ROCKWELL SOFTWARE

RSView® Machine Edition™

ENTERPRISE SERIES

Machine-Level HMI for Open and Embedded Solutions

User's Guide
Volume 2

Publication VIEWME-UM005D-EN-E - July 2005
Supersedes Publication VIEWME-UM005C-EN-E

Rockwell Automation

Contacting Rockwell Software

Technical Support Telephone—440-646-5800
Technical Support Fax—440-646-5801
World Wide Web—www.rockwellsoftware.com

Copyright Notice

Trademark Notice

Warranty

Doc ID ViewME-UM005D-EN-E
July 2005

18 Setting up trends

This chapter describes:

- what trends are.
- summary of steps for creating a trend graphic object.
- creating trends, and the Trend Object Properties dialog box.
- the parts of the trend graphic object.
- the different chart types.
- choosing colors, fonts, lines, and markers for the trend.
- testing the trend.
- using objects from the Trends graphic library.
- using buttons to control the trend at runtime.
- printing trend data.
- runtime errors for trends.

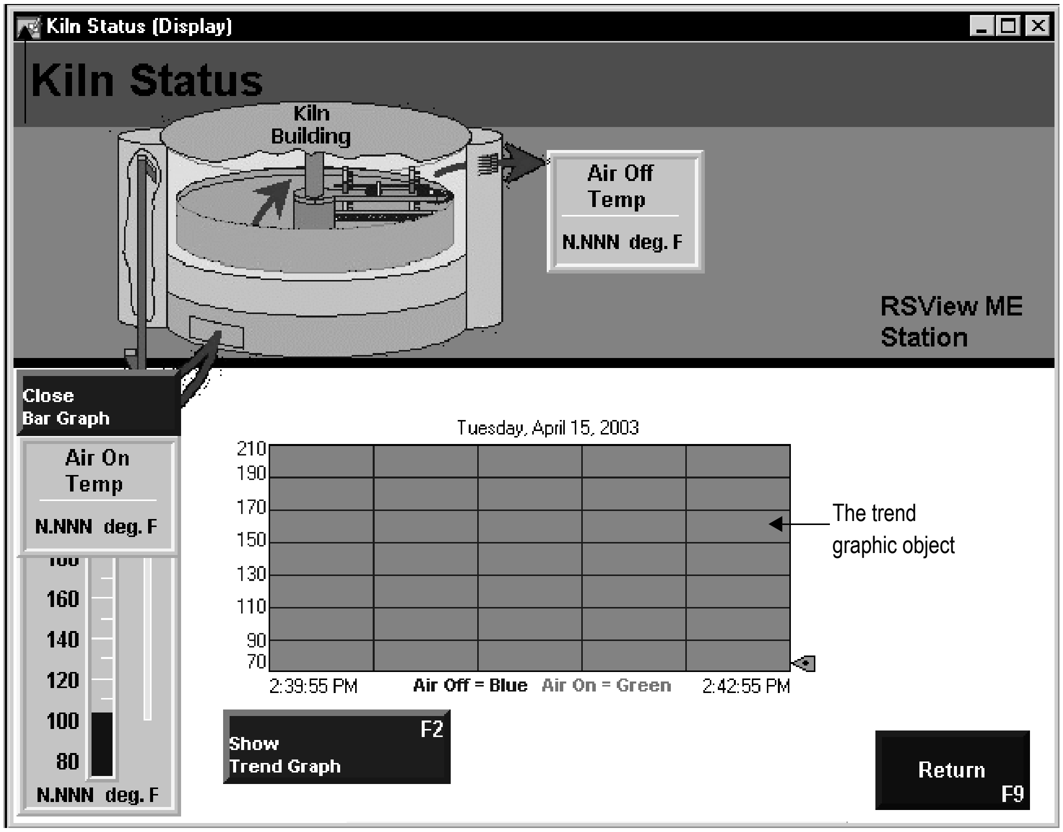
About trends

A trend is a visual representation of current or historical tag values. The trend provides operators with a way of tracking plant activity as it is happening.

You can:

- plot data for as many as eight tags or expressions on one trend.
- create a trend that is part of a graphic display or acts as the entire graphic display.
- plot data over time, or plot one variable against another in an XY Plot chart to show the relationship between them.
- display isolated or non-isolated graphs. Isolated graphing places each pen in a separate band of the chart. With non-isolated graphing, pen values can overlap.
- create buttons to allow the operator to pause, scroll, and print the trend data.

The illustration below shows a trend that has been added to a graphic display. You can view the Kiln Status display by opening the Malthouse sample application.



Current versus historical data

The data displayed in a trend can come from two sources. For current values, data comes from the value table as it is collected. The value table is a record of the most recent values collected from the data source, and is stored in temporary memory while the application is running.

For historical values, data comes from a data log model's log file, if a model is assigned to the trend. You can display both current and historical data in the same trend.

For information about data log models, see Chapter 11.

Time, date, and number formats

The trend is displayed using the time, date, and number formats of the current application language. For example, if the application language uses a comma for the decimal symbol, the scale on the y-axis uses commas for the decimal symbol.

For information about using multiple languages, see Chapter 23.

Summary of steps

These are the steps for creating a trend:

1. To plot historical data, create a data log model in the Data Log Models editor. For information, see Chapter 11.
2. Create a trend graphic object in the Graphics editor, as described on page 18-3.
3. Set up the trend in the Trend Object Properties dialog box. For details about the options in the dialog box, see Help.
4. If desired, create a next pen button, a pause button, or key buttons in the same graphic display, to allow the operator to switch between pens, pause the trend, or scroll the trend.

   For information about the buttons you can use with trends, see page 18-11.
5. To keep a printed record of the trend data, provide a way for the operator to print the graphic display. For information see page 18-12.

Creating trend objects

To create a trend object

1. In the Graphics editor, create or open a graphic display.
2. Select the Trend drawing tool by doing one of the following:
   - In the Objects toolbox, click the Trend tool.
   - On the Objects menu, select Trending, and then click Trend.
3. Drag the mouse to create a box approximately the size you want for the trend.
4. Double-click the trend to open the Trend Object Properties dialog box.
5. Set up the trend. For details, see Help.

Once you have set up the trend, you can edit it as you would any other graphic object. You can move it, resize it, attach animation to it, and so on. You can also use this object in other graphic displays by dragging it from one display and dropping it into another.

For more information about graphic objects, see Chapter 15.

Setting up trends

When you double-click a trend object, the Trend Object Properties dialog box opens. Use the dialog box to set up the trend.

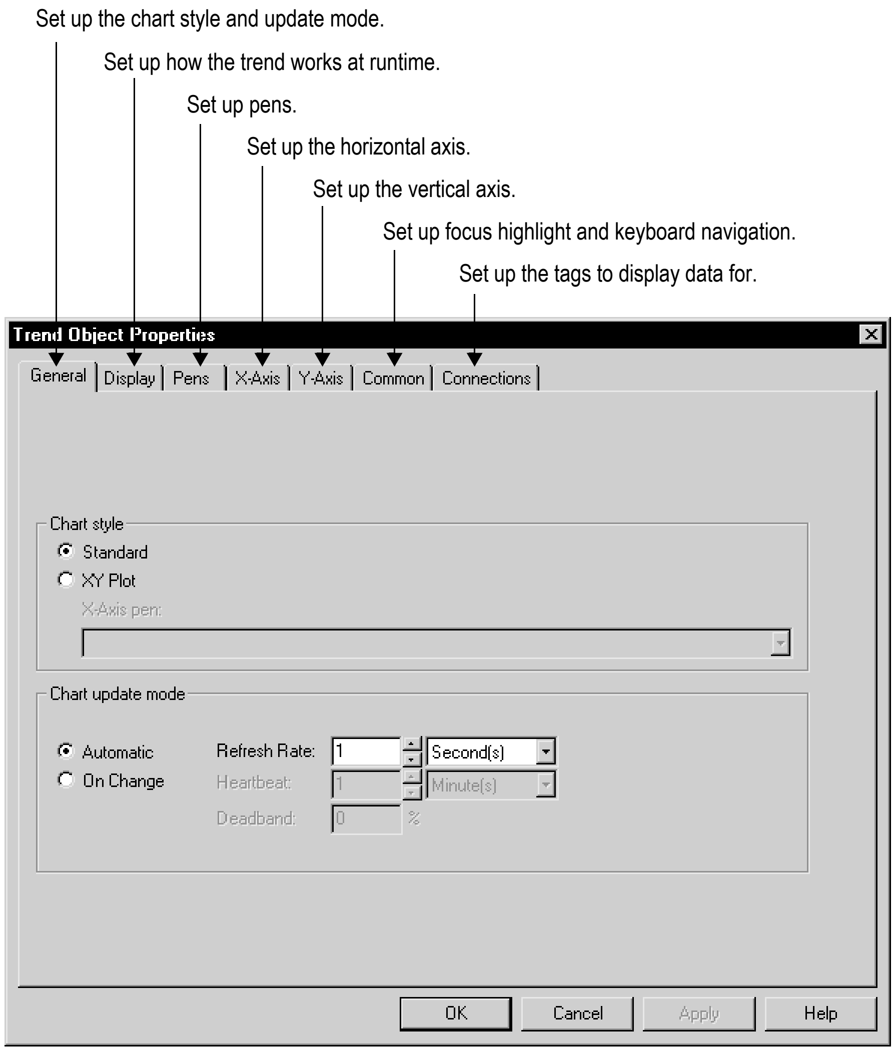

Set up the chart style and update mode.
Set up how the trend works at runtime.
Set up pens.
Set up the horizontal axis.
Set up the vertical axis.
Set up focus highlight and keyboard navigation.
Set up the tags to display data for.
Trend Object Properties
General
Display
Pens
X-Axis
Y-Axis
Common
Connections
OK
Cancel
Apply
Help

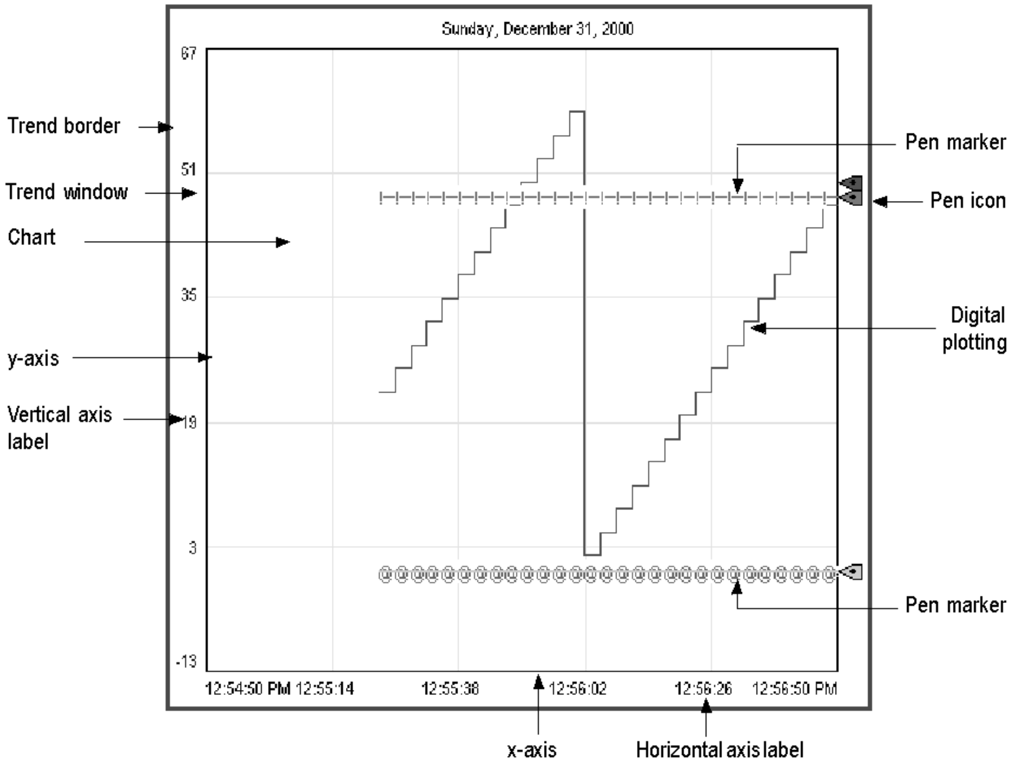
Chart style
- (●) Standard
- ( ) XY Plot
- X-Axis pen:

Chart update mode
- (●) Automatic — Refresh Rate: 1 Second(s)
- ( ) On Change — Heartbeat: 1 Minute(s)
- Deadband: 0 %

For details about the options in the Trend Object Properties dialog box, see Help.

The parts of a trend

The illustration below shows a standard trend chart, with three pens and a two-minute time span. Two of the pens have markers. The third uses digital plotting. For more information about chart types, see page 18-7.



Trend border

The border appears around the trend object at runtime when the trend is selected.

Trend window

The area around the chart, between the border and the chart, is the trend window.

Chart

The chart is the area of the trend in which values are plotted. It is bounded by the y-axis on the left and the x-axis on the bottom. It contains the plotted trend data (shown using pen lines and pen markers), as well as grid lines (if you choose to display them).

Y-axis

The y-axis is the left vertical edge of the chart. It is also known as the vertical axis.

Vertical axis labels

The vertical axis labels show the scale (range) of values for the pens. If desired, you can set up the trend to omit the vertical axis labels.

The minimum and maximum values for the scale can be determined automatically (using the best fit for the current data), be derived from a pen's minimum and maximum values, use a constant value, or be controlled by tags.

You can set up the trend so all pens use the same scale, or use individual ranges for each pen. If you choose the latter method, create a next pen button in the graphic display, to allow operators to view the range for each pen. When the operator presses the button, the vertical axis changes to the new pen's range.

For example, if Pen 1 has a minimum value of 10 and a maximum value of 100, the range on the vertical axis is 10 to 100 when the pen is selected. If Pen 2 has a minimum of -10 and a maximum of 50, the range on the vertical axis changes to -10 to 50 when the operator presses the next pen button.

X-axis

The x-axis is the bottom horizontal edge of the chart. It is also known as the horizontal axis.

Horizontal axis labels

For standard charts, the horizontal axis labels indicate the time span covered by the trend. For XY Plot charts, the horizontal axis labels show the scale (range) of values for the pen selected to serve as the x-axis pen.

If desired, you can set up the trend to omit the horizontal axis labels. The number of labels depends on the size of the trend object and the number of vertical grid lines.

Pens

Pens are the lines and symbols used to represent values. The values can be tags you are monitoring, expressions that manipulate tag values, or constants.

If there is no data for a pen, or if the data is outside the vertical axis range, the pen does not appear in the chart.

Pen icons

Pen icons appear at the right edge of the chart at runtime, if you choose to display them. The icon's position indicates the pen's most recent recorded value (from the value table), even if the trend is paused or if the most recent value has not been plotted yet.

Pen markers

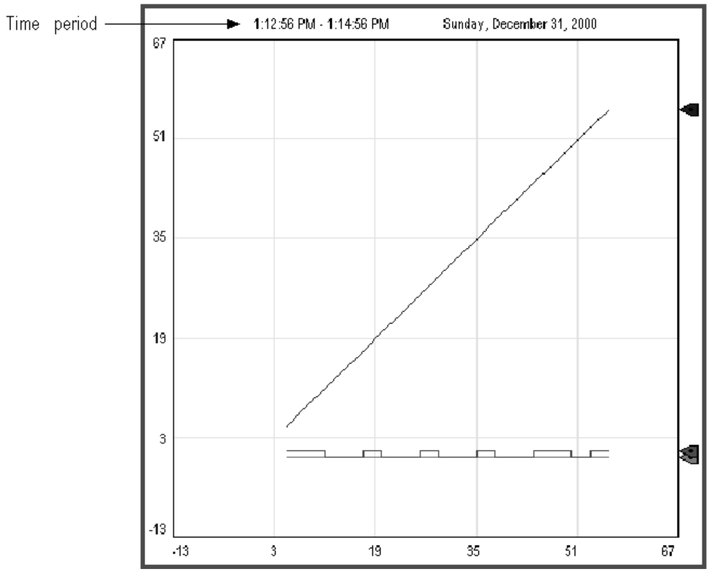
Pen markers are symbols that indicate data points. If data is plotted frequently, the markers might not appear as distinct, separate symbols. For example, see the lowest pen in the illustration on page 18-5.

Chart types

Standard vs. XY Plots

You can create a standard chart, which plots tag values against time, or an XY Plot chart, which plots one (or more) tag's values against another's.

This illustration shows what an XY Plot chart could look like:

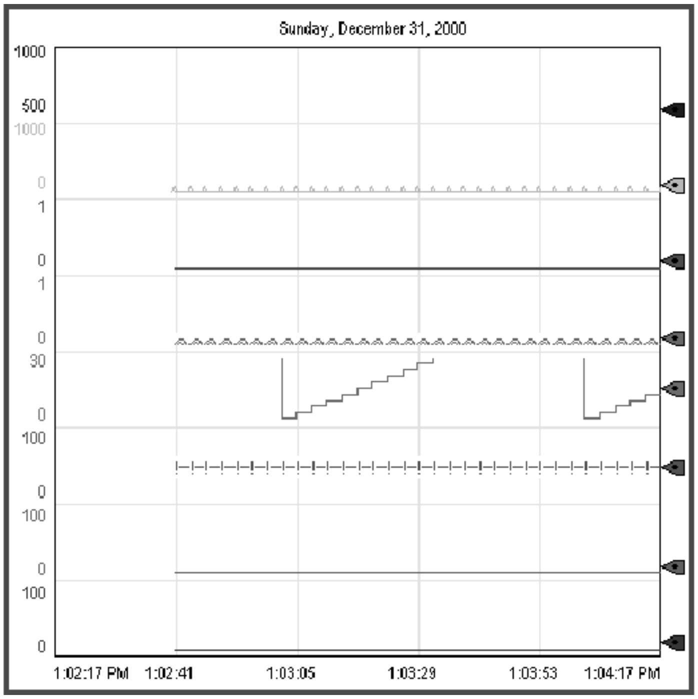


Notice that the horizontal axis labels display the range for the specified x-axis pen. The time period covered by the chart is at the upper left.

Isolated graphing

For charts with multiple pens, you can allow the pen values to overlap, or you can isolate each pen in its own horizontal band on the chart.

This is an example of isolated graphing, with a 10% buffer between each pen's band:



Notice that in this illustration each pen uses its own scale. If desired, you can use the same scale for all pens.

With isolated graphing, a grid line is automatically placed above each pen's band.

Plotting a value across the full width of the chart

Use horizontal lines to provide a frame of reference for your tag data. For example, if you define values that are the limits within which a tag must operate, and display horizontal lines in your trend to indicate the limits, when a tag crosses one of these limits the tag's alarm condition is obvious on the trend.

There are two ways to plot a value across the full width of the chart:

- In the Connections tab, assign a constant value to a pen.

When values for the pen have been plotted across the full width of the chart, the pen appears as a solid line.

- In the Connections tab, assign to a pen the tag, expression, or constant whose value will be used to determine the position of the line, and then in the Pens tab, choose the pen type Full Width.

  As soon as the trend is displayed, the pen appears as a horizontal line across the full width of the chart. Its vertical position is determined by the tag, expression, or constant's value. If the value changes, the position changes.

Choosing trend colors, fonts, lines, and markers

The following table summarizes where in the Trend Object Properties dialog box to specify colors, fonts, lines, and markers for a trend.

You can also specify these settings in the Properties tab of the Property Panel.

To specify this	**Use this box or column**	**In this tab**
Chart background color	Background color	Display
Horizontal label color	Text color	Display
Text font, style, and size	Font (button)	Display
Pen line, pen marker, pen icon, and vertical label color	Color	Pens
Pen line width	Width	Pens
Pen line style	Style	Pens
Pen marker	Marker	Pens
Vertical grid line color	Grid color	X-Axis
Horizontal grid line color	Grid color	Y-Axis

The trend border color

The trend border uses the highlight color for the graphic display, specified in the Behavior tab of the Display Settings dialog box.

The trend window color

By default, the trend window uses the background color of the display, specified in the General tab of the Display Settings dialog box.

To use a different window color

1. In the Property Panel, select the opaque WindowStyle, and then specify the WindowColor property.

For information about using the Property Panel, see page 15-31.

Testing the trend

Test Display tool

Edit Display tool

You can quickly test the trend by switching to test mode. If communications are active and there is data for the tags, the pens plot values in the trend. When you are finished testing, switch back to edit mode to continue editing.

To switch between test and edit modes

1. On the View menu, click Test Display or Edit Display, or click the Test Display and Edit Display tools.

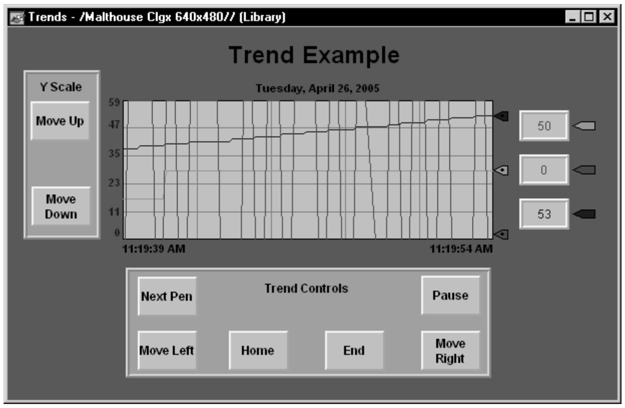
Test mode is not the same as running the display. Test mode does not change the appearance or position of the display as set up in the Display Settings dialog box. Also, data logging is not turned on in test mode.

Using the Trends graphic library

The Trends graphic library contains a trend graphic object and buttons for controlling the trend. It also contains numeric display objects that display the value of each tag used in the trend.

You can use the trend and objects as they are, or you can edit them to suit your needs. To use the objects, drag and drop (or copy and paste) them into your graphic display.

For information about copying and pasting objects from the graphic libraries, see page 15-45.



To use the Trends graphic library

1. Open the Graphics folder, and then open the Libraries folder.
2. Double-click the Trends library.
3. Drag and drop or copy and paste objects into your display.

Using buttons to control the trend at runtime

You can use button graphic objects with the trend, to allow the operator to pause the trend, switch between pens, or scroll the trend.

You can link buttons to a specific trend object, or set up a button to work with whichever object is selected in the graphic display. For information about linking buttons to objects, see page 16-10.

Use these buttons with trends:

This button	Does this
Pause	Toggles between pausing and automatic scrolling. When the trend is paused, the pen icons continue to move vertically to indicate the pens' current values. When the trend resumes scrolling, values that occurred while the trend was paused are filled in, bringing the trend up to the current time (unless you are scrolling historical data).
Next pen	Changes the vertical axis labels to the scale for the next pen. The color of the labels matches the color of the selected pen.
Move up	Scrolls up to display higher values on the vertical scale. For example, if the visible scale range is 0 to 100, pressing move up could change the visible range to 10 to 110. The incremental amount the axis scrolls depends on the pen's range and the number of horizontal grid lines. This button does not work if the "Minimum / maximum value option" in the Y-Axis tab is set to Automatic.
Move down	Scrolls down to display lower values on the vertical scale. This button does not work if the "Minimum /maximum value option" in the Y-Axis tab is set to Automatic.
Move left	Pauses the trend and scrolls to the left.
Move right	Pauses the trend and scrolls to the right.
Home	Pauses the trend and moves to the earliest data in the trend.

This button	Does this
End	Resumes trend scrolling and moves to the current (latest) data in the trend.

To see how the buttons work with the trend, open the Trends graphic library (see page 18-10), and start test mode.

For information about creating buttons, see Chapter 15. For details about setting up the buttons, see page 16-17.

Printing trend data

To print trend data at runtime, provide the operator with a method for printing the graphic display.

You can use these methods to print graphic displays at runtime:

- Create a display print button. For information about creating graphic objects, see Chapter 15.
- Assign a tag or expression to the Remote Display Print connection (in the Global Connections editor). When the value of the tag or expression changes from 0 to a non-zero value, the current display is automatically printed.

  Program the data source to trigger the change as often as you want the data printed.

  For more information about setting up remote display printing, see Chapter 8.

Everything on the screen is printed, including the current display, pop-up windows, and any visible background applications.

For information about specifying which printer to use at runtime for Windows 2000 or Windows XP applications, see page 26-10.

For information about specifying printer options for applications that will run on a PanelView Plus terminal, see the *PanelView Plus Terminals User Manual*.

For information about specifying printer options for applications that will run on a VersaView® CE terminal, see the *VersaView CE Terminals User Manual*.

Improving clarity of the trend printout

Depending on what type of printer you use, pen lines with a width of 1 pixel might not appear in the printout. Choose high-contrast colors and wider line widths to ensure that the trend data prints clearly.

Runtime errors for the trend

If data for the trend is not available at runtime due to communication errors, a message is sent to FactoryTalk® Diagnostics™.

See Help for information about solving common trend problems.

19 Setting up RecipePlus

This chapter describes:

- what recipes are.
- summary of steps for creating a recipe system.
- how the recipe system works.
- specifying the runtime location of recipe files.
- creating recipe files.
- comparing recipes.
- creating RecipePlus buttons, selectors, and tables.
- testing RecipePlus objects.
- using objects from the RecipePlus_Components graphic library.
- using buttons with the recipe objects.
- viewing data values that are saved at runtime.

About recipes

A recipe is a set of numeric and string data values (ingredients) that can be downloaded to their associated tags at the data source. Each ingredient has a pre-set data value assigned to it. The set of data values for all the ingredients in a recipe is called a data set. The set of numeric and string tags assigned to the ingredients in the recipe is called a tag set. The ingredients, data sets, and tag sets are stored together in a recipe file.

You can create different pairs of data sets and tag sets for the same set of ingredients. Each pairing of data set with tag set is called a unit. Each unit is like a unique recipe. At runtime, the operator can select the unit (recipe) that applies to the current operation.

For example, a bakery making whole wheat bread could use the same ingredients and tag sets, but depending on the type of crust desired, could use different data sets to specify different baking temperatures. As another example, you might want to have multiple production lines baking the same bread. In this case, the data set for all the production lines would be the same, but the tags receiving the recipe information would be different for each production line. Units allow you to combine different tag sets and data sets for the same set of ingredients.

The RSView® RecipePlus system allows you to create up to 15,000 ingredients, 50 data sets, 50 tag sets, and 2,500 units for each recipe file. You can create data sets at development time, edit them at runtime, and also create new data sets from tag values at runtime. You can write the data set values to tags, or write tag values to data sets.

The RecipePlus system can be used for manufacturing food and beverages, but it can also be used for any application where you want to display, edit, download, or save multiple values at once. For example, recipes are used in the petrochemical and pharmaceutical industries. In the pharmaceutical industry, you could use recipes to design flexible packaging, creating recipes that specify the number of tissues to put in a box or the number of milliliters of shampoo to put in a bottle.

Summary of steps

These are the steps for creating a recipe system:

1. In the RecipePlus Setup editor, specify the runtime location for recipe files. The files can be stored with the application or in a separate location. For details, see page 19-5.
2. In the RecipePlus Editor, set up ingredients, data sets, tag sets, and units. You can also specify a percent complete tag and a status tag for the recipe. For details, see page 19-6.
3. Create a display in the Graphics editor, containing a RecipePlus selector, table, and buttons. For details, see page 19-9.
4. If desired, create key buttons in the same graphic display, to allow the operator to use the selector and table without a keyboard.

   For information about the buttons you can use with recipes, see page 19-11.

How the recipe system works

A recipe system consists of a recipe file and the graphic objects used to work with the ingredients at runtime.

RecipePlus selector

Use the RecipePlus selector to select the recipe file and unit to work with.

RecipePlus table

Use the RecipePlus table to display the selected recipe file's ingredients, tag values, and data set values. The operator can edit data set values in the table, unless you select the View only option.

If desired, you can include a Compare column in the table, to compare tag values to data sets at a glance. If you choose this option, RSView displays an X in the Compare column when the tag value and data set value for an ingredient differ. Ingredients with an X are listed first.

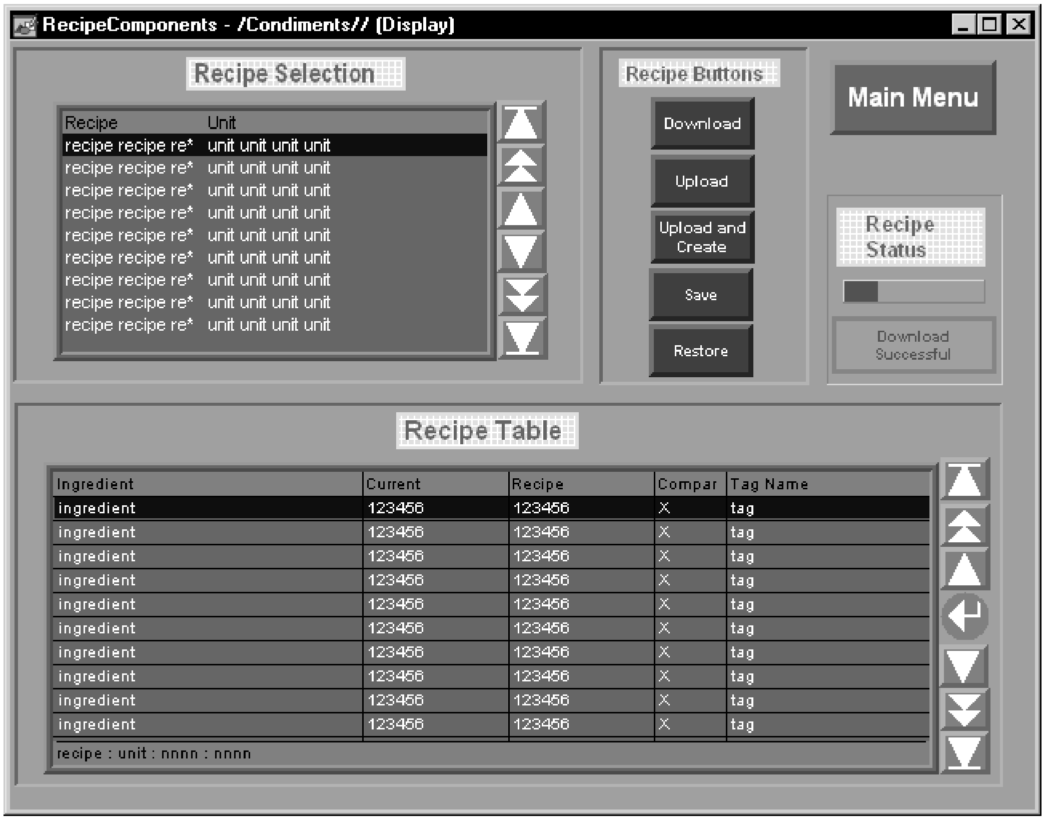
RecipePlus button

Use the RecipePlus button to perform actions on the selected recipe's ingredients. Set up a separate button for each action you want to perform:

- Restore—display the selected recipe in the RecipePlus table.
- Download—write the data set values to tags, for all the ingredients in the selected recipe.
- Upload—write tag values to the data set, for all the ingredients in the selected recipe. If all values are uploaded successfully, the recipe file is saved.
- Upload and create—write tag values to a new data set, for all the ingredients in the selected recipe. The operator is prompted for a name for the new unit. If all values are uploaded successfully, the recipe file is saved.

  The new data set is named Data Set *n*, where *n* is the next available number (starting at 1) that will create a unique data set name.
- Save—save the data set values for the recipe file and unit displayed in the RecipePlus table. If the operator made changes in the data set values using the string pop-up keyboard or numeric pop-up keypad, the new values in the table overwrite existing data set values (if any) for the unit in the recipe file.

