraA Symbol

You can remove the Trend Client from the canvas.

To remove the Trend Client from an ArcestraA symbol

- 1 Open the symbol you want to remove the Trend Client from.
- 2 Do either of the following:
 - Select the Trend Client and press **DELETE**.
 - Right-click the Trend Client and select **DELETE** from the menu.

Validating the Configuration of the Trend Client

You can validate the configuration of a Trend Client. If the configuration has an error, an exclamation mark appears next to the **Animation** icon.

Examples of errors include:

- Syntax errors in expressions
- Data type mismatches
- Required values not specified
- Specified values out of valid range

To validate the configuration of a Trend Client

- 1 Open the symbol containing the Trend Client.
- 2 Double-click the Trend Client. The **Edit Animations** dialog box appears.
- 3 Click the **Validate** icon on the right side of the **Edit Animations** dialog box. The Trend Client is validated. Possible errors are highlighted in the right side of the dialog box.



Clearing the Configuration from the Trend Client

You can clear all configuration data from the configuration boxes of and reset the settings to the defaults.

To clear all configuration data

- ◆ In the configuration panel, click the **Clear animation** icon.

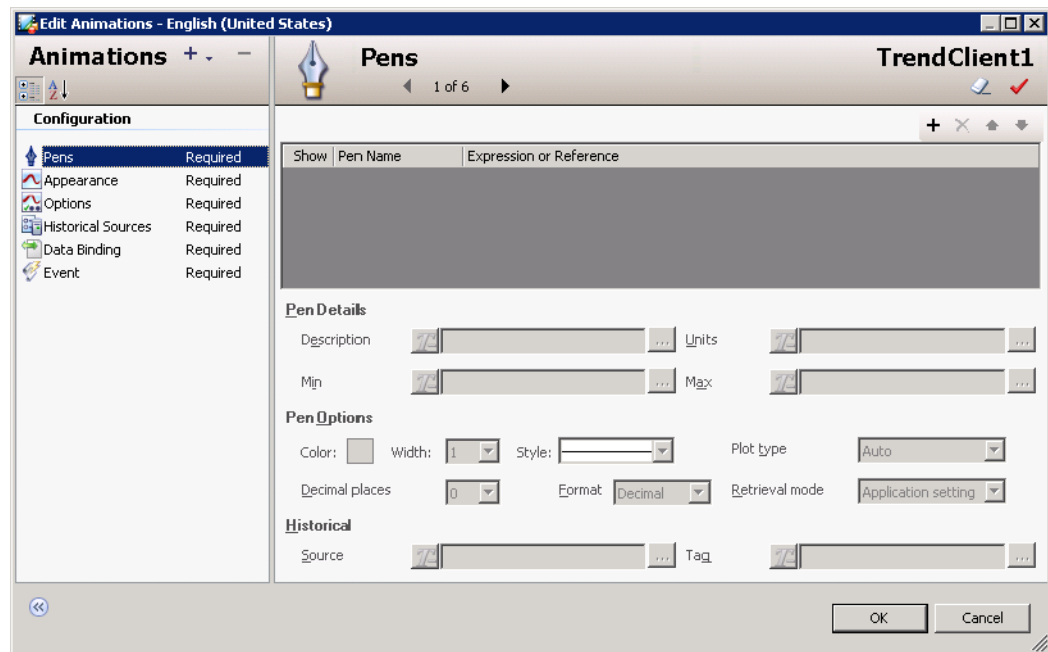


Adding a Pen

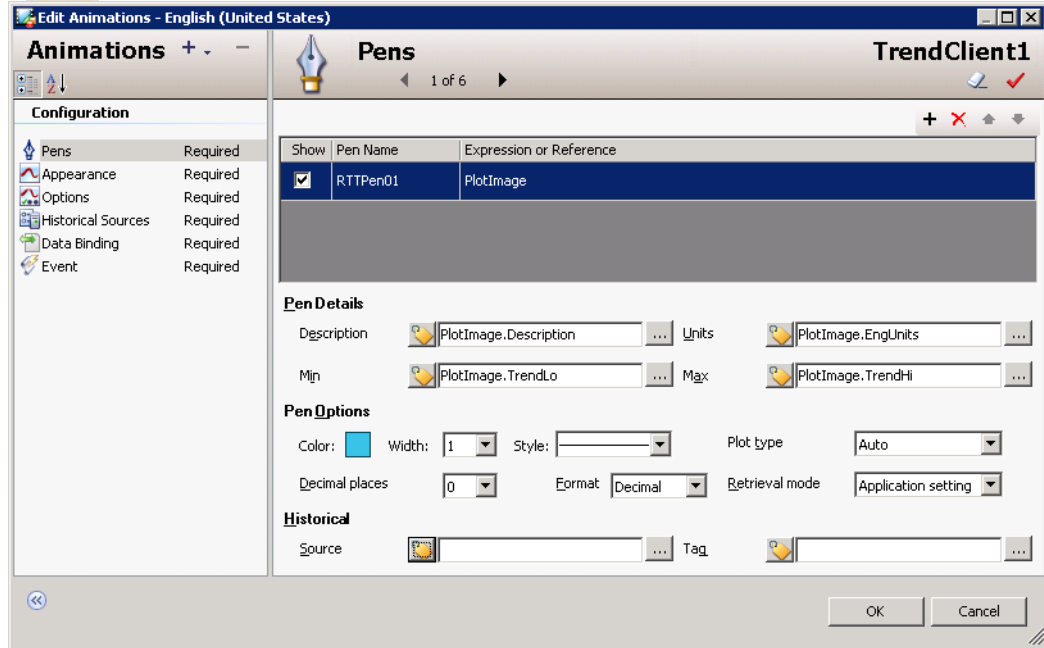
When you first add a Trend Client to the canvas, no pens are defined. To show a trend, you must add at least one pen and configure it so that the system can identify what trend information to chart. Each pen must have an associated expression or reference.

To add a pen

- 1 In the ArchestrA Symbol Editor, double-click the Trend Client. The **Edit Animations** dialog box appears with the pen configuration information in the right pane.



- 2 Click the **Add** icon on the right side of the **Edit Animations** dialog box. A new pen appears in the configuration panel.



- 3 In the row for the pen, configure the information for the new pen.

Show	Select the Show box to show the pen in the trend.
Pen Name	Type the name of the pen. (If you leave this blank, the pen's expression is used as the chart label.)
Expression or Reference	Type an expression or reference. Click the ellipsis button to show the Galaxy Browser.

To reorder pens



- ◆ Click the Up/Down icons to reorder the pens you have added.

Connecting a Pen to an InTouch Tag

You can connect the pen to an InTouch tagname through the **Expression or Reference** box. The InTouch tagname provides values at run time that control the animation and appearance of the pen.

This can be done by:

- Configuring a reference with the **intouch:tagname** syntax.
- Using a custom property and configuring the custom property in the embedded ArchestrA Symbol in InTouch to reference an InTouch tag. For more information, see the *InTouch HMI and ArchestrA Integration Guide*.

- Configuring an ArcestraA attribute reference to the managed InTouchViewApp object that contains the InTouch tagnames as attributes. The InTouchViewApp object uses the functionality of an InTouchProxy object.
- Configuring an ArcestraA attribute reference to an InTouchProxy object that contains the InTouch tagnames as items. This is a special case of configuring an ArcestraA attribute reference.

To be able to browse for InTouch tags, you must first:

- Create a managed InTouch application by deriving an InTouchViewApp template and configuring it in WindowMaker.
- Derive an instance of the InTouchViewApp derived template.

The InTouch tags are represented by attributes of the InTouchViewApp object instance.

You may not always be able to reference the local node if you use this method of connecting to the InTouch HMI software. As a result, you may have problems with some applications. If this is the case, use the **intouch:tagname** syntax.

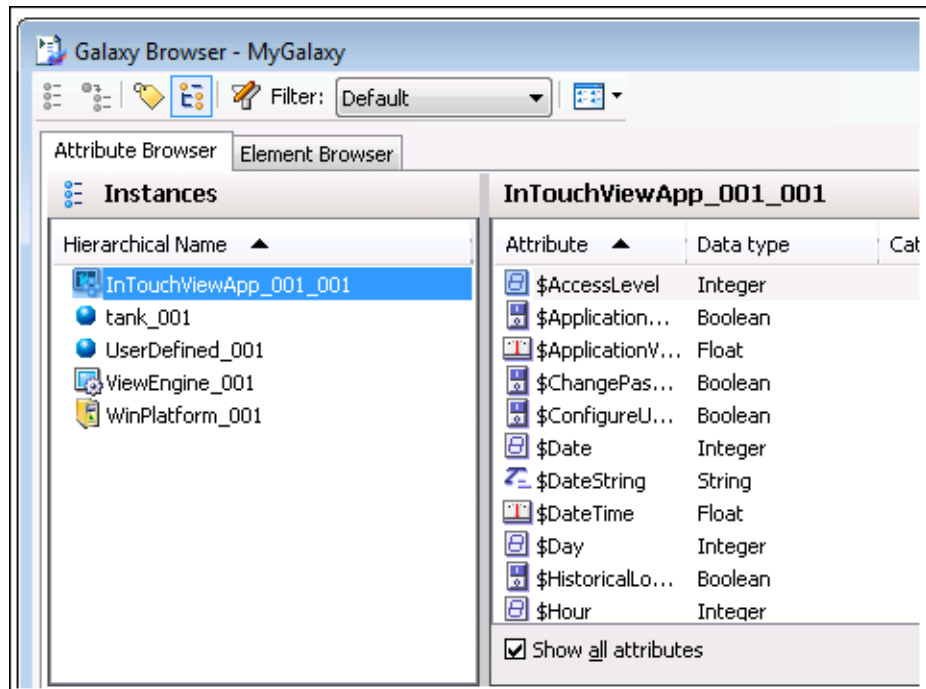
With the **intouch:tagname** syntax, the pen connects to the InTouch tag on the symbol's node. There are some restrictions on how you can use this syntax:

- Unlike in Application Server, you cannot use TRUE and FALSE as Boolean values. Use 1 and 0 instead.
- If you want to connect to an InTouch SuperTag with this syntax, use the following syntax instead:

```
attribute("intouch:SuperTag\Member")
```

To connect Trend Client to InTouch tags

- 1 In the Galaxy Browser, select the InTouchViewApp object that corresponds to the managed InTouch application. The right panel shows the InTouch tags.



- 2 Select a tag and click **OK**. The selected ArchestrA reference to an InTouch tag appears in **Expression or Reference**.

Configuring Pen Details and Options

After you add a pen and configure the expression or reference for the pen(s), you can set details and options for how the pen shows information on the trend chart. Pen details include the description, units, minimum and maximum for the pen. Pen options include the pen color, width, style, plot type, number of decimal places, format, and retrieval mode.

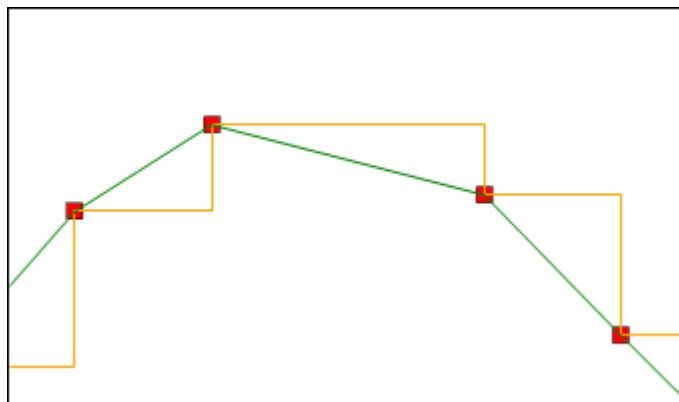
The default values for the pen details are formed by appending .Description, .EngUnits, .TrendLo, and .TrendHi to the pen expression. For example, if the pen expression is Me.PV for a field attribute named PV created on the user defined object, the default pen details are Me.PV.Description, Me.PV.EngUnits, Me.PV.TrendLo, and Me.PV.TrendHi. These are generally the best expressions to use for historized attributes. (If the expression is empty or invalid, the Trend Client uses default values of 0 and 100 for the minimum and maximum at run time.)

Note: For some input boxes on the configuration screens, you can specify if the configuration is a static value or a reference by setting the input mode. An input mode icon appears to the left of these boxes. Use static mode input to specify a literal static value such as "Temperature" or 3.141. Use reference mode to specify a reference to an attribute or symbol property such as Tank_001.PV. You can enclose static string values with double-quotes (such as "Description:" or "+Tank_001.Desc") with or without references in Reference mode. When you are in reference mode, you can also click the ellipsis button to the right of the box to show a selection tool such as the Galaxy Browser or the Tag Picker.

A line curve (plot type **Line**) is best suited for charting continuously-changing analog data. A step-line curve (plot type **Step**) is best suited for discrete data and box for analog data that is not continuous. When you select a plot type of **Auto**, the curve type is determined as follows:

- For tags retrieved from Wonderware Historian 9.0 (or higher), the type is based on the tag's effective interpolation setting. Tags that use stair-step interpolation are trended as a step line, and tags that use linear interpolation are trended as a line.
- For all other tags, the curve type is based on the tag type: step line for integer tags, and line for real tags.

The following illustration shows the same data drawn using each type of curve. The line curve is shown in green, the step line curve is shown in orange, and the point curve is shown in red.



To configure the options for a pen

1 Select the pen you want to configure.

2 In the **Pen Details** area, configure what the pen shows.

Description	The pen's description. The expression is evaluated and the result is used for the pen's description in the pen list (the legend) below the chart itself.
Units	The unit to chart in. This is the plot scale, rather than an instrument scaled from a percentage to engineering units.
Min	Type the minimum for the pen's range.
Max	Type the maximum for the pen's range.

3 In the **Pen Options** area, configure how the curve looks in the chart for the selected pen.

Color	The line color of the pen. Click the colored square to select the color from a palette or define a custom color.
Width	The thickness of the trend curve. Valid values are 0 through 10.
Style	The style of the trend curve; for example, a solid or dashed line.
Plot type	The type of trend curve to draw. Options are Line , Step line , Point , and Auto .
Decimal Places	The number of decimal places to show for the data value. This applies only to analog tags. Valid values are 0 through 15.
Format	The way the values for the pen appear, either in decimal format or scientific format.
Retrieval Mode	The data retrieval mode, either Cyclic , Full , or Application Setting .

Configuring Pens for Historical Sources and Tags

When you start up the Trend Client at run time with historical data and then plotting real-time data, you can configure a pen to initialize with a specific historical source and tag. For example, `MyEngine.Engine.Historian.Connection` is a common relative reference expression for the historical source on a template and, for a pen expression of `Me.PV`, `Me.TagName.PV` is a common relative reference expression for the historical tag.

If the historical tagname needs to use relative references, construct the string expression for the historical tagname that evaluates the relative part of the name and append a static part. For example, to reference a UDA named PV, use `Me.Tagname+".PV"` or a similar expression. You can define this expression on a symbol in a `$Reactor` template. There are instances of Historian tagnames are `Reactor1.PV` and `Reactor2.PV`. If you use the expression in the form `Me.PV`, it resolves to the current value instead of the tagname. For example, if PV is associated with a temperature, `Me.PV` would return a temperature rather than the string for the tagname.

You can also use valid relative references such as `Me`, `MyContainer`, `MyArea`, `MyEngine`, and `MyPlatform` when configuring the historical tag-names as relative references. For example, to reference the UDA named "Level" in a user-defined object named `$Reactor`, you can configure the historical tagname as an expression: `Me.TagName+".Level"`. At run time, when the object "Reactor_001" is viewed, the expression is resolved to "Reactor_001.Level" as the historical tag name.

To configure a pen for a historical source and tag

- 1 Select the pen you want to configure.
- 2 In the **Historical** area, configure the source and tag for the pen.

Source The historical source to use for the pen. This history source must already be configured in the **Historical Sources** pane. For more information, see "Configuring Historical Sources" on page 23.

If this is set to reference mode, click the ellipsis button to show the Galaxy Browser and use that to find the expression or reference.

Tag The historical tag to use for the pen. If this is set to reference mode, click the ellipsis button to show the Tag Picker. For more information, see "Using the Tag Picker" on page 28.

Note: To use the reference mode, you must have already configured a historical data source for the Trend Client. For more information, see "Configuring Historical Sources" on page 23.

Configuring Historical Sources

To use a Trend Client to chart historical data, you must connect to a data source. You can connect to a Wonderware Historian or an InTouch LGH file.

You can use one of the following authentication modes when connecting to Wonderware Historian:

- Windows Integrated Security (uses the running process credentials)
- Alternate Windows account (specifying the domain name, user name, and password).

The alternate credentials are used to do a Windows impersonation before connecting to the database using Windows Integrated Security.

- SQL Server Authentication

You specify the user name and password of a SQL Server account. These credentials are used to log in to the database.

Note: Ask your administrator what type of user account you must use to access the server.

Server connections are specific to the instance of the client and must be re-defined for each instance.

When you start a Wonderware Application Server application, you are not automatically logged on to every server that you configured before. You are only logged on to a server when a pen is configured to use that server.

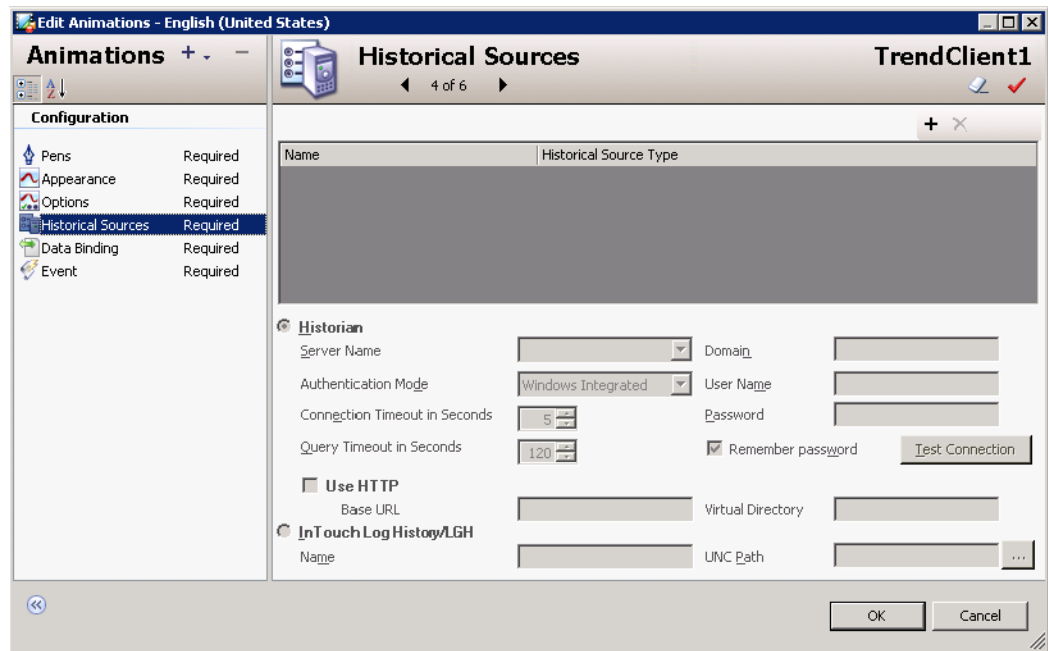
You need not run as an administrative user to operate in run time within an InTouch application. The user access control is enabled for you on Windows Vista, Windows Server 2008 and Windows 7 operating systems.

Creating a Wonderware Historian Connection

To create a Wonderware Historian connection, you need your assigned Wonderware Historian user name and password.

To create a Wonderware Historian connection

- 1 In the ArchestrA Symbol Editor, double-click the Trend Client. The **Edit Animations** dialog box appears
- 2 Click **Historical Sources**. The **Historical Sources** screen appears in the right pane.



- 3 Click the **Add** icon on the right side of the **Edit Animations** dialog box. A new server row appears in the configuration panel.
- 4 In **Server Name**, type the name of the server to which you want to connect. You can select from a list of the servers by clicking the arrow to the right of the box.

- 5 In **Authentication Mode**, select the authentication mode: Windows Integrated, Windows Account, or SQL Server. If you are logging on to the server using Windows Account credentials, configure the following login detail:

Domain The domain name in which your Windows account is validated.

If you are logging on to the server using Windows Account or SQL Server credentials, configure the following login details:

User Name Your assigned Wonderware Historian user name. If your system administrator has not assigned you a user name and password, you may use one of the default user accounts, which are automatically configured during a typical Wonderware Historian installation.

Password The password that is associated with the user name. Select **Remember password** to have the system remember your password.

- 6 In **Connection Timeout in Seconds**, type the connection timeout in seconds. Valid values are 1 to 600.
- 7 In **Query Timeout in Seconds**, type the query timeout in seconds. Valid values are 1 to 600.

Note: You can test the connection to the server with the information you've entered by clicking **Test Connection**. If necessary, you can edit the information before adding the server to the list.

- 8 Click **OK**.

Note: The Trend Client does not validate the connection when you click **OK**.

Creating an InTouch LGH Connection

You can get data from InTouch LGH files by connecting to them.

To create an InTouch LGH connection

- 1 In the ArchestrA Symbol Editor, double-click the Trend Client. The **Edit Animations** dialog box appears
- 2 Click **Historical Sources**. The **Historical Sources** screen appears in the right pane.
- 3 Click the **Add** icon on the right side of the **Edit Animations** dialog box. A new connection entry appears in the configuration panel.
- 4 Select **InTouch Log History/LGH**.



- 5 In **Name**, type the InTouch HMI software server.
- 6 In **UNC Path**, type the location of the InTouch LGH file using a UNC path name or a mapped drive path.
- 7 Click **OK**.

Note: The Trend Client does not validate the connection when you click OK.

Editing a Server Connection

You can edit an existing server connection.

To edit a server connection

- 1 On the **Historical Sources** screen, select the server you want to edit.
- 2 Edit the details for the server.
- 3 Click **OK**.

Removing a Server Connection

You can remove a server connection you no longer need.
After you delete a server, you cannot undelete it.

To remove a server connection

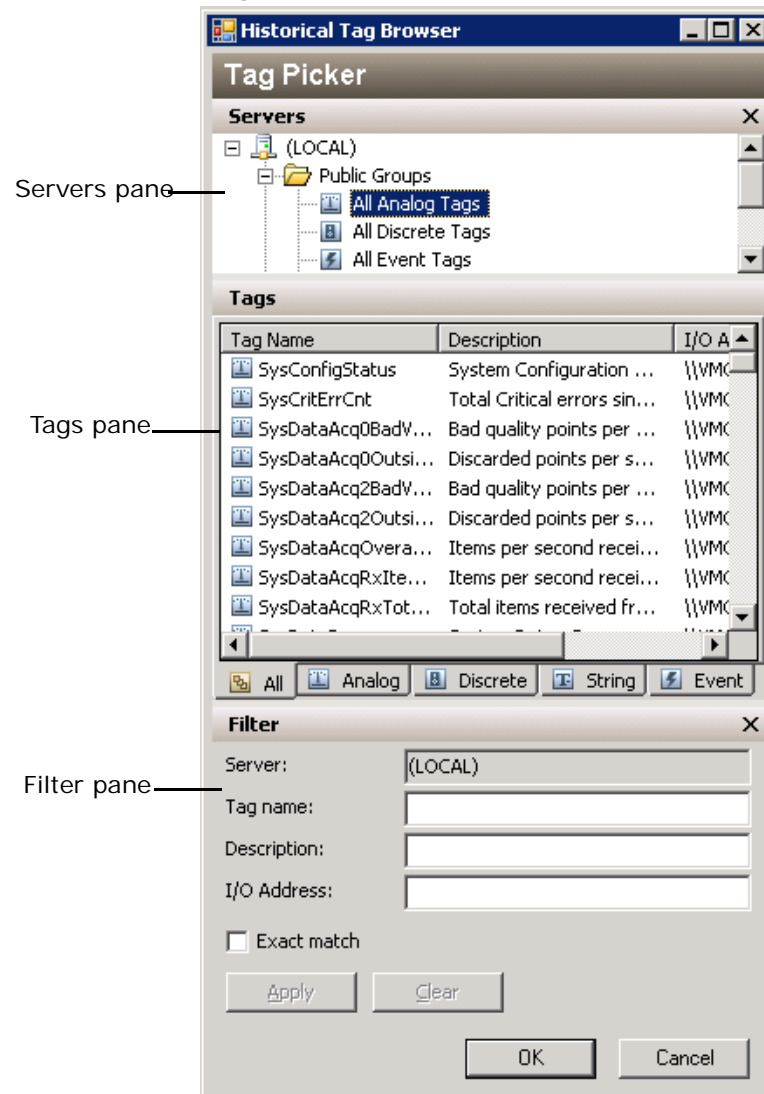
- 1 On the **Historical Sources** screen, select the server you want to delete.
- 2 Click the **Delete** icon on the right side of the **Edit Animations** dialog box. The server is deleted.



Using the Tag Picker

The Tag Picker shows which tag groups and tags exist in the Wonderware Historian database. It shows all tags that are visible to the currently logged-on user based on his permissions.

Using the Tag Picker, you can quickly search the database for tags of a certain type or for tags that match a particular search pattern. You can then select the ones you want to include in the Trend Client. You cannot select a tag from an LGH file.



For instructions to import the Tag Picker control and embed it in a symbol, see "Chapter 12 Using Client Controls" of the *Creating and Managing ArchestrA Graphics User's Guide*.

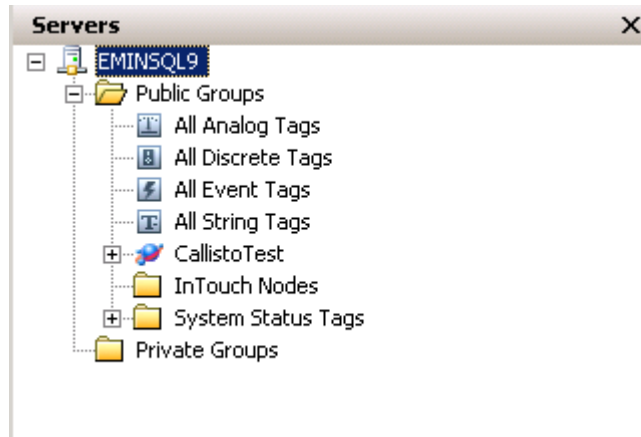
The Tag Picker includes the following panes:

- Servers pane
- Tags pane

- Filter pane

Using the Servers Pane

The **Servers** pane shows a list of Wonderware Historian folders. The **Servers** pane allows you to navigate through the folder structure (namespace) of one or more Wonderware Historians and select a group (folder) of tags.



The **Servers** pane shows the following items:

Category	Description
Servers	All objects that make up the basic Wonderware Historian system, such as tags, I/O Servers, defined engineering units, storage locations, and so on.
Public Groups	All objects that are visible to all clients. If you have administrative permissions, you can create, rename, and delete groups in the public groups folder.
Private Groups	All objects that are visible to the user that is currently logged on. Users can create, rename, and delete groups in the private groups folder.

Showing/Hiding the Servers Pane

To show the Servers pane

- Right-click in the **Servers** pane and then click **Servers pane** so that a check mark appears.

To hide the Servers pane

- Do one of the following:

- Right-click in the **Servers** pane and then click **Servers pane** so that no check mark appears.
- Click **Close**.

Adding a Group

You can add groups just as you would add a new folder in the Windows Explorer. For example, you can create the BoilerTags group under in the existing Private Groups group. You can also delete, cut, copy, paste, and drag objects from one folder to another.

To add a group

- 1 Right-click on the folder under which you want to create a group and then click **New Group**.

A new folder appears in the Tag Picker.

- 2 Type a name for the folder and press **ENTER**.

Adding a Tag to a Group

When you add tags to a new group, the original reference still appears in the default system group. Any tag can belong to any number of groups, and any group can contain any number of tags.

To add a tag to a group

- 1 Select the system group folder that contains the tag that you want to add to your new group.
- 2 In the **Tags** pane, select the tag to add.
- 3 Do any of the following:
 - Drag the tag from the **Tags** pane into the folder.
 - Use the **Copy** and **Paste** commands on the **Edit** menu to copy the tag to the target folder.
 - Right-click on the tag in the **Tags** pane. Use the **Copy** and **Paste** commands in the shortcut menu to copy the tag to the target folder.

Deleting a Group or Tag Reference

When you delete a private group or a tag reference in a private group, the group folder, any subfolders that the group folder may contain, and all references to tags are deleted. The tags themselves are not deleted, and the original references still appear in the default system group. You cannot delete public folders or the tag references contained in them.

To delete a group or tag

- 1 Select the group or tag in the pane.
- 2 Do one of the following:
 - Right-click on the group or tag and then click **DELETE**.
 - Press **DELETE**.

Renaming a Group

You can rename a group that you have created in the Tag Picker. However, you cannot rename a public folder.

To rename a group

- 1 Select the group in the pane.
- 2 Do one of the following:
 - Right-click on the group and then click **RENAME**.
 - Press **F2**.
- 3 Type a new name for the group and press **ENTER**.

Viewing Server Details

You can view information such as the version number, time zone, and security mode for any Historian server in the **Servers** pane.

To view server details

- 1 In the **Servers** pane, right-click on a Historian server and then click **Server details**. The **Server Details** dialog box appears.



- 2 Click **OK**.

Using the Filter Pane

Use the **Filter** pane to reduce the tags listed in the **Tags** pane according to criteria that you specify. You can filter the tags according to name, description, and I/O address.

The filter mechanism allows for the following wildcard characters as part of the filter criteria:

Wildcard Character	Filter Function
%	Any string of zero or more characters.
_	Any single character.
[]	Any single character within the specified range or set. For example: <ul style="list-style-type: none"> • [a-f] • [abcdef]
[^]	Any single character not within the specified range or set. For example: <ul style="list-style-type: none"> • [^a - f] • [^abcdef]

For example, to find all tagnames ending with “level”, type **%level**. Filter criteria are not case-sensitive.

When the **Servers** pane and the **Filter** pane are both visible, the filter conditions apply to the selected group in the **Servers** pane. When the **Servers** pane is hidden, the filter applies to all of the tags for the selected Wonderware Historian.

To apply a filter

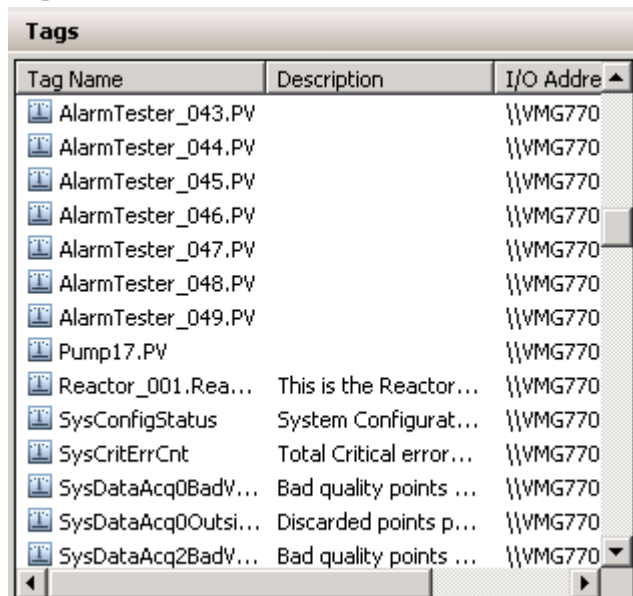
- 1 In the **Server** box, select the server.
This box is not available if the **Servers** pane is visible.
- 2 In the **Tag name** box, type the string to match for the tagname.
- 3 In the **Description** box, type the string to match for the description.
- 4 In the **I/O Address** box, type the string to match for the I/O address.
- 5 Select the **Exact match** check box to search for tags that exactly match the entire string that you provided for the tagname and description options.
For example, if you specify “level” as the tagname and do not select **Exact match**, any tagname that contains the string “level” appears. For example, “ReactLevel,” “ProdLevel,” and “\$AccessLevel.”

Note: The **Exact match** option does not apply to the I/O address.

- 6 Click **Apply** to apply the filter criteria.
- 7 Click **Clear** to clear the **Filter** pane.

Using the Tags Pane

The **Tags** pane shows all the tags for the currently selected group in the **Servers** pane.



Tag Name	Description	I/O Address
AlarmTester_043.PV		\\VMG770
AlarmTester_044.PV		\\VMG770
AlarmTester_045.PV		\\VMG770
AlarmTester_046.PV		\\VMG770
AlarmTester_047.PV		\\VMG770
AlarmTester_048.PV		\\VMG770
AlarmTester_049.PV		\\VMG770
Pump17.PV		\\VMG770
Reactor_001.Rea...	This is the Reactor...	\\VMG770
SysConfigStatus	System Configurat...	\\VMG770
SysCritErrCnt	Total Critical error...	\\VMG770
SysDataAcq0BadV...	Bad quality points ...	\\VMG770
SysDataAcq0Outsi...	Discarded points p...	\\VMG770
SysDataAcq2BadV...	Bad quality points ...	\\VMG770

To select multiple tags in the list, hold **Ctrl** and/or **Shift** while clicking.

To view only tags of a certain type, click the appropriate tab at the bottom of the pane.