The illustration below shows a graphic display that contains a RecipePlus selector, RecipePlus buttons, and a RecipePlus table. The display also contains key buttons for working with the selector and table, a bar graph that shows the percentage complete of the recipe operation, and a string display that shows the status of the recipe operation.



Recipe	Unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit

Ingredient	Current	Recipe	Compar	Tag Name
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag

recipe : unit : nnnn : nnnn

Number format

The values in the recipe table are displayed using the number format of the current application language. For example, if the application language uses a comma for the decimal symbol, floating-point values in the table use commas for the decimal symbol. For information about using multiple languages, see Chapter 23.

Numeric limits

RecipePlus supports the range of numbers allowed by the VARIANT data type. This range is -1.797693E+308 to 1.797693E+308. This range applies to the numbers that you enter in the RecipePlus Editor, and it also applies to the tag values that are uploaded to the recipe file at runtime.

Specifying the runtime file location

Use the RecipePlus Setup editor to specify the runtime file location.

Storing files outside the HMI project

If you store the recipe files outside the HMI project, the runtime application can use updated recipe files without creating a new runtime application (.mer) file.

Storing recipe files outside the HMI project also allows you to use RSView Studio to view and edit recipe data that is saved at runtime. For more information, see page 19-13.

If you want to store recipe files outside of the HMI project at runtime, make sure you move the files from the application's RecipePlus folder to the specified runtime location before running the application.

This is the path to the RecipePlus folder:

\Documents and Settings\All Users\Documents\RSView Enterprise\ME\HMI projects*Project name*\RecipePlus (Windows 2000)

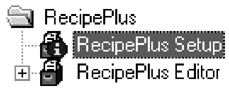
or

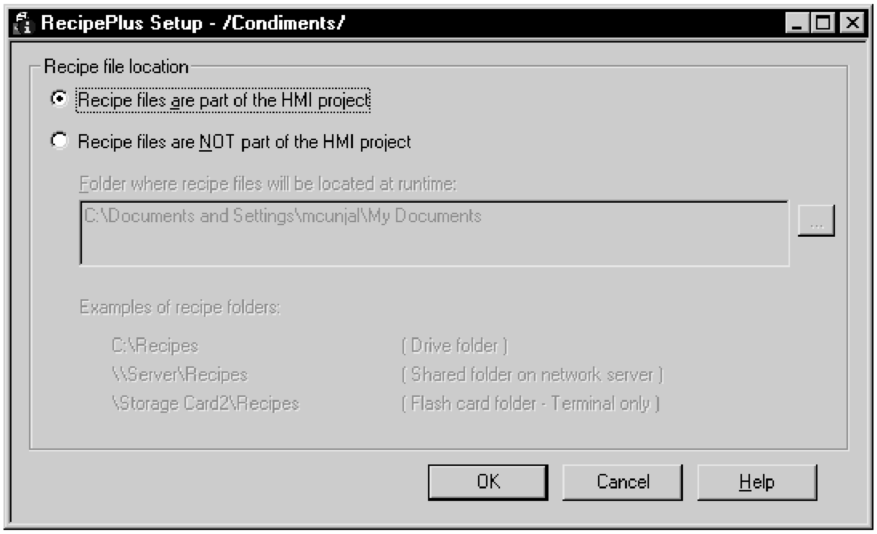
\Documents and Settings\All Users\Shared Documents\RSView Enterprise\ME\HMI projects*Project name*\RecipePlus (Windows XP)

If recipe files are stored outside the HMI project, when you perform an action on a recipe file at runtime, the file is locked until the action is completed. This prevents other users from making changes to a file while you are working with it.

Storing recipe files with the HMI project

If recipe files are part of the HMI project, when a recipe file is saved at runtime, RSView updates the .mer file with changes to the data sets. When you stop the runtime application, the changes are retained, and are displayed the next time you run the application and display the recipe file. However, you cannot upload the .mer file to RSView Studio to view the changes. Therefore, if you want to use the development computer to view runtime changes to recipe data, we recommend that you store recipe files outside the HMI project.




RecipePlus Setup - /Condiments/

Recipe file location

[✓ Recipe files are part of the HMI project]

Recipe files are NOT part of the HMI project

Folder where recipe files will be located at runtime:

C:\Documents and Settings\mcunjal\My Documents

Examples of recipe folders:

C:\Recipes	(Drive folder)
\\Server\Recipes	(Shared folder on network server)
\Storage Card2\Recipes	(Flash card folder - Terminal only)

For details about using the RecipePlus Setup editor, see Help.

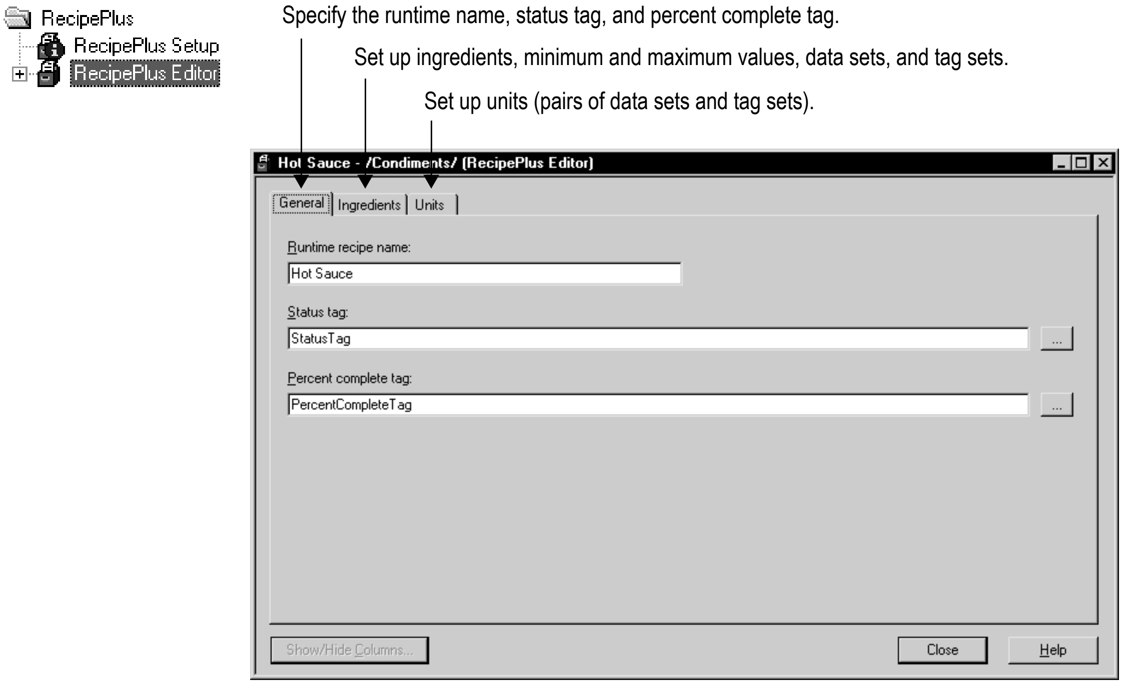
Setting up recipe files

Use the RecipePlus Editor to set up one or more recipe files. Each file is stored in the editor's folder. You can open and work on multiple recipe files at the same time.

The RecipePlus Editor has special items on the Edit menu that allow you to easily copy and paste from the spreadsheet in the Ingredients tab to Microsoft® Excel. This editor also has items on the Recipe menu, for adding, deleting, and renaming data sets and tag sets, and for comparing recipes.

To help you get started, RSView creates one data set, tag set, and unit. You can rename them and assign data values and tags to them, or delete them and create your own.

For information about comparing recipes, see the next section.


RecipePlus
RecipePlus Setup
RecipePlus Editor
Specify the runtime name, status tag, and percent complete tag.
Set up ingredients, minimum and maximum values, data sets, and tag sets.
Set up units (pairs of data sets and tag sets).
Hot Sauce - /Condiments/ (RecipePlus Editor)
General
Ingredients
Units
Show/Hide Columns...
Close
Help

Runtime recipe name: Hot Sauce

Status tag: StatusTag

Percent complete tag: PercentCompleteTag

For details about the options in the RecipePlus Editor, see Help.

You can also use the RecipePlus Editor to view the data values that are saved at runtime. For more information, see page 19-13.

Comparing recipes

You can use the RecipePlus Editor to compare data sets and tags sets within a single recipe, or between two recipes.

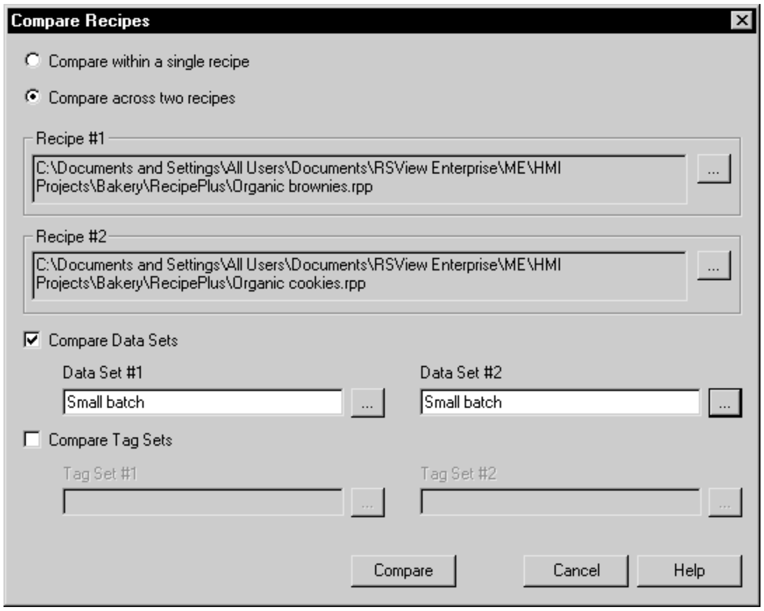
If you are comparing data sets or tag sets within a recipe, only ingredients with different values are displayed in the report.

If you are comparing data sets or tag sets between two recipes, both common ingredients and unique ingredients are listed in the report.

To compare recipes

1. Open the RecipePlus Editor.

2. On the Recipe menu, click Compare Recipes.


Compare Recipes

( ) Compare within a single recipe
(●) Compare across two recipes

Recipe #1: C:\Documents and Settings\All Users\Documents\RSView Enterprise\ME\HMI Projects\Bakery\RecipePlus\Organic brownies.rpp

Recipe #2: C:\Documents and Settings\All Users\Documents\RSView Enterprise\ME\HMI Projects\Bakery\RecipePlus\Organic cookies.rpp

[✓] Compare Data Sets

Data Set #1: Small batch

Data Set #2: Small batch

[ ] Compare Tag Sets

Tag Set #1:

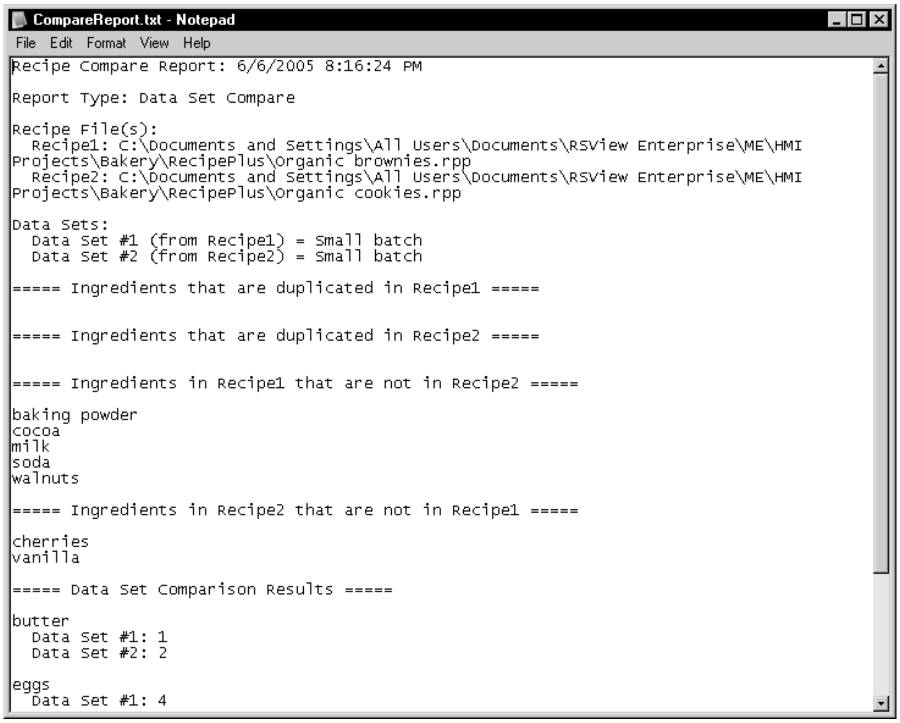
Tag Set #2:

3. Specify the recipe file or files, data sets, and tag sets to compare.

   For information about the options in the Compare Recipes dialog box, see Help.

4. Click Compare.

   A report is displayed in Windows® Notepad.



```
Recipe Compare Report: 6/6/2005 8:16:24 PM

Report Type: Data Set Compare

Recipe File(s):
  Recipe1: C:\Documents and Settings\All Users\Documents\RSView Enterprise\ME\HMI
Projects\Bakery\RecipePlus\Organic brownies.rpp
  Recipe2: C:\Documents and Settings\All Users\Documents\RSView Enterprise\ME\HMI
Projects\Bakery\RecipePlus\Organic cookies.rpp

Data Sets:
  Data Set #1 (from Recipe1) = Small batch
  Data Set #2 (from Recipe2) = Small batch

===== Ingredients that are duplicated in Recipe1 =====


===== Ingredients that are duplicated in Recipe2 =====


===== Ingredients in Recipe1 that are not in Recipe2 =====

baking powder
cocoa
milk
soda
walnuts

===== Ingredients in Recipe2 that are not in Recipe1 =====

cherries
vanilla

===== Data Set Comparison Results =====

butter
  Data Set #1: 1
  Data Set #2: 2

eggs
  Data Set #1: 4
```

Time and date formats

The time and date in the report use the time and short date format for the current application language. For information about using multiple languages, see Chapter 23.

Creating RecipePlus objects

You can create one RecipePlus table and RecipePlus selector per graphic display. You can create multiple RecipePlus buttons in a display, with a different action assigned to each.

The objects and button actions to use depend on how you want to use your recipe system. For example, if you just want to write data set values to tags, all you need is a RecipePlus selector and a RecipePlus button with the download action. For information about how the different objects in the recipe system work, see page 19-2.

To create a recipe object

1. In the Graphics editor, create or open a graphic display.
2. Select a RecipePlus drawing tool by doing one of the following:

- In the Objects toolbox, click the RecipePlus Button, RecipePlus Selector, or RecipePlus Table tool.
- On the Objects menu, select RecipePlus, and then click RecipePlus Button, RecipePlus Selector, or RecipePlus Table.

3. Drag the mouse to create a box approximately the size you want for the object.
4. Double-click the object to open its Properties dialog box.
5. Set up the object. For details, see Help.

Once you have set up a RecipePlus object, you can edit it as you would any other graphic object. You can move it, resize it, attach animation to it, and so on. You can also use the object in other graphic displays by dragging it from one display and dropping it into another.

For more information about graphic objects, see Chapter 15.

Testing RecipePlus objects

▶ Test Display tool

■ Edit Display tool

You can quickly test the recipe objects in a display by switching to test mode. If communications are active and there is data for the tags, you can download and upload recipe tag values. When you are finished testing, switch back to edit mode to continue editing.

To switch between test and edit modes

1. On the View menu, click Test Display or Edit Display, or click the Test Display and Edit Display tools.

Test mode is not the same as running the display. Test mode does not change the appearance or position of the display as set up in the Display Settings dialog box.

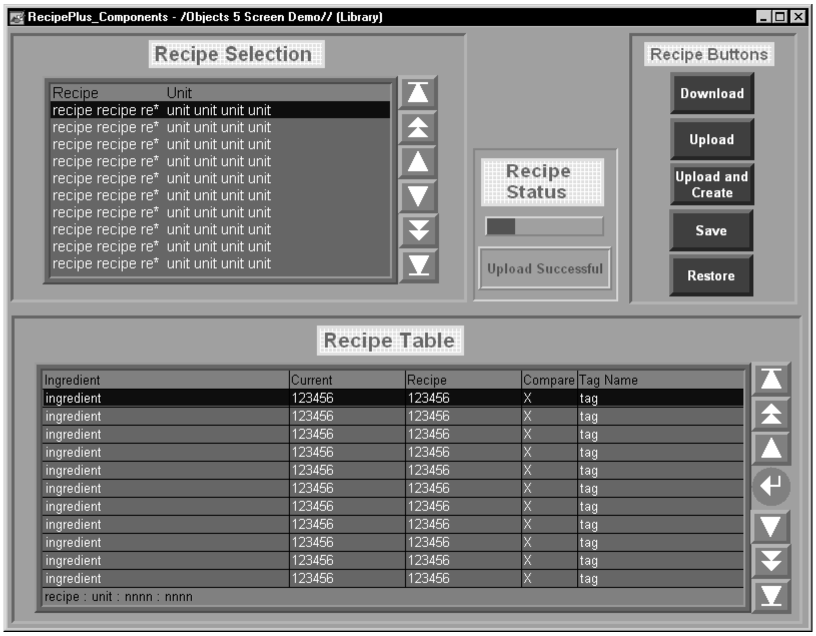
Using the RecipePlus_Components graphic library

The RecipePlus_Components graphic library contains a RecipePlus selector and table and buttons for working with the objects. It also contains a bar graph and multistate indicator that display the status of recipe operations.

Use test mode to see how the different RecipePlus objects work together. In test mode, the RecipePlus selector in the library displays any recipe files and units that you have created in your application.

You can use the objects in the library as they are, or you can edit them to suit your needs. To use the objects, drag and drop (or copy and paste) them into your graphic display.

For information about copying and pasting objects from the graphic libraries, see page 15-45.



Recipe	Unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit
recipe recipe re*	unit unit unit unit

Ingredient	Current	Recipe	Compare	Tag Name
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag
ingredient	123456	123456	X	tag

recipe : unit : nnnn : nnnn

To use the RecipePlus_Components graphic library

1. Open the Graphics folder, and then open the Libraries folder.
2. Double-click the RecipePlus_Components library.
3. Drag and drop or copy and paste objects into your display.

Using buttons with recipe objects

You can use button graphic objects with the RecipePlus selector and table, to select the recipe and unit to work with, and to select ingredients in the table.

You can link buttons to a specific recipe object, or set up a button to work with whichever object is selected in the graphic display. For information about linking buttons to objects, see page 16-10.

Use these buttons with recipe objects:

This button	Does this
Move up	Moves the highlight bar up one item in the list.

This button	Does this
Move down	Moves the highlight bar down one item in the list.
Page up	Moves the highlight bar up one page in the list.
Page down	Moves the highlight bar down one page in the list.
Home	Moves the highlight bar to the top item in the list.
End	Moves the highlight bar to the bottom item in the list.
Enter (table only)	Opens the numeric keypad or string keyboard for the operator to edit the data set value. If a numeric ingredient has a minimum and maximum value defined, these values are displayed in the numeric keypad. If the table is defined as View only, the operator cannot edit it.

To see how the buttons work with the RecipePlus selector and table, open the RecipePlus_Components graphic library (see page 19-10), and start test mode.

For information about creating buttons, see Chapter 15. For details about setting up the buttons, see page 16-17.

Example: Editing and downloading recipe values at runtime

This example shows how to use the RecipePlus graphic objects to edit and download recipe values at runtime.

1. In the RecipePlus editor, create a RecipePlus file containing ingredients, several data sets, a tag set, and several units combining the different data sets with the tag set.
2. Open the RecipePlus_Components library.
3. Start test mode.
4. Use the move up and move down buttons next to the RecipePlus selector to highlight a unit in the selector, and then press the Restore button.

   The unit's ingredients are displayed in the RecipePlus table, with the data set values in the Recipe column.
5. Use the move up and move down buttons next to the RecipePlus table to select an ingredient, and then press the Enter button.

   The numeric pop-up keypad opens, displaying the minimum and maximum values for the ingredient. If the ingredient is a string ingredient, the string pop-up keyboard opens.
6. Type a new value for the ingredient, and then press Enter.

The new value is displayed in the Recipe column.

7. Press the Save button to save the new value.
8. Press the Download button to write all the values in the Recipe column to the tags associated with the ingredients.

   The values are downloaded to the data source.

Viewing data values that are saved at runtime

Use the RecipePlus Editor in RSView Studio to view data values that have been saved at runtime.

The operator can save tag values at runtime by uploading to an existing data set or to a new data set. The operator can also edit data set values in the RecipePlus table and save the edited values (unless the table is View only).

If recipe files are stored with the HMI project, changes are saved in the .mer file only. You cannot view the changes in RSView Studio.

To view data values in modified recipe files

1. Do one of the following:
   - Add the recipe file (*.rpp) that you saved at runtime into the application using Add Component Into Application (for details, see page 2-15).
   - If the recipe file already exists in the application, you can just copy the modified file back into the application's RecipePlus folder. (For the path to the RecipePlus folder, see page 19-5.)
2. In the Explorer window in RSView Studio, double-click the modified recipe file.

   The RecipePlus Editor opens.
3. Click the Ingredients tab.
4. If the data set you want to view is not visible, scroll right to see more data sets.

20 Using expressions

This chapter describes:

- the types of expression components.
- using the Expression editor.
- which editors use expressions.
- formatting expressions.
- using tag names and tag placeholders with expressions.
- using constants.
- using operators.
- using math, security, and language functions.
- using if-then-else logic in expressions.
- the evaluation order of operators.
- using write expressions.

About expressions

Sometimes the data you gather from devices is meaningful only when you:

- compare it to other values.
- combine it with other values.
- create a cause-effect relationship with other values.

Expressions allow you to create mathematical or logical combinations of data that return more meaningful values. Depending on the components used in the expression, the value returned can be in the form of a numeric value, a true/false value, or a text string.

Expressions that result in floating-point values

If an expression results in a floating-point value but an integer value is required, the floating-point value is rounded.

For information about how values are rounded, see page 7-2.

Expression components

Expressions can be built from:

- tag values.
- tag placeholders.
- constants.
- arithmetic, relational, logical, and bitwise operators.
- mathematical and security functions.
- if-then-else logic.

Tags, arithmetic operators, bitwise operators, and mathematical functions such as SQRT (square root) return numeric values.

Relational and logical operators return true/false values. The security function CurrentUserHasCode(*x*) also returns a true/false value.

The security function CurrentUserName( ) returns a string value. The language function CurrentLanguage( ) returns a string value.

Expressions that use if-then-else logic can return numeric values, true/false values, or text strings, depending on how they are structured. These are called conditional expressions because the result of the expression depends on whether the If statement is true or false. When the If statement evaluates to true, the result is defined by the Then statement. When the If statement is false, the result is defined by the Else statement.

The Objects 5 Screen Demo sample application contains many examples of expressions. For example, see the alarm trigger expressions in the Alarm Setup editor.

Using the Expression editor

To create an expression, you can:

- type it directly in the "Tag or expression" column, for any connection that accepts expressions, or in the Expression box (for animation).
- open the Expression editor, and then create the expression in the editor.

Using the Expression editor versus typing expressions directly

Once you are familiar with expression syntax, you might find it quicker to create short expressions by typing them directly in the "Tag or expression" column.

The Expression editor allows you to see more text at once, which is useful for longer, more complicated expressions. Also, you can click buttons to enter tag names, operators, and functions, thus avoiding typing mistakes. Another advantage of using the Expression editor is that you can check whether the syntax of the expression you've created is valid.

To create an expression by typing it directly

1. Type an expression up to 999 characters long.

Expressions that you type directly are not checked for syntax.

To open the Expression editor, do one of the following

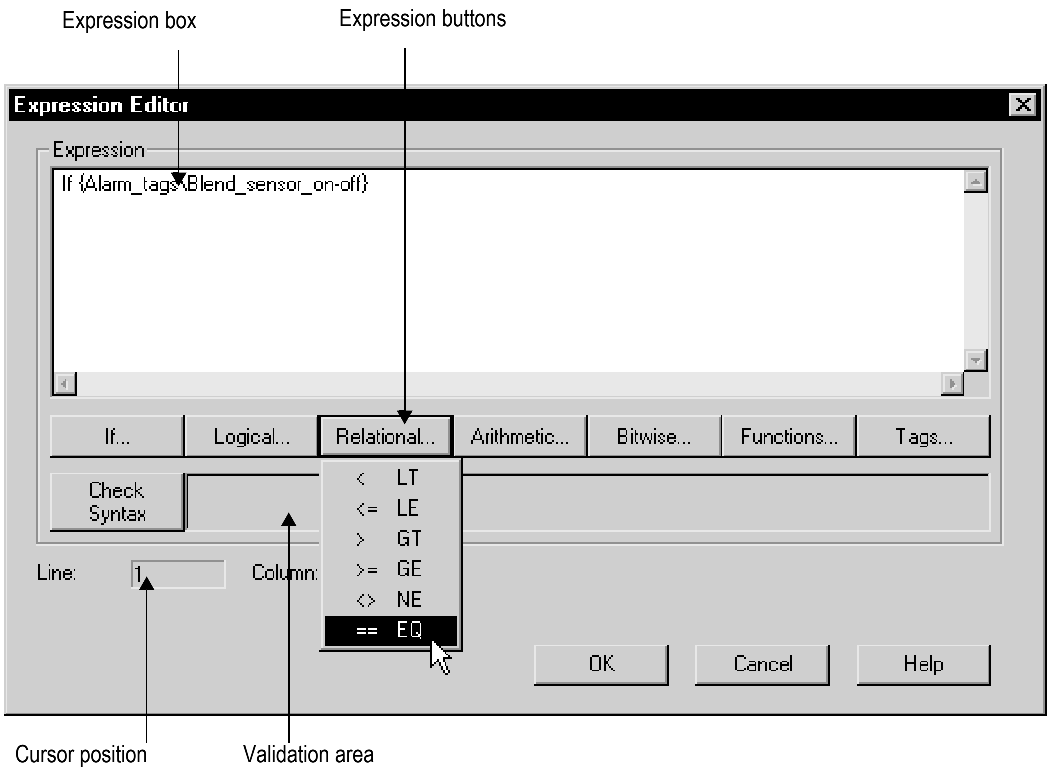
Browse button in the Exprn column

- Click the Browse button in the Exprn column for a connection that accepts expressions.

  The Browse button is not available for connections to which you can assign only tags.
- In the Animation dialog box, click the Expression button.

About the Expression editor

The Expression editor has these parts:



For details about using the options in the Expression editor, see Help.

Where you can use expressions

You can use expressions in these editors:

- Graphics—You can define an expression to control various aspects of a graphic object's appearance. For more information about assigning expressions to graphic objects, see page 15-40.

  You can also use expressions to attach animation to graphic objects. For more information, see Chapter 17.

- Global Connections—You can use expressions to remotely control when to open and print displays, as well as the date and time to display. For more information, see Chapter 8.
- Alarm Setup—When setting up alarms, you can use expressions for alarm triggers, and with some of the connections that silence and acknowledge alarms. For more information, see Chapter 9.
- Information Setup—You can use expressions to determine when to display information messages. For more information, see Chapter 12.
- Macros—You can use expressions in macros to assign values to tags. For more information, see page 22-1.

Formatting expressions

You can format expressions so they are easier to read. However, do not let tag names, function names, or function arguments span more than one line.

When formatting expressions, you can use line returns and multiple spaces.

Enclose strings in quotes. The string can contain any character, and can include spaces.

Example: Formatting an expression

To format this if-then-else statement, you can align the Else with the appropriate If, so the logic is easy to understand:

```
if (tag1 > tag2) then 0
else if (tag1 > tag3) then 2
else 4
```

Or you can condense it to the following:

```
if (tag1 > tag2) then 0 else if (tag1 > tag3) then 2 else 4
```

Using tag names and tag placeholders

A tag name can be included as part of an expression or can stand alone as the entire expression.

To supply a tag name, do one of the following:

- Type a tag name.

  You can type a tag name that does not exist in the tag database. When you click OK, you are prompted to create the tag. You can create it now, or write down the name and create it later.

- Click the Tags button and select a tag from the Tag Browser.

Enclose tag names that contain dashes or start with a number in braces { } when you use them in an expression. This distinguishes the characters in the tag name from the characters in the expression.

You can use string tags as operands with the plus (+) arithmetic operator and with the relational operators.

Using tag placeholders instead of tag names

The Graphics editor accepts tag placeholders instead of tag names. Placeholders allow you to use the same display with different sets of tags.

You can use tag placeholders in:

- the graphic display that opens when the application is first run.
- graphic displays that are opened using a goto display button.
- graphic displays that are opened using a display list selector.

Use parameter files to specify which tags to substitute for which placeholders at runtime. For information about using parameter files, see page 14-25.

The tag placeholder can replace any part of a tag name, including folder names. For example, you could create a parameter file specifying that the tag placeholder #1=Folder1. You could assign the folder and a tag name to a graphic object's connection: #1\Tag1.

To create a tag placeholder in an expression

1. Type the cross-hatch character followed by a number (no space in between). For example, #1.

Constants

A constant can have any of the following formats:

- integer (123)
- floating-point (123.45)
- string constant ("character string")

Arithmetic operators

Arithmetic operators perform math on two or more numeric values and calculate the result. The arithmetic operators are:

Symbol	Operator	Example (For these examples, tag1 = 5 and tag2 = 7)
+	addition	tag1 + tag2 returns a value of 12 You can also use this operator with string operands. See page 20-7.
-	subtraction	tag1 - tag2 returns a value of -2
*	multiplication	tag1 * tag2 returns a value of 35
/	division	tag1 / tag2 returns a value of 0.7142857
MOD,%	modulus (remainder)	tag2 MOD tag1 returns a value of 2 The modulus operator is the remainder of one number divided by another. In the example, the remainder of 7 divided by 5 is 2; so 7 % 5 = 2 **Important**: This operator is for integers only, not floating-point numbers.
**	exponent	tag1 ** tag2 returns a value of 78125

Be sure that any tag value you use as a divisor cannot at some point have a value of zero. Expressions that attempt to divide a number by zero produce an error at runtime.

String operands

The + operator can be used to join string operands. For example, the expression "hello" + "world" returns: helloworld.

You cannot join string tags to analog tags, whether they are HMI or data server tags.

Relational operators

Relational operators compare two numeric or string values to provide a true or false result. If the statement is true, a value of 1 is returned. If false, 0 is returned.

The relational operators are:

Symbols	**Operator**	**Numeric Example**	**String Example**
For the numeric examples, tag1 = 5 and tag2 = 7 For the string examples, serial_no = "ST009"			
EQ, ==	equal	tag1 == tag2 false	serial_no == "ST009" true
NE, <>	not equal	tag1 <> tag2 true	serial_no <> "ST011" true
LT, <	less than	tag1 < tag2 true	serial_no < "ST011" true
GT, >	greater than	tag1 > tag2 false	serial_no > "ST011" false
LE, <=	less than or equal to	tag1 <= tag2 true	serial_no <= "ST011" true
GE >=	greater than or equal to	tag1 >= tag2 false	serial_no >= "ST011" false

How string operands are evaluated

String operands are evaluated by case and by alphabetical order. Lower case letters are greater than upper case letters. For example, h is greater than H. Letters later in the alphabet are greater than those earlier in the alphabet. For example, B is greater than A.

Logical operators

Logical operators determine the validity of one or more statements. There are three logical operators: AND, OR, and NOT. The operators return a non-zero value if the expression is true, or a 0 if the expression is false.

Any statement that evaluates to a non-zero value is regarded as true. For example, the statement tag1 is false if the value of tag1 is 0, and true if tag1 has any other value.