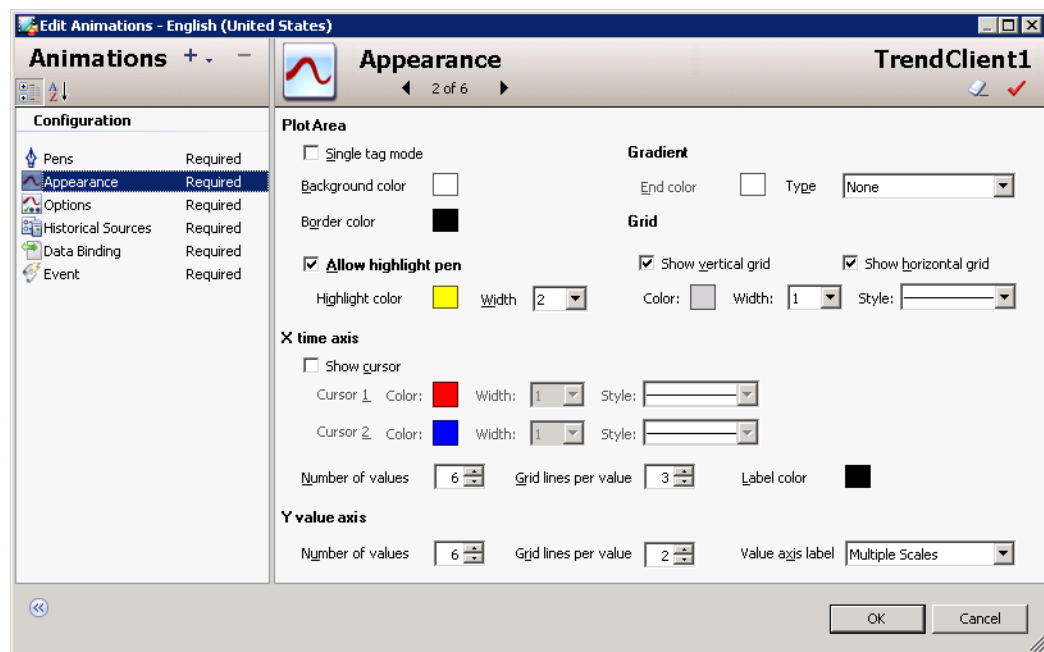
To sort the table by a particular column, click the column heading.

Configuring the Trend Appearance

You can set a number of general options for the trend appearance, including gradients, border and plot area background color, highlights, and colors for the X and Y axes.

To configure the trend appearance

- 1 In the ArchestrA Symbol Editor, double-click the Trend Client. The **Edit Animations** dialog box appears
- 2 Click **Appearance**. The trend appearance information appears in the right pane.



- 3 Select **Single tag mode** to set the trend to single tag mode.

When you initially create a tag list for a trend, all the tags are included in the display. Setting the trend to single tag mode allows you to exclude all tags but one from appearing in the trend chart, without removing them from the tag list.

Clear **Single tag mode** to view multiple tags again in the chart.

- 4 Click **Background color** to configure the main color of the background of the plot area. If you are using a gradient fill, this is the starting color for the gradient.
- 5 Click **End color** to select the ending color for the gradient. The gradient starts with the main color and fades to the gradient end color.

- 6 Click **Type** to specify the starting point for the flow of the gradient. Valid values are **LeftRight**, **TopBottom**, **Center**, **DiagonalLeft**, **DiagonalRight**, **HorizontalCenter**, and **VerticalCenter**. For example, if you select green as main color, white as the gradient end color, and center as the gradient type, the center of the chart is green and fades to white towards the surrounding edges.
- 7 Click **Border Color** to configure the color of the border for the entire chart area.
- 8 Select **Allow highlight pen** to highlight the pen. Configure the color and width to be used for pen highlighting.

Highlight color	Click to select or configure a color for highlighting the pen curve.
-----------------	--

Width	Specify the width (in pixels) of a highlighted curve.
-------	---

- 9 In the **Grid** area, configure the grid options.

Show horizontal grid	Check to show the horizontal grid.
----------------------	------------------------------------

Show vertical grid	Check to show the vertical grid.
--------------------	----------------------------------

Color	Click to select or configure a color.
-------	---------------------------------------

Width	The width, in pixels, of the border line.
-------	---

Type	The style of the border line.
------	-------------------------------

- 10 In the **X time axis** area, configure the properties for the horizontal axis.

Show horizontal cursor	Check to show the horizontal cursor.
------------------------	--------------------------------------

Cursor 1 color	Click to select or configure the color for time axis cursor 1.
----------------	--

Width	The width of time axis cursor 1.
-------	----------------------------------

Style	The line style of time axis cursor 1.
-------	---------------------------------------

Cursor 2 color	Click to select or configure the color for time axis cursor 2.
----------------	--

Width	The width of time axis cursor 2.
-------	----------------------------------

Style	The line style of time axis cursor 2.
-------	---------------------------------------

Number of values	The number of values that are shown along the time axis. The values are shown at evenly-spaced points along the axis. The number of values remain the same even if you zoom in and out. The valid range is from 2 to 15, with a default of 6.
------------------	---

Grid lines per value	The number of grid lines that appear between each pen value plotted on the chart. The valid range is from 1 to 20, with a default of 3.
----------------------	---

Label color	The color of the X axis label.
-------------	--------------------------------

11 In the **Y axis** area, configure the properties for the vertical axis.

Number of values	The number of values that are shown along the value axis. The timestamps are shown at evenly-spaced points along the axis. The number of values remain the same even if you zoom in and out. The valid range is from 2 to 15, with a default of 6.
------------------	--

Grid lines per value	The number of grid lines appearing between each pen value that is plotted on the chart. The valid range is from 1 to 20, with a default of 2.
----------------------	---

Value axis label	The label of the value axis.
------------------	------------------------------

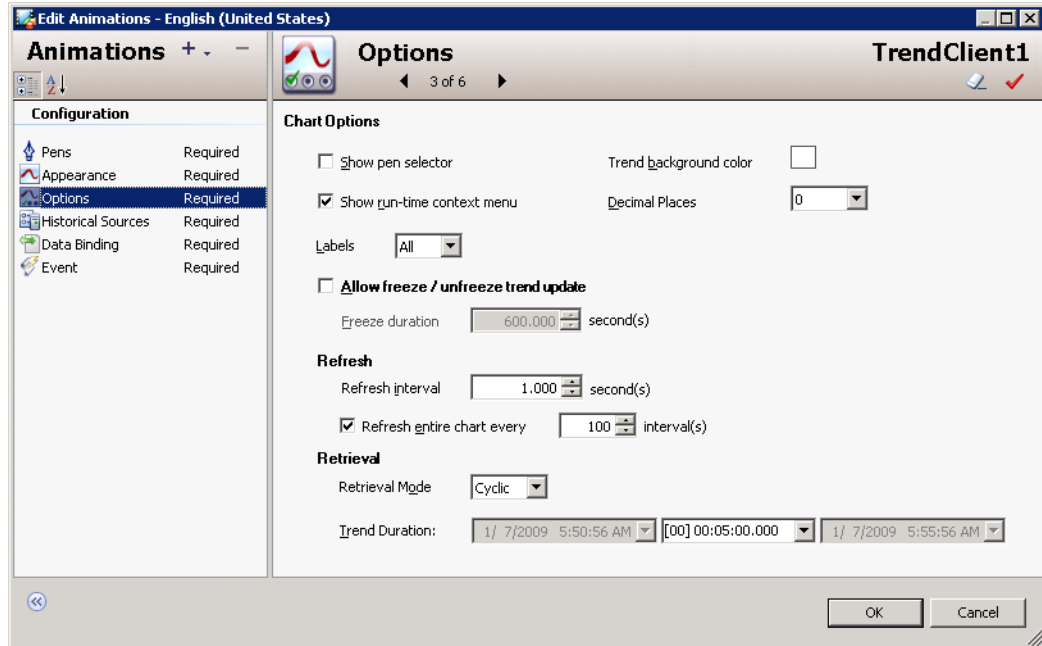
12 Click **OK**.

Setting Chart Options

You can set a number of general chart options, including the trend background color, freeze options, refresh interval, and retrieval options.

To set the chart options

- 1 In the ArchestrA Symbol Editor, double-click the Trend Client. The **Edit Animations** dialog box appears.
- 2 Click **Options**. The options information appears in the right pane.



- 3 Select the **Show pen selector** check box to show the legend or list of pens at the bottom the trend.
- 4 Click **Trend background color** to configure the main color of the background of the entire chart area.
- 5 Select **Show run-time context menu** to show the context menu when you right-click the trend at run time.
- 6 Set **Decimal Places** to the number of decimal places to use in the trend.
- 7 Set **Labels** to All to display the chart label, X and Y axis scales, and cursor information, or None to suppress the labels in the plot area
- 8 Select **Allow freeze/unfreeze trend update** to enable Freeze on the run-time context menu so you can freeze the trend chart from live updates. When this option is selected (the default), you can also set the freeze duration. The freeze duration determines how long the chart remains frozen when you freeze the chart at run time. After the freeze time has elapsed, the chart resumes the live updates. If the freeze duration is set to 0, you must manually unfreeze the chart.
- 9 Set **Refresh interval** to the number of seconds between chart updates. You can optionally select **Refresh entire chart** to refresh the entire chart after the number of refresh intervals you specify.

10 Set Retrieval Mode to Full or Cyclic. Full retrieval mode returns all stored data points for a tag within a trend period. Cyclic retrieval mode returns stored data at equal length intervals within a trend period. For more information, see Appendix A, "Understanding Data Retrieval".

Note: Retrieval mode is applicable for tags from Wonderware Historian. The value set here is the default setting for a pen; however, you can override it by setting the pen retrieval mode to any value other than application setting.

11 Set Trend Duration. You can set the trend duration for a maximum period of 2 years. All trend durations are calculated backward from an end date of the current time (except for the options "Yesterday" and "Previous Hour"). You can pick a duration from the list or type a specific duration in the list box. The time notation is [DD] HH:MM:SS.fff, where DD=days, HH=hours, MM=minutes, SS=seconds, and fff=milliseconds. To change a value in the duration list box, use the arrow keys on your keyboard to position the mouse cursor to the left of the number you want to change, and then type the new number. You can only change the duration and not the start and end time.

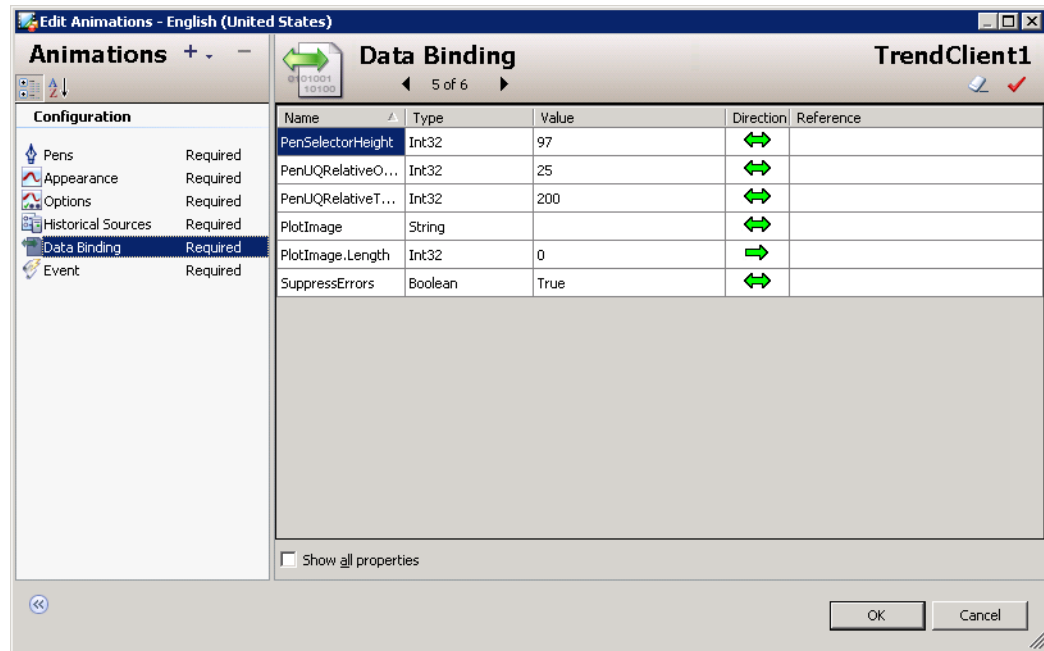
12 Click OK.

Setting Data Bindings

You can associate a reference to an attribute with the Data Binding pane instead of with scripting. Information can be passed in either direction or both directions as desired. Changes to the status of one can affect the other.

To set data bindings

- 1 In the ArchestrA Symbol Editor, double-click the Trend Client. The **Edit Animations** dialog box appears.
- 2 Click **Data Binding**. The data binding information appears in the right pane.



The Data Binding table contains the following read-only information:

Name	The name of the property.
Type	The .NET data type of the property.
Value	The value of the property.

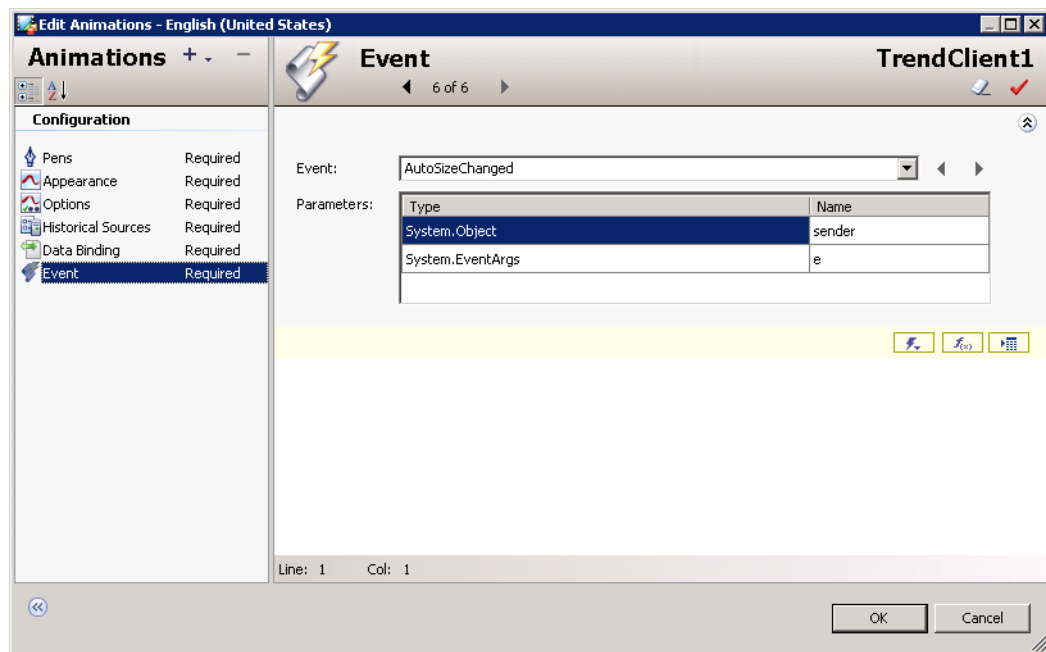
- 3 Click the **Direction** icon to disable the link entirely or change the direction of the data flow. If the arrow points to the right (away from the name of the property), the status of the Trend Client property affects the attribute you're binding to. If the arrow points to the left (toward the name of the event), the status of the Trend Client property is affected by the attribute you're binding to. If the arrow points both ways, a change in the status of one affects the other.
- 4 Double-click **Reference** to enter reference information. Click the ellipsis button to open the Galaxy Browser and use the Element Browser. For more information, see Chapter 3, *Working with Objects*, in the *Wonderware Application Server User's Guide*.
- 5 Select **Show all properties** to show all properties for which there is data binding information.
- 6 Click **OK**.

Handling Trend Events


You can use the custom event editor to build scripts to handle trend events.


To modify a trend event

- 1 In the ArchestrA Symbol Editor, double-click the Trend Client. The **Edit Animations** dialog box appears
- 2 Click **Events**. The events information appears in the right pane. For information on the different types of events, see "Trend Client Events" on page 92.



- 3 Select an event in the **Event** list. The parameters information appears in the lower portion of the pane.
- 4 Type a script for this event in the window. You can use the editor buttons to gather information and parameters.

 Browse Event Parameters inserts the event parameters and any associated event parameter information at the current insertion point in the script.

 Display Script Function Browser shows the available script functions. Select a function and click **OK** to insert the function and parameters at the current insertion point in the script.

 Display Attribute Browser opens the Galaxy Browser.

- 5 Click **OK**. The script information is saved for the event.

Chapter 3

Scripting the Trend Client

This section describes the properties, methods, and events for the Trend Client.

Trend Client Properties

The following are the properties of the Trend Client:

- Chart.AddMultiplePens
- Chart.BackgroundColor
- Chart.Freeze
- Chart.FreezeDurationMS
- Chart.HidePenList
- Chart.Labels
- Chart.PenPrecision
- Chart.RefreshEntireChartIntervals
- Chart.RetrievalMode
- Chart.UpdateRateMS
- HistorySources
- Pen.Color
- Pen.Count
- Pen.Description
- Pen.Expression

- Pen.Format
- Pen.HistorySource
- Pen.HistoryTagName
- Pen.Index
- Pen.Name
- Pen.Precision
- Pen.RetrievalMode
- Pen.Style
- Pen.TrendHi
- Pen.TrendLo
- Pen.TrendType
- Pen.Units
- Pen.Visible
- Pen.Width
- PenSelectorHeight
- PenUQRelativeOpacity
- PenUQRelativeThickness
- PlotArea.BackgroundColor
- PlotArea.BorderColor
- PlotArea.GradientEndColor
- PlotArea.GradientType
- PlotArea.GridColor
- PlotArea.GridHorizontal
- PlotArea.GridStyle
- PlotArea.GridVertical
- PlotArea.GridWidth
- PlotArea.HighlightCurrentPen
- PlotArea.PenHighlightColor
- PlotArea.PenHighlightWidth
- PlotArea.SingleTagMode
- PlotImage

- ShowContextMenu
- SuppressErrors
- TimeAxis.Cursor1.Color
- TimeAxis.Cursor1.Pos
- TimeAxis.Cursor1.Style
- TimeAxis.Cursor1.Width
- TimeAxis.Cursor2.Color
- TimeAxis.Cursor2.Pos
- TimeAxis.Cursor2.Style
- TimeAxis.Cursor2.Width
- TimeAxis.LabelColor
- TimeAxis.NumGridPerValue
- TimeAxis.NumValues
- TimeAxis.ShowCursors
- TimeSelector
- TimeSelector.DurationMS
- TimeSelector.EndDate
- TimeSelector.StartDate
- TimeSelector.TimeDuration
- ToolTipText
- TrendVersion
- ValueAxis.Label
- ValueAxis.NumGridPerValue
- ValueAxis.NumValues
- Visible

This section describes all the scriptable properties available in the Trend Client.