The logical operators are:

Symbols	Operator	Action	Example (For these examples, tag1 = 5 and tag2 = 7)
AND, &&	and	Returns a 1 if the statements to the right and left of the operator are both true.	(tag1 < tag2) AND (tag1 == 5) Both statements are true; returns a 1.
OR, ||	or	Returns a 1 if either the statement to the left or right of the operator is true.	(tag1 > tag2) OR (tag1 == 5) tag1 == 5 is true; returns a 1.
NOT	negation	Reverses the logical value of the statement it operates on.	NOT (tag1 < tag2) Although tag1 < tag2 is true, NOT reverses the logical value; returns a 0.

The parentheses are essential in the above expressions. They determine the evaluation order of the operators. For more information, see page 20-10.

Bitwise operators

Bitwise operators examine and manipulate individual bits within a value.

These operators are for integers only, not floating-point numbers. Do not use them with tags or expressions that return floating-point values.

Symbol	Operator	Action (for examples, see page 20-10)
&	And	Compares two integers or tags on a bit-by-bit basis. Returns an integer with a bit set to 1 if both the corresponding bits in the original numbers are 1. Otherwise, the resulting bit is 0.

Symbol	Operator	Action (for examples, see page 20-10)
|	inclusive OR	Compares two integers or tags on a bit-by-bit basis. Returns an integer with a bit set to 1 if either or both of the corresponding bits in the original numbers are 1. If both bits are 0, the resulting bit is 0.
^	exclusive OR (XOR)	Compares two integers or tags on a bit-by-bit basis. Returns an integer with a bit set to 1 if the corresponding bits in the original numbers differ. If both bits are 1 or both are 0, the resulting bit is 0.
>>	right shift	Shifts the bits within an integer or tag to the right. Shifts the bits within the left operand by the amount specified in the right operand. The bit on the right disappears. Either a 0 or a 1 is shifted in on the left, depending on whether the left-most bit is a 0 or a 1, and whether the operand consists of a signed or unsigned data type. For signed data types, if the left-most bit is 0, a 0 is shifted in. If the left-most bit is 1, a 1 is shifted in. In other words, the sign of the number is preserved. For unsigned data types, a 0 is always shifted in.
<<	left shift	Shifts the bits within an integer or tag to the left. Shifts the bits within the left operand by the amount specified in the right operand. The bit on the left disappears and 0 always shifts in on the right. See “Using the left shift operator,” later in this chapter.
~	complement	Returns one’s complement; that is, it toggles the bits within an integer or tag. Reverses every bit within the number so every 1 bit becomes a 0 and vice versa.

Using the left shift operator

If the left bit is a 1 an overflow occurs, and an error message is generated. To prevent this, use the bitwise AND operator with the left shift operator in an expression. For example:

(dev << 1) & 65535

where dev is a tag whose value is being shifted left, and 65535 is 1111 1111 1111 1111 in binary form.

Examples: Bitwise operators

For these examples, tag1 = 5 (binary 0000 0000 0000 0101) and tag2 = 2 (binary 0000 0000 0000 0010)

tag1 & tag2
Returns 0 (binary 0000 0000 0000 0000).

tag1 | tag2
Returns 7 (binary 0000 0000 0000 0111).

tag1 ^ tag2
Returns 7 (binary 0000 0000 0000 0111).

tag1 >> 1
Returns 2 (binary 0000 0000 0000 0010).

tag1 << 1
Returns 10 (binary 0000 0000 0000 1010).

~ tag1
Returns -6 (binary 1111 1111 1111 1010).

Evaluation order of operators

Expressions with more than one operator are evaluated in this order:

- Operators in parentheses are evaluated first.

  Therefore, to change the order of precedence, use parentheses.
- The operator with the highest precedence is evaluated next.
- When two operators have equal precedence, they are evaluated from left to right.

Operators are evaluated in this order:

Evaluation order	**Symbols**
1 (highest)	( )
2	NOT
	~
3	*
	/
	MOD, %
	**
	AND, &&
	&
	>>
	<<
4	+
	-
	OR, ||
	|
	^
5 (lowest)	EQ, ==
	NE, <>
	LT, <
	GT, >
	LE, <=
	GE, >=

Examples: Evaluation order

For these examples, tag1 = 5, tag2 = 7, and tag3 = 10.

(tag1 > tag2) AND (tag1 < tag3)

is evaluated in this sequence:

1. tag1 > tag2 = 0
2. tag1 < tag3 = 1
3. 0 AND 1 = 0

The expression evaluates to 0 (false).

---

tag1 > tag2 AND tag3

is evaluated in this sequence:

1. tag2 AND tag3 = 1
2. tag1 > 1 = 1

The expression evaluates to 1 (true).

NOT tag1 AND tag2 > tag3 ** 2

is evaluated in this sequence:

1. NOT tag1 = 0
2. 0 AND tag2 = 0
3. tag3 ** 2 = 100
4. 0 > 100 = 0

The expression evaluates to 0 (false).

Mathematical functions

Use math functions to calculate the square root, log (natural or base 10), or trigonometry ratios (in radians or degrees) of a tag.

This function	Returns this value
SQRT (expression)	The square root of the expression
LOG (expression)	The natural log of the expression
LOG10 (expression)	The base ten log of the expression
SIN (expression)	The sine of the expression in radians
COS (expression)	The cosine of the expression in radians
TAN (expression)	The tangent of the expression in radians
ARCSIN (expression)	The arc sine of the expression in radians
ARCCOS (expression)	The arc cosine of the expression in radians
ARCTAN (expression	The arc tangent of the expression in radians
SIND (expression)	The sine of the expression in degrees
COSD (expression)	The cosine of the expression in degrees
TAND (expression)	The tangent of the expression in degrees
ARCSIND (expression)	The arc sine of the expression in degrees
ARCCOSD (expression)	The arc cosine of the expression in degrees
ARCTAND (expression)	The arc tangent of the expression in degrees

Security functions

Use security functions to control access to your application.

These functions allow you to determine a user's identity or security rights in order to limit access to the application based on these criteria.

This function	Returns this value
CurrentUserHasCode (*Security Code Letters*)	True (1) if any of the specified security codes have been assigned to the user; false (0) if not. If checking multiple security codes, do not type a space between the security code letters. For example: CurrentUserHasCode (ABP) returns the value 1 if the user has been assigned one or more of the specified codes.
CurrentUserName( )	A string containing the name of the current user. This function is case sensitive. All RSView 3.x user names use uppercase letters.

For more information about setting up security for your application, see Chapter 13.

For an example of using the CurrentUserHasCode(*x*) function, see page 13-13. For examples of using the CurrentUserName( ) function, see page 13-24.

Language function

The language function shows you which language your application is currently using.

You can display the current language in a string display, or use it in expressions to generate language-specific messages for your users.

This function	Returns this value
CurrentLanguage( )	RFC1766 name of the current runtime language.

The RFC1766 name is a standard way of representing a language using the format: *languagecode-Country/RegionCode*

where *languagecode* is a lowercase two-letter code and *Country/RegionCode* is an uppercase two-letter code.

For example, U.S. English is en-US.

For more information about setting up languages for your application, see Chapter 23. For a list of RFC1766 names, see Appendix F.

If-then-else

If-then-else expressions carry out an action conditionally or branch actions depending on the statements in the expression. The if-then-else statements enable the expression to perform different actions in different situations and to repeat activities until a condition changes.

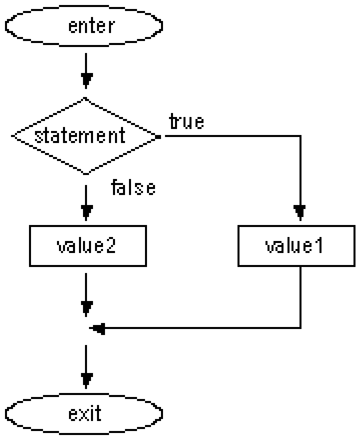
To build conditional expressions, use the relational operators and the logical operators for the statement and values.

The if-then-else structure is:

if *statement* then *value1* else *value2*

If the statement is true then the expression returns value1; if the statement is false then the expression returns value2. If the result of the statement is a non-zero value, the statement is true (and returns value1); if the result is 0, the statement is false (and returns value2).

The if-then-else structure is illustrated here.



Nested if-then-else

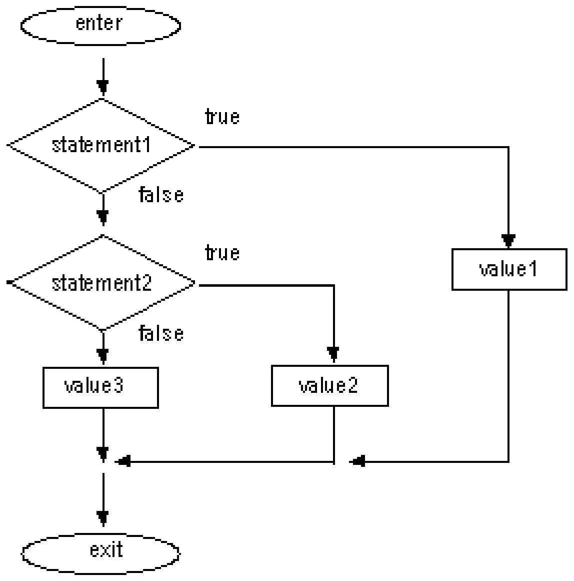
You can also nest an if-then-else structure inside the Then or Else part of an if-then-else structure.

Example 1: Nested if-then-else

This expression:

if *statement1* then *value1*
else if *statement2* then *value2*
else *value3*

has this interpretation:

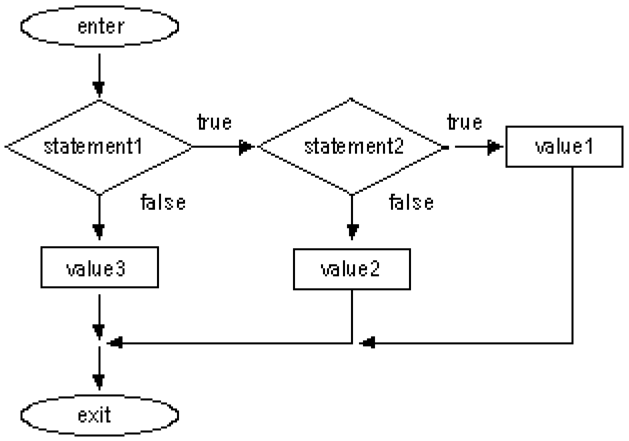


Example 2: Nested if-then-else

This expression:

if *statement1* then
if *statement2* then value1
else *value2*
else *value3*

has this interpretation:



Using write expressions

Write expressions allow the operator to enter a value which is manipulated by an expression before being sent to the data source. RSView® substitutes the value the operator enters for the placeholder in the expression, calculates the value of the expression, and writes the result to the Value connection. All write expressions must contain a question mark (?) as a placeholder for the value the operator enters.

You can use write expressions with the numeric input enable button and the numeric input cursor point. When the operator presses the button or cursor point, a keypad or scratchpad opens. The operator enters a value in the keypad or scratchpad, and this value is substituted for the ? placeholder in the write expression.

Example: Using write expressions

In this example, the operator regulates the speed of a conveyor belt by entering a value in feet or meters per second. When the operator enters the value in meters per second, the value is converted to feet per second before being passed to the data source.

The operator first indicates whether the value is in feet or meters by pushing a maintained push button. The push button has one state corresponding to feet per second, and the other state to meters per second.

Then the operator presses the numeric input enable button and enters the value for the conveyor speed in a numeric pop-up keypad. The ? character in the write expression is the placeholder for the value the operator enters.

To set up the maintained push button

1. In the Maintained Push Button Properties dialog box, in the States tab, set up these states:
   - State 0—Value: 0, Caption: Feet/S
   - State 1—Value: 1, Caption: Meters/S
2. In the Connections tab, assign a digital tag called Feet_or_meters to the Value connection (either an HMI tag or a data server tag).

To set up the numeric input enable button

1. In the Numeric Input Enable Properties dialog box, in the Label tab, type the caption "Enter conveyor speed".
2. In the Connections tab, assign a tag called Conveyor_speed to the Value connection.
3. Assign this expression to the Optional Exp connection:

   ```
   if Feet_or_meters == 0 then
   ?
   else
   ? * 3.281
   ```

RSView writes the result of the expression to the Conveyor_speed tag at the data source.

21 Using embedded variables

This chapter describes:

- the types of embedded variables.
- where you can create embedded variables.
- creating embedded variables.
- embedded variable syntax.
- how embedded variables are updated at runtime.
- how embedded variables are displayed at runtime.

About embedded variables

Embedded variables allow you to display values that change dynamically at runtime. You can use embedded variables in the text captions on graphic objects, and in message text. You can use multiple embedded variables in the same caption or message.

For example, you could embed a tag value and the time variable in a local message. At runtime when the local message is displayed, it is updated to reflect the tag's current value as the value changes. The time is also updated as the time changes.

Embedded variables can consist of:

- numeric (analog or digital) tags, including both HMI and data server tags.
- string tags, including both HMI and data server tags.
- tag placeholders. For information about tag placeholders, see page 15-41.
- the time.
- the date.

Where you can create embedded variables

You can create embedded variables in these editors:

- Graphics—Use this editor to insert embedded variables in the captions for graphic objects. For graphic objects with multiple states, you can insert different embedded variables in each state's caption.

  For information about specific graphic objects, see Chapter 16.

- Local Messages—Use this editor to insert embedded variables in local messages.

For more information about local messages, see page 14-29.

- Information Messages—Use this editor to insert embedded variables in information messages.

  For more information about information messages, see Chapter 12.

- Alarm Setup—Use this editor to insert embedded variables in alarm messages.

  For more information about alarms, see Chapter 9.

Creating embedded variables

To create an embedded variable in a graphic object's caption

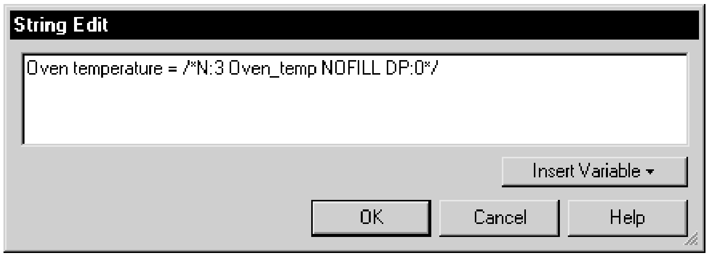
1. Open the graphic object's Properties dialog box.
2. Click the tab containing the Caption box.

   The Caption box is on the Label tab or the States tab, depending on the type of object.

   For text objects, use the Text box on the General tab.
3. Click Insert Variable.
4. Click the type of variable to insert.
5. Fill in the options in the dialog box that opens. For details about the options, see Help.

To create an embedded variable in a message

1. In the Message column of the Local Messages, Information Messages, or Alarm Setup editor, right click and then click Edit String.



Oven temperature = /*N:3 Oven_temp NOFILL DP:0*/

2. Click Insert Variable.
3. Click the type of variable to insert.
4. Fill in the options in the dialog box that opens. For details about the options, see Help.

Embedded variable syntax

Embedded variables are case sensitive, and must use specific syntax to work. Otherwise, the embedded variable is treated as a piece of text. Therefore, we do not recommend creating and editing embedded variables manually. Instead, use the Insert Variable and Edit Variable dialog boxes.

Numeric embedded variable syntax

Use numeric embedded variables to insert analog or digital tag values into text strings, including both HMI and data server tags.

Numeric embedded variables use this syntax:

/*N:# *Tag_name Fill_character* DP:#*/

where

N indicates it's a numeric embedded variable.

# indicates the number of digits.

Tag_name is the tag to display; you can also type a tag placeholder here.

Fill_character is the fill character to use: NOFILL, ZEROFILL, or SPACEFILL.

# indicates the number of decimal places.

Example: Numeric embedded variable syntax

To display the current value of a tag called Oven_temp, with 3 digits, no decimal places, and no fill character, you would type this:

/*N:3 Oven_temp NOFILL DP:0*/

String embedded variable syntax

Use string embedded variables to insert string tag values into text strings.

String embedded variables use this syntax:

/*S:# *Tag_name**/

where

S indicates it's a string embedded variable.

# indicates the number of characters if you select a fixed number of characters; type 0 if you don't want to use a fixed number.

Tag_name is the tag to display; you can also type a tag placeholder here.

Example: String embedded variable syntax

To display the current value of a string tag called Blower_status, with a fixed length of 20 characters, you would type this:

/*S:20 Blower_status*/

Time and date embedded variable syntax

Use time and date embedded variables to insert the current time or date into text strings.

Time and date embedded variables use this syntax:

/**Time_date_format**/

where

Time_date_format uses one of these character sequences:

These characters	Specify this format
SD	Short date
LD	Long date
SDT	Short date and time
LDT	Long date and time
T	Time
TSD	Time and short date
TLD	Time and long date

Example: Time and date embedded variable syntax

To display the time followed by the short date, you would type this:

/*TSD*/

A space is placed between the time and date when the embedded variable is displayed at runtime.

How embedded variables are updated at runtime

At runtime, this is how embedded variables are displayed and updated:

- Graphic objects—When a display containing a graphic object that uses an embedded variable is open, the value of the embedded variable is updated whenever a new tag value is read from the data source. For time and date embedded variables, the time and date are updated as the system time and date change.
- Local messages—When a display containing a local message display object is open, and the message the object is displaying contains an embedded variable, the value of the embedded variable is updated whenever a new tag value is read from the data source. For time and date embedded variables, the time and date are updated as the system time and date change.
- Information messages—The value of the embedded variable is read when the information message is first displayed. It is not updated after that.

  If the message is printed, it is printed using the value the variable had when the message was first displayed. This value is retained if you shut down and restart the application.
- Alarm messages—The value of the embedded variable is read when the alarm occurs, and is displayed in the message associated with the alarm. It is not updated after that.

  If the message is printed, it is printed using the value the variable had when the alarm first occurred. This value is retained if you shut down and restart the application.

How embedded variables are displayed at runtime

If there is no valid data available for the embedded variable, the variable is replaced with question marks (?). This could occur when a display first opens and the data has not arrived yet, or when there is a problem that prevents communication with the data source.

If a string or numeric embedded variable has been set up but no tag has been assigned, the embedded variable is replaced with asterisks (*).

Numeric embedded variables

The value shown for a numeric embedded variable depends on whether the tag value is a floating-point number or an integer. Integer values are displayed as is. Floating-point values are rounded to fit the specified number of digits for the variable.

For example, if the variable is set up to show 6 digits, 1234.56 is rounded to 1234.6. 1234.44 is rounded to 1234.4. The decimal counts as one of the digits.

For more information about how values are rounded, see page 7-2.

If the tag value, including the decimal point and minus sign, contains more digits than specified for the variable, the numeric variable is replaced with asterisks.

For example, if the variable is set up to show 6 digits, and the tag value is -123456, the variable will be replaced with asterisks.

Number formats

The numeric variable uses the number format of the current application language. For example, if the application language uses a comma for the decimal symbol, the numeric variable uses a comma for the decimal symbol.

For information about using multiple languages, see Chapter 23.

String embedded variables

For string embedded variables that do not use a fixed number of characters, the entire string tag value is displayed, unless a null character is read. Nothing after a null character is displayed.

If a fixed number of characters is used, the variable displays the value of the tag up to the number of characters specified, unless a null character is encountered before the specified length. Nothing is displayed after a null character. If necessary, spaces are used to make up the required number of characters.

Null characters have a hex value of 0. The null character indicates the end of string input. It does not add to the actual string length.

Time and date embedded variables

For embedded variables that show both the time and the date, a space is placed between the time and date when the embedded variable is displayed at runtime.

Time and date formats

Time and date embedded variables use the time and date formats for the current application language. For example, if you specify the short date format, at runtime the display uses the short date format that the application language uses.

For information about using multiple languages, see Chapter 23.

22 Using macros

This chapter describes:

- using macros to assign values to tags.
- using the Macros editor.
- when to use macros.
- running macros when tags or expressions change value.
- where to assign macros.

Using macros to assign values to tags

A macro is a list of tag assignments stored in a text file, in the format <tag>=<value>. Each assignment assigns a value to a tag. The value can be in the form of another tag, an expression, a numeric constant, or a string.

Examples: Using macros to set tag values

Tag1 = 8
Sets the value of Tag1 to 8.

Tank1\Message = “Tank1 overflow”
Sets the string tag Tank1\Message to Tank1 overflow.

Tag1 = Tag2
Sets the value of Tag1 to be the same as Tag2.

Tag1 = Tag1 + 1
Increases the value of Tag1 by 1.

Tag1 = if (Tag2 < Tag1) then 4 else 3
Performs the if-then-else calculation and stores the result in Tag1.

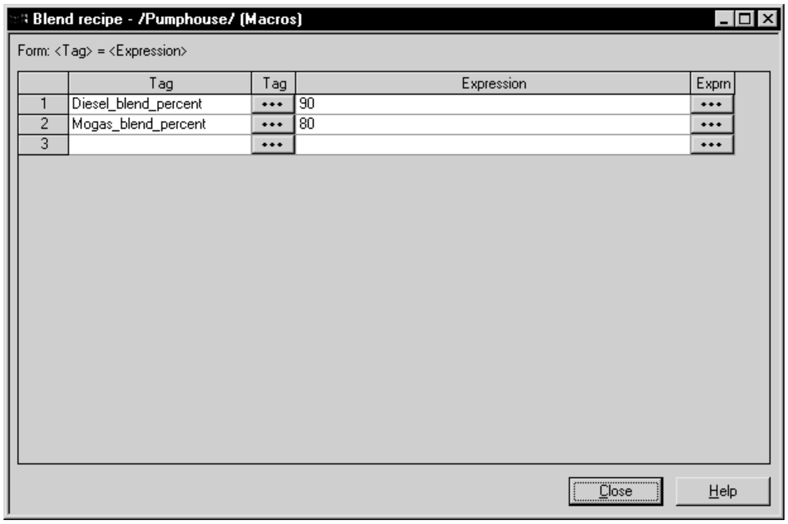
1Pump = {Industry-2} + {2Pump}
Adds the values of Industry-2 and 2Pump and stores the result in 1Pump.

Brackets surround Industry-2 because of the dash in the name. Brackets surround 2Pump because the name starts with a number. No brackets are used for 1Pump because this name is on the left side of the equal sign.

For more information about expression syntax, see Chapter 20.

Using the Macros editor

Use the Macros editor to create macros.


Blend recipe - /Pumphouse/ (Macros)
Form: <Tag> = <Expression>
Close
Help

	Tag	Tag	Expression	Exprn
1	Diesel_blend_percent	•••	90	•••
2	Mogas_blend_percent	•••	80	•••
3		•••		•••

For details about using the Macros editor, see Help.

When to use macros

You can assign macros to run when:

- the application starts or shuts down.
- a graphic display opens or closes.
- a user logs in or out. Macros can be assigned to individual users and to groups of users.
- a specified tag or expression changes to a new non-zero value (using global connections).
- an operator presses a macro button.

At runtime, when the macro runs, the values are sent to the tags at the data source.

At runtime, the tag assignments are executed asynchronously. That is, the system does not wait for the completion of one tag assignment before executing the next. Therefore, do not rely on the order of assignments to control your process.

Running macros when tags or expressions change value

You can use global connections to run macros when tags or expressions change value. This means you can use the data source to trigger the macro to run.

RSView® allows you to create up to five macros for use with global connections. The macros must be named Macro1, Macro2, Macro3, Macro4, and Macro5 in order to work with global connections.

For more information about global connections, see Chapter 8.

Example: Using macros to reset tag values

This example shows you how to run a macro whenever the operator needs to reset production information tags to known values.

The macro writes the desired values to the tags whenever the operator presses a momentary push button.

1. Create a memory tag called ResetProdData.
2. Create a momentary push button with the caption “Reset Production Data.” Assign the ResetProdData tag to the Value connection.
3. Create a macro called Macro1, with these tag assignments:

   TotalProductionUnits=0
   LineDownTime=0
   TotalRejects=0
4. In the Global Connections editor, assign the ResetProdData tag to the Remote Macro1 connection.

When the operator presses the Reset Production Data button, the value of the ResetProdData tag changes from 0 to 1. This tells RSView to run Macro1, which writes the specified values to the tags in the macro.

Where to assign macros

Once you’ve created the macros you want to use, assign the macros in these editors:

In this editor	Do this
Startup	Assign application startup and shutdown macros.

In this editor	Do this
Graphics	Assign macros to run when displays open or close, using the Display Settings dialog box.
User Accounts	Assign macros to run when users log in and log out. Macros that you assign to groups of users run each time any member of the group logs in or logs out.
Global Connections	Specify the tags or expressions that will run the macros named Macro1 to Macro5.

23 Setting up language switching

This chapter describes:

- what language switching is.
- summary of steps for setting up language switching.
- setting up Windows® to support language switching.
- adding languages to the application.
- exporting application text strings for translation.
- translating application text.
- importing translated text files.
- setting up multiple language support for graphic libraries.

About language switching

The RSView® language switching feature provides the ability to set up multiple languages for an application and switch languages dynamically at runtime. You specify an initial language for the runtime application when you create it, and select the languages that will be available at runtime. You can use up to 20 languages per application.

When the application runs, operators can change the language using a language switch button. Set up a different language switch button for each language.

With language switching you can:

- develop an application in one language, export the user-defined text strings for the application, and then import translated strings for up to 20 languages into the same application.
- use the same application in different countries, allowing operators in each location to view the application in their own language.
- allow operators in multilingual countries to use the language of their choice.
- import application components developed in different countries into a single application that supports multiple languages.

Summary of steps

Follow these steps to set up language switching for an application:

1. For applications that will run on a Windows 2000 or Windows XP operating system, install the Windows languages that the application will use.
2. For applications that will run on a PanelView Plus™ or VersaView® CE terminal, set up the fonts that the application will use.
3. Create, open, or import the application in the language of your choice. For details, see Chapter 4.
4. Add languages to the application. For details, see page 23-3.
5. Create graphic objects and specify the text strings that they will use.
6. Create a language switch button for each language you plan to use at runtime. For details, see Help.
7. Export the application text strings for translation. For details, see page 23-4.
8. Translate the text strings. For details, see page 23-6.
9. Import translated text strings for each of the translation languages. For details, see page 23-10.
10. Open the application in each language, to check the layout of the translated text.
11. Create the runtime application, specifying the startup language and the languages that the operator can switch to. For details, see Chapter 25.

Setting up Windows for language switching

Installing Windows languages

We recommend that you install all the languages that your application will use, on both the development and runtime computers. Installing languages turns on the Windows font support features, which allow applications to display characters for different languages using a single application font.

For details about installing languages, see Windows Help.

PanelView Plus and VersaView CE terminals are shipped with languages already installed.

Setting up Windows fonts

For applications that will run on a PanelView Plus or VersaView CE terminal, install the fonts that the application will use. For applications that use the Windows 2000 or Windows XP operating system, the fonts you want are generally installed when you install the languages for the application.

For details about choosing fonts that work well with language switching, see page 15-14.

Make sure that the development and runtime computers are set up to use the fonts you want for the title bar and inactive title bar. Specify the font in the Windows Control Panel, in the Display Properties dialog box.

For information about setting up fonts on a PanelView Plus or VersaView CE terminal, see the *PanelView Plus Terminals User Manual* or the *VersaView CE Terminals User Manual*. These manuals are available on the RSView Machine Edition™ CD.

Windows locale settings

Windows locale formatting determines how the application displays time, date, and floating point values at design time and runtime.

When the application language is switched, the locale settings for the new language are used even if that language has not been installed.

You do not need to edit the default locale settings.

Adding languages to the application

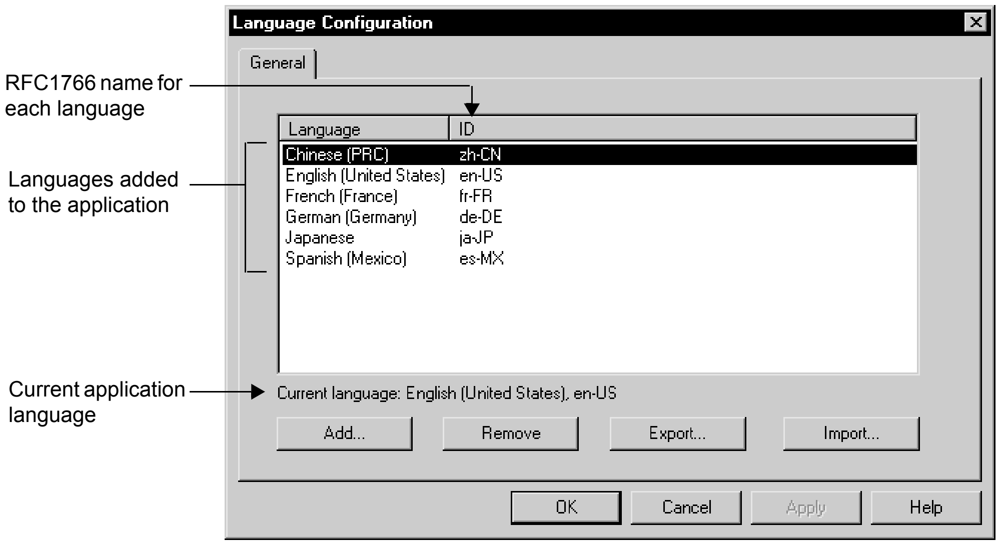
Use the Language Configuration dialog box in RSView Studio to add up to 20 languages to the application.

Add languages before you:

- create language switch buttons.
- create the runtime application.

To add languages to an application

1. On the Tools menu, click Languages.


RFC1766 name for each language
Languages added to the application
Current application language
Language Configuration
General
Add...
Remove
Export...
Import...
OK
Cancel
Apply
Help

Language	ID
Chinese (PRC)	zh-CN
English (United States)	en-US
French (France)	fr-FR
German (Germany)	de-DE
Japanese	ja-JP
Spanish (Mexico)	es-MX

Current language: English (United States), en-US

2. Add languages. For details, see Help.

Removing languages

You can also use the Language Configuration dialog box to remove languages. When you remove a language, all the strings for the language are deleted the next time you save the application. Saving a component deletes strings for the removed language from the component.

Exporting application text strings for translation

Exported text is saved to a tab-delimited text file in Unicode format. The file can be translated and then imported back into the application.

These text strings allow language switching and are exported for translation:

- display titles for On Top displays (defined in the Display Settings dialog box)
- text graphic objects
- captions that you define for graphic objects, including objects in global object displays
- text descriptions and error messages for the Change Password window. These strings are exported if you use a password button in your application.*
- local, information, and alarm messages*

* The text descriptions and error messages for the Change Password window, as well as the text strings for local, information, and alarm messages, are exported in the file. You

can translate these strings, but you cannot change their language dynamically at runtime the way you can for graphic object strings.

For the Change Password window, the runtime application will use the strings that are in the application when the runtime application is created. For example, if the current application language is French, and you have imported French text strings for the Change Password window, then the French strings will be compiled in the runtime application. If the operator opens the window, the strings will be displayed in French. If the operator changes to another language, the strings will still be displayed in French.

For local, information, and alarm messages, you can use the CurrentLanguage( ) function to generate messages in multiple languages at runtime. For information about using the language function with local messages, see page 14-33. For information about using the language function with information messages, see page 12-5. For information about using the language function with alarm messages, see page 9-16.

Exported text file locations

Exported text files are saved at this default location:

C:\Documents and Settings\All Users\Documents\RSView Enterprise\Strings (Windows 2000)

or

C:\Documents and Settings\All Users\Shared Documents\RSView Enterprise\Strings (Windows XP)

You can specify another location if desired.

File name format

The format for the exported text file name is <*ApplicationName*><*HMIServerName*><*RFC1766*>.txt, where:

- <*ApplicationName*> is the name of the application.
- <*HMIServerName*> is the name of the HMI server containing the text strings you exported. This is always the same name as the application name.
- <*RFC1766*> is the RFC1766 name associated with the language selected for the export operation.

For details about the schema of exported text files, see page 23-8.

For a list of Windows languages and the corresponding RFC1766 names, see page F-1.

To export application text to a text file

1. On the Tools menu, click Languages.

2. In the Language Configuration dialog box, select the application language for which to export text strings, and then click the Export button.
3. Follow the instructions in the String Import Export Wizard.

   For details about options in the String Import Export wizard, click Help.

If you cancel the export while it is in progress, any text files created prior to canceling might be incomplete. If you export the file multiple times to the same location, previously exported versions are overwritten.

Troubleshooting export problems

You do not have to check every text file created during an export to verify that text strings were exported correctly. If errors occur, or if you cancel the export while it is in progress, a message appears in the Diagnostics List, and in the FactoryTalk® Diagnostics™ log file.

In addition, errors are displayed when they occur in a log file called ExportErrors.txt, which is saved at this location:

C:\Documents and Settings\All Users\Documents\RSView Enterprise\Strings (Windows 2000)

or

C:\Documents and Settings\All Users\Shared Documents\RSView Enterprise\Strings (Windows XP)

Existing error log files are overwritten for each subsequent export operation that generates errors.

Translating application text

This section contains information about the format and schema of the exported file. It also contains information about using Microsoft Excel or Windows Notepad to edit the file.

Because RSView requires that parts of the text file remain the way they were exported, give the information in this section to the translator, to ensure that the file can be imported after it has been modified.

File name and format

You will likely want to rename the file before translating it, to avoid confusing it with the original file. You can use any file name ending with the file extension .txt.

To import text into an RSView application, you must save the file as tab-delimited text, in Unicode text format.

Opening the text file in Microsoft Excel

When you open the text file in Microsoft Excel, the Text Import Wizard appears.

To specify the file format (Step 1 of the wizard)

1. Click Delimited.
2. In the Start import at row box, type or select 1.
3. In the File origin list, click Windows (ANSI).
4. Click Next.

To specify the field delimiter (Step 2 of the wizard)

1. Select the Tab check box. If any other check boxes are selected, clear them.
2. Make sure the “Treat consecutive delimiters as one” check box is cleared.
3. Click Next.

To specify the column data format (Step 3 of the wizard)

1. If it is not selected already, under Column data format, click General.
2. Click Finish.

Saving the text file in Microsoft Excel

To save the file

1. On the File menu, click Save.

   Excel warns that the file may contain features that are not compatible with Unicode text.
2. When prompted to keep the workbook in Unicode format that leaves out incompatible features, click Yes.
3. Close the file.
4. When prompted to save changes, click Yes.
5. When prompted again to keep the workbook in Unicode format that leaves out incompatible features, click Yes.

Differences in file format for files saved in Excel

If you use Notepad to open a Unicode text file that was saved in Excel, you will notice some differences from a file edited and saved in Notepad.

You do not have to change the format of the file before you import it into RSView.

The differences are:

- Double quotes surrounding the string definitions are removed for most strings.
- String definitions containing embedded double quotes or other characters that Excel treats as special characters, such as commas, are enclosed within double quotes.
- Any embedded double quotes are converted to a pair of double quotes.

Saving the Unicode text file in Notepad

When saving the file, save it using the Unicode encoding option in the Save As dialog box.

File schema

Comments

The text file uses the # symbol as a comment delimiter if it is the first character on a line.

Header

The first seven lines of the text file contain header information that must not be translated or modified.

Body

The body of the text file starts on line eight, and includes the following columns:

Component type	**Component name**	**String reference**	**"String definition"**
Graphic Display	Pump station	1	"Stop motor"

The file is sorted alphabetically by component name, and then numerically by string reference number.

Each string reference number refers to a different object in the component. In the example shown above, string reference 1 might refer to a momentary push button in the graphic display called Pump station.

The string definition is enclosed in quotes in Notepad, but not in the spreadsheet column in Excel.

In the translated text file, the only text that can be modified is the text inside the quotation marks in the string definition column. For example, translated into German, the file would contain these changes:

[redacted]	**Component type**	**Component name**	**String reference**	**“String definition”**
[redacted]	Graphic Display	Pump station	1	“Motor abschalten”

Do not change the entries in the component type or component name columns, unless the component was renamed in the application after the text was exported.

Do not modify the string reference number. The string reference number is a unique number that is assigned to an object by RSView. Modifying the string reference number prevents RSView from identifying the object correctly when you import the text.

Working with pairs of double quotes

If a text string contains double quotes, the whole string definition must also be enclosed in double quotes. For example:

Call "Duty Manager"

must be entered in the string file as:

"Call "Duty Manager""

Importing text containing multiple sets of double quotes

If the string definition contains an odd number of double quotes, the number of double quotes is rounded down to an even number, and then each pair is imported as one double quote. For example, the string:

"Call "Duty Manager"

appears in the application as:

Call Duty Manager

Working with backslashes and new-line characters

To force text to begin on a new line, precede the text with the characters **\n**. For example:

Motor\nabschalten

appears in the application as:

Motor
abschalten

To make the characters \n appear as part of the text, type \\n.To make a backslash appear in the application, type two backslashes (\\).

Importing text containing multiple backslashes

If the imported text file contains an odd number of backslashes next to each other, one of the backslashes will be ignored.

For example, the string:

Seven\\\Eight

is imported into the application as:

Seven\Eight

Importing text

To import text from a file into your application, the file must be saved in Unicode text format.

To import text into your application from a text file

1. Create a backup of the text currently in your application, for the language you are about to import, by exporting it.

   Save the backup file in a different location than the translated file you are about to import.

   Backing up the language text is recommended because this allows you to restore the original text to your application if an error occurs while importing, or if you cancel the import before it is complete.

   For information about exporting text, see page 23-4.
2. On the Tools menu, click Languages.
3. In the Language Configuration dialog box, select the application language for which to import text strings, and then click the Import button.
4. Follow the instructions in the String Import Export Wizard.

   For details about options in the String Import Export wizard, click Help.

If you cancel the import before it is complete, any text strings that were changed are not restored to their original values. To restore the text that was originally in the application (for the language you just imported), import the text from the backup text file you created in step 1.

Troubleshooting importing

You do not have to check every graphic display in your application to verify that text was imported correctly. If errors occurred while importing text, they are displayed automatically from a log file called ImportErrors.txt in the following folder:

C:\Documents and Settings\All Users\Documents\
RSView Enterprise\Strings (Windows 2000)

or

C:\Documents and Settings\All Users\Shared Documents\
RSView Enterprise\Strings (Windows XP)

Each time errors occur while importing text into an application, the ImportErrors.txt file is overwritten.

If errors occurred while importing text, or if the import was canceled, a message appears in the Diagnostics List, and in the FactoryTalk Diagnostics log file.

Common errors and their causes

If text seems to have been imported for some components and not for others, see the ImportErrors.txt file for an error message. The causes and solutions for common errors are described below.

If some, but not all, of the text in an application seems to have been modified, the import might have been canceled. If you cancel the import before it is complete, any text strings that were changed are not restored to their original values. To restore the text originally in the application, import the text from the backup text file you created in step 1 on page 23-10.

In the error messages that follow, *x* is the line number in the text file.

Error message	**Cause and solution**
Server defined in *FileName*.txt does not exist. None of the strings in this file were imported.	The name of the HMI server is invalid. If the HMI server was renamed, open the file, and then correct the name of the HMI server.
File "*FileName*", Line *x*. *ComponentType* is not a valid component type for the application. The string was not imported.	The component type has been modified in the text file. Open the text file, and then correct the text for the component type.
File "*FileName*", Line *x*. *ComponentName* is not a valid component for the application. The string was not imported.	The component name has been modified in the text file, or the component in the application has been renamed. Open the text file, and then correct the text for the component name.

Error message	Cause and solution
File "*FileName*", Line *x*. The string reference must be an unsigned long integer value between 1 and 4294967295. The string was not imported.	The string reference number has been modified in the text file, and the new string reference number is invalid. Open the backup text file, and then copy the correct the string reference number into the translated text file.
File "*FileName*", Line *x*. The string was not used in the application and was not imported.	This error occurs if: ■ the string reference number has been modified in the text file, and the new string reference number is not used in the application. Open the backup text file, and then copy the correct the string reference number into the translated text file. ■ the object was deleted from the application after the text was exported. If this is correct, ignore the error.
File "*FileName*", Line *x*. The string definition must be contained within double quotes. The string was not imported.	The translated string definition includes embedded double quotes, but the string definition itself was not enclosed in double quotes. Open the text file, and then enclose all string definitions containing embedded double quotes in double quotes. For example, the string definition Start "Backup motor" must be enclosed in double quotes, like this: **"Start "Backup motor""**
File "*FileName*", Line *x*. Invalid line format!	A line in the import file does not contain all the component name or string reference number fields. The import continues with the next line in the file. Open the backup text file, and then copy the missing fields into the translated text file.
Unable to open {*FileName*.txt}. None of the strings from this file were imported.	The text file could not be opened. Make sure the text file is in the folder from which you are importing files, and that you can open the text file in Notepad or Microsoft® Excel.

Setting up multiple language support for graphic libraries

When you create an application, the graphic libraries installed with RSView are "language neutral" by default.

This means that graphic objects in the libraries always display their text strings as shipped, regardless of the current application language. Since the current application language is ignored, text strings never appear as undefined.

You can change the default for any graphic library, so that it supports multiple languages instead of a single language. When you do this, when you open the library you can view the text strings that are defined for the current application language. If the library's text strings have not been defined for the current language, the undefined strings appear as single question marks.

If you turn off a graphic library's multiple language support, and then save the library, only strings for the current application language are saved. The current application language therefore becomes the “language neutral” language. Any strings for other languages are deleted.

To turn on support for multiple languages in a graphic library

1. Right-click an empty area in the graphic library display, and then click Display Settings.
2. In the General tab, select Support Multiple Languages.

After you turn on support for multiple languages, when you save the graphic library, all strings that support language switching are saved in the current application language. For information about saving libraries in multiple languages, see page 14-17.

Using graphic libraries that support multiple languages

To use the graphics libraries in an application, you can add a graphic library into the Displays folder or copy objects from a graphic library into a graphic display. If the graphic library supports multiple languages:

- when you add the graphic library into the Displays folder, all strings, including strings for languages that are not supported by the application, are included with the display.
- if you copy an object from the graphic library into a graphic display, only strings for languages supported by the application are copied.

For more information about using graphic libraries, see Chapter 14.

24 Setting up display navigation

This chapter describes:

- what display navigation is.
- developing a hierarchy of displays.
- testing display navigation.
- using graphic objects to navigate.
- controlling display changes remotely.

This chapter describes methods for navigating between graphic displays. For information about navigating between objects in a graphic display, see page 16-9.

About display navigation

The term display navigation refers to the way the operator moves between the graphic displays that make up an application.

Use these methods to set up display navigation for your application:

- Develop a hierarchy of graphic displays, to chart how users will navigate the application.
- Determine which users will have access to which parts of the application.
- Create graphic objects that the operator can use to navigate the application.
- Use the Remote Display Number connection to automatically control display changes. The use of this connection is optional. Assign it in the Global Connections editor.
- In the Startup editor, specify the graphic display to open when the application starts.
- Set up security so that only authorized users have access to the application or parts of the application.

  For information about setting up security, see Chapter 13.

Developing a hierarchy of displays

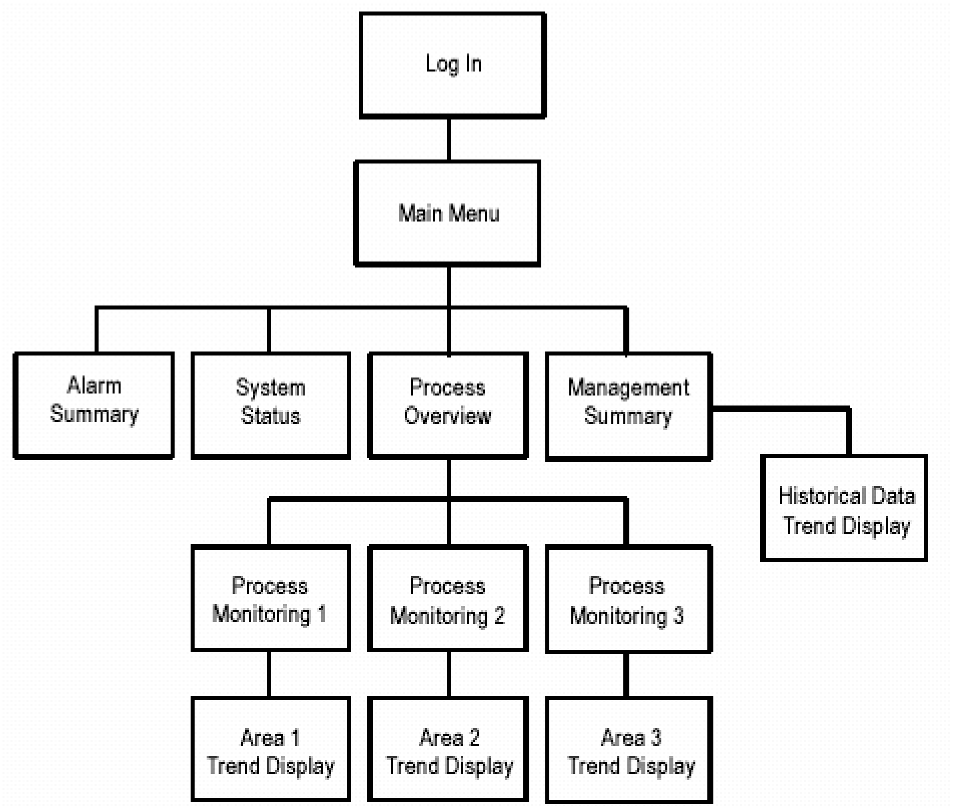
A display hierarchy is a series of graphic displays that provide progressively more detail as users move through them. Design your display hierarchy to meet the needs of the various users, including managers, supervisors, and operators.

If you plan to use security, determine which groups of users need access to which displays, and decide where in the hierarchy to locate login, logout, password, and shutdown buttons. For information about setting up security, see Chapter 13.

A display hierarchy could include:

- an initial graphic display for logging in.
- a graphic display that serves as a menu.
- an overview of the plant.
- a comprehensive display of each process being monitored.
- process-specific displays that provide more detail.
- management summary displays.
- trend displays of historical and current data.

The following illustration shows a sample display hierarchy.



Testing display navigation

Once you set up display navigation for your application, test the application to make sure that navigation flows smoothly and that you have avoided problems like these:

- A graphic display contains no buttons for moving forward or back.
- When a graphic display closes, no other display is open and there is no way to continue using the application.

For information about testing your application, see page 25-2.

Using graphic objects to navigate

Use these graphic objects to navigate through the displays in the application:

Use this object	To do this
Goto display button	Open the specified graphic display.
Goto configure mode button	Stop the application and open the RSView® ME Station™ dialog box.
Return to display button	Close the current display and open the previous display.
Close display button	Close the current display. Can send a value to a tag when the display closes.
Display list selector	Provide a list of graphic displays so the operator can select which display to open.
Shutdown button	Stop the application and exit RSView® ME Station.

The behavior of these graphic objects depends on which types of graphic displays are used. See the following sections for details.

For detailed information about setting up graphic objects, see Help.

Switching languages

If the application uses multiple languages and operators will be switching languages at runtime, place language switch buttons in a display that the operators have access to. For example, put the buttons in the display that opens when the operators log in. Create a language switch button for each language that the operators will be using.

For more information about setting up multiple languages, see Chapter 23.

Display type

The display type you use gives you additional control over how the operator navigates between displays. For example, use the On Top type to keep a graphic display on top at all times, even when another display has focus. Or use the Replace type if you want a display to replace all other open displays when it opens.

For more information about display types, see page 14-12.

Goto display buttons

When you set up a goto display button, specify the graphic display to open when the operator presses the button at runtime.

The specified display doesn't open if the operator does not have security access for the display.

You can also assign a parameter file that assigns tags to tag placeholders in the display when the display opens. For more information about parameter files, see page 14-25.

For information about setting up goto display buttons, see Help.

How display types affect the button's behavior

The goto display button's behavior at runtime also depends on which types of graphic displays are already open and which type of display it is opening:

- If the graphic display assigned to the button is a Replace display, it closes any open On Top or Replace displays. It does not close any On Top displays that use the Cannot Be Replaced option.
- If the display assigned to the button is already open, but does not have focus, pressing the button gives the display focus.

The operator cannot use the goto display button to open Replace displays if display change is currently controlled remotely (using global connections). But the operator can still open On Top displays.

Goto configure mode buttons

When the operator presses a goto configure mode button at runtime, the current application stops running and the RSView ME Station dialog box opens.

While in configure mode, the operator can use RSView ME Station to change applications, application settings, and terminal settings. The operator can also delete log files. For more information about the RSView ME Station dialog box, see Chapter 26.

Return to display buttons

When the operator presses a return to display button at runtime, the graphic display that the button is on closes and the display that was previously open reopens.

The current display does not close if:

- the display change is currently controlled remotely (using global connections).
- there were no previously opened Replace displays.
- the operator does not have security access for the previous display. This can only occur if a new user logs in using a login button in the current display.

The return to display button only goes back to the most recent display. It doesn't go back through a series of displays.

For information about setting up return to display buttons, see Help.

How display types affect the button's behavior

Both the current display and the previous display must be Replace displays. When the operator presses the return to display button:

- If the graphic display that is closing is a Replace display, the display closes and the previously opened Replace display opens. Any On Top displays that were previously open with the Replace display are not reopened.
- If the graphic display that is closing is an On Top display, the display closes but no display is reopened.

We therefore recommend that you use return to display buttons in Replace displays only.

Example: Navigating through displays

This example uses the display hierarchy illustrated on page 24-2, and shows what happens as the operator navigates through the hierarchy. The graphic displays are all Replace displays.

1. In the Main Menu display, the operator uses a display list selector to open the Process Overview display.
2. In the Process Overview display, the operator presses a goto display button to open the Process Monitoring 1 display.
3. After viewing the state of the process, the operator presses a return to display button to close the current display and reopen the Process Overview display.

4. In the Process Overview display, the operator presses a return to display button. Which display opens? The Process Monitoring 1 display (because this was the previously opened display).

   To return to the Main Menu display from the Process Overview display, the operator would have to press a goto display button that is set up to open the Main Menu display.

Close display buttons

When the operator presses a close display button at runtime, the graphic display that the button is on closes. You can set up the button to write out a value when the display closes.

If the graphic display that is closing is a Replace display, and the display change is controlled remotely, the display does not close. If the display does not close, the close value, if any, is not written out.

If the display change is controlled by the operator and the graphic display that is closing is a Replace display, if there are no On Top displays open, an empty window is displayed. The operator will not be able to use the application again (unless a remote display change occurs or an alarm, activity, or information message display opens).

We therefore recommend that you use close display buttons in On Top displays only.

For information about setting up close display buttons, see Help.

Display list selectors

Use the display list selector to show a list of graphic displays that the operator can choose from. The operator can scroll through the list and select the graphic display to open.

The specified display doesn't open if the operator does not have security access for the display.

You can also assign a parameter file that assigns tags to tag placeholders in the display when the display opens. For more information about parameter files, see page 14-25.

For information about setting up display list selectors, see Help.

How display types affect the selector's behavior

The display list selector's behavior at runtime also depends on which types of graphic displays are already open and which type of display it is opening:

- If the selected graphic display is a Replace display, it closes any open On Top and Replace displays. It does not close On Top displays that use the Cannot Be Replaced option.
- If the selected display is an On Top display, it opens on top of the current display. The current display does not close.

The operator cannot use the display list selector to open Replace displays if display change is currently controlled remotely. But the operator can still open On Top displays (with or without the Cannot Be Replaced option).

Selecting the display to open

The operator can scroll through the list and select displays using the key button graphic objects, or, if the list has the input focus, by using the arrow keys and Enter key on a keypad or external keyboard.

You can link key buttons to a specific display list selector, or set up the buttons to work with whichever object is selected in the graphic display.

For information about input focus, see page 16-9. For information about linking buttons to the display list selector, see page 16-10.

Shutdown buttons

When the operator presses the shutdown button at runtime, the application stops and RSView ME Station closes.

To prevent an unauthorized user from stopping the application, assign visibility animation to the shutdown button. For details, see page 13-15. Or, place the button in a display that only authorized users have access to.

For information about setting up shutdown buttons, see Help.

Controlling display changes remotely

To control display changes remotely, you can set up the data source to open graphic displays using global connections.

Global connections are connections that apply to your entire runtime application. Global connections allow the data source to control or interact with your application at runtime.

For example, the Remote Display Number connection is a global connection that you can use to control display changes from the data source. You can also use global connections to print graphic displays from the data source, to run macros from the data source, and to control the date and time displayed on the runtime terminal. For more information about global connections, see Chapter 8.

25 Specifying startup settings, testing applications, and creating runtime applications

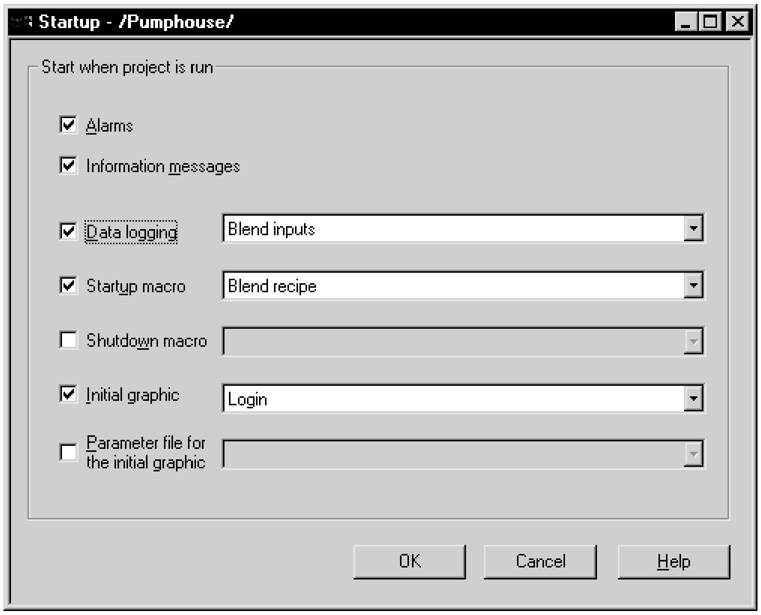
This chapter describes:

- specifying startup settings.
- testing your application.
- creating runtime application files.

Specifying startup settings

Use the Startup editor to specify which application processes and components to start when the application starts at runtime.

You can specify startup settings once you've set up all the parts of the application, or you can specify processes and select components in the Startup editor as you create them.



Start when project is run

- [✓] Alarms
- [✓] Information messages
- [✓] Data logging: Blend inputs
- [✓] Startup macro: Blend recipe
- [ ] Shutdown macro
- [✓] Initial graphic: Login
- [ ] Parameter file for the initial graphic

For detailed information about the options in the Startup editor, see Help.

Testing your application

You can test your application in RSView® Studio™ at any time during the development process, to make sure that everything works the way you intend.

If the development computer is connected to the data source, you can test all functions of the application, including security settings, language switching, communications, and alarm monitoring.

An RSView ME Station™ emulator opens on the development computer and runs the application. This runtime version of the application is a temporary version for testing use only. You cannot run it on another computer.

There is a two-hour time limit for test running the application in RSView Studio.

The procedure in this section shows you how to test your entire application. For information about testing a single graphic display, see page 14-10.

To test your application in RSView Studio

Test Application tool

1. On the Application menu, click Test Application, or click the Test Application tool.
2. If your application uses multiple languages, specify the languages to include and the initial runtime language, and then press Finish. For details, see Help.
3. Test your application.
4. To stop your application, press a shutdown button, or type the character 'x.'

   Make sure you provide the operator with a method for shutting down the application at runtime. For more information about methods for shutting down applications, see page 26-5.

Once you've tested your application to make sure everything works the way you intend, create the runtime application file and transfer the file to the runtime computer.

Creating runtime application files

Before you can run your application, you must create a runtime version. When you create the runtime version, RSView Studio compiles all of the necessary application information into a single file with the extension .mer.

Creating .mer files for previous versions

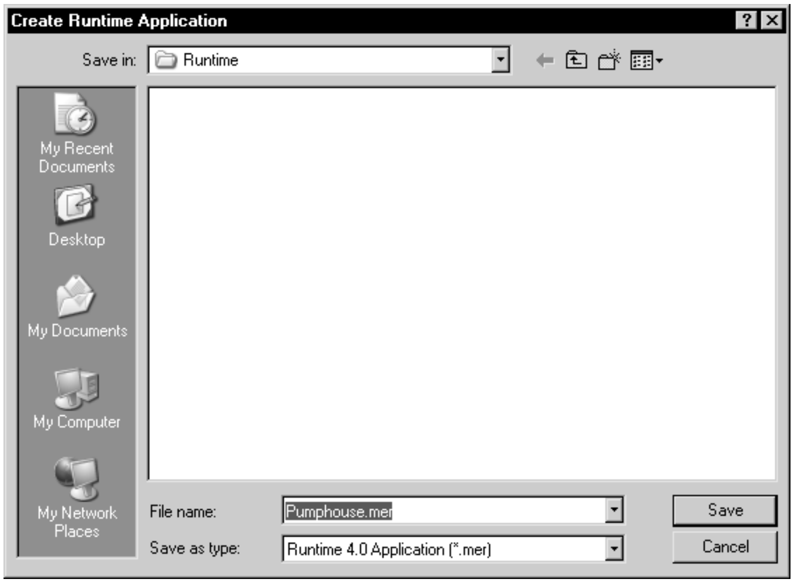
You can specify the version of RSView ME Station for which to create the .mer file. For example, if the application will run on a terminal that uses RSView ME Station version 3.2, you can specify that version for the .mer file.

If the application contains features that are not supported by the version you select, RSView displays a validation report that lists the unsupported features. The runtime application file is not created. You must remove or turn off the unsupported features before you can create the runtime application file.

For information about the features supported in different versions of RSView, and how to remove or replace them, see Appendix G.

To create a runtime application

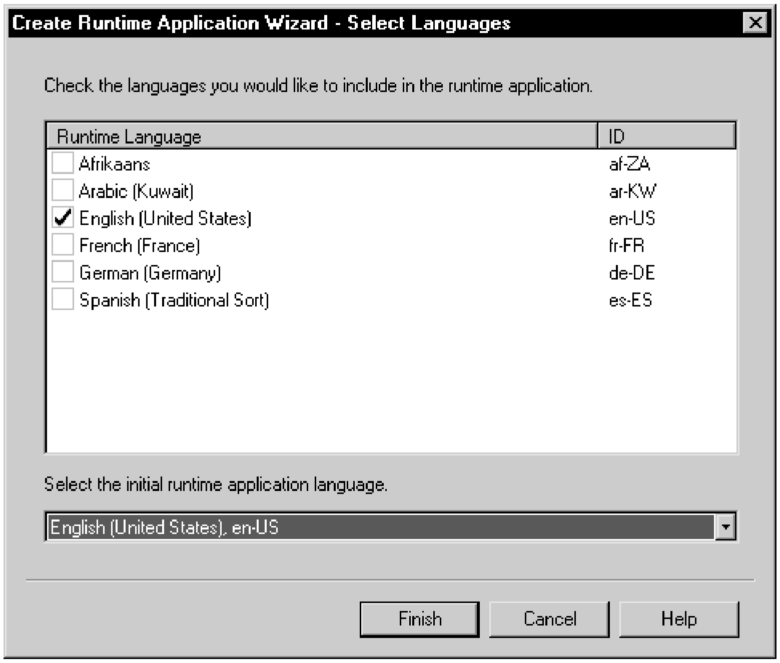
1. In RSView Studio, with the application open, on the Application menu click Create Runtime Application.



File name: Pumphouse.mer

Save as type: Runtime 4.0 Application (*.mer)

2. Specify the folder and file name for the runtime application.
3. In the Save as type box, specify the version of RSView ME Station for which to create the .mer file.
4. Click Save.


Create Runtime Application Wizard - Select Languages

Check the languages you would like to include in the runtime application.

Runtime Language	ID
☐ Afrikaans	af-ZA
☐ Arabic (Kuwait)	ar-KW
✓ English (United States)	en-US
☐ French (France)	fr-FR
☐ German (Germany)	de-DE
☐ Spanish (Traditional Sort)	es-ES

Select the initial runtime application language.

English (United States), en-US

5. If your application uses multiple languages, specify the languages to include and the initial runtime language, and then press Finish. For details, see Help.

   This wizard is not displayed if your application uses one language only.

For information about transferring the runtime application:

- to a Windows® 2000 or Windows XP platform, see Chapter 26.
- to a PanelView Plus™ or VersaView® CE terminal, see Chapter 27.

26 Running applications in Windows 2000 or Windows XP

This chapter describes:

- moving applications to the runtime computer.
- starting RSView® ME Station™.
- loading and running applications.
- shutting down applications.
- editing device short cuts.
- looking up contact information for technical support.
- setting up FactoryTalk® Diagnostics™ at runtime.
- setting up serial ports for KEPServerEnterprise.
- setting up RSLinx® Enterprise™ communication drivers.
- specifying the printers to use at runtime.
- specifying startup options for RSView ME Station.
- deleting log files on the runtime computer.
- specifying the time, date, and number formats to use at runtime.
- using the DeskLock tool.

Summary of steps

Follow these steps to:

- install the necessary hardware and software on the runtime computer.
- transfer your Windows® 2000 or Windows XP application to the runtime computer.
- set up options in RSView ME Station.

For information about installing RSView ME Station, see the *RSView Machine Edition Installation Guide*.

Installing hardware and software on the runtime computer

1. If you will be printing displays, alarms, or diagnostics messages, set up printer connections on the runtime computer.

For more information, see page 26-10.

2. If you are using RSLinx Enterprise, set up communications as described in Chapter 5.
3. If you are using RSLinx® Classic as the OPC® server on the runtime computer, install RSLinx Classic on the runtime computer.
4. If you are using RSLinx Classic on a remote computer, install RSLinx Classic on the remote computer.
5. If you are using an OPC server other than RSLinx Enterprise or RSLinx Classic, install the OPC server software on the runtime computer or on another computer on the network.

   For installation information, see the documentation supplied by your OPC server vendor. For information about OPC, see Chapter 5.
6. If your application uses third-party ActiveX® objects, install and register the Windows 2000 or Windows XP version of the objects on the runtime computer.

   For information about ActiveX objects, see page 15-22.
7. Install on the runtime computer all languages used by the runtime application.
8. If the runtime computer uses different time, date, or number formats than the development computer, specify the time, date, and number formats to use at runtime.

   For more information, see page 26-16.
9. If desired, use the DeskLock tool to prevent users from switching to another software application or using the desktop at runtime.

   For more information, see page 26-16.

Transferring the application

- Move the application to the Windows 2000 or Windows XP runtime computer.

  For more information, see page 26-3.

Setting up options in RSView ME Station

1. On the runtime computer, start RSView ME Station.

   For more information, see page 26-3.
2. Load the application.

   For more information, see page 26-4.
3. Edit device shortcuts, if necessary.

For more information, see page 26-6.

4. Set up FactoryTalk Diagnostics on the runtime computer (if you have not already done so).

   For more information, see page 10-6.

5. If you will be using KEPServerEnterprise™, specify serial port IDs.

   For more information, see page 26-8.

6. If you will be using RSLinx Enterprise, set up communication drivers (if you have not already done so).

   For more information, see page 26-9.

7. Specify the printers to use.

   For more information, see page 26-10.

8. Specify startup options for RSView ME Station.

   For more information, see page 26-11.

Once you've completed these steps, you're ready to run the application. For information about running your application, see Chapter 28.