You can access the properties of an individual pen by setting it as the current pen and then applying the pen properties.

Chart.AddMultiplePens

The `Chart.AddMultiplePens` property is a read-write Boolean property that suspends or allows the chart refresh while multiple pens are added to the chart.

Syntax

```
Chart.AddMultiplePens = bool;  
Result = Chart.AddMultiplePens;
```

Remarks

The default is `FALSE` (allows the chart refresh while multiple pens are added to the chart).

You can set this property to `TRUE`, then add multiple properties using a script without refreshing the graph. After adding the final tag, set this property back to `FALSE`. The graph is automatically refreshed and shows all the pens you have added.

Chart.BackgroundColor

The `Chart.BackgroundColor` property is a read-write property that gets or sets the background color of the chart.

Syntax

```
Chart.BackgroundColor = color;  
Result = Chart.BackgroundColor;
```

Remarks

The default is white.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, `FromARGB()`, `FromKnownColor()`, and `FromName()`. For a list of the .NET color names and the corresponding hexadecimal codes, see [.NET Colors](#) on page 94. For more information on the color methods, see the Microsoft documentation for .NET Framework development.

Chart.Freeze

The `Chart.Freeze` property is a read-write Boolean property that stops (`TRUE`) or starts (`FALSE`) the trend display from updating.

Syntax

```
Chart.Freeze = bool;  
Result = Chart.Freeze;
```

Remarks

The default is FALSE, which continues (or resumes) the trend display updates. Setting the property to TRUE stops the trend display updates.

Chart.FreezeDurationMS

The Chart.FreezeDurationMS property is a read-write integer property that freezes the chart from live updates for the specified duration in milliseconds.

Note: When freeze duration is set to a value other than 0, in run time after a freeze has been initiated, live updates appear on the Trend Client automatically after the configured time has elapsed. If Freeze Duration for chart is set to 0, the auto resume feature is disabled.

Syntax

```
Chart.FreezeDurationMS = int;  
Result = Chart.FreezeDurationMS;
```

Remarks

The default is 600000 (10 minutes).

Chart.HidePenList

The Chart.HidePenList property is a read-write Boolean property that shows or hides the pen list in the chart.

Syntax

```
Chart.HidePenList = bool;  
Result = Chart.HidePenList;
```

Remarks

The default is TRUE.

Chart.Labels

The Chart.Labels property is a read-write integer property that sets the visibility of labels (chart label, X and Y axes scales, and cursor information) in the chart. Supported options are All and None (where 0 = All and 1 = None).

Syntax

```
Chart.Labels = int;  
Result = Chart.Labels;
```

Remarks

The default is 0 (All).

Chart.PenPrecision

The `Chart.PenPrecision` property is a read-write integer property that gets or sets the number of decimal places to show by default for the data value.

Syntax

```
Chart.PenPrecision = int;  
Result = Chart.PenPrecision;
```

Remarks

The default is 0.

Chart.RefreshEntireChartIntervals

The `Chart.RefreshEntireChartIntervals` property is a read-write integer property that refreshes the entire chart at every specified number of intervals. (The interval is specified with the `Chart.UpdateRateMS` property.) If the value is '0' the chart only adds new data to the chart; however, it does not refresh the data already present on the chart.

Syntax

```
Chart.RefreshEntireChartInterval = int;  
Result = Chart.RefreshEntireChartIntervals;
```

Remarks

The default is 100 intervals.

Chart.RetrievalMode

The `Chart.RetrievalMode` property is a read-write integer property that gets or sets the data retrieval mode for retrieving the data from IndustrialSQL Server at the chart level. Supported options are cyclic and full (where 0 = cyclic and 2 = full).

Syntax

```
Chart.RetrievalMode = int;  
Result = Chart.RetrievalMode;
```

Remarks

The default is 0 (cyclic).

Chart.UpdateRateMS

The `Chart.UpdateRateMS` property is a read-write integer property that gets or sets the chart refresh interval in milliseconds.

Syntax

```
Chart.UpdateRateMS = int;
Result = Chart.UpdateRateMS;
```

Remarks

The default is 1000.

HistorySources

The HistorySources property is a read-only object property that gets a list of history sources. For more information about the HistorySources properties, see "Trend Client History Sources" on page 68.

Syntax

```
Result = HistorySources;
```

Example

```
Dim histSources = TrendClient1.HistorySources;
Dim histSource = histSources.GetSource("<history source
name>");
```

Remarks

No default.

Pen.Color

The Pen.Color property is a read-write property that gets or sets the color of the currently selected pen.

Syntax

```
Pen.Color = color;
Result = Pen.Color;
```

Remarks

The default is color.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, FromARGB(), FromKnownColor(), and FromName(). For a list of the .NET color names and the corresponding hexadecimal codes, see .NET Colors on page 94. For more information on the color methods, see the Microsoft documentation for .NET Framework development.

Pen.Count

The Pen.Count property is a read-only integer property that gets the number of the pens in the pen list.

Syntax

```
Result = Pen.Count;
```

Remarks

The default is 0.

Pen.Description

The Pen.Description property is a read-write string property that gets or sets the description of the currently selected pen.

Syntax

```
Pen.Description = string;  
Result = Pen.Description;
```

Remarks

If the pen expression was configured by browsing to the attribute in the Attribute Browser, the default value is obtained from the corresponding description property of the associated ArcestrA attribute. Otherwise, the default value is blank if the pen expression is manually configured.

Pen.Expression

The Pen.Expression property is a read-write string property that gets or sets the expression of the currently selected pen. The expression can be a Galaxy attribute, InTouch tag reference, or an expression.

Syntax

```
Pen.Expression = string;  
Result = Pen.Expression;
```

Remarks

The default is blank.

Pen.Format

The Pen.Format property is a read-write integer property that gets or sets the format of the current pen (where 0 is decimal and 1 is scientific).

Syntax

```
Pen.Format = int;  
Result = Pen.Format;
```

Remarks

The default is 0 (decimal).

Pen.HistorySource

The Pen.HistorySource property is a read-only string property that gets the history source of the currently selected pen. For more information about the HistorySources properties, see "Trend Client History Sources" on page 68.

Note: This value is empty when no historical source is configured, if an expression is configured, or if no historical tag is configured.

Syntax

```
Result = Pen.HistorySource;
```

Remarks

The default is blank.

Pen.HistoryTagName

The Pen.HistoryTagName property is a read-write string property that gets or sets the historical tag name for the currently selected pen.

Syntax

```
Pen.HistoryTagName = string;  
Result = Pen.HistoryTagName;
```

Remarks

The default is blank.

Pen.Index

The Pen.Index property is a read-only integer property that gets the index of the currently selected pen, or -1 if no pen is selected. The array is zero-based.

Syntax

```
Result = Pen.Index;
```

Remarks

The default is -1.

Pen.Name

The Pen.Name property is a read-write string property that gets or sets the name of the currently selected pen.

Syntax

```
Pen.Name = string;  
Result = Pen.Name;
```

Remarks

The default is blank.

Pen.Precision

The Pen.Precision property is a read-write integer property that gets or sets the decimal precision of the currently selected pen.

Syntax

```
Pen.Precision = int;  
Result = Pen.Precision;
```

Remarks

The default is the trend chart's pen precision property.

Pen.RetrievalMode

The Pen.RetrievalMode property is a read-write enumerated property that gets or sets the mode for retrieving the data from IndustrialSQL Server of the currently selected pen. Supported options are:

0 = Cyclic

2 = Full

12 = ApplicationSetting

Syntax

```
Pen.RetrievalMode = enum;  
Result = Pen.RetrievalMode;
```

Remarks

The default is the trend chart's retrieval mode property (12).

Pen.Style

The Pen.Style property is a read-write integer property that gets or sets the line style of the currently selected pen.

Valid values are:

0 = Solid

1 = Dash

2 = DashDot

3 = DashDotDot

4 = Dot

Syntax

```
Pen.Style = int;
```

```
Result = Pen.Style;
```

Remarks

The default is 0.

Pen.TrendHi

The Pen.TrendHi property is a read-write double property that gets or sets the high value of the value range to be used for the currently selected pen.

Syntax

```
Pen.TrendHi = double;  
Result = Pen.TrendHi;
```

Remarks

If the pen expression was configured by browsing to the attribute in the Attribute Browser, the default value is set to the corresponding TrendHi property of the associated ArchestrA attribute. Otherwise, the default value is 100 if the pen expression is manually configured.

Pen.TrendLo

The Pen.TrendLo property is a read-write double property that gets or sets the low value of the value range to be used for the currently selected pen.

Syntax

```
Pen.TrendLo = double;  
Result = Pen.TrendLo;
```

Remarks

If the pen expression was configured by browsing to the attribute in the Attribute Browser, the default value is set to the corresponding TrendLo property of the associated ArchestrA attribute. Otherwise, the default value is 0 if the pen expression is manually configured.

Pen.TrendType

The Pen.TrendType property is a read-write integer property that gets or sets the plot type of the currently selected pen.

0 = Point

1 = Line

2 = StepLine

3 = Auto

When `Pen.TrendType` is set to `Auto` and there is no Historian default available, the Trend Client uses a plot type of **Line** for real tags and **StepLine** for integer tags.

Syntax

```
Pen.TrendType = int;  
Result = Pen.TrendType;
```

Remarks

The default is 3.

Pen.Units

The `Pen.Units` property is a read-write string property that gets or sets the units of the currently selected pen.

Syntax

```
Pen.Units = string;  
Result = Pen.Units;
```

Remarks

If the pen expression was configured by browsing to the attribute in the Attribute Browser, the default value is set to the corresponding `EngUnits` property of the associated `ArchestraA` attribute. Otherwise, the default value is blank if the pen expression is manually configured.

Pen.Visible

The `Pen.Visible` property is a read-write Boolean property that gets or sets the visible property for the currently selected pen.

Syntax

```
Pen.Visible = bool;  
Result = Pen.Visible;
```

Remarks

The default is `TRUE` (the pen is visible on the chart). When this property is set to `FALSE`, the pen is hidden from the chart display.

Pen.Width

The `Pen.Width` property is a read-write integer property that gets or sets the line width (from 1 to 10) of the currently selected pen.

Syntax

```
Pen.Width = int;  
Result = Pen.Width;
```

Remarks

The default is 1, when a pen is added to the pen list. If there is no pen in the pen list, the default is 0.

PenSelectorHeight

The PenSelectorHeight property is a read-write integer property that gets or sets the height of the pen selector in pixels.

Syntax

```
PenSelectorHeight = int;  
Result = PenSelectorHeight;
```

Remarks

The default is 97.

PenUQRelativeOpacity

The PenUQRelativeOpacity property is a read-write integer property that gets or sets the line's relative opacity when the data quality is uncertain.

The two properties, relative opacity and relative thickness, create a visual distinction for values with uncertain quality. When the pen type is point, no visual change is shown on the trend.

Syntax

```
PenUQRelativeOpacity = int;  
Result = PenUQRelativeOpacity;
```

Remarks

The default is 25%.

For example, if the pen has an opacity of 80% and the data quality is "uncertain," then the Trend Client calculates the opacity of the pen as $80\% \times 25\% = 20\%$ opaque.

PenUQRelativeThickness

The PenUQRelativeThickness property is a read-write integer property that gets or sets the line's relative thickness when data quality is uncertain.

The two properties, relative opacity, and relative thickness, create a visual distinction for values with uncertain quality. When the pen type is point, no visual change is shown on the trend.

Syntax

```
PenUQRelativeThickness = int;  
Result = PenUQRelativeThickness;
```

Remarks

The default is 200%.

For example, if a line is 1 pixel wide and 200% thick, then the Trend Client calculates the thickness of the line with a data quality of “uncertain“ as 1 pixel x 200% = 2 pixels wide.

PlotArea.BackgroundColor

The `PlotArea.BackgroundColor` property is a read-write property that gets or sets the color of the plot area of the graph.

Syntax

```
PlotArea.BackgroundColor = color;  
Result = PlotArea.BackgroundColor;
```

Remarks

The default is white.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, `FromARGB()`, `FromKnownColor()`, and `FromName()`. For a list of the .NET color names and the corresponding hexadecimal codes, see `.NET Colors` on page 94. For more information on the color methods, see the Microsoft documentation for .NET Framework development.

PlotArea.BorderColor

The `PlotArea.BorderColor` property is a read-write property that gets or sets the color of the plot area's border.

Syntax

```
PlotArea.BorderColor = color;  
Result = PlotArea.BorderColor;
```

Remarks

The default is black.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, `FromARGB()`, `FromKnownColor()`, and `FromName()`. For a list of the .NET color names and the corresponding hexadecimal codes, see `.NET Colors` on page 94. For more information on the color methods, see the Microsoft documentation for .NET Framework development.

PlotArea.GradientEndColor

The `PlotArea.GradientEndColor` property is a read-write property that gets or sets the gradient end color of the plot area of the graph.

Syntax

```
PlotArea.GradientEndColor = color;
Result = PlotArea.GradientEndColor;
```

Remarks

The default is white.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, `FromARGB()`, `FromKnownColor()`, and `FromName()`. For a list of the .NET color names and the corresponding hexadecimal codes, see [.NET Colors](#) on page 94. For more information on the color methods, see the Microsoft documentation for [.NET Framework development](#).

PlotArea.GradientType

The `PlotArea.GradientType` property is a read-write integer property that gets or sets the gradient type of the plot area of the graph.

0 = None (no gradient)

1 = LeftRight

2 = TopBottom

3 = Center

4 = DiagonalLeft

5 = DiagonalRight

6 = HorizontalCenter

7 = VerticalCenter

Syntax

```
PlotArea.GradientEndType = int;
Result = PlotArea.GradientEndType;
```

Remarks

The default is 0.

PlotArea.GridColor

The `PlotArea.GridColor` property is a read-write property that gets or sets the color of grid.

Syntax

```
PlotArea.GridColor = color;
Result = PlotArea.GridColor;
```

Remarks

The default is grey.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, `FromARGB()`, `FromKnownColor()`, and `FromName()`. For a list of the .NET color names and the corresponding hexadecimal codes, see [.NET Colors](#) on page 94. For more information on the color methods, see the Microsoft documentation for .NET Framework development.

PlotArea.GridHorizontal

The `PlotArea.GridHorizontal` property is a read-write Boolean property that shows or hides the horizontal grid.

Syntax

```
PlotArea.GridHorizontal = bool;  
Result = PlotArea.GridHorizontal;
```

Remarks

The default is TRUE.

PlotArea.GridStyle

The `PlotArea.GridStyle` property is a read-write integer property that gets or sets the grid line style.

0 = Solid

1 = Dash

2 = DashDot

3 = DashDotDot

4 = Dot

Syntax

```
PlotArea.GridStyle = int;  
Result = PlotArea.GridStyle;
```

Remarks

The default is 0.

PlotArea.GridVertical

The `PlotArea.GridVertical` property is a read-write Boolean property that shows or hides the vertical grid.

Syntax

```
PlotArea.GridVertical = bool;  
Result = PlotArea.GridVertical;
```


Remarks

The default is TRUE.

PlotArea.GridWidth

The `PlotArea.GridWidth` property is a read-write integer property that gets or sets the width of the grid. Allowed values are 1 to 10.

Syntax

```
PlotArea.GridWidth = int;  
Result = PlotArea.GridWidth;
```

Remarks

The default is 1.

PlotArea.HighlightCurrentPen

The `PlotArea.HighlightCurrentPen` property is a read-write Boolean property that highlights the currently selected pen.

Syntax

```
PlotArea.HighlightCurrentPen = bool;  
Result = PlotArea.HighlightCurrentPen;
```

Remarks

The default is TRUE.

PlotArea.PenHighlightColor

The `PlotArea.PenHighlightColor` property is a read-write property that gets or sets the pen's highlight color.

Syntax

```
PlotArea.PenHighlightColor = color;  
Result = PlotArea.PenHighlightColor;
```

Remarks

The default is yellow.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, `FromARGB()`, `FromKnownColor()`, and `FromName()`. For a list of the .NET color names and the corresponding hexadecimal codes, see [.NET Colors](#) on page 94. For more information on the color methods, see the [Microsoft documentation for .NET Framework development](#).

PlotArea.PenHighlightWidth

The `PlotArea.PenHighlightWidth` property is a read-write integer property that gets or sets the width of the pen's highlight color. Allowed values are 1 to 5.