Moving applications to the runtime computer

The runtime application file has the extension .mer. You can use any standard file transfer method to copy your runtime application from the development computer to the runtime computer.

You can:

- copy the application file from the development computer to a floppy disk, and then from the floppy disk to the runtime computer.
- if the application file is too large to fit on a floppy disk, use a larger storage device such as a Zip® disk.
- if the development and runtime computers are on the same network, use Windows Explorer or My Computer to move the file.

For information about creating the runtime application file, see Chapter 25.

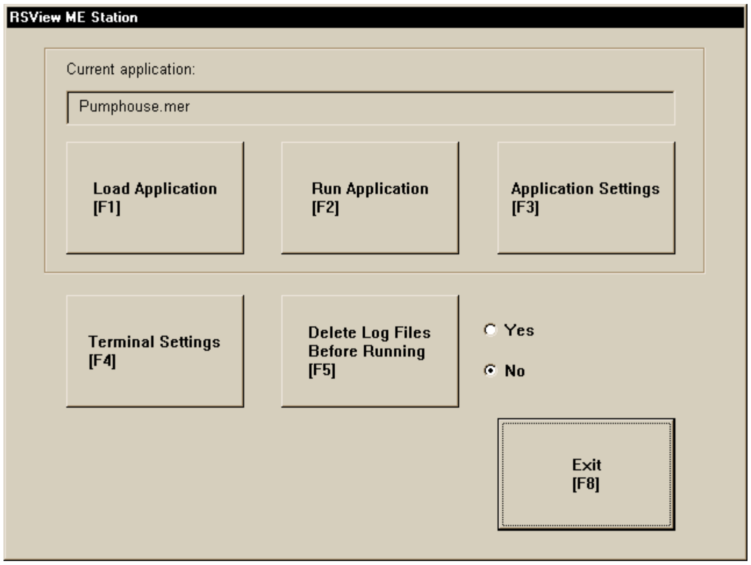
Starting RSView ME Station

If you are running an application on the development computer, we recommend that you exit RSView Studio before starting RSView ME Station.

To start RSView ME Station

1. On the Windows Start menu, select Programs, Rockwell Software, RSView Enterprise, and then click RSView ME Station.

   The RSView ME Station dialog box opens.



Current application: Pumphouse.mer

Yes [✓ No]

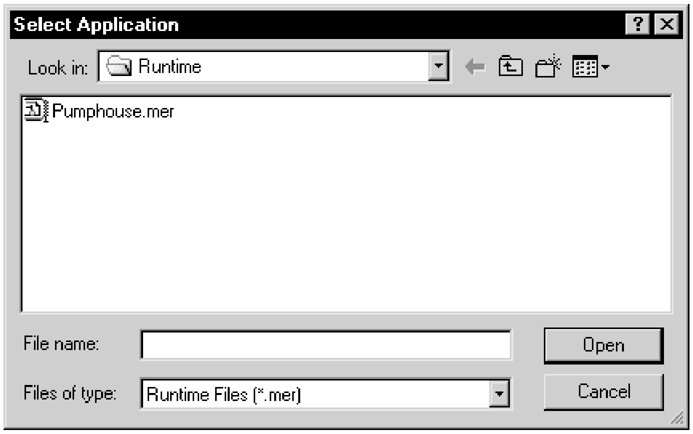
For information about specifying startup options for when RSView ME Station starts, see page 26-11.

Loading and running applications

You can run any runtime application that is on the runtime computer. Runtime applications have the extension .mer.

To load and run the application

1. In the RSView ME Station dialog box, click Load Application, or press F1.



2. Navigate to the folder containing the application's .mer file, and then click the file name.
3. Click Open.
4. To replace the runtime computer's communication settings with the application's communication settings, click Yes. To keep the runtime computer's communication settings, click No.

   You are notified that the application's FactoryTalk System Directory of users and security policies will be loaded on this computer. This is the set of users and policies that have been set up for the application and are contained in the .mer file.

   The computer's existing FactoryTalk System Directory will be archived, and will be restored when you stop the application. To turn off this warning, see page 26-15.
5. To continue, click Yes. To stop loading the application, click No.

   If you continue, the application name is displayed in the Current application box in the RSView ME Station dialog box.
6. To run the application, click Run Application.

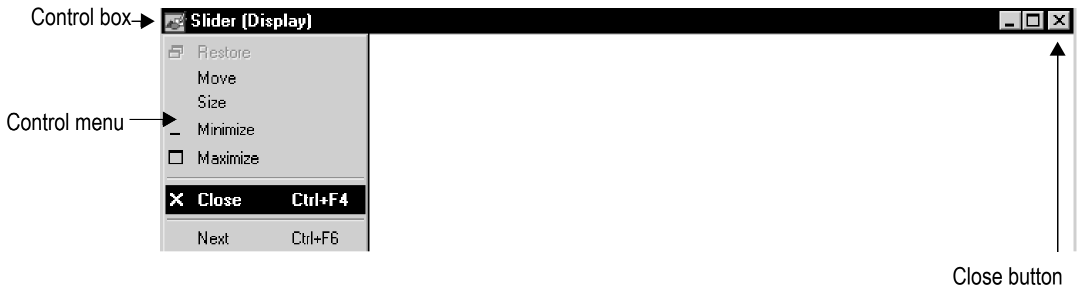
   The DEFAULT user is logged in. If a macro is assigned to the DEFAULT user, the macro runs.

Shutting down applications

To shut down an application, use one of these methods

- Press a shutdown button in a graphic display.

- If the application is set up to use a title bar with a Control box, click the Close button at the right end of the title bar.
- If the application is set up to use a title bar with a Control box, on the Control menu at the left end of the title bar, click Close.



For information about using a title bar in graphic displays, see page 4-14. For information about preventing unauthorized users from shutting down applications, see page 13-15.

What happens when the application shuts down

When the application shuts down:

- if you assigned a shutdown macro (in the Startup editor), the macro runs, assigning values to tags, and then the application stops.

  For information about the Startup editor, see Chapter 25.
- RSView ME Station closes.
- The computer's FactoryTalk System Directory is restored.

Changing application settings

Editing device shortcuts

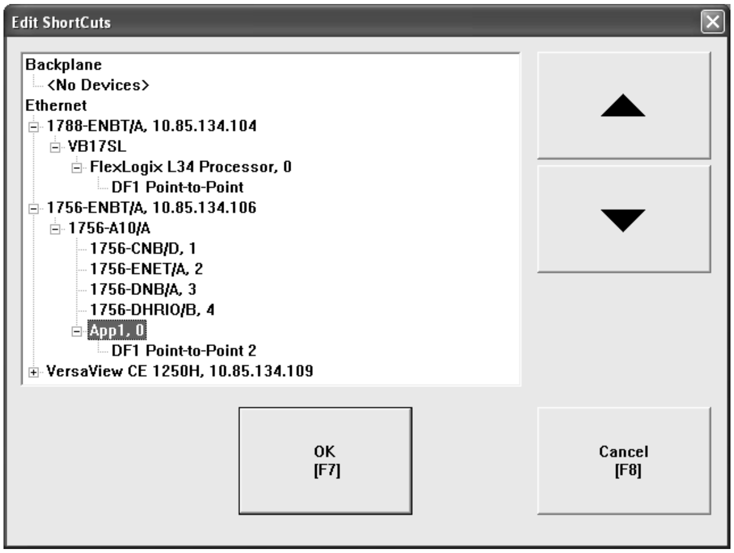
You can use RSView ME Station to edit device shortcuts that have been set up in the application.

Before editing device shortcuts, load the application containing the device shortcut, as described on page 26-4.

To edit device shortcuts

1. In the RSView ME Station dialog box, click Application Settings.
2. Double-click Device Shortcuts.

3. Double-click the name of the shortcut to edit.



```
Backplane
  <No Devices>
Ethernet
  1788-ENBT/A, 10.85.134.104
    VB17SL
      FlexLogix L34 Processor, 0
        DF1 Point-to-Point
  1756-ENBT/A, 10.85.134.106
    1756-A10/A
      1756-CNB/D, 1
      1756-ENET/A, 2
      1756-DNB/A, 3
      1756-DHRIO/B, 4
      App1, 0
        DF1 Point-to-Point 2
  VersaView CE 1250H, 10.85.134.109
```

4. In the Edit ShortCuts dialog box, click the device you want the shortcut to point to, and then click OK.

Looking up contact information for technical support

To look up technical support contact information

1. In RSView ME Station, click Terminal Settings.
2. Double-click About RSView ME Station.
3. Click Technical Support.

   The telephone number, fax number, and URL for technical support are displayed.

Setting up FactoryTalk Diagnostics on the runtime computer

You can set up FactoryTalk Diagnostics on the runtime computer using the RSView ME Station dialog box.

To set up FactoryTalk Diagnostics on the runtime computer

1. In RSView ME Station, click Terminal Settings.
2. Double-click Diagnostics Setup.

   The FactoryTalk Diagnostics Setup dialog box opens.

3. Set up FactoryTalk Diagnostics, as described on page 10-6.

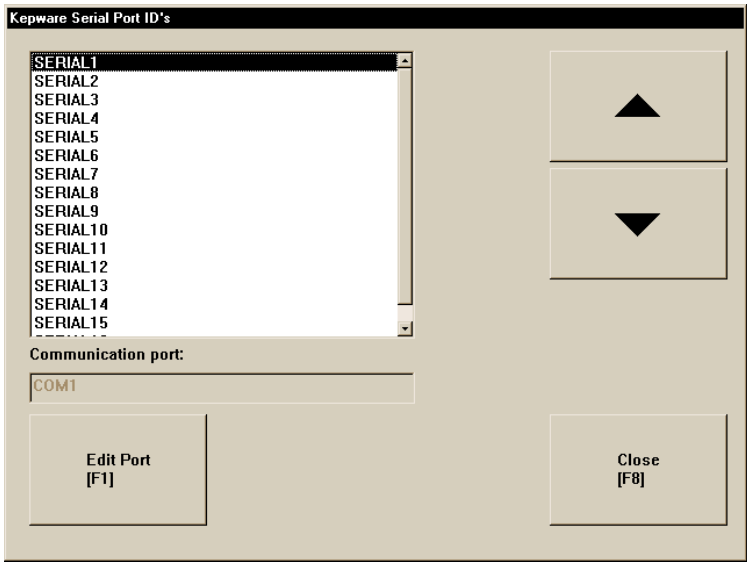
Setting up serial ports for use with KEPServerEnterprise

If you plan to use KEPServerEnterprise and serial communications, you must specify which COM port to use.

For information about setting up communications in KEPServerEnterprise, see KEPServerEnterprise Help.

To specify the COM port to use for serial communications

1. In the RSView ME Station dialog box, click Terminal Settings.
2. Double-click Networks and Communications.
3. Double-click KEPServer Serial Port ID's.



Communication port: COM1

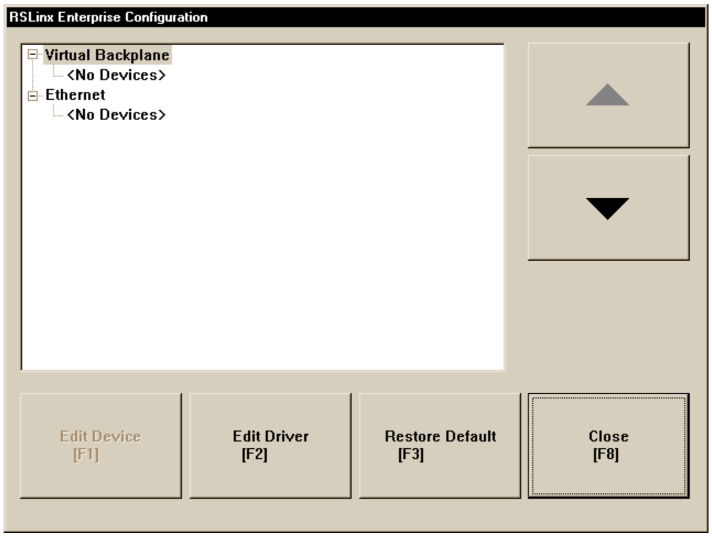
4. In the Kepware Serial Port ID's dialog box, click the serial port ID you specified when you set up the KEPServerEnterprise channel.
5. Click Edit Port.
6. In the Communication Ports dialog box, click the COM port to use for KEPServerEnterprise communications.

Setting up RSLinx Enterprise communication drivers

Use RSLinx Enterprise to set up communication drivers for your runtime application. You can set up the drivers directly in RSLinx Enterprise, or open RSLinx Enterprise by using the RSView ME Station dialog box.

To set up the RSLinx Enterprise communication driver to use at runtime

1. In the RSView ME Station dialog box, click Terminal Settings.
2. Double-click Networks and Communications.
3. Double-click RSLinx Enterprise Communications.



4. To edit a driver, select it and then click Edit Driver.
5. To edit a device, select it and then click Edit Device.

For information about setting up RSLinx Enterprise drivers and devices, see the RSLinx documentation.

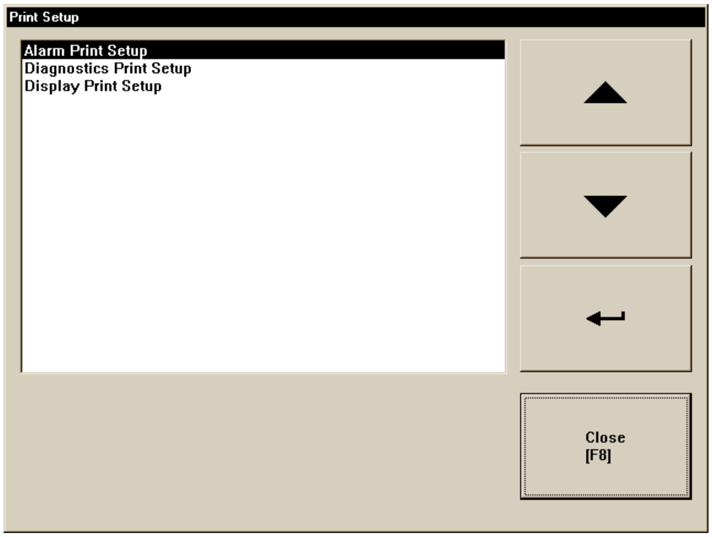
Once the driver is set up, RSView ME Station automatically starts the driver software when you run the application.

Specifying the printers to use at runtime

You can use local or network printers to print alarm messages, reports, diagnostics messages, and graphic displays at runtime. If desired, you can use a different printer for each type of printing.

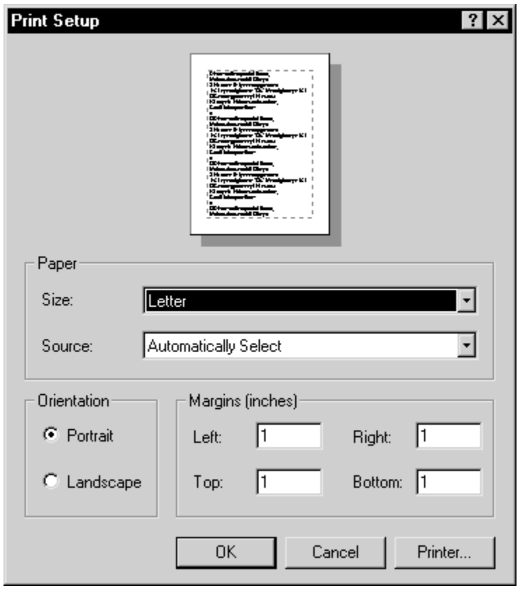
To specify the printers to use at runtime

1. In the RSView ME Station dialog box, click Terminal Settings.
2. Double-click Print Setup.



Alarm Print Setup
Diagnostics Print Setup
Display Print Setup

3. Double-click the type of printing to set up.



Paper

Size: Letter

Source: Automatically Select

Orientation

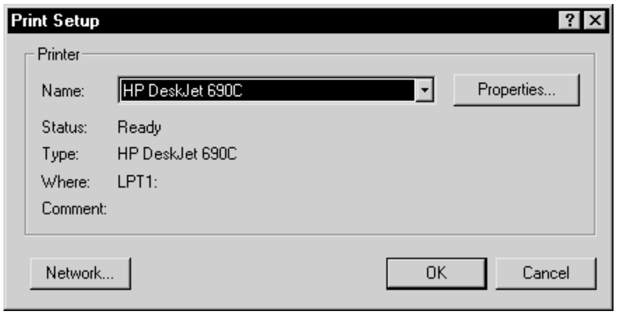
[✓ Portrait] Landscape

Margins (inches)

Left: 1 Right: 1

Top: 1 Bottom: 1

4. Click Printer.



Printer

Name: HP DeskJet 690C

Status: Ready

Type: HP DeskJet 690C

Where: LPT1:

Comment:

5. Modify the print options as required.

For detailed information about printer options, refer to your Windows documentation.

Specifying startup options for RSView ME Station

This section describes how to start RSView ME Station automatically when Windows starts, and describes startup options for RSView ME Station.

When you start RSView ME Station, you can:

- run an application.

- load an application.
- automatically delete an application's log files before running the application.
- replace RSLinx Enterprise communications on the runtime computer with the application's settings.

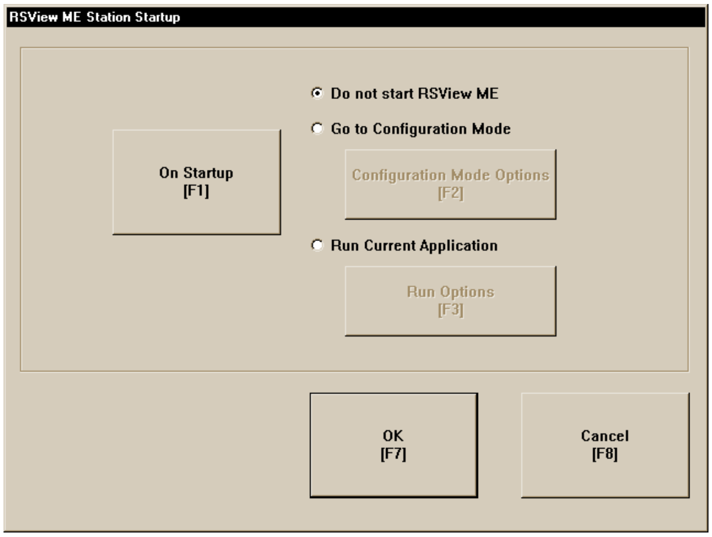
By default, the option to start RSView ME Station when Windows starts is turned off. The settings in this section apply only if you want RSView ME Station to start automatically when Windows starts.

To start RSView ME Station when Windows starts and run an application

1. In the RSView ME Station dialog box, load the application that you want to run.

   For details, see page 26-4.

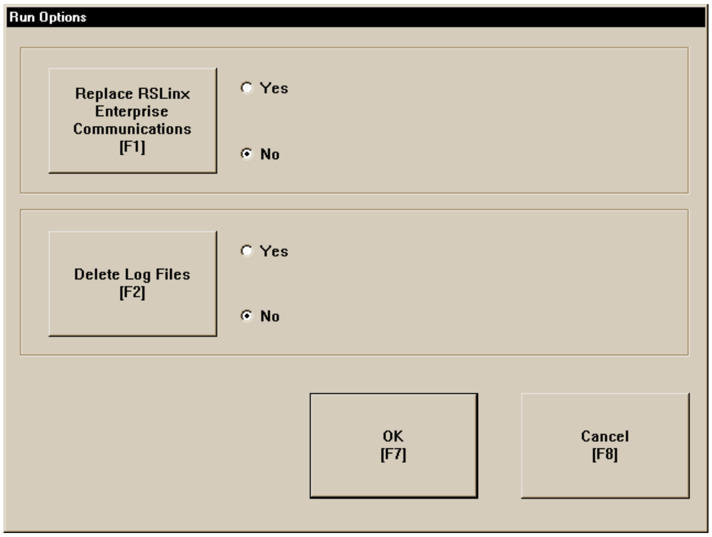
2. Click Terminal Settings.
3. Double-click RSView ME Station Startup.



4. Click Run Current Application.

   This option is not available if you have not loaded an application yet.

5. Click Run Options.



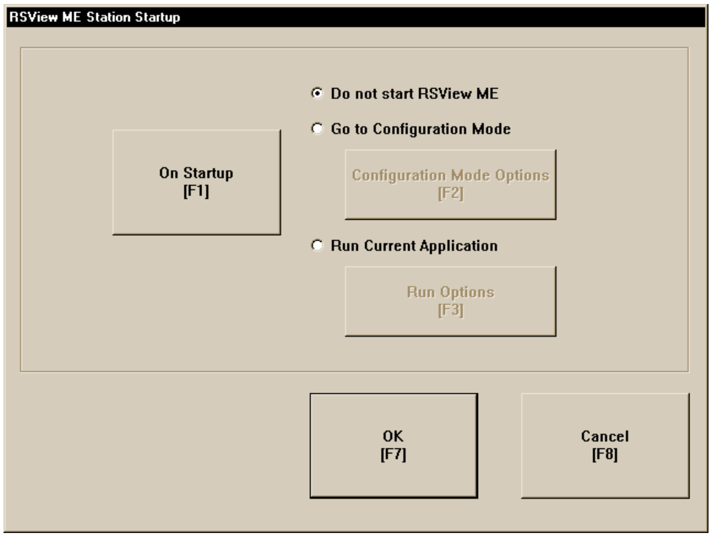
6. Specify whether to replace the RSLinx Enterprise communication settings on the runtime computer with the application's settings when the application starts.

7. Specify whether to delete the application's log files on startup.

To start RSView ME Station when Windows starts without running an application

1. In the RSView ME Station dialog box, click Terminal Settings.

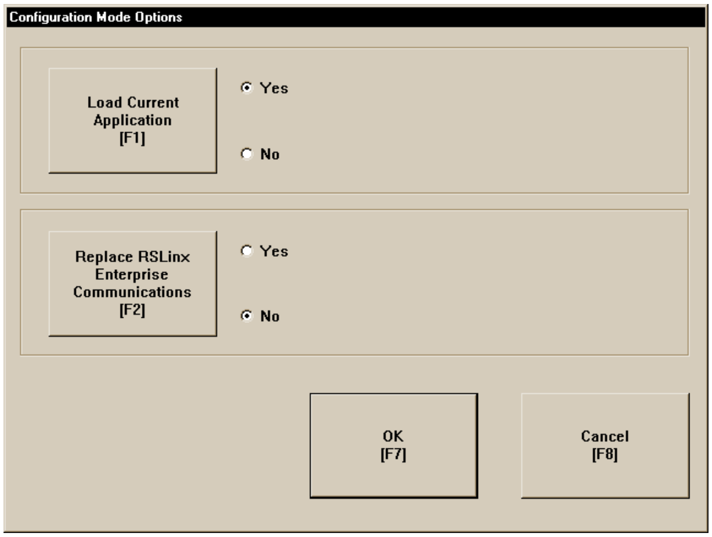
2. Double-click RSView ME Station Startup.



[✓ Do not start RSView ME] Go to Configuration Mode Run Current Application

3. Click Go to Configuration Mode.

4. Click Configuration Mode Options.


Configuration Mode Options
Load Current Application [F1]
Replace RSLinx Enterprise Communications [F2]
OK [F7]
Cancel [F8]

Load Current Application: [✓ Yes] No

Replace RSLinx Enterprise Communications: Yes [✓ No]

5. Specify whether to load the current application when RSView ME Station starts.

   This option is not available if you have not loaded an application.

6. Specify whether to replace the RSLinx Enterprise communication settings on the runtime computer with the application's settings when the application starts.

   This option is not available if you have not loaded an application.

Deleting log files on the runtime computer

When you run your application, RSView ME Station stores log files for alarms and data logging (if you use these features). When you start RSView ME Station, you can delete the alarm and data log files for the loaded application, or for all the applications on the runtime computer.

Running a newer version of the application

If you run a newer version of an application, the alarm log file for the older version is deleted automatically. The data log file for the older version is retained, to allow the display of historical data in trends.

For more information about the alarm log file, see page 9-9. For more information about data log files, see page 11-5.

Deleting log files manually

To delete log files for the loaded application

1. In the RSView ME Station dialog box, click Yes beside the Delete Log Files Before Running button.

All alarm and data log files for the loaded application are deleted.

To delete log files for all applications on the runtime computer

1. In the RSView ME Station dialog box, click Terminal Settings.
2. Double-click Delete Log Files.
3. Click Yes.

Log files for data log models that use the default path are deleted. All alarm log files are deleted.

Turning off the FactoryTalk Directory Server warning

When you load an application, you are notified that the application's FactoryTalk System Directory of users and security policies will be loaded on the computer. The computer's existing FactoryTalk System Directory is archived while you run the application. It is restored when you stop the application.

You can turn off this notification warning.

To turn off the overwrite warning

1. In the RSView ME Station dialog box, click Terminal Settings.
2. Double-click System Directory Overwrite Warning.
3. Click No.

Specifying time, date, and number formats

Time, date, and number formats are used by these graphic objects:

- numeric display
- gauge
- time and date display
- trend
- alarm banner
- alarm list

The time and date formats are also used when printing the alarm history report (using the print alarm history button). The numeric and time and date embedded variables also use number, time, and date formats.

All objects use the time, date, and number formats of the current application language. For example, if the application language uses a comma for the decimal symbol, numeric variables use a comma for the decimal symbol.

For information about using multiple languages, see Chapter 23.

Using the DeskLock tool

Use the DeskLock tool to prevent users from switching to another software application or having access to the desktop at runtime.

This tool can have far-reaching effects on your operating system. The DeskLock tool replaces the standard Windows 2000 or Windows XP non-configured desktop with a customized desktop. The customized desktop is intended to prevent operators from having access to other applications and operating system functions such as restarting Windows or shutting down tasks.

Before using the tool, read the DeskLock Help file completely.

To open the DeskLock tool

1. On the Windows Start menu, select Programs, Rockwell Software, RSView Enterprise, Tools, and then click DeskLock.

For details about using the tool, see the tool's Help.

27 Transferring applications to a PanelView Plus or VersaView CE terminal

This chapter describes:

- starting RSView® ME Station™.
- setting up communication drivers to use to transfer applications to a PanelView Plus™ or VersaView® CE terminal.
- transferring applications and TrueType™ fonts to PanelView Plus or VersaView CE terminals.
- uploading applications from a terminal to the development computer.
- comparing applications on the development computer with applications on the terminal.

For information about using your application at runtime, see Chapter 28.

For information about using RSView ME Station on the terminal, including setting up communications, printers, and input devices, see the *PanelView Plus Terminals User Manual* or the *VersaView CE Terminals User Manual*. These manuals are available on the RSView Machine Edition™ CD.

Summary of steps

Follow these steps to:

- install hardware and software on a PanelView Plus or VersaView CE terminal.
- transfer your application to the PanelView Plus or VersaView CE terminal.

For information about installing RSView ME Station, see the *RSView Machine Edition Installation Guide*.

Installing hardware and software on a VersaView CE terminal

1. If you will be printing displays, alarms, reports, or diagnostics messages, connect a printer to the USB or Network port.

   VersaView CE terminals support printing using the Microsoft® Windows® CE PCL3 printer driver, which is already installed on the terminal. Printing is supported for most laser and ink-jet printers. If you have problems printing, check that your printer is compatible with the PCL3 printer driver.

2. If you are using an OPC® server other than RSLinx Enterprise, for example, KEPServerEnterprise™, install the OPC server software on the terminal.

   For installation information, see the documentation supplied by your OPC server vendor. For information about OPC data servers, see Chapter 5.

3. If your application uses third-party ActiveX® objects, install and register the Windows CE version of the objects on the terminal.

   For information about ActiveX objects, see page 15-22.

Installing printers on a PanelView Plus terminal

1. If you will be printing displays, alarms, reports, or diagnostics messages, connect a printer to the USB or Network port.

   PanelView Plus terminals support printing using the Microsoft Windows CE PCL3 printer driver, which is already installed on the terminal. Printing is supported for most laser and ink-jet printers. If you have problems printing, check that your printer is compatible with the PCL3 printer driver.

The PanelView Plus terminal is a "closed box," which means you must order any software your application will need when you order the system. All software comes pre-installed.

Transferring applications

1. On the PanelView Plus or VersaView CE terminal, start RSView ME Station.

   For more information, see page 27-3.

2. In RSView ME Station, if you're transferring via a serial connection, specify and set up the serial driver. If you are using Ethernet® for the transfer, you can skip this step.

   For more information, see page 27-3.

3. On the development computer, set up the RSLinx Enterprise driver to use for the transfer.

   For more information, see page 27-5.

4. Move the application to the PanelView Plus or VersaView CE terminal using the Transfer Utility tool.

   For more information, see page 27-5.

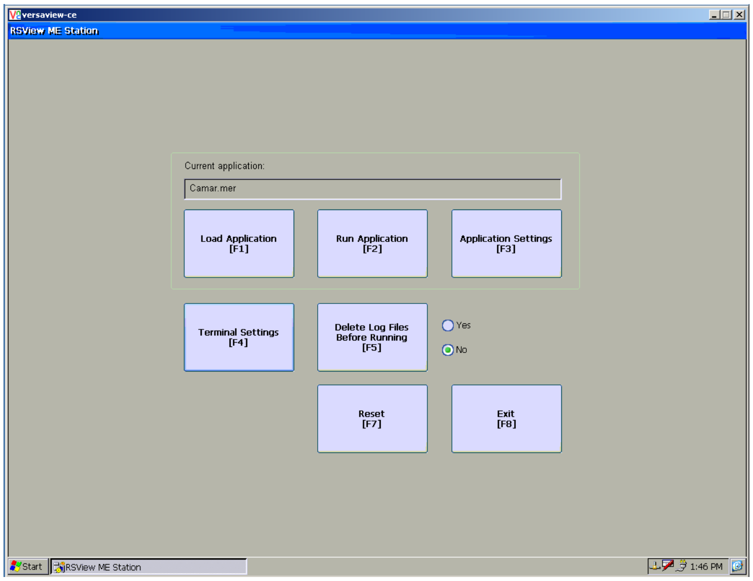
You can also transfer applications to the terminal using a Flash Card. For information about this method, see the *PanelView Plus Terminals User Manual* or the *VersaView CE Terminals User Manual*.

Starting RSView ME Station

To start RSView ME Station on a VersaView CE terminal

1. On the Windows Start menu, select Programs, Rockwell Software, and then click RSView ME Station.

   The RSView ME Station dialog box opens.



Starting RSView ME Station on a PanelView Plus terminal

If you are running a PanelView Plus terminal, the RSView ME Station dialog box opens automatically when the terminal starts up.

Specifying the driver to use for the transfer

You can download the runtime application file to the PanelView Plus or VersaView CE terminal using:

- Ethernet
- a direct serial connection

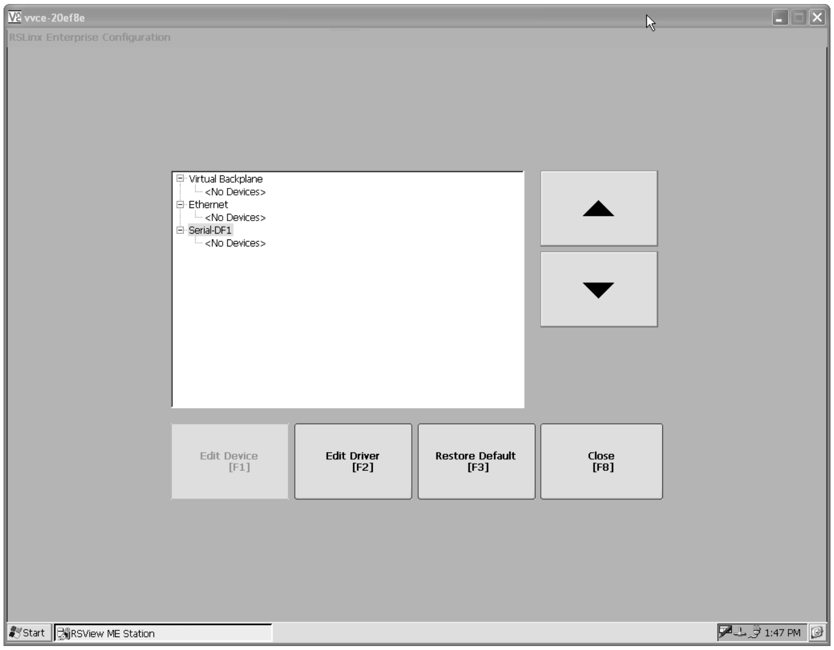
If you are using an Ethernet connection for the transfer, you don’t need to set up a driver for the transfer, since the Ethernet driver is selected and loaded by default.

If you are using a serial connection for the transfer, specify and set up the driver.

You can also transfer applications to the terminal using a Flash Card. For information about this method, see the *PanelView Plus Terminals User Manual* or the *VersaView CE Terminals User Manual*.

To specify and set up a serial driver for the transfer

1. In the RSView ME Station dialog box, click Terminal Settings.
2. Double-click Networks and Communications.
3. In the Networks and Communications dialog box, double-click RSLinx Enterprise Communications.



4. Click the Serial-DF1 driver, and then click Edit Driver.
5. Select Use Auto-Configuration, and then click Edit.
6. In the dialog box that opens, select Yes, and then click OK.

   Auto configuration works with most devices, including ControlLogix® and PLC-5®. If auto configuration is not successful for your device, return to these steps, select No, and continue to step 7.
7. Set up the driver, and then click OK. If you are using auto configuration, skip this step.

Setting up a driver for the transfer on the development computer

On the development computer, set up the RSLinx Enterprise driver that you will use to download your application to the PanelView Plus or VersaView CE terminal.

Use one of these drivers for the transfer:

- Serial-DF1—Use for a serial transfer.
- Ethernet—Use with an Ethernet connection.

For information about setting up drivers in RSLinx Enterprise, see RSLinx Help.

Downloading applications and Windows TrueType fonts

Use the Transfer Utility to download your runtime application and Windows TrueType font files from the development computer to the PanelView Plus or VersaView CE terminal.

Runtime applications have the extension .mer. Windows TrueType font files have the extension .ttf or .ttc.

For information about creating the runtime application file, see Chapter 25.

About the download

You can use a serial or Ethernet connection between the development computer and the PanelView Plus or VersaView CE terminal destination. You must use one of the drivers listed in the previous section for the download.

You can download while an application is running on the runtime computer.

You can download:

- multiple applications (one at a time) to the same runtime computer.
- a new copy of the application currently running on the runtime computer, for use the next time the application is started.

If the download process fails or is cancelled, the application file will be deleted from the destination directories. Make a copy of the application file before you begin and make sure there is enough storage space on the destination computer.

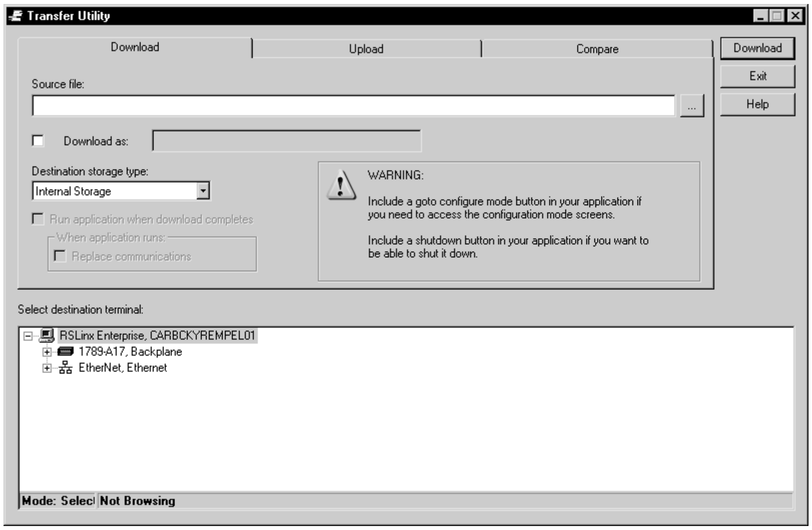
Serial downloads

To perform a serial download, make sure the terminal is connected to the development computer using the correct cable. Connect a PanelView Plus or VersaView CE terminal to the PC using the Allen-Bradley® serial cable 2711-NC13.

To start the Transfer Utility tool

1. On the development computer, do one of the following:

   - In RSView Studio™, on the Tools menu, click Transfer Utility.
   - On the Windows Start menu, select Programs, Rockwell Software, RSView Enterprise, Tools, and then click ME Transfer Utility.



WARNING:

Include a goto configure mode button in your application if you need to access the configuration mode screens.

Include a shutdown button in your application if you want to be able to shut it down.

For details about using the utility, see the utility's Help.

Uploading applications from the PanelView Plus or VersaView CE terminal

Use the Transfer Utility to upload your runtime application from the PanelView Plus or VersaView CE terminal to the development computer. Runtime applications have the extension .mer.

Any password changes that were made while the application was running are saved in the .mer file and will be uploaded.

If the upload process fails or is cancelled, the application file will be deleted from the destination directory. Make sure there is enough storage space on the destination computer.

About the upload

You can upload while an application is running on the runtime computer. You can upload an existing copy of the application currently running on the runtime computer.

Serial uploads

To perform a serial upload, make sure the terminal is connected to the development computer using the correct cable. Connect a PanelView Plus or VersaView CE terminal to the PC using the Allen-Bradley serial cable 2711-NC13.

For details about performing the upload, see Help for the Transfer Utility.

Comparing applications

You can also use the Transfer Utility tool to compare an application on the development computer with an application on the terminal. The comparison tool can tell you whether the files are identical or different.

For details about comparing applications, see Help for the Transfer Utility.

28 Using your application

This chapter describes:

- logging in and logging out.
- changing passwords.
- entering numeric and string values.
- changing tag values.
- viewing tag data.
- viewing alarms and messages.
- viewing information about runtime communication errors.

For information about navigating between displays, see Chapter 24. For information about navigating between and using the graphic objects in a display, see page 16-5.

Logging in to a Windows domain

To log data to a network location, the runtime computer must be logged in to the same domain as the computer on the network. To do this, the user must have access rights for the domain.

To log in to a domain from Windows 2000 or Windows XP

1. When Windows® starts, type a user name, password, and domain name.

The PanelView Plus™ or VersaView® CE terminals cannot be part of a domain. However, you can verify that the user who is logged in to the PanelView Plus or VersaView CE terminal is on a user list that is part of a domain.

On the PanelView Plus or VersaView CE terminal, logging in hard codes a domain member's credentials. This gives the terminal access to permissible network resources such as network folders.

To authenticate a user on a PanelView Plus or VersaView CE terminal

1. In the RSView® ME Station™ dialog box, click Terminal Settings.
2. Double-click Networks and Communications.
3. Double-click Network Connections.
4. Double-click Network Identification.

5. Type a user name, password, and domain name, and then click OK.

Logging in to the application

Users log in using the Login window. They must have a user account in order to log in.

For information about setting up user accounts and passwords, see Chapter 13.

4.0 applications

To log in, the user must have an account in the User Accounts editor, in the RSView 4.0 Accounts tab. The user must have a password, which is set up in RSAssetSecurity, using the Users folder. For details, see Chapter 13.

Passwords are case sensitive.

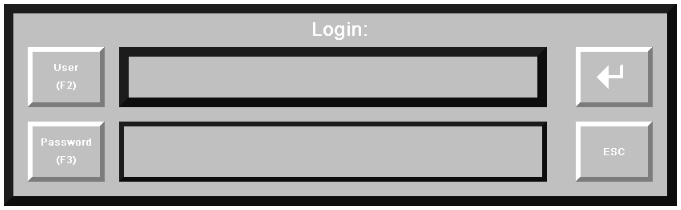
3.x applications

To log in, the user must have an account in the User Accounts editor, in the RSView 3.x Accounts tab. User passwords are optional in RSView 3.x. If you use them, they are not case sensitive.

To log in

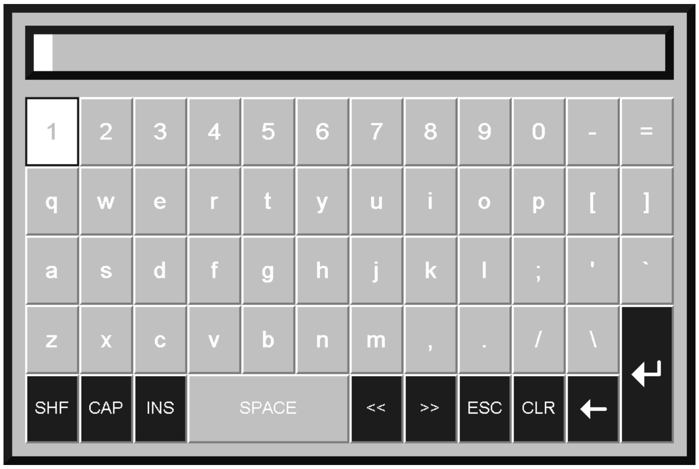
1. Press the login button.

   The Login window opens.



2. Press the User button, or press F2 on an external keyboard.

   The string pop-up keyboard opens. For details about using the keyboard, see page 28-11.



3. Type your user name in the string pop-up keyboard, or on an external keyboard, and then press Enter.

   For RSView 4.0 applications, the name is not case sensitive. For RSView 3.x applications, the name must use ALL CAPS.

4. To enter a password, press the Password button, or press F3 on an external keyboard.
5. Type the password in the string pop-up keyboard, or on an external keyboard, and then press Enter.

   For RSView 4.0 applications, the password is case sensitive. For RSView 3.x applications, the password is not case sensitive.

6. To close the Login window and complete the login, press Enter.

What happens when a user logs in

When a user logs in:

- If the previous user did not log out, the previous user is logged out now. If a logout macro is set up for the previous user, the logout macro runs, assigning values to tags. If the previous user belongs to a group, and a logout macro is set up for the group, the logout macro runs.
- The new user is logged in.
- If a login macro is set up for the new user, the macro runs, assigning values to tags. If the new user belongs to a group, and a login macro is set up for the group, the login macro runs.

Problems with logging in

Login is unsuccessful under these circumstances:

- If the graphic display changes remotely before the user has finished logging in, the login is cancelled.
- If the user name doesn't match the settings in the User Accounts editor, the user is not logged in.
- If the user password doesn't match the password set up in RSAssetSecurity, in the Users folder (4.0 applications), or the password set up in the User Accounts editor (for 3.x applications), the user is not logged in.
- If the screen resolution is smaller than 280 pixels wide by 84 pixels high, the Login window cannot open.
- If the screen resolution is smaller than 236 pixels wide by 208 pixels high, the string pop-up keyboard cannot open.

Changing passwords

You can use the password button to change your password at runtime. Any password changes that are made at runtime are saved in the .mer file.

When passwords are changed at runtime, be sure to change the passwords on the development computer as well. Otherwise, the next time you create the .mer file the password changes will be lost.

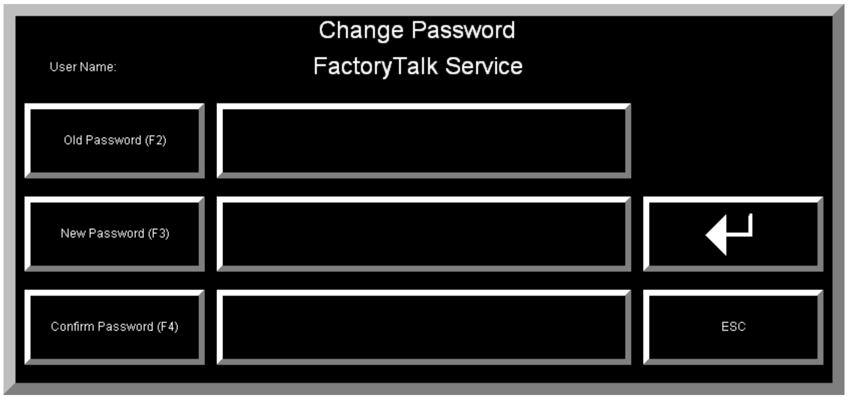
You cannot use the password button to change passwords for Windows-linked users. The passwords for these users must be changed in Windows.

For 3.x applications, you cannot change passwords in RSView ME Station. To change a password for an RSView user, use the User Accounts editor in RSView Studio™, then recreate the runtime application and download the .mer file again. To change a password for a Windows user, use the Windows Control Panel.

To change your password

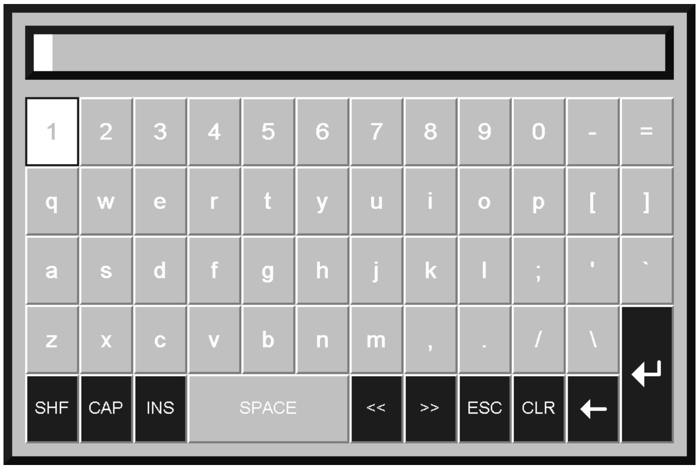
1. Press the password button.

   The Change Password window opens.



2. Press the Old Password button, or press F2 on an external keyboard.

   The string pop-up keyboard opens. For details about using the keyboard, see page 28-11.



3. Type your old password in the string pop-up keyboard, or on an external keyboard, and then press Enter.

   The password is case sensitive.

4. Press the New Password button, or press F3 on an external keyboard.
5. Type the new password in the string pop-up keyboard, or on an external keyboard, and then press Enter.
6. Press the Confirm Password button, or press F4 on an external keyboard.

7. Type the new password again, and then press Enter.
8. To close the Change Password window and complete the change, press Enter.

Logging out

When the current user logs out, if a logout macro is assigned to the user, the macro runs. If the user belongs to a group, and a logout macro is set up for the group, the logout macro runs.

After the current user is logged out, the DEFAULT user is logged in. If a login macro is assigned to the DEFAULT user, the macro runs.

You can also set up the application to automatically log out the current user after a specified period of inactivity. For more information, see page 13-14.

To log out

1. Press the logout button.

Entering numeric values

The operator can enter or ramp numeric values at runtime using the numeric input enable button or the numeric input cursor point.

When the operator presses the button or activates the cursor point, the numeric pop-up keypad or scratchpad opens. If you set up the button or cursor point to ramp, pressing the object gives it focus, but does not open a pop-up window.

To use the numeric pop-up scratchpad, a keyboard must be attached to the runtime computer, or the terminal must be a keypad terminal.

For information about setting up the numeric input enable button and numeric input cursor point, see Help.

You can also use a ramp button to ramp numeric values. For details, see Help for the button.

Activating the cursor point

When the numeric input cursor point has focus, the operator can activate the cursor point by pressing any of these keys on the keyboard or keypad:

- numbers from 0 to 9
- the minus sign (-) or decimal point (.)
- the Enter key, or an enter button
- the Backspace key, or a backspace button

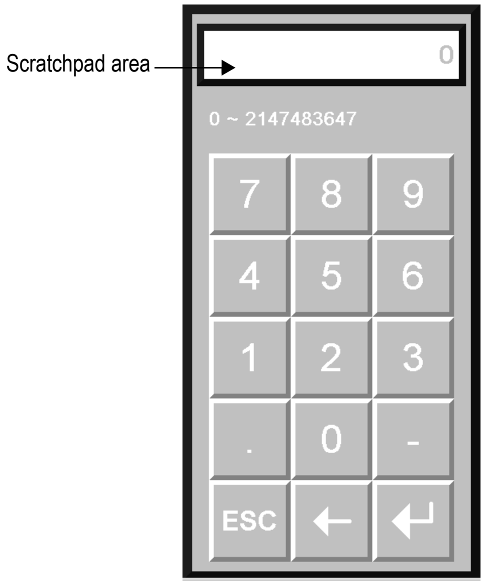
When the operator activates the numeric input cursor point, the numeric pop-up keypad or scratchpad opens.

Ramping numeric values

If you set up the numeric input enable button or numeric input cursor point to ramp values, the operator presses the object to give it focus. When the object has focus, the operator can press a move up or move down button, or the Up Arrow or Down Arrow key on a keyboard or keypad, to ramp the value at the Value connection.

Using the numeric pop-up keypad

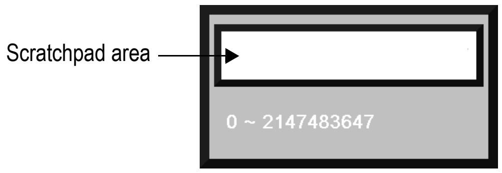
The keypad can accept up to 17 digits, including the decimal point, leading zero, and minus sign.



When the keypad is open, no other objects in the graphic display can accept input.

Using the numeric pop-up scratchpad

The scratchpad can accept up to 17 digits, including the decimal point, leading zero, and minus sign.



The scratchpad does not contain any buttons. If the runtime computer does not have an external keyboard or keypad attached, the operator will be unable to enter values or close the scratchpad.

When the scratchpad is open, no other objects in the graphic display can accept input.

Using buttons and keys with the numeric pop-up windows

You can use the following buttons and their keyboard or keypad equivalents with the numeric pop-up keypad. For the pop-up scratchpad, you can use the keys on an external keyboard or keypad only.

Use this button or key	To do this
Decimal (.)	Type a decimal point. This button is visible only if the decimal point type for the numeric input enable button or numeric input cursor point is Implicit.
Minus (-)	Toggle the value between positive and negative.
ESC	Close the pop-up window without writing the value to the object's Value connection.
Backspace	Delete the right-most digit in the scratchpad. If no digits are left, the minus sign, if any, is removed.
Enter	Close the pop-up window and write the value to the object's Value connection. If the Optional Expression connection is assigned to the button or cursor point, RSView calculates the value of the expression and writes the result to the Value connection. For details, see page 20-17. If Enter key handshaking is set up for the button or cursor point, the object's Enter connection is set to 1 and the handshaking timer begins timing. For details, see page 16-14.
Delete	Clear the scratchpad. There is no Delete button in the pop-up keypad, but you can use the Delete key on an external keyboard or keypad.

How values are ramped

If the button or cursor point is being used to ramp a value at the Value connection:

- When the tag assigned to the Value connection is an integer tag, but the ramp value is a floating point value, the ramp value is added to (or subtracted from) the Value connection value, and then the result is rounded and written to the Value connection.

  For information about how values are rounded, see page 7-2.

- If the operator presses and holds down the move up or move down button, or the Up Arrow or Down Arrow key on the keyboard or keypad, the button or key goes into auto repeat mode. The ramp value is added to (or subtracted from) the last value sent to the Value connection.

  For more information about auto repeat mode, see page 16-14.

- If ramping the button or cursor point would cause the value at the Value connection to exceed the maximum value, or be less than the minimum value, a message is written to the Diagnostics List and the value at the Value connection is not changed.

How values are calculated

When the pop-up keypad or scratchpad has focus and the operator presses the Enter button, the value that is sent to the Value connection depends on how the button or cursor point is set up.

- The value the operator enters is compared to the minimum and maximum range for the object. If the value is within the range, the value is written to the Value connection.

  If the Optional Expression connection is assigned, and the original value is within the range but the expression result is a value outside the range, the value is still written to the Value connection.

- If the decimal point type is Fixed Position, with a “Digits after decimal” value that is greater than 0, the decimal point is stripped from the value before comparing it to the minimum and maximum range.

  For example, if the entered value is 9.25, the stripped value is 925.

  If the stripped value is within the range, the stripped value is sent to the Value connection (or substituted for the ? in the Optional Expression).

- If the decimal point type is Implicit, and the tag assigned to the Value connection uses an integer data type, floating-point values are rounded.

  If the Optional Expression connection is assigned, the floating-point value is substituted for the ? in the expression, and the expression result is rounded.

  For information about how values are rounded, see page 7-2.

Problems with the numeric pop-up windows

Problems with the numeric pop-up keypad and scratchpad occur under these circumstances:

- If the graphic display changes remotely before the user has pressed the Enter button, the pop-up window closes without writing out a value.
- If the screen resolution is smaller than 124 pixels wide by 240 pixels high, the pop-up keypad cannot open.
- If the screen resolution is smaller than 124 pixels wide by 68 pixels high, the pop-up scratchpad cannot open.
- If the value is too large for the data type of the tag assigned to the Value connection, the pop-up window remains open and the value is not written to the connection. The scratchpad area changes to red to alert the operator of the error.
- If the value is outside the minimum and maximum range specified for the object, the pop-up window remains open and the value is not written to the connection. The scratchpad area changes to red to alert the operator of the error.
- If Enter key handshaking is still in effect, the pop-up window closes but the value is not written to the connection.

Entering string values

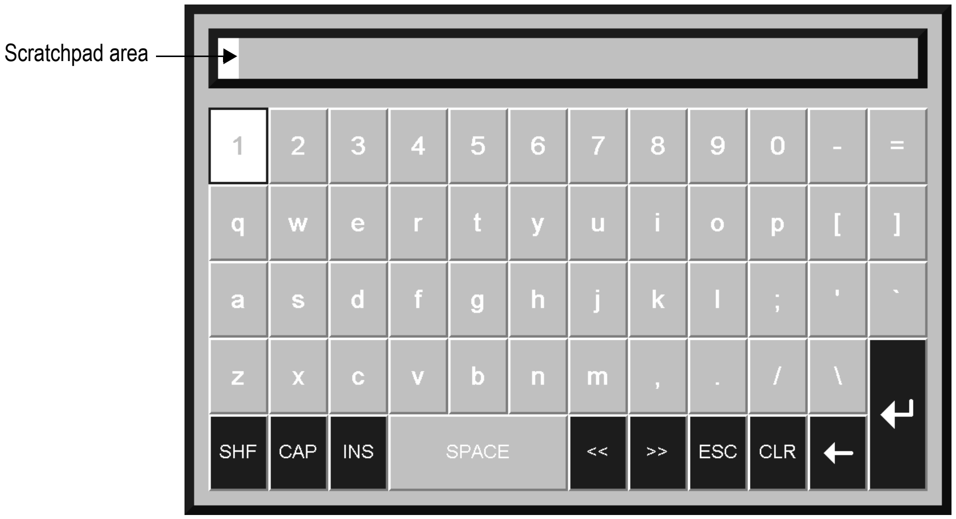
The operator can enter string values at runtime using the string input enable button.

When the operator presses the button, the string pop-up keyboard or scratchpad opens, depending on how you set up the button. To use the scratchpad, a keyboard must be attached to the runtime computer.

For information about setting up the string input enable button, see Help.

Using the string pop-up keyboard

The string pop-up keyboard opens when the operator presses a string input enable button or the User or Password button in the Login dialog box. The pop-up is also used with the password button graphic object.



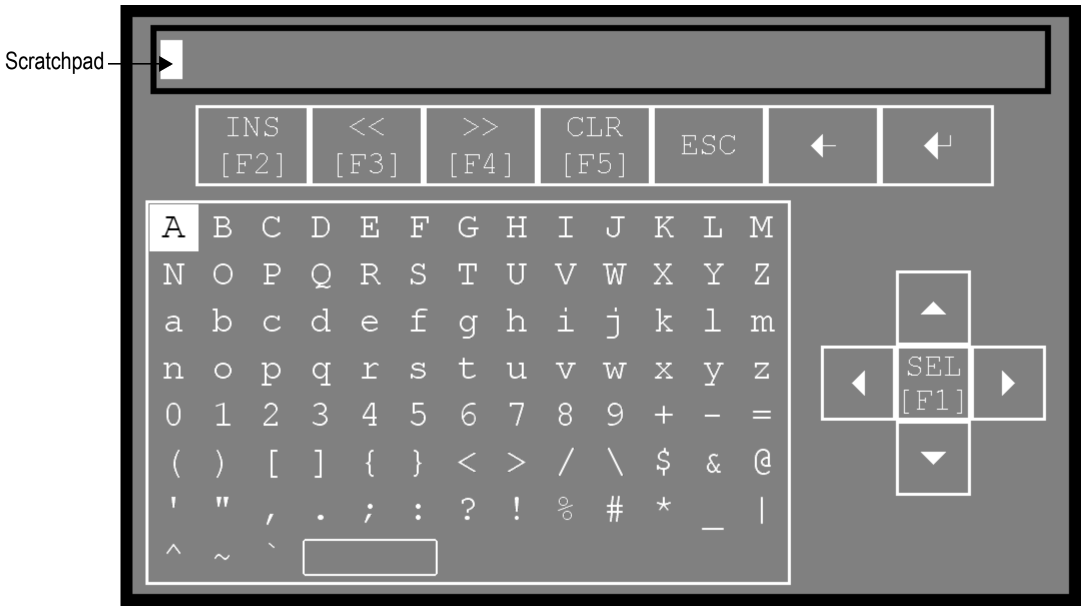
If the operator opens the pop-up keyboard by pressing a string input enable button, the number of characters the keyboard accepts depends on how you set up the button.

When the keyboard is open, no other objects in the graphic display can accept input.

Using the string pop-up character input

You can use a string pop-up character input instead of the string pop-up keyboard. With the string pop-up character input, you use the arrow keys to select the characters to input.

The string pop-up character input opens when the operator presses a string input enable button or the User or Password button in the Login dialog box. The pop-up is also used with the password button graphic object.



If the operator opens the string pop-up character input by pressing a string input enable button, the number of characters the input accepts depends on how you set up the button.

To use the string pop-up character input in Windows 2000 and Windows XP applications

1. In RSView Studio, on the Tools menu, select Options.
2. Click the String Pop-Up tab.
3. Select Use the string pop-up character input.

To use the string pop-up character input on a PanelView Plus or VersaView CE terminal

1. In RSView ME Station, click Terminal Settings.
2. Double-click Input Devices.
3. Select String Pop-Up, and then click Enter.

4. Specify whether to invoke the pop-up character input instead of the pop-up keyboard, and then click OK.

Using the string pop-up scratchpad

If the operator opens the string pop-up scratchpad by pressing a string input enable button, the number of characters the scratchpad accepts depends on how you set up the string input enable button.

Scratchpad area

The scratchpad does not contain any buttons. If the runtime computer does not have an external keyboard attached, the operator will be unable to enter characters or close the scratchpad (unless the runtime terminal is a keypad terminal). If the runtime terminal is a keypad terminal, the operator can enter numbers (not letters) in the scratchpad, and close the scratchpad.

When the scratchpad is open, no other objects in the graphic display can accept input.

Using buttons and keys with the string pop-up windows

You can use the following buttons and their keyboard equivalents with the string pop-up keyboard. For the pop-up scratchpad, you can use the keys on an external keyboard only. The string pop-up character input doesn't have a SHF or CAP key, and it has additional arrow keys for selecting the characters to input.

Use this pop-up keyboard button	Or this keyboard equivalent	To do this
SHF	none	Capitalize a single letter, or type a shifted character such as #.
CAP	none	Capitalize multiple letters.
INS	Insert	Toggle between insert and overstrike modes.
SPACE	Spacebar	Insert a space.
<<	Left Arrow	Move the cursor to the left.
>>	Right Arrow	Move the cursor to the right.

Use this pop-up keyboard button	Or this keyboard equivalent	To do this
ESC	Esc	Close the pop-up window without writing the string to the Login dialog box, the Change Password dialog box, or the string input enable button's Value connection.
CLR	Delete	Clear the scratchpad.
Backspace	Backspace	Delete the character in front of the cursor.
Enter	Enter	Close the pop-up window and write the string to the Login dialog box, the Change Password dialog box, or the string input enable button's Value connection. If Enter key handshaking is set up for the string input enable button, the button's Enter connection is set to 1 and the handshaking timer begins timing. For details, see page 16-14.

What is written to the Value connection

When the operator presses the Enter button in the pop-up keyboard or scratchpad, the string that is sent to the Value connection depends on how the string input enable button is set up.

- If a fill character is set up for the button, and the operator enters fewer than the maximum number of input characters, the fill characters are placed after the string the operator enters.
  - Spaces have a hex value of 20.
  - Zeroes have a hex value of 30.
  - FF characters have a hex value of FF.
  - Null characters have a hex value of 0. The null character indicates the end of string input. It does not add to the actual string length.
- If the number of input characters is fewer than the number of characters in the length of the string tag assigned to the Value connection, the remaining spaces are padded with the null character.

When the string is written to the Value connection, the first character is placed in the high order byte of the first word at the tag address, the second character is placed in the low order byte of the first word, and so on.

Problems with the string pop-up windows

Problems with the string pop-up keyboard and scratchpad occur under these circumstances:

- If the graphic display changes remotely before the user has pressed the Enter button, the pop-up window closes without writing out a string.
- If the screen resolution is smaller than 236 pixels wide by 208 pixels high, the pop-up keyboard cannot open.
- If the screen resolution is smaller than 236 pixels wide by 44 pixels high, the pop-up scratchpad cannot open.
- If the string pop-up window is set up to accept more characters than the Value connection tag length, the pop-up window remains open and the string is not written to the connection. The scratchpad area changes to red to alert the operator of the error.
- If Enter key handshaking is still in effect, the pop-up window closes but the value is not written to the connection.

Changing tag values

This section gives an overview of the graphic objects you can use to change tag values. For information about setting up the objects, see Chapter 16 and Help.

The operator uses these objects to start and stop plant operations, and to control machines and processes. Choose the objects that best suit your process. Set up the data source to carry out the desired actions in response to the changes in tag values.

The operator can use function keys with all of these objects except control list selectors, third-party ActiveX® objects, and drawing objects with slider animation. The last three types of objects can be pressed using a mouse or touch screen.

For information about assigning function keys to graphic objects, see page 16-7.

Do not use push buttons for emergency stops. Emergency stop buttons must be hard-wired.

You can also use macros to assign values to tags. For more information, see Chapter 22.

The operator can change tag values at runtime using these graphic objects:

Use this graphic object	To do this
Momentary push button	Start a process or action by sending one value to the tag when pressed, and another value when released.

Use this graphic object	**To do this**
Maintained push button	Toggle between two values by sending one value to the tag when pressed, and a second value the next time the button is pressed and released. This button is useful for changing a setting within a machine or process, but not for starting the machine or process.
Latched push button	Start a machine or process and remain set (latched) until the process is completed, by sending a value to the tag when pressed, and retaining this value until reset (unlatched) by the Handshake connection.
Multistate push button	Cycle through a series of values. Each time the operator presses the button, the value for the next state is sent to the tag. When the button is in its last state, pressing it causes the button to change to its first state and write out the first state value. This button is useful when you want the operator to see and select multiple options in sequence, using a single button. The button displays the current state of a process or operation by showing a different color, caption, or image to reflect the different states.
Interlocked push button	Use a group of buttons to send values to the same tag. When the operator presses a button in the group, the button's value is sent to the tag, and the button remains highlighted as long as the tag value is the same as the button's value. Pressing a new button in the group releases the other button and sends a new value to the tag. You can also use a single interlocked push button to send a value to a tag.
Ramp button	Increase or decrease the value of a tag by a specified integer or floating-point value. Use two ramp buttons together to create a raise/lower control.
Numeric input enable button	Enter a numeric value and write the value to a tag. You can also use this object to ramp values.
Numeric input cursor point	Enter a numeric value and write the value to a tag. You can also use this object to ramp values.
String input enable button	Enter a string value and write the value to a tag.
RecipePlus button	Write values for all the ingredients in the selected recipe to a set of tags. The button works with the RecipePlus table and RecipePlus selector graphic objects.