Syntax

```
PlotArea.PenHighlightWidth = int;  
Result = PlotArea.PenHighlightWidth;
```

Remarks

The default is 2.

PlotArea.SingleTagMode

The `PlotArea.SingleTagMode` property is a read-write Boolean property that sets whether to show only the currently selected pen or all pens.

Syntax

```
PlotArea.SingleTagMode = bool;  
Result = PlotArea.SingleTagMode;
```

Remarks

The default is `FALSE`.

PlotImage

The `PlotImage` property is a read-write string property that gets or sets the plot background image for the chart.

Syntax

```
PlotImage = string;  
Result = PlotImage;
```

Remarks

The value of this property is the folder path and filename for the image. Supported image types are .jpeg, .gif, .bmp, and .png.

The default is blank.

ShowContextMenu

The `ShowContextMenu` property is a Boolean property that shows the context menu in run time.

Syntax

```
ShowContextMenu = bool;  
Result = ShowContextMenu;
```

Remarks

The default is TRUE.

SuppressErrors

The SuppressErrors property is a read-write Boolean property that suppresses or allows error messages.

Note: All errors are logged to ArcestraA Logger regardless of this setting.

Syntax

```
SuppressErrors = bool;  
Result = SuppressErrors;
```

Remarks

The default is TRUE.

TimeAxis.Cursor1.Color

The TimeAxis.Cursor1.Color property is a read-write property that gets or sets the color of the left time axis cursor.

Syntax

```
TimeAxis.Cursor1.Color = color;  
Result = TimeAxis.Cursor1.Color;
```

Remarks

The default is red.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, FromARGB(), FromKnownColor(), and FromName(). For a list of the .NET color names and the corresponding hexadecimal codes, see .NET Colors on page 94. For more information on the color methods, see the Microsoft documentation for .NET Framework development.

TimeAxis.Cursor1.Pos

The TimeAxis.Cursor1.Pos property is a read-write property that gets or sets the time position of the left time axis cursor.

Syntax

```
TimeAxis.Cursor1.Pos = datetime;  
Result = TimeAxis.Cursor1.Pos;
```

Remarks

No default.

TimeAxis.Cursor1.Style

The `TimeAxis.Cursor1.Style` property is a read-write integer property that gets or sets the style of the left time axis cursor. Valid values are:

- 0 = Solid
- 1 = Dash
- 2 = DashDot
- 3 = DashDotDot
- 4 = Dot

Syntax

```
TimeAxis.Cursor1.Style = int;  
Result = TimeAxis.Cursor1.Style;
```

Remarks

The default is 0.

TimeAxis.Cursor1.Width

The `TimeAxis.Cursor1.Width` property is a read-write integer property that gets or sets the width of the left time axis cursor. Allowed values are 1 to 10.

Syntax

```
TimeAxis.Cursor1.Width = int;  
Result = TimeAxis.Cursor1.Width;
```

Remarks

The default is 1.

TimeAxis.Cursor2.Color

The `TimeAxis.Cursor2.Color` property is a read-write property that gets or sets the color of the right time axis cursor.

Syntax

```
TimeAxis.Cursor2.Color = color;  
Result = TimeAxis.Cursor2.Color;
```

Remarks

The default is blue.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, `FromARGB()`, `FromKnownColor()`, and `FromName()`. For a list of the .NET color names and the corresponding hexadecimal codes, see .NET Colors on page 94. For more information on the color methods, see the Microsoft documentation for .NET Framework development.

TimeAxis.Cursor2.Pos

The `TimeAxis.Cursor2.Pos` property is a read-write property that gets or sets the time position of the right time axis cursor.

Syntax

```
TimeAxis.Cursor2.Pos = datetime;  
Result = TimeAxis.Cursor2.Pos;
```

Remarks

No default.

TimeAxis.Cursor2.Style

The `TimeAxis.Cursor2.Style` property is a read-write integer property that gets or sets the style of the right time axis cursor. Valid values are:

0 = Solid
1 = Dash
2 = DashDot
3 = DashDotDot
4 = Dot

Syntax

```
TimeAxis.Cursor2.Style = int;  
Result = TimeAxis.Cursor2.Style;
```

Remarks

The default is 0.

TimeAxis.Cursor2.Width

The `TimeAxis.Cursor2.Width` property is a read-write integer property that gets or sets the width of the right time axis cursor. Allowed values are 1 to 10.

Syntax

```
TimeAxis.Cursor2.Width = int;  
Result = TimeAxis.Cursor2.Width;
```

Remarks

The default is 1.

TimeAxis.LabelColor

The `TimeAxis.LabelColor` property is a read-write property that gets or sets the color of the time axis labels.

Syntax

```
TimeAxis.LabelColor = color;  
Result = TimeAxis.LabelColor;
```

Remarks

The default is black.

Color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, `FromARGB()`, `FromKnownColor()`, and `FromName()`. For a list of the .NET color names and the corresponding hexadecimal codes, see [.NET Colors](#) on page 94. For more information on the color methods, see the Microsoft documentation for .NET Framework development.

TimeAxis.NumGridPerValue

The `TimeAxis.NumGridPerValue` property is a read-write integer property that gets or sets the number of grid lines that appear between each value shown on time axis. The valid range is 1 to 20.

Syntax

```
TimeAxis.NumGridPerValue = int;  
Result = TimeAxis.NumGridPerValue;
```

Remarks

The default is 3.

TimeAxis.NumValues

The `TimeAxis.NumValues` property is a read-write integer property that gets or sets the number of time labels that are shown along the time axis. The valid range is 2 to 15.

Syntax

```
TimeAxis.NumValues = int;  
Result = TimeAxis.NumValues;
```

Remarks

The default is 6.

TimeAxis.ShowCursors

The `TimeAxis.ShowCursors` property is a read-write Boolean property that shows or hides the time axis cursors.

Syntax

```
TimeAxis.ShowCursors = bool;  
Result = TimeAxis.ShowCursors;
```

Remarks

The default is `FALSE`.

TimeSelector

The `TimeSelector` property is a read-only property that gets the `Time Range Picker` object used in the Trend Client. You can use it in scripting to shorten the code using its properties and methods.

For the individual properties and methods, see the following properties, or the methods starting at "`TimeSelector.GetStartAndEndTimes`" on page 86.

Example 1

```
dim TRP as object;  
  
TRP = TrendClient1.TimeSelector;  
  
StartDate = TRP.StartDate;  
  
EndDate = TRP.EndDate;  
  
duration = TRP.TimeDuration;
```

Example 2

```
dim TRP as object;  
  
TRP = TrendClient1.TimeSelector;  
  
TRP.SetStartAndEndTimes(StartDate, EndDate, Duration);
```

Remarks

The return value is a `TimeRangePicker`.

TimeSelector.DurationMS

The `TimeSelector.DurationMS` property is a read-write integer property that gets the time duration measured in milliseconds.

The start time of the Trend Client (`TimeSelector.StartDate`) is calculated as the end time (`TimeSelector.EndDate`) minus the new time duration (`TimeSelector.DurationMS`).

When you set the value of the `TimeSelector.DurationMS` property, the `TimeSelector.TimeDuration` property is set to 0.

Syntax

```
result = TimeSelector.DurationMS;  
TimeSelector.DurationMS = Value;
```

Remarks

The default value is 300000.

TimeSelector.EndDate

The `TimeSelector.EndDate` property is a read-only string property that gets the end date and time of the Trend Client.

The default value is the time the Trend Client is placed on the canvas. If the **Update to Current Time** option is enabled, the `TimeSelector.EndDate` property is updated with the current time.

Note: To set the end date and time of the Trend Client, use the `TimeSelector.SetStartAndEndTimes` method.

Syntax

```
result = Trend01.TimeSelector.EndDate;
```

TimeSelector.StartDate

The `TimeSelector.StartDate` property is a read-only string property that gets the start date and time of the Trend Client.

The default value is the time the Trend Client is placed on the canvas minus the duration.

Note: To set the start date and time of the Trend Client, use the `TimeSelector.SetStartAndEndTimes` method.

Syntax

```
result = TimeSelector.StartDate;
```

TimeSelector.TimeDuration

The `TimeSelector.TimeDuration` property is a read-write enumerated property that gets or sets the enumerated duration of the time axis of the chart.

Syntax

```
TimeSelector.TimeDuration = enum;  
Result = TimeSelector.TimeDuration;
```


The `TimeSelector.TimeDuration` property is a read-write integer property that gets or sets the time duration. The start time of the Trend Client (`TimeSelector.StartDate`) is calculated as the end time (`TimeSelector.EndDate`) minus the new time duration.

The `TimeSelector.TimeDuration` can have one of the following values:

Value	Description
0	Custom
1	The last minute.
2	The last five minutes.
3	The last ten minutes.
4	The last 15 minutes.
5	The last 30 minutes.
6	The last hour.
7	The last two hours.
8	The last four hours.
9	The last eight hours.
10	The last 12 hours.
11	The last 24 hours.
12	The last two days.
13	The last week.
14	The last two weeks.
15	The last month.
16	The last three months.
17	One minute.
18	Five minutes.
19	Ten minutes.
20	15 minutes.
21	30 minutes.
22	One hour.
23	Two hours.
24	Four hours.
25	Eight hours.

Value	Description
26	12 hours.
27	24 hours.
28	Two days.
29	One week.
30	Two weeks.
31	One month.
32	Three months.
33	Yesterday: 0:00:00 of the previous day to 0:00:00 of the current day.
34	Current day: 0:00:00 of the current day to the current time.
35	Previous hour: The start of the previous hour to the start of the current hour.
36	Current hour: The start of the current hour to the current time.

Syntax

```
result = TimeSelector.TimeDuration;  
TimeSelector.TimeDuration = Value;
```

Example

```
Trend01.TimeSelector.TimeDuration = 5;  
// The trend is now set to show the last 30 minutes.
```

Remarks

The default is 18 (5 minutes).

ToolTipText

The ToolTipText property is a read-write string property that gets or sets the pop-up text that appears when the mouse is hovered over the chart at run time.

Syntax

```
ToolTipText = string;  
Result = ToolTipText;
```

Remarks

The default is blank.

TrendVersion

The TrendVersion property is a read-only string property that gets the version of the trend.

Syntax

```
Result = TrendVersion;
```

Remarks

The return value is 1.0.0 for the first release.

ValueAxis.Label

The ValueAxis.Label property is a read-write integer property that gets or sets which labels are shown on the value axis.

0 = MultipleScales

1 = SingleScale

2 = ValuesAtCursor

Syntax

```
ValueAxis.Label = int;  
Result = ValueAxis.Label;
```

Remarks

The default is 0.

ValueAxis.NumGridPerValue

The ValueAxis.NumGridPerValue property is a read-write integer property that gets or sets the number of grid lines that appear between each value shown along the Y-axis. The valid range is 1 to 20.

Syntax

```
ValueAxis.NumGridPerValue = int;
```

```
Result = ValueAxis.NumGridPerValue;
```

Remarks

The default is 2.

ValueAxis.NumValues

The ValueAxis.NumValues property is a read-write integer property that gets or sets the number of value labels that are shown along the Y-axis. The valid range is 2 to 15.

Syntax

```
ValueAxis.NumValues = int;  
Result = ValueAxis.NumValues;
```

Remarks

The default is 6.

Visible

The Visible property is a read-write Boolean property that shows or hides the Trend Client at run time.

Syntax

```
Visible = bool;  
Result = Visible;
```

Remarks

The default is TRUE.

Trend Client History Sources

The Trend Client has the following scriptable properties for the history sources:

- HistorySource.Authentication
- HistorySource.Domain
- HistorySource.Password
- HistorySource.RetainPassword
- HistorySource.ServerName
- HistorySource.Type
- HistorySource.UNCPATH
- HistorySource.UserID
- HistorySources.Count

- HistorySources.Items

HistorySource.Authentication

The HistorySource.Authentication property is a read-write string property that gets or sets the authentication mode for the connection to the server. Available options are SQL Server, Windows Account, and Windows Integrated.

Syntax

```
HistorySource.Authentication = string;  
Result = HistorySource.Authentication;
```

Remarks

The default is Windows Integrated.

HistorySource.Domain

The HistorySource.Domain property is a read-write string property that gets or sets the domain name for the connection to the Wonderware Historian.

Syntax

```
HistorySource.Domain = string;  
Result = HistorySource.Domain;
```

Remarks

The default is an empty message value ("").

This property is not applicable if you are using an InTouch LGH file as the data source.

HistorySource.Password

The HistorySource.Password property is a read-write string property that gets and sets the password for the connection to the server.

Syntax

```
HistorySource.Password = string;  
Result = HistorySource.Password;
```

Remarks

The default is wwUser.

HistorySource.RetainPassword

The HistorySource.RetainPassword property is a read-write Boolean property that indicates whether the password is stored in persistent storage.

Syntax

```
HistorySource.RetainPassword = bool;  
Result = HistorySource.RetainPassword;
```

Remarks

The default is TRUE (the password is stored in persistent storage).

This property is not applicable if you are using an InTouch LGH file as the data source.

HistorySource.ServerName

The HistorySource.ServerName property is a read-write string property that gets or sets the name of the history source.

Syntax

```
HistorySource.ServerName = string;  
Result = HistorySource.ServerName;
```

Remarks

The default is an empty message value ("").

HistorySource.Type

The HistorySource.Type property is a read-write string property that gets or sets the type of the history source. Possible values are “InSQL” (Wonderware Historian Server) and InTouch (InTouch History Files).

Syntax

```
HistorySource.Type = string;  
Result = HistorySource.Type;
```

Remarks

The default is “InSQL.”

HistorySource.UNCPPath

The HistorySource.UNCPPath property is a read-write string property that gets or sets the UNC path of the InTouch Log History/LGH file.

Syntax

```
HistorySource.UNCPPath = string;  
Result = HistorySource.UNCPPath;
```

Remarks

The default is an empty message value ("").

HistorySource.UserID

The `HistorySource.UserID` property is a read-write string property that gets and sets the user ID for the Wonderware Historian.

Syntax

```
HistorySource.UserID = string;
Result = HistorySource.UserID;
```

Remarks

The default is `wwUser`.

This property is not applicable if you are using an InTouch LGH file as the data source.

HistorySources.Count

The `HistorySources.Count` property is a read-only integer property that shows the number of history sources configured.

Syntax

```
Result = HistorySources.Count;
```

Remarks

The default is 0.

HistorySources.Items

The `HistorySources.Items` property is a read-only array property that shows the array of history source names.

Syntax

```
Result = HistorySources.Items;
```

Remarks

No default.

Example 1

Populate a string array `uda` of a userdefined instance with the server names configured in the Trend Control, and then add them to a list box control on the canvas)

```
dim i as integer;
dim b as object;
ListBox1.Clear();
for i = 1 to TrendClient1.HistorySources.Count;
b = TrendClient1.HistorySources.Items[i];
```

```
UserDefined_001.items[i] = b.ServerName;  
ListBox1.AddItem(UserDefined_001.items[i]);  
next;
```

Trend Client Methods

The following are the methods used by the Trend Client:

- AddHistorianSource
- AddPen
- ClearPens
- DeleteCurrentPen
- GetHistorianSource
- GetPenValAtX1
- GetStartAndEndTimes
- MoveNextPen
- MovePrevPen
- RefreshData
- RefreshTimes
- RemoveHistorianSource
- RemovePen
- ScaleAllPens
- ScaleAutoAllPens
- ScaleAutoPen
- ScaleDownAllPens
- ScaleDownPen
- ScaleMoveAllPensDown
- ScaleMoveAllPensUp
- ScaleMovePenDown
- ScaleMovePenUp
- ScalePen
- ScaleUpAllPens
- ScaleUpPen

- SetCurrentPen
- SetDuration
- SetStartAndEndTimes
- TimeSelector.GetStartAndEndTimes
- TimeSelector.RefreshTimes
- TimeSelector.SetStartAndEndTimes
- UpdateHistorianSource

The following section describes the scriptable methods available in the Trend Client.

AddHistorianSource

AddHistorianSource method adds a new history source.

Example

```
Dim b as object;
b = Trend1.AddHistorianSource("idc_insql12", "InSQL");
```

Syntax

```
[Result =] Trend1.AddHistorianSource(string HistorySourceName,
string Type);
```

Parameters

HistorySourceName

The name of the history source.

Type

The type of history source "InSQL" or "InTouch".

Return Value

If there is already a server with the given name in the list, the object for that server is returned. Otherwise, a new server with the given name is added to the list and the object for the new server is returned.

AddPen

The AddPen method adds a named pen to the trend.

Examples

```
dim b as boolean;
b = Trend1.AddPen("MyPen01", "UserDefined_001.Value",
    "InSQL01", "UserDefined_001.Value", 1);
```

```
b = Trend1.AddPen("MyPen02", "InTouch:$Second",  
  "InSQL01", "$Second", 1);  
b = Trend1.AddPen("MyPen03",  
  "MyContainer.InletPump1+MyContainer.InletPump2", "", "", 1);
```

Syntax

```
[Result=] AddPen(string PenName, string TagName, string  
  HistorySource, string HistoryTagName, int HistoryTagType);
```

Parameters

PenName

The name of the pen to add.

TagName

The name of the tag associated with the pen being added.

HistorySource

The history source for the pen being added.

HistoryTagName

The name of the history tag for the pen being added.

HistoryTagType

The type of history tag for the pen being added: 1 for Wonderware Historian or 3 for LGH.

Return Value

Returns TRUE if the pen was successfully added; otherwise, returns FALSE.

ClearPens

The ClearPens method removes all pens from the trend.

Example

```
dim b as boolean;  
b = Trend1.ClearPens();
```

Syntax

```
[Result=] Trend1.ClearPens();
```

Parameters

None.

Return Value

Returns TRUE if the pens were successfully deleted from the trend; otherwise, returns FALSE.

DeleteCurrentPen

The DeleteCurrentPen method removes the currently selected pen from the trend.

Example

```
dim b as boolean;  
b = Trend1.DeleteCurrentPen();
```

Syntax

```
[Result=] Trend1.DeleteCurrentPen();
```

Parameters

None.

Return Value

Returns TRUE if the pen was successfully removed; otherwise, returns FALSE.

GetHistorianSource

GetHistorianSource method gets the named history source object for a server from the server list.

Example

```
Dim b as object;  
b = Trend1.GetHistorianSource("idc_insql15");
```

Syntax

```
[Result =] Trend1.GetHistorianSource(string HistorySourceName);
```

Parameters

HistorySourceName

The name of the history source.

Return Value

If the server exists, the object is returned; otherwise, a NULL is returned.

GetPenValAtX1

The GetPenValAtX1 method gets the value of the expression associated with the specified pen at the point at which its curve intersects the axis cursor the first time.

Example

```
double val = Trend1.GetPenValAtX1("MyPen01");
```

Syntax

```
[Result=] Trend1.GetPenValAtX1 (string PenName);
```

Parameters

PenName

The pen you want to check the value of.

Return Value

Returns a double if the point is analog or discrete; returns the fixed position of the point if the point is message or other type. If the specified pen is shown in the chart multiple times, then the method uses the first instance that was added.

GetPenValAtX2

The GetPenValAtX2 method gets the value of the expression associated with the specified pen at the point at which its curve intersects the axis cursor the second time.

Example

```
double val = Trend2.GetPenValAtX2 ("MyPen01");
```

Syntax

```
[Result=] Trend2.GetPenValAtX2 (string PenName);
```

Parameters

PenName

The pen you want to check the value of.

Return Value

Returns a double if the point is analog or discrete; returns the fixed position of the point if the point is message or other type. If the specified pen is shown in the chart multiple times, then the method uses the first instance that was added.

GetStartAndEndTimes

The GetStartAndEndTimes method gets the start and end times for the query.

Example

```
Dim SDate as System.DateTime;  
Dim EDate as System.DateTime;  
Dim b as System.Int32  
b = Trend1.GetStartAndEndTimes (SDate, EDate);
```

```

StartTime = SDate.ToString();
EndTime = EDate.ToString();
Ret = b;

```

Syntax

```

[Result =] Trend1.GetStartAndEndTimes(DateTime StartTime,
    DateTime EndTime);

```

Parameters

StartTime

The start time for the query.

EndTime

The end time for the query.

Return Value

Returns the time range enumeration.

MoveNextPen

The MoveNextPen method sets the current pen focus to the next pen in the list of trended attributes or tags. If the current pen is the last pen, calling the MoveNextPen method sets the first pen as the current pen.

Example

```

dim b as boolean;
b = Trend1.MoveNextPen();

```

Syntax

```

[Result=] Trend1.MoveNextPen();

```

Parameters

None.

Return Value

Returns TRUE if the current pen focus was successfully set to the next pen (or wrapped around to the first pen in the list); otherwise, returns FALSE.

MovePrevPen

The MovePrevPen method sets the current pen focus to the previous pen in the list of trended attributes or tags. If the current pen is the first pen, calling the MovePrevPen method has no effect.

Example

```

dim b as boolean;
b = Trend1.MovePrevPen();

```

Syntax

```
[Result=] Trend1.MovePrevPen();
```

Parameters

None.

Return Value

Returns TRUE if the current pen focus was successfully set to the previous pen; otherwise, returns FALSE.

RefreshData

The RefreshData method refreshes the trend chart by retrieving new data for all pens.

Example

```
bool b = Trend1.RefreshData();
```

Syntax

```
[Result=] Trend1.RefreshData();
```

Parameters

None.

Return Value

Returns TRUE if the trend was successfully updated; otherwise, FALSE is returned.

RefreshTimes

The RefreshTimes method sets the time period for the query by updating the end time to current time and recalculates the start time based on the new end time and duration.

Example

```
dtag = 1;  
Trend1.RefreshTimes(dtag);
```

Syntax

```
Trend1.RefreshTimes(bool TriggerEvent);
```

Parameters

TriggerEvent

If you set the Boolean parameter to TRUE, the OnChange event is triggered if the time is updated.

RemoveHistorianSource

RemoveHistorianSource method removes the specified history source from the list.

Example

```
Dim b as Boolean;
b = Trend1.RemoveHistorianSource("IDC_INSQL15");
```

Syntax

```
[Result] = Trend1.RemoveHistorianSource(string
    HistorySourceName);
```

Parameters

HistorySourceName

The name of the history source.

Return Value

This method returns true if the instance was removed from the list. This method returns false if the exact instance is not in the list, and the list remains unchanged.

RemovePen

The RemovePen method removes the specified pen from the trend chart.

Note: If the same pen is available multiple times, then only the first occurrence is removed.

Example

```
dim b as boolean;
b = Trend1.RemovePen("MyPen01");
```

Syntax

```
[Result=] Trend1.RemovePen(string PenName);
```

Parameters

PenName

The pen name to be removed.

Return Value

Returns TRUE if the pen was successfully removed from the trend chart; otherwise, returns FALSE.

ScaleAllPens

The ScaleAllPens method sets the Y-axis (value axis) scale for all pens.

Example

```
dim b as boolean;  
b = Trend1.ScaleAllPens(-100, 100);
```

Syntax

```
[Result=] Trend1.ScaleAllPens(double Min, double Max);
```

Parameters*Min*

Type the minimum value for the Y-axis.

Max

Type the maximum value for the Y-axis.

Return Value

Returns TRUE if the Y-axis scale is set successfully; otherwise, returns FALSE.

ScaleAutoAllPens

The ScaleAutoAllPens method sets a suitable Y-axis scale for all tags in accordance with the current minimum and maximum values.

Example

```
dim b as boolean;  
b = Trend1.ScaleAutoAllPens();
```

Syntax

```
[Result=] Trend1.ScaleAutoAllPens();
```

Parameters

None.

Return Value

Returns TRUE if the Y-axis is successfully scaled; otherwise, returns FALSE.

ScaleAutoPen

The ScaleAutoPen method sets a suitable Y-axis scale for the currently selected pen in accordance with the current minimum and maximum values.

Example

```
dim b as boolean;  
b = Trend1.ScaleAutoPen();
```


Syntax

```
[Result=] Trend1.ScaleAutoPen();
```

Parameters

None.

Return Value

Returns TRUE if the Y-axis is successfully scaled; otherwise, returns FALSE.

ScaleDownAllPens

The ScaleDownAllPens method increases the value range of all pens by one third.

Example

```
dim b as boolean;  
b = Trend1.ScaleDownAllPens();
```

Syntax

```
[Result=] Trend1.ScaleDownAllPens();
```

Parameters

None.

Return Value

Returns TRUE if the value range is successfully increased by one third; otherwise, returns FALSE.

ScaleDownPen

The ScaleDownPen method increases the value range of the currently selected pen by one third.

Example

```
dim b as boolean;  
b = Trend1.ScaleDownPen();
```

Syntax

```
[Result=] Trend1.ScaleDownPen();
```

Parameters

None.

Return Value

Returns TRUE if the scaling is successful; otherwise, returns FALSE.

ScaleMoveAllPensDown

The ScaleMoveAllPensDown method moves the value scale down for all pens.

Example

```
dim b as boolean;  
b = Trend1.ScaleMoveAllPensDown();
```

Syntax

```
[Result=] Trend1.ScaleMoveAllPensDown();
```

Parameters

None.

Return Value

Returns TRUE if the scaling is successful; otherwise, returns FALSE.

ScaleMoveAllPensUp

The ScaleMovesAllPensUp method moves the value scale up for all pens.

Example

```
dim b as boolean;  
b = Trend1.ScaleMoveAllPensUp();
```

Syntax

```
[Result=] Trend1.ScaleMoveAllPensUp();
```

Parameters

None.

Return Value

Returns TRUE if the scaling is successful; otherwise, returns FALSE.

ScaleMovePenDown

The ScaleMovePenDown method moves the value scale down for the currently selected pen.

Example

```
dim b as boolean;  
b = Trend1.ScaleMovePenDown();
```

Syntax

```
[Result=] Trend1.ScaleMovePenDown();
```

Parameters

None.

Return Value

Returns TRUE if the scaling is successful; otherwise, returns FALSE.

ScaleMovePenUp

The ScaleMovePenUp method moves the value scale up for the currently selected pen.

Example

```
dim b as boolean;  
b = Trend1.ScaleMovePenUp();
```

Syntax

```
[Result=] Trend1.ScaleMovePenUp();
```

Parameters

None.

Return Value

Returns TRUE if the scaling is successful; otherwise, returns FALSE.

ScalePen

The ScalePen method sets the Y-axis scale for the currently selected pen.

Example

```
dim b as boolean;  
b = Trend1.ScalePen(-100, 100);
```

Syntax

```
[Result=] Trend1.ScalePen(double Min, double Max);
```

Parameters

Min

Type the minimum value for the Y-axis.

Max

Type the maximum value for the Y-axis.

Return Value

Returns TRUE if the Y-axis scale is set successfully; otherwise, returns FALSE.

ScaleUpAllPens

The `ScaleUpAllPens` method decreases the value range of all pens by one fourth.

Example

```
dim b as boolean;  
b = Trend1.ScaleUpAllPens();
```

Syntax

```
[Result=] Trend1.ScaleUpAllPens();
```

Parameters

None.

Return Value

Returns `TRUE` if the value range is successfully reduced by one fourth; otherwise, returns `FALSE`.

ScaleUpPen

The `ScaleUpPen` method decreases the value range of the currently selected pen by one fourth.

Example

```
dim b as boolean;  
b = Trend1.ScaleUpPen();
```

Syntax

```
[Result=] Trend1.ScaleUpPen();
```

Parameters

None.

Return Value

Returns `TRUE` if the value range is successfully reduced by one fourth; otherwise, returns `FALSE`.

SetCurrentPen

The `SetCurrentPen` method sets the specified pen as the currently selected pen. If the pen is available multiple times, then the first occurrence is selected.

Example

```
dim b as boolean;  
b = Trend1.SetCurrentPen("MyPen01");
```

Syntax

```
[Result=] Trend1.SetCurrentPen(string PenName);
```

Parameters*PenName*

The name of the pen being set.

Return Value

Returns TRUE if the pen was set successfully; otherwise, returns FALSE.

SetDuration

The SetDuration method sets the duration of the trend. Calling this method first sets the end time to the current time and then the start time to current time minus the specified duration. Default duration is 5 minutes. Maximum duration is 24 hours.

Example

```
dim b as boolean;
b = Trend1.SetDuration("00:05:00");
```

Syntax

```
[Result=] Trend1.SetDuration(string DateTimeDuration);
```

Parameters*DateTimeDuration*

The time of the duration in HH:MM:SS format.

Return Value

Returns TRUE if the duration is set successfully; otherwise, returns FALSE.

SetStartAndEndTimes

The SetStartAndEndTimes method sets the start and end times for the query.

Example

```
Dim b as Boolean;
b = Trend1.SetStartAndEndTimes("08/31/2008
15:33:43","09/01/2009 15:33:43",0);
```

Syntax

```
[Result =] Trend1.SetStartAndEndTimes(DateTime StartTime,
DateTime EndTime, integer Duration);
```

Parameters

StartTime

The start time for the query. Only considered if the duration is set to Custom. For other durations, the start time is calculated automatically based on the end time and duration.

EndTime

The end time for the query. Only considered if the duration is set to Custom or an option from 17 to 32 (OneMinute to ThreeMonths). Otherwise, the end time is set based on the duration.

Duration

The time1 range duration. If the duration is set to Custom, the specified start and end times are used. For other duration options, the time indicated by the duration is used, and the start and/or end times are updated as necessary. For more information on valid values for the duration, see "TimeSelector.TimeDuration" on page 64.

TimeSelector.GetStartAndEndTimes

The TimeSelector.GetStartAndEndTimes method gets the start and end times for the query.

Syntax

```
Trend01.GetStartAndEndTimes(DateTime StartTime, DateTime EndTime);
```

Parameters

StartTime

String attribute, custom property, or element property to retrieve the start time.

EndTime

String attribute, custom property, or element property to retrieve the end time.

Example

```
dim SDate as string;
dim EDate as string;
Trend01.TimeSelector.GetStartAndEndTimes(SDate, EDate);
StartDate = SDate;
EndDate = EDate;
```

TimeSelector.RefreshTimes

The `TimeSelector.RefreshTimes` method sets the time period for the query by updating the end time to current time and recalculates the start time based on the new end time and duration.

If you set the Boolean parameter to `TRUE`, the `OnChange` event is triggered if the time is updated.

Only use this method, if the **Update to Current Time** option is cleared or the **UpdateToCurrentTime** property is `FALSE`.

Note: This method does not work if the **UpdatetoCurrentTime** property value is `TRUE`.

Syntax

```
Trend01.TimeSelector.RefreshTimes (TriggerEvent);
```

Example

```
dtag = 1;
Trend01.TimeSelector.RefreshTimes (dtag);
```

TimeSelector.SetStartAndEndTimes

The `TimeSelector.SetStartAndEndTimes` method sets the start and end times for the query.

You must specify one of the following parameter combinations:

- Start time and end time. Set the `Duration` parameter to `Custom (0)`.
- Start time and duration. Set the `EndTime` parameter to "".
- End time and duration. Set the `StartTime` parameter to "".
- Start time, duration, and end time. The Trend Client shows an error message if start time plus duration is not equal to end time.

Syntax

```
Trend01.SetStartAndEndTimes (DateTime StartTime, DateTime
    EndTime, integer Duration);
```

Parameters

StartTime

The start time for the query. Only considered if the duration is set to `Custom (0)` by the `Duration` parameter. For other durations, the start time is calculated automatically based on the end time and duration.

EndTime

The end time for the query. Only considered if the duration is set to Custom or an option from 17 to 32 (OneMinute to ThreeMonths) by the Duration parameter. Otherwise, the end time is set based on the duration.

Duration

Duration enum. For more information on possible duration values, see "TimeSelector.TimeDuration" on page 64.

Example

```
Trend01.TimeSelector.SetStartAndEndTimes("08/31/2008  
15:33:43", "09/01/2009 15:33:43", 0);
```

UpdateHistorianSource

UpdateHistorianSource method updates the specified history source in the list.

Example

```
Dim a as object;  
Dim b as Boolean;  
  
a = Trend1.GetHistorianSource ("IDC_INSQL15");  
a.Authentication = "Windows Integrated";  
b = Trend1.UpdateHistorianSource("IDC_INSQL15");
```

Syntax

```
[Result =] Trend1.UpdateHistorianSource(string  
HistorySourceName);
```

Parameters*HistorySourceName*

The name of the history source.

Return Value

Returns true if the given instance is currently in the server list; otherwise, it returns false.

Trend Client History Sources

The Trend Client has the following scriptable methods for the history sources:

- HistorySource.Connect
- HistorySource.Disconnect
- HistorySources.Add

- HistorySources.GetSource
- HistorySources.Remove
- HistorySources.Update

HistorySource.Connect

HistorySource.Connect method logs on to the current history source.

Example

```
Dim histSources = Trend1.HistorySources;  
Dim histSource = histSources.GetSource("idc_insql12");  
Dim statusMsg as string;  
Dim ret = histSource.Connect(statusMsg);
```

Syntax

```
[Result =] HistorySource.Connect(string statusMessage);
```

Parameters

statusMessage

Information of the result of the log on attempt.

Return Value

Returns TRUE if the log on was successful; otherwise, returns FALSE. The server must be configured before calling the Connect method. Changes made to the server configuration after a connect do not take effect until after a disconnect and subsequent connect.

HistorySource.Disconnect

HistorySource.Disconnect method logs off the connection to the current history source.

Example

```
Dim histSources = Trend1.HistorySources;  
Dim histSource = histSources.GetSource("idc_insql12");  
histSource.Disconnect();
```

Syntax

```
[Result =] HistorySource.Disconnect();
```

Parameters

None.

Return Value

None.

Remarks

Repeated calls to this method are harmless and do not result in further state change events.

HistorySources.Add

HistorySources.Add method adds a new history source.

Example

```
Dim histSources = Trend1.HistorySources;  
Dim ret = histSources.Add("idc_insql12", "InSQL");
```

Syntax

```
[Result =] HistorySources.Add(string HistorySourceName, string  
    Type);
```

Parameters

HistorySourceName

The name of the history source.

Type

The type of history source “InSQL“ or “InTouch“.