Use this graphic object	To do this
Drawing object with horizontal or vertical slider animation	Control the value of a tag by dragging the slider object with a mouse. The pixel position of the slider is translated into a value that is written to the tag. If the value of the tag is changed externally, the position of the slider changes to reflect this.
Control list selector	Select from a list of states for a process or operation. The list is highlighted to show the current state, and the operator can scroll through the list to select a different state. The value assigned to the selected state is written to the tag. If the value of the tag is changed externally, the position of the highlight changes to reflect this.
ActiveX object	A third-party object, connected to an analog, digital, or string tag, including both HMI and data server tags. When the object's property value changes, the new value is written to the associated tag.

You can attach visibility animation to these graphic objects, to display or hide the objects based on changes in tag or expression values. For information about visibility animation, see page 17-8.

For information about creating graphic objects, see Chapter 15. For information about setting up graphic objects, see Chapter 16 and Help.

Viewing tag data

This section gives an overview of the graphic objects you can use to display tag data. For information about setting up the objects, see Chapter 16 and Help.

The operator can view tag data at runtime using these graphic objects:

Use this graphic object	To display this
Numeric display	Numeric tag values. For example, display the current temperature of an oven.
Numeric input cursor point	Numeric tag values. For example, display the current temperature of an oven.
String display	String tag values. For example, set up the data source to generate strings that report on the state of a process or operation, or that provide the operator with instructions about what to do next.

Use this graphic object	To display this
Bar graph	Numeric values in bar graph format. The bar graph increases or decreases in size to show the changing value.
Gauge	Numeric values in dial format. The gauge's needle moves around the dial to show the changing value.
Multistate indicator	The state of a process, on a panel that changes its color, image, or caption to indicate the current state. Each state is set up to correspond to a numeric tag value or least significant bit.
Symbol	The state of a process, using a monochrome image that changes color to indicate the current state. Each state is set up to correspond to a numeric tag value or least significant bit. This object is useful for showing the state of a process or operation at a glance.
List indicator	The state of a process, using a list of possible states with the current state highlighted. Each state is represented by a caption in the list, and is set up to correspond to a numeric tag value or least significant bit. This indicator is useful if you want to view the current state but also want to see the other possible states. For sequential processes, the list can alert the operator about what happens next in the process.
Trend	Historical or current numeric tag values, plotted against time or displayed in an XY plot where one or more tags' values are plotted against another tag's values to show the relationship between them.
RecipePlus table	Current tag values and data set values of the ingredients in the selected recipe, and the number of ingredients in the recipe. The table works with the RecipePlus button and RecipePlus selector graphic objects.
Drawing object with rotation, width, height, fill, color, or horizontal or vertical position animation	Display the value of a tag using a pictorial representation that shows the current value in relation to a range of possible values. For example, use rotation animation to show the tag value as a needle's position on a dial. For color animation, assign different colors to represent different values.
ActiveX object	A third-party object, connected to an analog, digital, or string tag, including both HMI and data server tags. The data displayed depends on the object.

Many of these objects can be set up to manipulate tag values using expressions, and display the expression result rather than the original tag value. For information about expressions, see Chapter 20.

You can also attach visibility animation to these graphic objects, to display or hide the objects based on changes in tag or expression values. For information about visibility animation, see page 17-8.

For information about creating graphic objects, see Chapter 15. For information about setting up graphic objects, see Chapter 16 or Help.

Displaying the date and time

To display the current date and time, create a time and date display. This object uses the operating system's date and time, in the format of the application's current language, and therefore does not require tags or expressions.

Viewing alarms and messages

The operator can view alarms and other messages at runtime using these graphic objects and graphic displays:

This information	Appears in this object	In this default graphic display	For details, see
Alarm messages	Alarm banner	[ALARM].	page 9-26
Alarm messages	Alarm list	No default, although this object appears in the [ALARM MULTI-LINE] and [HISTORY] graphic libraries.	page 9-25, page 9-26, and page 9-28
Alarm messages	Alarm status list	No default, although this object appears in the [STATUS] graphic library.	page 9-27
System activity	Diagnostics list	[DIAGNOSTICS]	page 10-11
Information messages	Information message display	[INFORMATION]	page 12-8
Local messages	Local message display	No default.	page 14-33

The default alarm and information displays open automatically when the assigned tags match messages' trigger values. The default diagnostics display opens automatically when system activity occurs. If desired, you can set up your own graphic displays to open automatically, instead of the default displays. You can also set up any of the displays to open when an operator presses a goto display button or selects a display in the display list selector.

The operator can acknowledge alarm and information messages. The operator can clear alarm and diagnostics messages. The operator can sort alarms and reset their status.

Viewing information about runtime communication errors

To display communication errors in the diagnostics list object, set up message routing so that messages are sent to the RSView Diagnostics List.

For information about setting up diagnostics message routing, see Chapter 10.

Changing languages

You can change languages at runtime. The languages available depend on what has been set up for the runtime application. There is a separate language switch button for each language that you can change to.

For information about setting up language switching, see Chapter 23.

To change languages

1. Press a language switch button.

   Text strings in the application change to the language specified by the button.

Appendix A

Converting PanelBuilder 1400e applications

This appendix describes:

- terms that are different in PanelBuilder™ 1400e and RSView®.
- steps for converting PanelBuilder 1400e applications.
- names of equivalent graphic objects in the two products.
- PanelBuilder 1400e graphic objects that are not supported in RSView.
- PanelBuilder 1400e settings and controls that are not supported in RSView.
- how communications are converted and which PanelBuilder 1400e communication protocols are not supported in RSView.
- converting PanelBuilder 1400e Remote I/O communications.
- PanelBuilder 1400e graphic object features that are not supported in RSView, with information about how to achieve the same result when possible.
- converting PanelBuilder 1400e expressions.

PanelBuilder 1400e applications are applications you create using PanelBuilder 1400e Configuration Software for Windows®. For information about converting applications from PanelBuilder or PanelBuilder32, see Appendix B.

You can convert PanelView 1200 applications to PanelBuilder 1400e applications, and then convert the PanelBuilder 1400e applications to RSView Machine Edition™ applications.

Terminology

This section describes terms that are different in PanelBuilder 1400e and RSView.

PanelBuilder 1400e term	RSView term
screen	display, graphic display
Optional Keypad Write Expression	Optional Expression
programmable controller	data source

PanelBuilder 1400e term	RSView term
control	connection

In RSView, the data source can be memory or a device such as a programmable controller or an OPC® server. RSView writes values to and reads values from the data source. The data source is configured to exchange information (in the form of numeric or string values) between RSView and the machine that your application is controlling. The general term data source is used unless specifically discussing a programmable controller.

Summary of steps

Follow these steps to convert PanelBuilder 1400e applications:

1. Prepare the application in PanelBuilder 1400e, and then convert the application file, as described in the next section.
2. Specify additional project settings, as described on page 4-10.

   For example, if you want the application to have a border around its graphic displays, or to use a title bar, you can specify these options in the Project Settings editor.

   We recommend that you use the Project Settings editor to change the project window size, rather than using the Convert to new window size option in the Machine Edition Import Wizard.

3. If you use the Convert to new window size option in the Machine Edition Import Wizard, check the position of the graphic objects in each display.
4. Set up communications and edit tags that don't convert directly.

   For more information, see page A-9.

5. Set up graphic object features that don't convert directly.

   For more information, see page A-12.

6. Check each expression you used in PanelBuilder 1400e.

   For more information, see page A-13.

7. If you are going to use a printer at runtime, set it up for Ethernet® or USB printing. Adjust the printer settings on the PanelView Plus™ or VersaView® CE terminal.

   For information about setting up printers on the terminal, see the *PanelView Plus Terminals User Manual* or the *VersaView CE Terminals User Manual*. These manuals are available on the RSView Machine Edition CD.

Converting PanelBuilder 1400e application files

Follow these steps to convert a PanelBuilder 1400e application file, with the extension .pvc, to an RSView application file, with the extension .med. The original PanelBuilder 1400e application file is not modified by the conversion.

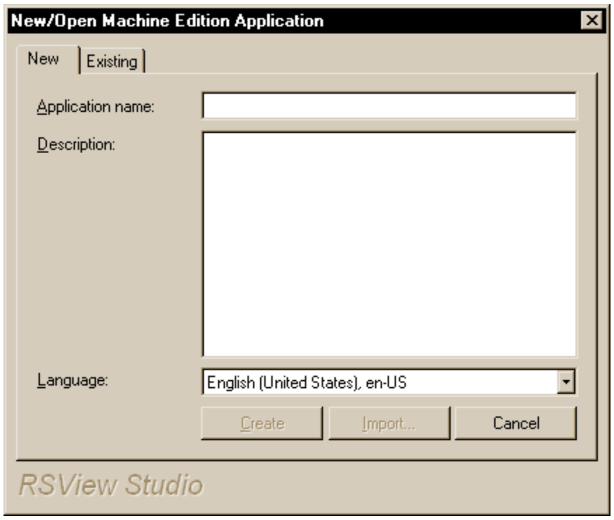
Steps to take in PanelBuilder 1400e before you convert the application

1. Delete the Pass-Through file assignment. RSView Studio™ does not support pass-through file transfers.
2. Make sure the block transfer file numbers are sequential without gaps. If necessary, renumber the block transfer file numbers so there are no missing numbers. Tag addresses in the application will change automatically to match the new number.
3. Save the application.

You can convert the PanelBuilder 1400e application when you open RSView Studio, or once RSView Studio is already open.

To convert a PanelBuilder 1400e application when you open RSView Studio

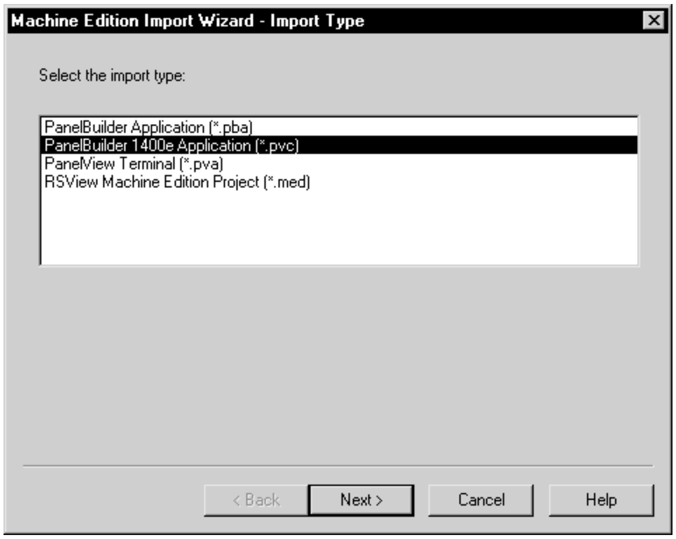
1. Open RSView Studio.



2. In the New tab, in the Application name box, type a name for your converted application, up to 32 characters long.
3. If desired, type a description of the application.

If the PanelBuilder 1400e application contains an Application File Comment, the Application File Comment will overwrite the description you type here. You can add or change the description later, as described on page 4-17.

4. Specify a language for the converted application. For information about using different languages, see Chapter 23.
5. Click Import.


Machine Edition Import Wizard - Import Type

Select the import type:

PanelBuilder Application (*.pba)
PanelBuilder 1400e Application (*.pvc)
PanelView Terminal (*.pva)
RSView Machine Edition Project (*.med)

6. Follow the steps in the Machine Edition Import Wizard.

   For details about the options in the Machine Edition Import Wizard, see Help.

When you complete the steps of the wizard, RSView Studio converts the PanelBuilder 1400e application, creates the converted application's folders and files, and then displays the converted application in the Explorer window in RSView Studio.

If there are any messages about conversion, they are displayed automatically in the Project Status dialog box.

The converted application is created in the ME\HMI projects directory, in a folder with the same name as the application name you specified in step 2.

This is the path to the ME\HMI projects directory:

\Documents and Settings\All Users\Documents\RSView Enterprise\ME\HMI projects (Windows 2000)

or

\Documents and Settings\All Users\Shared Documents\RSView Enterprise\ME\HMI projects (Windows XP)

Conversion messages are saved in a file called Convert.log, in the HMI projects directory.

To convert a PanelBuilder 1400e application when RSView Studio is already open

New Application

1. On the File menu, click New Application, or click the New Application tool.

   If an application is already open, RSView Studio asks you whether to close the application that is currently open. Click Yes.

2. Follow steps 2 through 6 in the previous procedure.

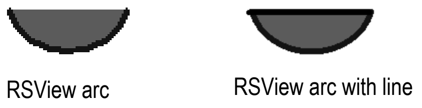
Equivalent graphic objects

This section describes graphic objects that are equivalent in PanelBuilder 1400e and RSView, but have different names in the two products.

This PanelBuilder 1400e object	Is converted to this RSView object	Notes
Increment Value Button	Ramp button	During conversion the button is set up to increment.
Decrement Value Button	Ramp button	During conversion the button is set up to decrement.
Increment Value Button with Display	Ramp button and numeric display	The Increment Value Button with Display is divided into two separate RSView objects.
Decrement Value Button with Display	Ramp button and numeric display	The Decrement Value Button with Display is divided into two separate RSView objects.
ASCII Input (small and large)	String input enable button	
Numeric Entry Keypad (small and large)	Numeric input enable button	
Screen List Selector's list	Display list selector	The PanelBuilder 1400e Screen List Selector is divided into four separate RSView graphic objects.
Screen List Selector's Enter Key	Enter button	
Screen List Selector's Down Cursor	Move down button	
Screen List Selector's Up Cursor	Move up button	
Control List Selector's list	Control list selector	The PanelBuilder 1400e Control List Selector is divided into four separate RSView graphic objects.

This PanelBuilder 1400e object	Is converted to this RSView object	Notes
Control List Selector's Enter Key	Enter button	
Control List Selector's Down Cursor	Move down button	
Control List Selector's Up Cursor	Move up button	
Screen Select Keypad (small and large)	Display list selector	Specify the graphic displays that the display list selector can open.
Screen Keypad Enable Button	Display list selector	Specify the graphic displays that the display list selector can open.
Goto Screen Button	Goto display button	
Return to Previous Screen Button	Return to display button	
ASCII Display	String display	
Numeric Keypad Enable Button	Numeric input enable button	
Normally Open Momentary Push Button	Momentary push button	During conversion the button is set up to be normally open.
Normally Closed Momentary Push Button	Momentary push button	During conversion the button is set up to be normally closed.
Screen Print Button	Display print button	
Alarm History Sort By Time/Sort By Value Button	Sort alarms button	
Alarm Status Reset Qty/Time Button	Reset alarm status button	
Alarm Panel	Alarm banner	
Single Line Alarm Window	Alarm banner	
Alarm Status Screen	Alarm status list	
Clear All Button	Clear alarm history button	
Print Button (in Alarm History screen)	Print alarm history button	

This PanelBuilder 1400e object	Is converted to this RSView object	Notes
Print Button (in Alarm Status screen)	Print alarm status button	
Alarm Status Button/Alarm History Button	Goto display button	
Exit Button	Close display button	
Alarm History List	Alarm list	
Display Mode Button	Alarm status mode button	
Time Display	Time and date display	During conversion the display is set up to show the time only. The PanelBuilder 1400e time format is not converted. For details about the RSView time format, see page 16-17.
Date Display	Time and date display	During conversion the display is set up to show the date only. The PanelBuilder 1400e date format is not converted. For details about the RSView date format, see page 16-17.
Arc (with solid fill style)	Arc (with solid back style) and line	The line graphic object is added because the solid RSView arc shape does not have a line between the two points of the arc.



Unsupported graphic objects

These PanelBuilder 1400e objects are not supported in RSView:

- Scrolling List (includes Cursor List, Multistate Indicator Object List, Local Message Object List, Numeric Data Display Object List)
- Set Bit Cursor Point

Unsupported settings and controls

This section describes PanelBuilder 1400e settings and controls that are not used in RSView.

Controls for transferring runtime application files

PanelBuilder 1400e uses these optional controls for transferring files to the runtime terminal:

- Transfer Inhibit control
- Transfer Request control
- Transfer Status control

These controls are not necessary in RSView because the ME Transfer Utility allows you to transfer the runtime project file while running a project on the runtime terminal.

Settings and controls for alarms

RSView does not use these PanelBuilder 1400e features and settings to manage alarms:

- alarm relays
- bit alarm acknowledgement
- Remote Alarm Operation Hold Time. The PanelBuilder 1400e Remote Alarm Ack Control Hold Time will be used for all alarm hold times. You can change the hold time in the RSView Alarm Setup editor, in the Advanced tab.
- Remote Alarm Control Delay Time. In RSView, if an Ack connection is assigned, when an alarm is acknowledged the Ack connection is set immediately, without waiting for a delay time.

RSView does not use these PanelBuilder 1400e controls to manage alarms:

- PLC Controlled Relay control
- PLC Controlled Audio control
- Acknowledge to PLC control (if the Alarm Acknowledge to PLC option is set to Bit)

Invalid characters in screen names

Characters in PanelBuilder 1400e screen names that are not supported in RSView are replaced with the underscore character.

Screen security settings

PanelBuilder 1400e screen security settings are not converted, because RSView uses a different method to assign security to graphic displays. For information about setting up security in RSView, see Chapter 13.

Block tags

Block tags are not supported in RSView. Block tags that are monitored for alarms in your PanelBuilder 1400e application are converted to bit arrays. For information about monitoring bit arrays for alarm conditions, see Chapter 9.

Converting non-RIO communications

This section describes how communications that do not use Remote I/O (RIO) are converted. For information about converting RIO communications, see page A-10.

RSView does not use nodes for communications. Nodes are converted to RSLinx topics. Topics are then converted into device shortcuts, to run with RSLinx Enterprise. You must have both RSLinx Classic and RSLinx Enterprise installed to make this two-step conversion.

Tags are converted to HMI tags within the RSLinx topics. The Unsolicited_Msgs node is not converted.

If you import an application multiple times, delete the device shortcuts in RSLinx Enterprise before re-importing. Otherwise, multiple unused device shortcuts will be created in RSLinx Enterprise.

For more information about setting up communications, see Chapter 5.

Unsupported communication protocols

These communication protocols are not supported in RSView:

- ControlNet scheduled communications
- Modbus communications

Tags that use these protocols are converted to HMI memory tags. Once you have set up communications for your converted application, change the memory tags to device tags that point to the correct addresses. For information about editing HMI tags, see Chapter 7.

Unsupported tag data types

These tag data types are not supported in RSView:

- Binary (used with Remote I/O communications)
- Bit Position
- 1-BCD, 2-BCD, 5-BCD, 6-BCD, 7-BCD, 8-BCD
- BIN3, BIN4, BIN6, BIN8 (used with Modbus communications)

Tags that use these data types are converted to analog HMI tags with the Default data type. The Default data type uses floating point values.

For Bit Position and Binary data types, use the bitwise expression operators to display data that does not reference supported lengths. For more information, see the Rockwell Automation KnowledgeBase.

To open the KnowledgeBase

1. In RSView Studio, on the Help menu, select Rockwell Software on the Web, and then click Rockwell Automation KnowledgeBase.

For information about using bitwise expression operators, see page 20-8.

Unsupported initial values

Device tags in RSView do not use initial values. Memory tags are converted with their initial values.

Converting RIO communications

You can use Remote I/O (RIO) communications on the PanelView Plus and VersaView CE runtime platforms.

RIO communications are not supported for applications that use the Windows 2000 or Windows XP operating systems. However, you can test run your RIO applications on the development computer.

To convert an RIO application from PanelBuilder 1400e to RSView

1. Convert the application, as described on page A-3.
2. Open the RSLinx Enterprise data server, and then double-click Communication Setup.
3. In the Communication Setup editor, add an RIO driver.
   - For PanelView Plus 400 and 600 terminals, use the 2711P-RN1 driver.
   - For all other PanelView Plus or VersaView CE terminals, use the 2711P-RN6 driver.

   For information about adding drivers in RSLinx, see RSLinx Help.
4. Expand the RIO tree, right-click RIO Data, and then click Configure RIO.
5. In the RIO Configuration dialog box, right-click RIO, and then click Import.
6. Browse to the location of the RIO configuration file.

   The file is saved in the root of the application's directory.
7. In the Communications Setup editor, create a device shortcut that points to the RIO data device.

For information about creating a device shortcut, see RSLinx Help.

8. Apply the shortcut to the RIO driver.
9. Correct any invalid RIO configurations. Invalid RIO configurations are highlighted with red “x” icons.
10. Create an alias for any data that is not a 16-bit integer or bit.
11. Save the converted RIO application.

RIO configurations are not saved with the application when you exit RSView Studio. However, they are backed up with the application in the Application Manager. For information about handling multiple applications with different RIO settings, see the Rockwell Automation KnowledgeBase. For information about using the Application Manager, see page 4-10.

Unsupported PanelBuilder 1400e RIO tags

A PanelBuilder 1400e RIO tag will be converted to an HMI memory tag and an error will be logged to the conversion log file if the RIO tag:

- has a blank address.
- has a data type of 1-BCD, 2-BCD, 3-BCD, 5-BCD, 6-BCD, or 7-BCD.
- has a data type of Bit Position and its address does not reference a single bit.
- has a data type of Binary and its address does not reference a single bit, a single word, or a length or range of 8 or 16 bits.

  For Bit Position and Binary data types, use the bitwise expression operators to display data that does not reference supported lengths. For more information, see the Rockwell Automation KnowledgeBase.

- for block transfer tags and tags using SLC™ optional addressing, has a data type of Byte and its address does not have a bit offset of 0 or 8.

  For other tag types, has a data type of Byte and its address does not have a bit offset of 0 or 10.
- has a data type of Binary and its address has a length or range of 8 bits, but its address does not have a bit offset of 0 or 8 (for block transfer addresses and SLC I/O addresses).
- has a data type of Binary and its address has a length or range of 8 bits, but its address does not have a bit offset of 0 or 10 (for non-SLC I/O addresses).
- has a data type of Binary and its address has a length or range of 16 bits, but its address does not have a bit offset of 0.

- has a data type of Default, Unsigned Integer, Signed Integer, Long Integer, Float, 4 Digit BCD, or 8 Digit BCD and its address has a bit offset assigned that is not 0.
- has a tag type of Block.
- has an invalid PanelBuilder 1400e address.

Unsupported graphic object features

This section describes features of PanelBuilder 1400e graphic objects that are not supported in RSView. The Notes column provides additional information and describes methods for achieving the same result when possible.

Graphic object	Unsupported feature in RSView	Notes
Image, text, arc, ellipse, line, panel, rectangle, wedge	Blinking wallpaper objects	If you want an object to blink at runtime, unlock the wallpaper. In RSView, all of the listed objects except images and panels use color animation to blink. For details, see page 17-9. Panels use the Blink property to blink. Color images do not blink. Monochrome images use the Blink property to blink.
Numeric Display	Polarity	If a PanelBuilder 1400e application was set up with the Polarity control requiring a negative number to display the minus sign, the numeric display will not work properly after the application is converted to RSView Machine Edition.
Numeric Input Cursor Point, Numeric Data Display	Fixed Position and PLC Controlled decimal display options	Use an expression to achieve the same result. Assign the expression to the object's Value connection. For information about expressions, see Chapter 20.
Numeric Input Cursor Point, Numeric Keypad Enable Button, Numeric Keypad	PLC Controlled and Decimal Key Controlled input options	Objects are converted with the Decimal Point property set to Implicit.
Numeric Input Cursor Point	Retain Cursor on Cancel	The numeric input cursor point retains focus when the operator cancels entering a numeric value.
Maintained Push Button, Multistate Push Button, Control List Selector	Initial state values	If you want to set these objects' states on application startup, create a macro to set the appropriate tag values for the objects' connections. For information about macros, see Chapter 22. Assign the macro in the Startup editor. For details, see Help.

Graphic object	Unsupported feature in RSView	Notes
Trend	Blinking pens Date labels on the X-Axis Background screen plotting	The date is displayed in the title. You can plot tag values in the background by assigning the tags to a data log model. Tags set up for background screen plotting are automatically assigned to a data log model on conversion. However, data log models do not plot expression values. Therefore, expressions set up for background screen plotting are not converted. For information about data logging, see Chapter 11.
All objects	PanelBuilder 1400e object name	Object names are replaced with the RSView default object names. The PanelBuilder 1400e object name is used for the object's description. You can view and edit the name and description in the Property Panel. For details, see Help.
All objects	Caption and image placement	RSView supports one, three, or nine positions for captions and images, depending on the type of object. On conversion, captions and images are positioned using the closest match. Therefore some captions might overlap images, some captions might be truncated, and some images might be clipped to fit the object.
All objects	Multiple image labels	RSView supports one image label per object or state. If a PanelBuilder 1400e object is set up to use multiple image labels, only the top left image is converted.

Converting expressions

Some PanelBuilder 1400e expression syntax is not supported in RSView. Expressions are converted without modification, and then turned off by placing warning text at the beginning of the first line of the expression. In addition, exclamation marks (!) are placed at the beginning of each subsequent line of the expression. To turn on the expression, you must remove the warning text and exclamation marks, and revise the syntax if necessary.

The maximum expression length in RSView is 1,024 characters. If a PanelBuilder 1400e expression contains more than 1,024 characters, the excess characters are not converted.

Some PanelBuilder 1400e objects support both tags and expressions. For these objects, if the text assigned to a connection could be valid syntax for both a tag and an expression, the connection is treated as an expression, and is therefore turned off.

For example, N20-0_String_64 could be the name of a tag, or it could be an expression that subtracts “0_String_64” from the tag “N20.” The text would be converted as an expression, and turned off.

To turn on an expression

1. Select the object containing the expression.
2. Open the Property Panel, and then click the Connections tab.
3. In the Exprn column, click the Browse button beside the expression to turn on.
4. In the Expression editor, delete the warning text and exclamation marks.
5. Revise the expression, if necessary, using the tables in the following three sections as guides.
6. Click Check Syntax.

...

Browse button

For more information about using the Expression editor, see Chapter 20 or Help.

Equivalent expression syntax

This table describes RSView expression syntax that is equivalent to PanelBuilder 1400e syntax. When you edit the converted expressions, replace the PanelBuilder 1400e syntax with the RSView equivalent.

Syntax that is not listed in this table or in the next section is okay the way it is.

Type of expression component or operator	**PanelBuilder 1400e syntax**	**RSView syntax**
Comment	REM or ‘	!
Line continuation	_ (underscore)	Not needed.
Equality	(=)	EQ or ==
Bitwise Not	Not	~ (tilde)
Bitwise And	And	&
Bitwise Or	Or	| (pipe)
Bitwise XOr	XOr	^
If both operands are Byte, Integer, Long, Variant, or any combination of these data types, use the RSView syntax. For other data types, no change is needed.		

Unsupported expression syntax

This table describes the PanelBuilder 1400e expression syntax that is not supported in RSView with information about how to achieve the same result where possible.

Type of expression component or operator	PanelBuilder 1400e syntax	Equivalent RSView syntax (if any)
Exit statement	Exit	
Local variables	DIM *varname* AS ... *varname* =	
Integer division	\	(*x* - (*x* MOD *y*))/*y*
Endif	If then endif If then else endif	If then else 0 If then else
Select case	Select Case Case1...CaseN CaseElse EndSelect	Use nested if-then-else.
Logical Xor (if one or both operands are Boolean or Single data types)	Xor	NOT ((*x* AND *y*) OR NOT (*x* OR *y*))

Order of precedence

The order of precedence is slightly different in RSView. Check your expressions to make sure the result is what you intend.

PanelBuilder 1400e order of precedence	RSView order of precedence
( )	( )
- (negation)	NOT, ~ (tilde)
*, / (floating point division)	*, /, MOD, %, **, AND, &&, &, >>, <<
\ (integer division)	+, -, OR, ||, |, ^
MOD	EQ, ==, NE, <>, LT, <, GT, >, LE, <=, GE, >=
+, - (subtraction)	
=, <>, <, >, <=, >=	

PanelBuilder 1400e order of precedence	RSView order of precedence
Not	
And	
Or	
Xor	

For more information about order of precedence, see page 20-10.

Appendix B

Converting PanelBuilder and PanelBuilder32 applications

This appendix describes:

- terms that are different in PanelBuilder™ and RSView®.
- steps for converting PanelBuilder applications.
- names of equivalent graphic objects in PanelBuilder and RSView.
- PanelBuilder graphic objects that are not supported in RSView.
- PanelBuilder settings and controls that are not supported in RSView.
- how communications are converted and which PanelBuilder communication protocols are not supported in RSView.
- converting PanelBuilder Remote I/O communications.
- PanelBuilder graphic object features that are not supported in RSView, with information about how to achieve the same result when possible.

This appendix uses the term PanelBuilder to refer to both PanelBuilder and PanelBuilder32 features.

For information about converting applications from PanelBuilder 1400e, see Appendix A.

Terminology

This section describes terms that are different in PanelBuilder and RSView.

PanelBuilder term	RSView term
screen	display, graphic display
programmable controller	data source
control	connection

In RSView, the data source can be memory or a device such as a programmable controller or an OPC® server. RSView writes values to and reads values from the data source. The data source is configured to exchange information (in the form of numeric or string values) between RSView and the machine that your application is controlling. The general term data source is used unless specifically discussing a programmable controller.

Summary of steps

Follow these steps to convert PanelBuilder applications:

1. Convert the application file, as described in the next section.
2. Specify additional project settings, as described on page 4-10.

   For example, if you want the application to have a border around its graphic displays, or to use a title bar, you can specify these options in the Project Settings editor.
3. If you select Convert to new window size, check the position of the graphic objects in each display.
4. Set up communications and edit tags that don't convert directly.

   For more information, see page B-7.
5. Set up graphic object features that don't convert directly.

   For more information, see page B-10.

Converting PanelBuilder application files

Follow these steps to convert a PanelBuilder application file, with the extension .pba or .pva, to an RSView application file, with the extension .med. The original PanelBuilder application file is not modified by the conversion.

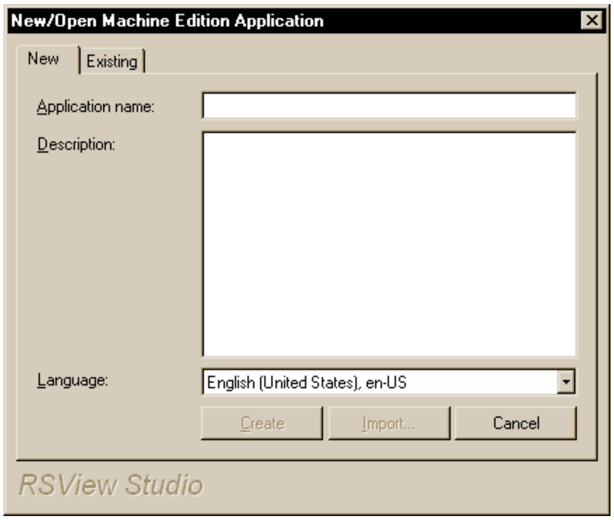
Steps to take in PanelBuilder before you convert the application

1. Semicolons (;) in tag addresses are supported in PanelBuilder, but not in RSView Studio™. Before importing the PanelBuilder application, in the PanelBuilder Tag Editor, change the semicolons to colons (:).
2. Dashes (-) in tag names are supported in PanelBuilder, but not in RSView Studio. Before importing the PanelBuilder application, in the PanelBuilder Tag Editor, locate any tags whose names contain dashes and duplicate the tags. Then rename the tags without the dash, or replace the dash with an underscore (_). Once the tags have been renamed, use the Tag Search feature to find the graphic objects using the original tag names and edit the objects to replace the old tag names with the new ones.