Return Value

If there is already a server with the given name in the list, the object for that server is returned. Otherwise, a new server with the given name is added to the list and the object for the new server is returned.

HistorySources.GetSource

HistorySources.GetSource method gets the named history source object for a server from the server list.

Example

```
Dim histSources = Trend1.HistorySources;  
Dim histSource = histSources.GetSource("IDC_INSQL15");
```

Syntax

```
[Result =] HistorySources.GetSource(string HistorySourceName);
```

Parameters

HistorySourceName

The name of the history source.

Return Value

If the server exists, the object is returned; otherwise, a NULL is returned.

HistorySources.Remove

HistorySources.Remove method removes the specified history source from the list.

Example

```
Dim histSources = Trend1.HistorySources;  
Dim ret = histSources.Remove("IDC_INSQL15");
```

Syntax

```
[Result =] HistorySources.Remove(string HistorySourceName);
```

Parameters

HistorySourceName

The name of the history source.

Return Value

This method returns TRUE if the instance was removed from the list. This method returns FALSE if the exact instance is not in the list, and the list remains unchanged.

HistorySources.Update

HistorySources.Update method updates the specified history source in the list.

Example

```
Dim histSources = Trend1.HistorySources;  
Dim ret = histSources.Update("IDC_INSQL15");
```

Syntax

```
[Result =] HistorySources.Update(string HistorySourceName);
```

Parameters

HistorySourceName

The name of the history source.

Return Value

Returns TRUE if the given instance is currently in the server list; otherwise, it returns FALSE.

Trend Client Events

The following events are relevant to the Trend Client.

- CurrentPenChanged
- DatesChanged
- MouseClick
- PenDisplayChanged
- PenlistChanged
- ShutDown
- SizeChanged
- StartUp
- StateChanged

CurrentPenChanged

The CurrentPenChanged event is triggered when a different pen is selected.

Syntax

```
TrendClient.CurrentPenChanged();
```

DatesChanged

The DatesChanged event is triggered when a date/time for the trend chart changes because there is a scripted change to the chart's duration.

Syntax

```
TrendClient.DatesChanged();
```

MouseClick

The MouseClick event is triggered when there is a mouse click in the trend region.

Syntax

```
TrendClient.MouseClick();
```

PenDisplayChanged

The PenDisplayChanged event is triggered when the display options for a pen in the pen list are changed.

Syntax

```
TrendClient.PenDisplayChanged();
```

PenlistChanged

The PenlistChanged event is triggered when a pen is added to or removed from the pen list.

Syntax

```
TrendClient.PenlistChanged();
```

ShutDown

The ShutDown event is triggered when the client is closed.

Syntax

```
TrendClient.ShutDown();
```

SizeChanged

The SizeChanged event is triggered when a resize event is received.

Syntax

```
TrendClient.SizeChanged();
```

StartUp

The StartUp event is triggered when the client is started.

Syntax

```
TrendClient.StartUp();
```

StateChanged

The StateChanged event is triggered when a change has been made to the configuration for a pen in the pen list.

Syntax

```
TrendClient.StateChanged();
```

Colors in Trend Client

Trend Client and ActiveFactory use different color rendering schemes. ActiveFactory uses BGR and ABGR color rendering,

where:

A = Transparency (for ABGR only)

B = Blue

G = Green

R = Red

Information on how colors are rendered in ActiveFactory appears in Chapter 19, Common Properties, Methods, Events, Enums, and Data Types, of the *ActiveFactory Software User's Guide*.

In contrast, in the Trend Client, color is a .NET Framework data type. You can use various color methods to set the color, such as a predefined color name, FromARGB(), FromKnownColor(), and FromName(). For example, to set the pen color on a Trend Client Trend named **RealtimeTrend1** to **DarkGreen**, you use the following script statement:

```
RealtimeTrend1.Pen.Color =
    System.Drawing.Color.FromName("DarkGreen");
```

For more information on the color methods, see the Microsoft documentation for .NET Framework development.

.NET Colors

The following table is an overview of the color .NET color names with hexadecimal code.

Color with Hex Code	Color with Hex Code	Color with Hex Code
AliceBlue #F0F8FF	AntiqueWhite #FAEBD7	Aqua #00FFFF
Aquamarine #7FFFD4	Azure #F0FFFF	Beige #F5F5DC
Bisque #FFE4C4	Black #000000	BlanchedAlmond #FFEBCD
Blue #0000FF	BlueViolet #8A2BE2	Brown #A52A2A
BurlyWood #DEB887	CadetBlue #5F9EA0	Chartreuse #7FFF00
Chocolate #D2691E	Coral #FF7F50	CornflowerBlue #6495ED
Cornsilk #FFF8DC	Crimson #DC143C	Cyan #00FFFF
DarkBlue #00008B	DarkCyan #008B8B	DarkGoldenrod #B8860B
DarkGray #A9A9A9	DarkGreen #006400	DarkKhaki #BDB76B

Color with Hex Code	Color with Hex Code	Color with Hex Code
DarkMagenta #8B008B	DarkOliveGreen #556B2F	DarkOrange #FF8C00
DarkOrchid #9932CC	DarkRed #8B0000	DarkSalmon #E9967A
DarkSeaGreen #8FBC8B	DarkSlateBlue #483D8B	DarkSlateGray #2F4F4F
DarkTurquoise #00CED1	DarkViolet #9400D3	DeepPink #FF1493
DeepSkyBlue #00BFFF	DimGray #696969	DodgerBlue #1E90FF
Firebrick #B22222	FloralWhite #FFF0F0	ForestGreen #228B22
Fuchsia #FF00FF	Gainsboro #DCDCDC	GhostWhite #F8F8FF
Gold #FFD700	Goldenrod #DAA520	Gray #808080
Green #008000	GreenYellow #ADFF2F	Honeydew #F0FFF0
HotPink #FF69B4	IndianRed #CD5C5C	Indigo #4B0082
Ivory #FFFFFF0	Khaki #F0E68C	Lavender #E6E6FA
LavenderBlush #FFF0F5	LawnGreen #7CFC00	LemonChiffon #FFFACD
LightBlue #ADD8E6	LightCoral #F08080	LightCyan #E0FFFF
LightGoldenrodYellow #FAFAD2	LightGray #D3D3D3	LightGreen #90EE90
LightPink #FFB6C1	LightSalmon #FFA07A	LightSeaGreen #20B2AA
LightSkyBlue #87CEFA	LightSlateGray #778899	LightSteelBlue #B0C4DE
LightYellow #FFFFE0	Lime #00FF00	LimeGreen #32CD32
Linen #FAF0E6	Magenta #FF00FF	Maroon #800000
MediumAquaMarine #66CDA A	MediumBlue #0000CD	MediumOrchid #BA55D3
MediumPurple #9370DB	MediumSeaGreen #3CB371	MediumSlateBlue #7B68EE
MediumSpringGreen #00FA9A	MediumTurquoise #48D1CC	MediumVioletRed #C71585
MidnightBlue #191970	MintCream #F5FFFA	MistyRose #FFE4E1
Moccasin #FFE4B5	NavajoWhite #FFDEAD	Navy #000080
OldLace #FDF5E6	Olive #808000	OliveDrab #6B8E23
Orange #FFA500	OrangeRed #FF4500	Orchid #DA70D6
PaleGoldenrod #EEE8AA	PaleGreen #98FB98	PaleTurquoise #AFEEEE
PaleVioletRed #DB7093	PapayaWhip #FFEFD5	PeachPuff #FFDAB9

Color with Hex Code	Color with Hex Code	Color with Hex Code
DarkMagenta #8B008B	DarkOliveGreen #556B2F	DarkOrange #FF8C00
DarkOrchid #9932CC	DarkRed #8B0000	DarkSalmon #E9967A
DarkSeaGreen #8FBC8B	DarkSlateBlue #483D8B	DarkSlateGray #2F4F4F
DarkTurquoise #00CED1	DarkViolet #9400D3	DeepPink #FF1493
DeepSkyBlue #00BFFF	DimGray #696969	DodgerBlue #1E90FF
Firebrick #B22222	FloralWhite #FFF0F0	ForestGreen #228B22
Fuchsia #FF00FF	Gainsboro #DCDCDC	GhostWhite #F8F8FF
Gold #FFD700	Goldenrod #DAA520	Gray #808080
Green #008000	GreenYellow #ADFF2F	Honeydew #F0FFF0
HotPink #FF69B4	IndianRed #CD5C5C	Indigo #4B0082
Ivory #FFFFFF0	Khaki #F0E68C	Lavender #E6E6FA
LavenderBlush #FFF0F5	LawnGreen #7CFC00	LemonChiffon #FFFACD
LightBlue #ADD8E6	LightCoral #F08080	LightCyan #E0FFFF
LightGoldenrodYellow #FAFAD2	LightGray #D3D3D3	LightGreen #90EE90
LightPink #FFB6C1	LightSalmon #FFA07A	LightSeaGreen #20B2AA
LightSkyBlue #87CEFA	LightSlateGray #778899	LightSteelBlue #B0C4DE
LightYellow #FFFFE0	Lime #00FF00	LimeGreen #32CD32
Linen #FAF0E6	Magenta #FF00FF	Maroon #800000
MediumAquamarine #66CDA4	MediumBlue #0000CD	MediumOrchid #BA55D3
MediumPurple #9370DB	MediumSeaGreen #3CB371	MediumSlateBlue #7B68EE
MediumSpringGreen #00FA9A	MediumTurquoise #48D1CC	MediumVioletRed #C71585
MidnightBlue #191970	MintCream #F5FFFA	MistyRose #FFE4E1
Moccasin #FFE4B5	NavajoWhite #FFDEAD	Navy #000080
OldLace #FDF5E6	Olive #808000	OliveDrab #6B8E23
Orange #FFA500	OrangeRed #FF4500	Orchid #DA70D6
PaleGoldenrod #EEE8AA	PaleGreen #98FB98	PaleTurquoise #AFEEEE
PaleVioletRed #DB7093	PapayaWhip #FFEFD5	PeachPuff #FFDAB9

Color with Hex Code	Color with Hex Code	Color with Hex Code
DarkMagenta #8B008B	DarkOliveGreen #556B2F	DarkOrange #FF8C00
DarkOrchid #9932CC	DarkRed #8B0000	DarkSalmon #E9967A
DarkSeaGreen #8FBC8B	DarkSlateBlue #483D8B	DarkSlateGray #2F4F4F
DarkTurquoise #00CED1	DarkViolet #9400D3	DeepPink #FF1493
DeepSkyBlue #00BFFF	DimGray #696969	DodgerBlue #1E90FF
Firebrick #B22222	FloralWhite #FFF0F0	ForestGreen #228B22
Fuchsia #FF00FF	Gainsboro #DCDCDC	GhostWhite #F8F8FF
Gold #FFD700	Goldenrod #DAA520	Gray #808080
Green #008000	GreenYellow #ADFF2F	Honeydew #F0FFF0
HotPink #FF69B4	IndianRed #CD5C5C	Indigo #4B0082
Ivory #FFFFFF0	Khaki #F0E68C	Lavender #E6E6FA
LavenderBlush #FFF0F5	LawnGreen #7CFC00	LemonChiffon #FFFACD
LightBlue #ADD8E6	LightCoral #F08080	LightCyan #E0FFFF
LightGoldenrodYellow #FAFAD2	LightGray #D3D3D3	LightGreen #90EE90
LightPink #FFB6C1	LightSalmon #FFA07A	LightSeaGreen #20B2AA
LightSkyBlue #87CEFA	LightSlateGray #778899	LightSteelBlue #B0C4DE
LightYellow #FFFFE0	Lime #00FF00	LimeGreen #32CD32
Linen #FAF0E6	Magenta #FF00FF	Maroon #800000
MediumAquaMarine #66CDAA	MediumBlue #0000CD	MediumOrchid #BA55D3
MediumPurple #9370DB	MediumSeaGreen #3CB371	MediumSlateBlue #7B68EE
MediumSpringGreen #00FA9A	MediumTurquoise #48D1CC	MediumVioletRed #C71585
MidnightBlue #191970	MintCream #F5FFFA	MistyRose #FFE4E1
Moccasin #FFE4B5	NavajoWhite #FFDEAD	Navy #000080
OldLace #FDF5E6	Olive #808000	OliveDrab #6B8E23
Orange #FFA500	OrangeRed #FF4500	Orchid #DA70D6
PaleGoldenrod #EEE8AA	PaleGreen #98FB98	PaleTurquoise #AFEEEE
PaleVioletRed #DB7093	PapayaWhip #FFEFD5	PeachPuff #FFDAB9

Color with Hex Code	Color with Hex Code	Color with Hex Code
Peru #CD853F	Pink #FFC0CB	Plum #DDA0DD
PowderBlue #B0E0E6	Purple #800080	Red #FF0000
RosyBrown #BC8F8F	RoyalBlue #4169E1	SaddleBrown #8B4513
Salmon #FA8072	SandyBrown #F4A460	SeaGreen #2E8B57
SeaShell #FFF5EE	Sienna #A0522D	Silver #C0C0C0
SkyBlue #87CEEB	SlateBlue #6A5ACD	SlateGray #708090
Snow #FFFAFA	SpringGreen #00FF7F	SteelBlue #4682B4
Tan #D2B48C	Teal #008080	Thistle #D8BFD8
Tomato #FF6347	Transparent #FFFFFF	Turquoise #40E0D0
Violet #EE82EE	Wheat #F5DEB3	White #FFFFFF
WhiteSmoke #F5F5F5	Yellow #FFFF00	YellowGreen #9ACD32

Scripting Differences between Trend Client and ActiveFactory Trend Control

The Trend Client maintains the same support for run-time reference binding as all ArcestrA graphic elements.

The Trend Client properties and methods are very similar to the ActiveFactory Trend control properties and methods, as shown in the following table:

ActiveFactory Trend property or method	Trend Client property or method
(no equivalent)	"AddHistorianSource" on page 73
AddTag	AddPen on page 73
AddMultipleTags	Chart.AddMultiplePens on page 44
BackColor	Chart.BackgroundColor on page 44
(no equivalent)	Chart.Freeze on page 44
(no equivalent)	Chart.FreezeDurationMS on page 45
(no equivalent)	Chart.HidePenList on page 45
(no equivalent)	"Chart.Labels" on page 45
DefaultTagPrecision	Chart.PenPrecision on page 46

ActiveFactory Trend property or method	Trend Client property or method
(no equivalent)	Chart.RefreshEntireChartIntervals on page 46
RetrievalOptionsRetrievalMode	Chart.RetrievalMode on page 46
LiveModeRate	Chart.UpdateRateMS on page 46
ClearTags	ClearPens on page 74
DeleteCurrentTag	DeleteCurrentPen on page 75
(no equivalent)	"GetHistorianSource" on page 75
CurrentTagValAtX1	GetPenValAtX1 on page 75
(no equivalent)	"GetStartAndEndTimes" on page 76
(no equivalent)	HistorySource.Authentication on page 69
(no equivalent)	HistorySource.Connect on page 89
(no equivalent)	HistorySource.Disconnect on page 89
(no equivalent)	HistorySource.Domain on page 69
(no equivalent)	HistorySource.Password on page 69
(no equivalent)	HistorySource.RetainPassword on page 69
(no equivalent)	HistorySource.ServerName on page 70
(no equivalent)	HistorySource.Type on page 70
(no equivalent)	HistorySource.UNCPATH on page 70
(no equivalent)	HistorySource.UserID on page 71
Servers	HistorySources on page 47
(no equivalent)	HistorySources.Add on page 90
(no equivalent)	HistorySources.Count on page 71
(no equivalent)	HistorySources.GetSource on page 90
(no equivalent)	"HistorySources.Items" on page 71
(no equivalent)	HistorySources.Remove on page 91
(no equivalent)	HistorySources.Update on page 91
MoveNextTag	MoveNextPen on page 77
MovePrevTag	MovePrevPen on page 77
CurrentTagColor	Pen.Color on page 47

ActiveFactory Trend property or method	Trend Client property or method
(no equivalent)	Pen.Count on page 47
(no equivalent)	Pen.Description on page 48
(no equivalent)	Pen.Expression on page 48
CurrentTagFormat	Pen.Format on page 48
(no equivalent)	Pen.HistorySource on page 49
(no equivalent)	Pen.HistoryTagName on page 49
CurrentTagIndex	Pen.Index on page 49
(no equivalent)	Pen.Name on page 49
(no equivalent)	"Pen.Precision" on page 50
CurrentTagPrecision	Pen.RetrievalMode on page 50
CurrentTagRetrievalStyle	Pen.Style on page 50
(no equivalent)	Pen.TrendHi on page 51
(no equivalent)	Pen.TrendLo on page 51
CurrentTagTrendType	Pen.TrendType on page 51
(no equivalent)	Pen.Units on page 52
(no equivalent)	Pen.Visible on page 52
CurrentTagPenWidth	Pen.Width on page 52
CurrentTagRetrievalMode	PenSelectorHeight on page 53
(no equivalent)	PenUQRelativeOpacity on page 53
(no equivalent)	PenUQRelativeThickness on page 53
BackColor	PlotArea.BackgroundColor on page 54
BorderColor	PlotArea.BorderColor on page 54
PlotGradientEndColor	PlotArea.GradientEndColor on page 54
PlotGradient	PlotArea.GradientType on page 55
GridColor	PlotArea.GridColor on page 55
GridHorizontal	PlotArea.GridHorizontal on page 56
(no equivalent)	PlotArea.GridStyle on page 56
GridVertical	PlotArea.GridVertical on page 56
(no equivalent)	PlotArea.GridWidth on page 57

ActiveFactory Trend property or method	Trend Client property or method
HighlightCurrentTag	PlotArea.HighlightCurrentPen on page 57
(no equivalent)	PlotArea.PenHighlightColor on page 57
(no equivalent)	PlotArea.PenHighlightWidth on page 58
SingleTagMode	PlotArea.SingleTagMode on page 58
PlotImage	PlotImage on page 58
RefreshData	RefreshData on page 78
(no equivalent)	"RefreshTimes" on page 78
(no equivalent)	"RemoveHistorianSource" on page 79
RemoveTag	RemovePen on page 79
ScaleAllTags	ScaleAllPens on page 79
ScaleAutoAllTags	ScaleAutoAllPens on page 80
ScaleAutoTag	ScaleAutoPen on page 80
ScaleDownAllTags	ScaleDownAllPens on page 81
ScaleDownTag	ScaleDownPen on page 81
ScaleMoveAllTagsDown	ScaleMoveAllPensDown on page 82
ScaleMoveAllTagsUp	ScaleMoveAllPensUp on page 82
ScaleMoveTagDown	ScaleMovePenDown on page 82
ScaleMoveTagUp	ScaleMovePenUp on page 83
ScaleTag	ScalePen on page 83
ScaleUpAllTags	ScaleUpAllPens on page 84
ScaleUpTag	ScaleUpPen on page 84
SetCurrentTag	SetCurrentPen on page 84
SetDuration	SetDuration on page 85
(no equivalent)	"SetStartAndEndTimes" on page 85
AllowContextMenu	ShowContextMenu on page 58
SupressErrors	SuppressErrors on page 59
XCursor1Color	TimeAxis.Cursor1.Color on page 59
XCursor1Pos	TimeAxis.Cursor1.Pos on page 59
(no equivalent)	TimeAxis.Cursor1.Style on page 60

ActiveFactory Trend property or method	Trend Client property or method
(no equivalent)	TimeAxis.Cursor1.Width on page 60
XCursor2Color	TimeAxis.Cursor2.Color on page 60
XCursor2Pos	TimeAxis.Cursor2.Pos on page 61
(no equivalent)	TimeAxis.Cursor2.Style on page 61
(no equivalent)	TimeAxis.Cursor2.Width on page 61
NumDataPointLabels	TimeAxis.LabelColor on page 62
NumTimeAxisGridPerValue	TimeAxis.NumGridPerValue on page 62
NumTimeAxisValues	TimeAxis.NumValues on page 62
(no equivalent)	TimeAxis.ShowCursors on page 63
(no equivalent)	"TimeSelector" on page 63
(no equivalent)	"TimeSelector.DurationMS" on page 63
(no equivalent)	"TimeSelector.EndDate" on page 64
(no equivalent)	"TimeSelector.GetStartAndEndTimes" on page 86
(no equivalent)	"TimeSelector.RefreshTimes" on page 87
(no equivalent)	"TimeSelector.SetStartAndEndTimes" on page 87
(no equivalent)	"TimeSelector.StartDate" on page 64
TimeSelector.TimeDuration	TimeSelector.TimeDuration on page 64
ToolTipText	ToolTipText on page 67
(no equivalent)	TrendVersion on page 67
(no equivalent)	"UpdateHistorianSource" on page 88
NumXValueAxisLabels	ValueAxis.Label on page 67
NumYAxisGridPerValue	ValueAxis.NumGridPerValue on page 67
NumYAxisValues	ValueAxis.NumValues on page 68
Visible	Visible on page 68

Appendix A

Understanding Data Retrieval

The Trend Client supports both Cyclic and Full retrieval modes when retrieving data from Wonderware Historian. This appendix explains how both retrieval modes work.

Understanding Retrieval Modes

Different retrieval modes enable you to access data stored on an Wonderware Historian in different ways. For example, if you retrieve data over a long trend duration, you can use equal length data collection intervals to minimize response time. For a shorter period, you might want to retrieve all stored values to produce more accurate trends.

Although Wonderware Historian with a version of 9.0 or higher supports a variety of retrieval modes, the Trend Client only supports Cyclic and Full data retrieval.

Cyclic Retrieval

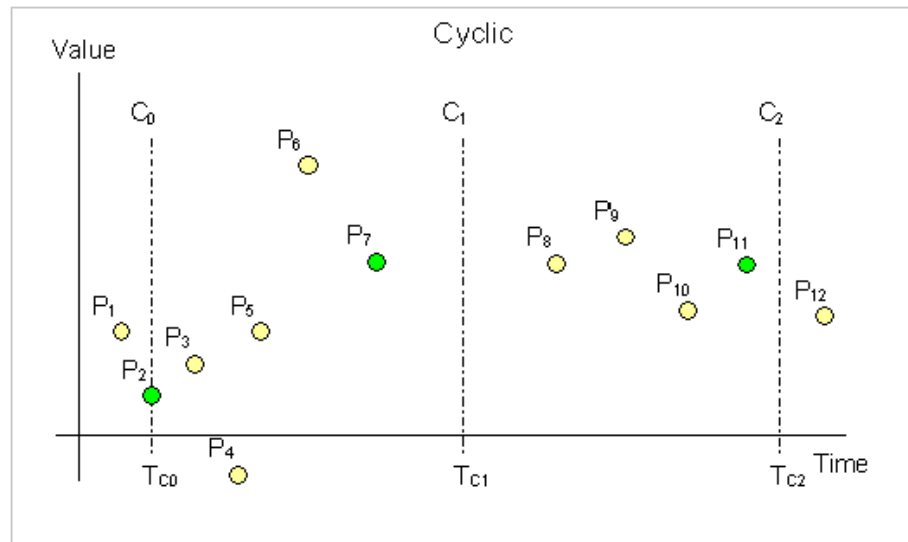
Cyclic retrieval mode returns stored data over a trend duration at equal length cyclic intervals. In Cyclic retrieval mode, a single value is returned at the beginning of each cyclic interval.

If no data is stored at the start of a cyclic interval, the last recorded value before the start of an interval is returned. If two consecutive intervals have the same value, no value is returned for the second interval.

Cyclic retrieval works with all types of tags and produces a virtual rowset, which may or may not correspond to the actual data rows stored on the Wonderware Historian.

The length of a cyclic interval is dynamic and determined by the duration of a trend and the resolution of a trend in pixels: 1 cyclic interval for every 2 pixels.

The following graph shows how tag values are retrieved from the Wonderware Historian using Cyclic retrieval mode.



Data is retrieved over a period starting at T_{C0} and an ending at T_{C2} . Each dot in the graph represents an actual data point stored in the Historian. The Historian can return data from three cyclic intervals at T_{C0} , T_{C1} , and T_{C2} .

The following data points are returned by Cyclic retrieval mode:

- At T_{C0} : P₂, because it falls on the start boundary of an interval
- At T_{C1} : P₇, because it is the last data point before the start of the next cyclic interval
- At T_{C2} : none because P₇ and P₁₁ are the same value

Cyclic retrieval mode is fast and consumes little Historian resources. However, cyclic retrieval mode may not accurately reflect stored data because important process values (data gaps, value spikes) may fall outside of the retrieval intervals and not appear as values shown in the trend.

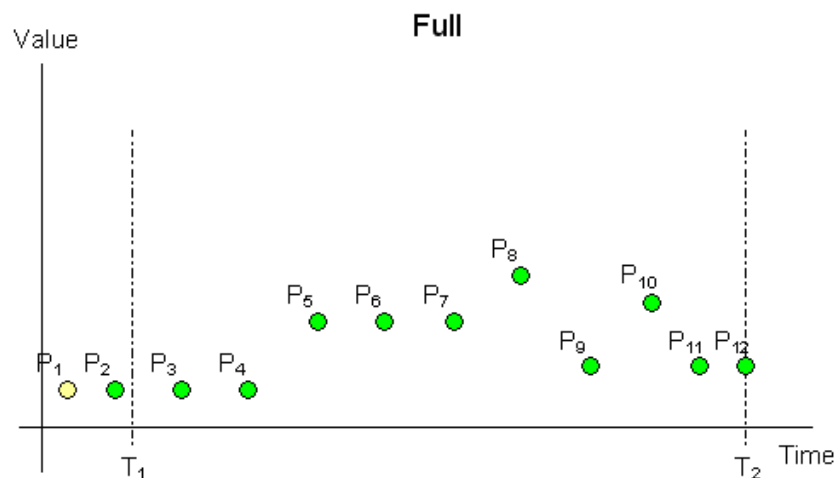
Full Retrieval

In Full retrieval mode, all stored data points are returned, regardless of whether a value or quality has changed since the last value. This mode enables the same value and quality pair (or NULL value) to be returned consecutively with their actual timestamps. It works with all types of tags.

By using Full retrieval in conjunction with storage without filtering (that is, no delta or cyclic storage mode is applied at the historian), you can retrieve all values that originated from the plant floor data source or from another application.

Full retrieval best represents the process measurements recorded by the Wonderware Historian. However, it imposes a higher load on the server, the network, and the client system because a very large number of records may be returned for longer time periods.

The following illustration shows how values are returned for Full retrieval:



Data is retrieved in full mode with a start time of T₁ and an end time of T₂. Each dot in the graphic represents an actual data point stored on the historian. From these points, the following are returned:

- P₂, because there is no actual data point at T₁
- P₃ through P₁₂, because they represent stored data points during the time period

Index

A

ActiveFactory Trend control 12, 98
AddHistorianSource 73
adding a group 30
adding a pen 17
adding a tag to a group 30
AddPen 73
animations 16, 19
applying a filter 33
attributes
 referencing a SuperTag 19

B

browser 28

C

Chart.AddMultiplePens 44
Chart.BackgroundColor 44
Chart.Freeze 44
Chart.FreezeDurationMS 45
Chart.HidePenList 45
Chart.Labels 45
Chart.PenPrecision 46
Chart.RefreshEntireChartIntervals 46

Chart.RetrievalMode 46
Chart.TimeRangePicker.TimeDuration 64
Chart.TimeSelector.TimeDuration 64
Chart.UpdateRateMS 46
clearing configuration data 16
ClearPens 74
colors in Trend Client 94
colors, .NET 94
configuring data sources 22, 23
configuring historical sources 23
configuring pen details and options 20
configuring pens for historical sources and tags 22
configuring trend appearance 34
configuring Trend Client 15
connecting a pen to an InTouch tag 18
creating a group 30
creating a new InTouch LGH connection 26
creating a Wonderware Historian connection 24, 25
creating an InTouch LGH connection 26
creating server connections 24, 26
CurrentPenChanged 92
custom properties 18
cyclic retrieval 103

D

data sources for the Trend Client 12
DatesChanged 92
DeleteCurrentPen 75
deleting a group or tag 31
differences between Trend Client and
ActiveFactory Trend control 12
Documentation 9
documentation conventions 9
domain 69

E

editing a group 30
editing a server connection 27
enabling and disabling Trend Client 16

F

filter pane 32
full retrieval 105

G

GetHistorianSource 75
GetPenValAtX1 75
GetStartAndEndTimes 76
groups 29
 adding a tag 30
 creating 30
 deleting 30
 editing 30
 renaming 31

H

handling trend events 40
hiding the Servers pane 29
hiding/showing tags 34
HistorySource.Authentication 69
HistorySource.Connect 89
HistorySource.Disconnect 89
HistorySource.Domain 69
HistorySource.Name 69, 70
HistorySource.Password 69
HistorySource.RetainPassword 69
HistorySources 47
HistorySources.Add 90
HistorySources.Count 71
HistorySource.ServerName 70
HistorySources.GetSource 90
HistorySources.Items 71

HistorySources.LogOff 89
HistorySources.LogOn 89
HistorySources.Remove 91
HistorySources.TestConnection 91
HistorySources.Update 91
HistorySource.Type 70
HistorySource.UNCPATH 70
HistorySource.UserID 71

I

InTouch 18
InTouchViewApp 19

M

managed InTouch application object 19
modifying a trend event 40
MouseClicked 92
MoveNextPen 77
MovePrevPen 77

N

.NET colors 94

O

objects 29

P

password 69
Pen.RetrievalMode 50
Pen.Color 47
Pen.Count 47
Pen.Description 48
Pen.DisplayChanged 92
Pen.Expression 48
Pen.Format 48
Pen.HistorySource 49
Pen.HistoryTagName 49
Pen.Index 49
PenlistChanged 93
Pen.Name 49
Pen.Precision 50
Pen.RetrievalMode 50
PenSelectorHeight 53
Pen.Style 50
Pen.TagName 48
Pen.TrendHi 51
Pen.TrendLo 51
Pen.TrendType 51

- Pen.Units 52
 - PenUQRelativeOpacity 53
 - PenUQRelativeThickness 53
 - Pen.Visible 52
 - Pen.Width 52
 - placing the Trend Client into an Arcestra
 - symbol 15
 - PlotArea.BackgroundColor 54
 - PlotArea.BorderColor 54
 - PlotArea.GradientEndColor 54
 - PlotArea.GradientType 55
 - PlotArea.GridColor 55
 - PlotArea.GridHorizontal 56
 - PlotArea.GridStyle 56
 - PlotArea.GridVertical 56
 - PlotArea.GridWidth 57
 - PlotArea.HighlightCurrentPen 57
 - PlotArea.PenHighlightColor 57
 - PlotArea.PenHighlightWidth 58
 - PlotArea.SingleTagMode 58
 - PlotImage 58
 - private groups 29
 - properties, methods, and events 41
 - public groups 29
- R**
- RefreshData 78
 - RefreshTimes 78
 - RemoveHistorianSource 79
 - RemovePen 79
 - removing a server connection 27
 - removing the Trend Client from an Arcestra
 - symbol 16
 - removing, server connections 27
 - renaming a group 31
 - RetainPassword 69
 - retrieval modes
 - cyclic 103
 - full 105
- S**
- ScaleAllPens 79
 - ScaleAutoAllPens 80
 - ScaleAutoPen 80
 - ScaleDownAllPens 81
 - ScaleDownPen 81
 - ScaleMoveAllPensDown 82
 - ScaleMoveAllPensUp 82
 - ScaleMovePenDown 82
 - ScaleMovePenUp 83
 - ScalePen 83
 - ScaleUpAllPens 84
 - ScaleUpPen 84
 - scripting differences between Trend Client
 - and ActiveFactory Trend Control 98
 - server connections
 - creating 24, 26
 - editing 27
 - removing 27
 - server details, viewing 31
 - server, connecting to 23
 - servers pane
 - showing/hiding 29
 - using 29
 - SetCurrentPen 84
 - SetDuration 85
 - SetStartAndEndTimes 85
 - setting chart options 36
 - setting data bindings 38, 39
 - setting trend options 36, 38, 40
 - ShowContextMenu 58
 - showing the Servers pane 29
 - showing/hiding the servers pane 29
 - ShutDown 93
 - SizeChanged 93
 - Startup 93
 - StateChanged 93
 - SuppressErrors 59
 - system objects 29
- T**
- tag picker 28
 - tags
 - adding to a group 30
 - attribute reference to a SuperTag 19
 - connecting to element animations 18
 - hiding/showing in a trend 34
 - SuperTags 19
 - tags pane, using 33
 - technical support, contacting 9
 - TimeAxis.Cursor1.Color 59
 - TimeAxis.Cursor1.Pos 59
 - TimeAxis.Cursor1.Style 60
 - TimeAxis.Cursor1.Width 60
 - TimeAxis.Cursor2.Color 60
 - TimeAxis.Cursor2.Pos 61

- TimeAxis.Cursor2.Style 61
- TimeAxis.Cursor2.Width 61
- TimeAxis.LabelColor 62
- TimeAxis.NumGridPerValue 62
- TimeAxis.NumValues 62
- TimeAxis.ShowCursors 63
- TimeSelector 63
- TimeSelector.DurationMS 63
- TimeSelector.EndDate 64
- TimeSelector.GetStartAndEndTimes 86
- TimeSelector.RefreshTimes 87
- TimeSelector.SetStartAndEndTimes 87
- TimeSelector.StartDate 64
- TimeSelector.TimeDuration 64
- ToolTipText 67
- Trend Client
 - differences from ActiveFactory Trend control 12
 - events 92
 - history sources 68, 88
 - methods 72
 - properties 41
 - scripting differences from ActiveFactory Trend control 98

- validating configuration 16
- Trend Client events 92
- Trend Client history sources 68, 88
- Trend Client methods 72
- Trend Client properties 41
- TrendVersion 67

U

- understanding retrieval modes 103
- understanding the Trend Client 11
- UpdateHistorianSource 88
- using the filter pane 32
- using the servers pane 29
- using the tag picker 28
- using the tags pane 33

V

- validating the configuration of a Trend Client 16
- ValueAxis.Label 67
- ValueAxis.NumGridPerValue 67
- ValueAxis.NumValues 68
- viewing server details 31
- Visible 68