You can convert the PanelBuilder application when you open RSView Studio, or once RSView Studio is already open.

To convert a PanelBuilder application when you open RSView Studio

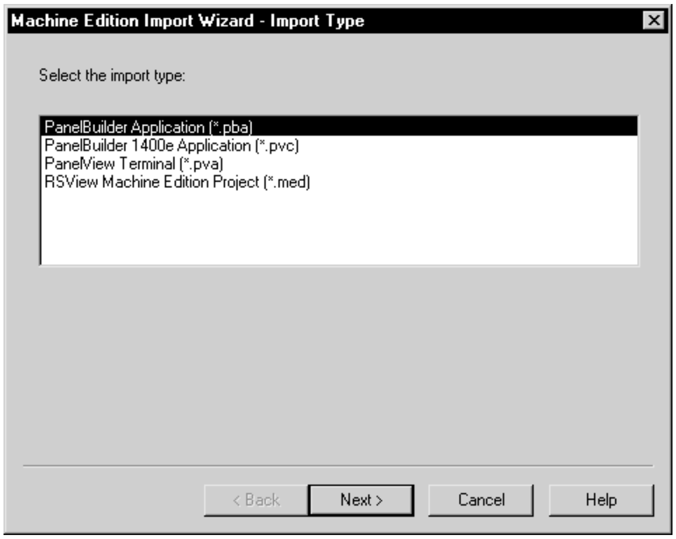
1. Open RSView Studio.



2. In the New tab, in the Application name box, type a name for your converted application, up to 32 characters long.
3. If desired, type a description of the application.

   If the PanelBuilder application contains an Application Description, the Application Description will overwrite the description you type here. You can add or change the description later, as described on page 4-17.
4. Specify the last language that was used to edit the application. This will be used for the converted application. For information about using different languages, see Chapter 23.
5. Click Import.

You can only import one language for your application, even if the original application uses multiple languages. The imported language will be the last language used to edit the application.



PanelBuilder Application (*.pba)
PanelBuilder 1400e Application (*.pvc)
PanelView Terminal (*.pva)
RSView Machine Edition Project (*.med)

6. Follow the steps in the Machine Edition Import Wizard.

   For details about the options in the Machine Edition Import Wizard, see Help.

When you complete the steps of the wizard, RSView Studio converts the PanelBuilder application, creates the converted application's folders and files, and then displays the converted application in the Explorer window in RSView Studio.

If there are any messages about conversion, they are displayed automatically in the Project Status dialog box.

The converted application is created in the ME\HMI projects directory, in a folder with the same name as the application name you specified in step 2.

This is the path to the ME\HMI projects directory:

\Documents and Settings\All Users\Documents\RSView Enterprise\ME\HMI projects (Windows® 2000)

or

\Documents and Settings\All Users\Shared Documents\RSView Enterprise\ME\HMI projects (Windows XP)

Conversion messages are saved in a file called Convert.log, in the HMI projects directory.

To convert a PanelBuilder application when RSView Studio is already open

New Application

1. On the File menu, click New Application, or click the New Application tool.

   If an application is already open, RSView Studio asks you whether to close the application that is currently open. Click Yes.

2. Follow steps 2 through 6 in the previous procedure.

Equivalent graphic objects

This section describes graphic objects that are equivalent in PanelBuilder and RSView, but have different names in the two products.

This PanelBuilder object	Is converted to this RSView object	Notes
Numeric Entry Keypad Enable Button	Numeric input enable button	
Numeric Entry Cursor Point	Numeric input enable button	
Increment/Decrement Entry Button	Numeric input enable button	The numeric input enable button is set up to work as a ramp button, using the Fine Step value. The Coarse Step value is not converted.
ASCII Entry Keypad Enable button	String input enable button	The Show Current String on ASCII Scratchpad setting is not converted. The pop-up scratchpad or keyboard is always blank when opened.
ASCII Entry Cursor Point	String input enable button	The Show Current String on ASCII Scratchpad setting is not converted. The pop-up scratchpad or keyboard is always blank when opened.
Message Display	Multistate indicator	
Numeric Data Display	Text	The text object contains a numeric embedded variable that displays the read tag.
Connected Line	Polyline	
Circle	Ellipse	The ellipse has a circular shape.
Freeform	Freehand	
Screen List Selector	Display list selector	

This PanelBuilder object	Is converted to this RSView object	Notes
Goto Screen Button	Goto display button	
Return Screen Button	Return to display button	
New Password Button	Password button	
Print Alarm List Button	Print alarm history button	
Clear Alarm List Button	Clear alarm history button	

Unsupported graphic objects

These PanelBuilder objects are not supported in RSView:

- Print Only Object
- Circular Scale
- Scrolling Text
- Print Alarm Button
- Horn Silence Button
- Lamp/Horn Test Button
- Select Operator Button
- Enable/Disable Security Button
- Verify Password Button. The RSView Password button opens a dialog box that allows the user to type and verify a new password.

Unsupported settings and controls

This section describes PanelBuilder settings and controls that are not used in RSView.

Settings and controls for alarms

RSView does not use these PanelBuilder features and settings to manage alarms:

- Ack setting for alarm messages; in RSView, all alarms can be acknowledged
- bit alarm acknowledgement

RSView does not use these PanelBuilder controls to manage alarms:

- Remote Ack All Handshake Tag
- Remote Clear All Alarm Tag

- Remote Clear All Alarm Handshake Tag

Invalid characters in screen names and tag names

Characters in PanelBuilder screen names and tag names that are not supported in RSView are replaced with the underscore character.

Time and date

PanelBuilder time and date formats are not converted. For details about RSView time and date formats, see page 16-17.

External fonts

PanelBuilder external fonts are not converted. When you convert your application you can specify the font to use instead. For details, see Help for the Machine Edition Import Wizard.

Screen security settings

PanelBuilder screen security settings are not converted, because RSView uses a different method to assign security to graphic displays. For information about setting up security in RSView, see Chapter 13.

Power-up options

These PanelBuilder power-up options are not imported into RSView:

- Write Last Terminal State to Controller
- Display Last User Screen
- Use Terminal Presets

Converting non-RIO communications

This section describes how communications that do not use Remote I/O (RIO) are converted. For information about converting RIO communications, see page B-8.

RSView does not use nodes for communications. Nodes are converted to RSLinx topics. Topics are then converted into device shortcuts, to run with RSLinx Enterprise. You must have both RSLinx® Classic and RSLinx Enterprise installed to make this two-step conversion.

If you import an application multiple times, delete the device shortcuts in RSLinx Enterprise before re-importing. Otherwise, multiple unused topics will be created in RSLinx Enterprise.

For more information about setting up communications, see Chapter 5.

Unsupported communication protocols

These communication protocols are not supported in RSView:

- ControlNet scheduled communications
- Modbus communications
- Profibus DP communications
- DeviceNet communications
- DHPlus nodes that use the AutoMax node type
- invalid PanelBuilder node types and non-existent node names that are saved with the PanelBuilder application

Tags that use these protocols are converted to HMI memory tags. Once you have set up communications for your converted application, change the memory tags to device tags that point to the correct addresses. For information about editing HMI tags, see Chapter 7.

Bit array tags

You can monitor bit arrays for alarm conditions in RSView, but you can't assign bit arrays to most graphic objects or write to bit arrays. (The only exception is the piloted control list selector object. For this object, you can assign a bit array tag to the Visible States connection.)

All bit array tags in your PanelBuilder application are converted to HMI memory tags.

For information about monitoring bit arrays for alarm conditions, see Chapter 9. For information about editing HMI tags, see Chapter 7. For information about the piloted control list selector, see Help.

Converting RIO communications

Remote I/O (RIO) communications are not supported for applications that use the Windows 2000 or Windows XP operating systems. You can use RIO communications on the PanelView Plus™ and VersaView® CE runtime platforms.

To convert an RIO application from PanelBuilder to RSView

1. Convert the application, as described on page B-2.
2. Open the RSLinx Enterprise data server, and then double-click Communication Setup.
3. In the Communication Setup editor, add an RIO driver.
    - For PanelView Plus 400 and 600 terminals, use the 2711P-RN1 driver.

- For all other PanelView Plus or VersaView CE terminals, use the 2711P-RN6 driver.

For information about adding drivers in RSLinx, see RSLinx Help.

4. Expand the RIO tree, right-click RIO Data, and then click Configure RIO.
5. In the RIO Configuration dialog box, right-click RIO, and then click Import.
6. Browse to the location of the RIO configuration file.

   The file is saved in the root of the application’s directory.
7. In the Communications Setup editor, create a device shortcut named “PVRIO” that points to the RIO data device.

   For information about creating a device shortcut, see RSLinx Help.
8. Correct any invalid RIO configurations. Invalid RIO configurations are highlighted with red “x” icons.
9. Save the converted RIO application.

Unsupported PanelBuilder RIO tags

A PanelBuilder RIO tag will be converted to an HMI memory tag and an error will be logged to the conversion log file if the RIO tag:

- has a blank address.
- has a data type of Bit Array and its address does not have an array size of 1, 8, 16, or 32.
- has a data type of Bit Array, and its array size is 16 or 32, but its address does not have a bit offset of 0.
- has a data type of Bit or BOOL, and its address does not contain the bit delimiter character “/”.
- has a data type of 4-BCD, Unsigned Integer, Signed Integer or INT, Character Array, or DINT, and its address contains the bit delimiter character “/”.
- is a block transfer tag with a data type of Bit Array, and its array size is 8, but its address does not have a bit offset of 0 or 8.
- is a block transfer tag with a data type of SINT, and its address does not have a bit offset of 0 or 8.
- is an I/O tag with an address that references an undefined rack.

- is an I/O tag with a data type of SINT, and its address does not have a bit offset of 0 or 10.
- does not have a valid I/O address or block transfer address.

Unsupported graphic object features

This section describes features of PanelBuilder graphic objects that are not supported in RSView. The Notes column provides additional information and describes methods for achieving the same result when possible.

Graphic object	Unsupported feature in RSView	Notes
Image text, arc, ellipse, freehand, line, polyline, rectangle, wedge	Blink property	In RSView, all of the listed objects except images use color animation to blink. For details, see page 17-9. Color images do not blink. Monochrome images use the Blink property to blink.
Increment/Decrement Entry Button (Converted to numeric input enable button)	Allow Home/End Allow Wrap Ramping by coarse steps	
Maintained Push Button, Multistate Push Button, Standard Control List Selector	Initial state values	If you want to set these objects' states on application startup, create a macro to set the appropriate tag values for the objects' connections. For information about macros, see Chapter 22. Assign the macro in the Startup editor. For details, see Help.
Multistate Indicator, Message Display	Print Setting	
Bar Graph	Inner text and inner graphic	Converted to a separate text object and image object.
Gauge	Inner text and inner graphic	Converted to a separate text object and image object.
	Scale clipping	If the scale doesn't fit within the height or width of the gauge, it is not clipped. Check the position of the scale to ensure it doesn't overlap other objects.
	Needle	Converted to a separate gauge object; if the gauge had 2 needles, each needle is converted to a separate gauge object.
Alarm List	No Acknowledgement Required	All alarms can be acknowledged.

Graphic object	**Unsupported feature in RSView**	**Notes**
All objects	Image placement	RSView supports one, three or nine positions for images, depending on the type of object. On conversion, images are positioned using the closest match. Therefore some images might be clipped to fit the object.
All objects	Turn Object View On property	If this property is set to False, the converted object has a transparent background, no border, no caption, and no image.
All objects	Blinking inner graphics	If the inner graphic uses a color image, it will not blink. Use a monochrome image if you want the inner graphic to blink.

System tags

This appendix describes system tags.

System tags are preconfigured HMI tags created by RSView®. System tags are read-only. Display them as needed in your application.

Alarms

The following tag contains the time and date when the status of alarms was last reset. The date uses the long date format.

Tag name	Type	Function
system\AlarmReset DateAndTimeString	String	Contains the date and time of the last alarm reset.

For information about resetting alarms, see page 9-8.

Graphics

The following HMI tags can be used to make graphic objects appear as though they are blinking on and off:

Tag name	Type	Function
system\BlinkFast	Digital	Toggles on and off every 100 ms (10 times per second).
system\BlinkSlow	Digital	Toggles on and off every 500 ms (twice per second).

A more efficient way to make graphic objects blink is to use the blinking color option in color animation. For details, see page 17-9.

Also, many objects have a Blink property that you can set up. For information about specific objects, see Help.

Time

These HMI tags record time and date information in various formats:

Tag Name	Type	Provides this data	Read or write
system\Date	String	System date.	Read only
system\DateAndTime Integer	Analog	Number of seconds elapsed since midnight (00:00:00) January 1, 1970, coordinated universal time.	Read only
system\DateAndTime String	String	Complete date and time display. For example: Monday, December 12 2001 10:47:50 AM	Read only
system\DayOfMonth	Analog	Day of the month (1 - 31).	Read only
system\DayOfWeek	Analog	Day of the week (1-7); Sunday = 1.	Read only
system\DayOfYear	Analog	Day of the year (1-366).	Read only
system\Hour	Analog	Hour of the day (0-23).	Read and write
system\Minute	Analog	Minutes (0 - 59).	Read and write
system\Month	Analog	Number for month (1-12).	Read only
system\MonthString	String	Name of the month.	Read only
system\Second	Analog	Seconds (0 - 59).	Read and write
system\Time	String	System Time.	Read only
system\Year	Analog	The year (1980-2099).	Read only

For information about using the data source to update the system date and time, or about sending the runtime computer's date and time to the data source, see Chapter 8.

User

This tag contains the name of the current user:

Tag Name	Type	Function
system\User	String	Contains name of logged-in user.

We recommend that you use the expression security function CurrentUserName( ) instead of the system\User tag, especially if you intend to convert the application to RSView Enterprise Supervisory Edition. In distributed applications, system\User returns the name of the user logged into the HMI server, not the user logged into the display client.

For more information about the security functions, see page 20-14.

Appendix D

ODBC database schema

This appendix describes the ODBC database format, or schema, for messages from FactoryTalk® Diagnostics™. The target table of the ODBC database to which you are sending messages must use the format shown in this appendix.

The option of logging FactoryTalk Diagnostics messages to an ODBC database is available for computers running Windows® 2000 and Windows XP only.

For information about setting up FactoryTalk Diagnostics, see Chapter 10.

FactoryTalk Diagnostics log table

FactoryTalk Diagnostics log data in ODBC format uses one table.

This column	Contains	SQL data type	Length
TimeStmp	The time and date data was logged, in coordinated universal time format. Encoded as a date variant.	SQL_TIMESTAMP	Driver dependent
MessageText	Message to be logged.	SQL_VARCHAR, *or* SQL_CHAR	254
Audience	A number representing the message audience: 0 for Operator 1 for Engineer 2 for Developer 3 for Secure	SQL_SMALLINT, *or* SQL_INTEGER	1
Severity	A number representing the severity of the diagnostics message: 0 for Error 1 for Warning 2 for Information 3 for Audit	SQL_SMALLINT, *or* SQL_INTEGER	1
Area	The FactoryTalk path to the area in which the activity occurred. Used for RSView® Supervisory Edition™ only.	SQL_VARCHAR, *or* SQL_CHAR	80
Location	The name of the computer where the message was generated.	SQL_VARCHAR, *or* SQL_CHAR	15

This column	Contains	SQL data type	Length
UserID	The name of the user (including domain name, if there is one) that initiated the action that caused the diagnostics message. If the diagnostics message was caused by an HMI server, the user column contains "System."	SQL_VARCHAR, *or* SQL_CHAR	38
UserFullName	The full name of the user that was logged in when the activity occurred.	SQL_VARCHAR, *or* SQL_CHAR	255
Provider	The name of the product that generated the message.	SQL_VARCHAR, *or* SQL_CHAR	20

APPENDIX E

Importing and exporting XML files

This appendix describes:

- creating alarm and graphics XML files.
- exporting, editing, and importing XML files.
- the alarm XML file structure.
- the graphics XML file structure.

About XML

XML is the Extensible Markup Language used to create documents with structured text information. It has a standardized format and structure. You can use XML to edit the elements and attributes needed to create an alarm setup file or to modify graphic displays.

For example, if you have a list of 100 tags to monitor for alarms, with multiple messages for each tag, you might prefer to enter all the information in a text editor, and then import the alarm setup information into RSView®.

Another example of using XML files is to export the alarm setup information you develop in one application, import the setup information to another RSView application, and then modify the alarm setup as needed. Or, you could modify the information in the XML file before importing it.

For more information about XML, see the World Wide Web Consortium's web page about XML at:
http://www.w3.org/XML.

Creating XML files by exporting

The quickest way to create an XML file for your application's alarm setup or graphic displays is to export the data from RSView. You can then open the XML file in Notepad, make your changes, and import the file back into RSView.

The strings for the application's current language are exported to the XML file. To export strings for another language, reopen the application in the new language and repeat the XML export.

The strings for the application's current language are exported to the XML file.

To export alarm information to an XML file

1. In the Explorer window, right-click the Alarm Setup editor.

2. Click Import and Export.

   The Alarm Import Export Wizard opens.

3. Follow the instructions in the wizard.

   For information about using the Alarm Import Export Wizard, see Help.

RSView creates a file with the name you specify, in the location you specify.

To export graphic display information to an XML file

1. In the Explorer window, right-click the Displays editor or the Global Objects editor.
2. Click Import and Export.

   The Graphics Import Export Wizard opens.

3. Follow the instructions in the wizard.

   For information about using the Graphics Import Export Wizard, see Help.

RSView creates XML files for the selected graphic displays, in the location you specify.

RSView also creates a file called BatchImport_*Application name*.xml, in the same location. You can use this file to import multiple displays at the same time. To import a different set of displays than you exported, edit the list of display names in the BatchImport_*Application name*.xml file.

Editing XML files

We recommend that you use Notepad to edit your XML files.

If you do not want to change a property, you don't need to include it in the XML file. When you import the file, if you select the option "Create new objects in the display," properties that are not listed in the file are set to their default values. If you select the option "Update existing objects on the display," only properties that are listed in the file are updated with imported information.

If you include attributes for an object whose name does not match one of those in the graphic display, the attributes for that object are not imported. Attributes for all other objects in the file whose names do match the ones in the graphic display are imported.

Saving XML files in Notepad

Save XML files created or edited in Notepad using either UTF-8 or UTF-16 file format. Notepad's Unicode file type corresponds to UTF-16 file format. For files containing strings in English or other Latin-based languages, UTF-8 is recommended, to reduce the size of the XML file. For other languages such as Chinese, Japanese, or Korean, UTF-16 is recommended.

The first line of every XML file contains XML version and encoding attributes. Make sure the encoding attribute matches the format that you are going to use when you save the file. For example, if the original file was saved in UTF-8 format and you plan to save it in UTF-16 format, make sure the first line specifies encoding="UTF-16".

Testing XML files

An XML file must be well-formed to be imported. To find out whether your XML file is well-formed, test it.

To test an XML file

1. Open the XML file in Internet Explorer.

If you can see the XML code, your file is well-formed. If the XML code is not well-formed, Internet Explorer displays an error message.

Importing XML files

You can import an alarm setup, graphic display, or global object display that has been created using an external programming tool or editor, or you can import an XML file that you originally exported from RSView and then modified.

When you import an alarm setup, graphic display, or global object display, your existing alarm setup or display will be overwritten. Back up your application first, using the Application Manager tool. Or, you can save a copy of your existing alarm setup or display by exporting it to an XML file before you import the new one.

Error log file

If errors occur during importing, the errors are logged to a text file. The file opens automatically when importing is finished. The last paragraph of the file lists the location of the file.

Importing alarm XML files

To import alarm information from an XML file

1. In the Explorer window, right-click the Alarm Setup editor.
2. Click Import and Export.

   The Alarm Import Export Wizard opens.
3. Follow the instructions in the wizard.

   For more information about using the Alarm Import Export Wizard, see Help.

Importing graphics XML files

You can import a single graphic or global object display XML file at a time, or import multiple displays. You can also choose whether to import new objects or update existing objects.

To import multiple displays, specify the names of the displays in the file BatchImport_*Application name*.xml. RSView creates this file when you export multiple displays. For details, see page E-2.

To import display information from an XML file

1. In the Explorer window, right-click the Displays or Global Objects editor.
2. Click Import and Export.

   The Graphics Import Export Wizard opens.
3. Follow the instructions in the wizard.

   For more information about using the Graphics Import Export Wizard, see Help.

Alarm setup XML file structure

The alarm setup XML file is an RSView XML document that describes the alarm setup for an application. The root element of the XML document is called **alarms**. It represents the Alarm Setup editor. An XML document can contain only one root element. All other elements in the document must be contained or nested within the root element.

In an XML document, the start of an element is marked **<element name>**. The end is marked **</element name>**.

If the element contains no subelements, the end can be marked />. For example, <trigger id="T1" type="value" ack-all-value="0" />.

The syntax for specifying an attribute for an element is **attribute="value"**. The attribute value must be enclosed in single or double quotes.

Here is a sample structure for an alarm XML document:.

Element	**Description**
<alarms>	Root element.
<alarm>	Contains attributes from the Advanced tab of the Alarm Setup editor, as well as the triggers and messages elements.
<triggers>	Contains a trigger element for each trigger in the Triggers tab of the Alarm Setup editor.

Element	Description
<trigger id="T1" />	Contains attributes for the first alarm trigger.
<trigger id="T2" />	Contains attributes for the second alarm trigger.
</triggers>	Indicates the end of the triggers element.
<messages>	Contains a message element for each message in the Messages tab of the Alarm Setup editor.
<message id="M1" />	Contains attributes for the first alarm message.
<message id="M2" />	Contains attributes for the second alarm message.
</messages>	Indicates the end of the messages element.
</alarm>	Indicates the end of the alarm element.
</alarms>	Indicates the end of the alarms element.

You can specify multiple attributes for an element. For example, the alarm element contains 11 possible attributes from the Advanced tab of the Alarm Setup editor.

For more information about alarm elements and their attributes, see Help for the Alarm Import Export Wizard.

Graphics XML file structure

The graphic display or global object display XML file is an RSView XML document that describes the objects and settings for a display. The root element of the XML document is called **gfx**. It represents the display. An XML document can contain only one root element. All other elements in the document must be contained or nested within the root element.

In an XML document, the start of an element is marked **<element name>**. The end is marked **</element name>**.

If the element contains no subelements, the end can be marked />. For example, <caption fontFamily="Arial" fontSize="8" bold="false" />.

The syntax for specifying an attribute for an element is **attribute="value"**. The attribute value must be enclosed in single or double quotes.

Here is a sample structure for a graphic display XML document containing two graphic objects. The second object has states:

Element	**Description**
<gfx>	Root element.
<displaySettings />	Contains attributes from the Display Settings dialog box in the Graphics editor.
<object1>	Contains attributes from the General and Common tabs in the object's Properties dialog box, as well as elements for the object's caption, image, animation, and connections.
<caption />	Contains attributes for the object's caption.
<imageSettings />	Contains attributes for the object's image.
<animations>	Contains an animation element for each type of animation set up for the object.
<animateVisibility />	Contains attributes for Visibility animation.
<animateColor />	Contains attributes for Color animation.
</animations>	Indicates the end of the animations element.
<connections>	Contains a connection element for each connection assigned to the object.
<connection name= "Value" />	Contains attributes for the Value connection.
<connection name= "Indicator" />	Contains attributes for the Indicator connection.
</connections>	Indicates the end of the connections element.
</object1>	Indicates the end of the *object1* element.
<object2>	Contains attributes from the General and Common tabs in the object's Properties dialog box, as well as elements for the object's states and connections.
<states>	Contains state elements for each of the object's states.
<state stateid="0">	Contains attributes for the object's first state, as well as elements for the state's caption and image.
<caption />	Contains attributes for the state's caption.
<imageSettings />	Contains attributes for the state's image.
</state>	Indicates the end of the state element.

Element	Description
<state stateid="1">	Contains attributes for the object's second state, as well as elements for the state's caption and image.
<caption />	Contains attributes for the state's caption.
<imageSettings />	Contains attributes for the state's image.
</state>	Indicates the end of the state element.
</states>	Indicates the end of the states element.
<connections>	Contains a connection element for each connection assigned to the object.
<connection name= "Value" />	Contains attributes for the Value connection.
<connection name= "Indicator" />	Contains attributes for the Indicator connection.
</connections>	Indicates the end of the connections element.
</*object2*>	Indicates the end of the *object2* element.
</gfx>	Indicates the end of the gfx element.

You can specify multiple attributes for an element. For example, the caption element contains 13 possible attributes.

Elements for group objects begin with **<group>** and end with **</group>**. The <group> element contains all the elements for each object in the group.

For more information about graphic object elements and their attributes, see Help for the Graphics Import Export Wizard.

RFC1766 names

This appendix describes RFC1766 names for Windows languages.

Mapping languages to RFC1766 names

The following table lists the languages that Windows supports and the RFC1766 name associated with each language.

You can use the codes to name the translated application files before importing them. The codes are also used with the CurrentLanguage function.

RFC1766 Name	**Language – Country/Region**
af–ZA	Afrikaans – South Africa
sq–AL	Albanian – Albania
ar–DZ	Arabic – Algeria
ar–BH	Arabic – Bahrain
ar–EG	Arabic – Egypt
ar–IQ	Arabic – Iraq
ar–JO	Arabic – Jordan
ar–KW	Arabic – Kuwait
ar–LB	Arabic – Lebanon
ar–LY	Arabic – Lybia
ar–MA	Arabic – Morocco
ar–OM	Arabic – Oman
ar–QA	Arabic – Qatar
ar–SA	Arabic – Saudi Arabia
ar–SY	Arabic – Syria
ar–TN	Arabic – Tunisia
ar–AE	Arabic – United Arab Emirates
ar–YE	Arabic – Yemen
hy–AM	Armenian – Armenia
az–AZ–Cyrl	Azeri (Cyrillic) – Azerbaijan
az–AZ–Latn	Azeri (Latin) – Azerbaijan

RFC1766 Name	Language – Country/Region
eu–ES	Basque – Basque
be–BY	Belarusian – Belarus
bg–BG	Bulgarian – Bulgaria
ca–ES	Catalan – Catalan
zh–HK	Chinese – Hong Kong SAR (Default Sort Order – Stroke Count)
zh–HK	Chinese – Hong Kong SAR (Alternate Sort Order – Stroke Count)
zh–MO	Chinese – Macau SAR (Default Sort Order – Pronunciation)
zh–MO	Chinese – Macau SAR (Alternate Sort Order – Stroke Count)
zh–CN	Chinese – China (Default Sort Order – Pronunciation)
zh–CN	Chinese – China (Alternate Sort Order – Stroke Count)
zh–SG	Chinese – Singapore (Default Sort Order – Pronunciation)
zh–SG	Chinese – Singapore (Alternate Sort Order – Stroke Count)
zh–TW	Chinese – Taiwan (Default Sort Order – Stroke Count)
zh–TW	Chinese – Taiwan (Alternate Sort Order – Bopomofo)
hr–HR	Croatian – Croatia
cs–CZ	Czech – Czech Republic
da–DK	Danish – Denmark
div–MV	Dhivehi – Maldives
nl–BE	Dutch – Belgium
nl–NL	Dutch – The Netherlands
en–AU	English – Australia
en–BZ	English – Belize
en–CA	English – Canada
en–CB	English – Caribbean
en–IE	English – Ireland
en–JM	English – Jamaica
en–NZ	English – New Zealand
en–PH	English – Philippines
en–ZA	English – South Africa
en–TT	English – Trinidad and Tobago
en–GB	English – United Kingdom

RFC1766 Name	Language – Country/Region
en–US	English – United States
en–ZW	English – Zimbabwe
et–EE	Estonian – Estonia
fo–FO	Faroese – Faroe Islands
fa–IR	Farsi – Iran
fi–FI	Finnish – Finland
fr–BE	French – Belgium
fr–CA	French – Canada
fr–FR	French – France
fr–LU	French – Luxembourg
fr–MC	French – Monaco
fr–CH	French – Switzerland
mk–MK	FYRO Macedonian
gl–ES	Galician – Galician
ka–GE	Georgian – Georgia (Default Sort Order – Traditional)
ka–GE	Georgian – Georgia (Alternate Sort Order – Modern Sort)
de–AT	German – Austria
de–DE	German – Germany (Default Sort Order – Dictionary)
de–DE	German – Germany (Alternate Sort Order – Phone Book Sort DIN)
de–LI	German – Liechtenstein
de–LU	German – Luxembourg
de–CH	German – Switzerland
el–GR	Greek – Greece
gu–IN	Gujarati – India
he–IL	Hebrew – Israel
hi–IN	Hindi – India
hu–HU	Hungarian – Hungary (Default Sort Order)
hu–HU	Hungarian – Hungary (Alternate Sort Order – Technical Sort)
is–IS	Icelandic – Iceland
id–ID	Indonesian – Indonesia
it–IT	Italian – Italy

RFC1766 Name	Language – Country/Region
it–CH	Italian – Switzerland
ja–JP	Japanese – Japan (Default Sort Order)
ja–JP	Japanese – Japan (Alternate Sort Order – Unicode)
kn–IN	Kannada – India
kk–KZ	Kazakh – Kazakhstan
kok–IN	Konkani – India
ko–KR	Korean – Korea (Default Sort Order)
ko–KR	Korean – Korea (Alternate Sort Order – Korean Xwansung Unicode)
ky–KZ	Kyrgyz – Kazakhstan
lv–LV	Latvian – Latvia
lt–LT	Lithuanian – Lithuania
ms–BN	Malay – Brunei
ms–MY	Malay – Malaysia
mr–IN	Marathi – India
mn–MN	Mongolian – Mongolia
nb–NO	Norwegian (Bokml) – Norway
nn–NO	Norwegian (Nynorsk) – Norway
pl–PL	Polish – Poland
pt–BR	Portuguese – Brazil
pt–PT	Portuguese – Portugal
pa–IN	Punjabi – India
ro–RO	Romanian – Romania
ru–RU	Russian – Russia
sa–IN	Sanskrit – India
sr–SP–Cyrl	Serbian (Cyrillic) – Serbia
sr–SP–Latn	Serbian (Latin) – Serbia
sk–SK	Slovak – Slovakia
sl–SI	Slovenian – Slovenia
es–AR	Spanish – Argentina
es–BO	Spanish – Bolivia
es–CL	Spanish – Chile

RFC1766 Name	Language – Country/Region
es–CO	Spanish – Colombia
es–CR	Spanish – Costa Rica
es–DO	Spanish – Dominican Republic
es–EC	Spanish – Ecuador
es–SV	Spanish – El Salvador
es–GT	Spanish – Guatemala
es–HN	Spanish – Honduras
es–MX	Spanish – Mexico
es–NI	Spanish – Nicaragua
es–PA	Spanish – Panama
es–PY	Spanish – Paraguay
es–PE	Spanish – Peru
es–PR	Spanish – Puerto Rico
es–ES	Spanish – Spain (Default Sort Order – International)
es–ES	Spanish – Spain (Alternate Sort Order – Traditional)
es–UY	Spanish – Uruguay
es–VE	Spanish – Venezuela
sw–KE	Swahili – Kenya
sv–FI	Swedish – Finland
sv–SE	Swedish – Sweden
syr–SY	Syriac – Syria
ta–IN	Tamil – India
tt–RU	Tatar – Russia
te–IN	Telugu – India
th–TH	Thai – Thailand
tr–TR	Turkish – Turkey
uk–UA	Ukrainian – Ukraine
ur–PK	Urdu – Pakistan
uz–UZ–Cyrl	Uzbek (Cyrillic) – Uzbekistan
uz–UZ–Latn	Uzbek (Latin) – Uzbekistan
vi–VN	Vietnamese – Vietnam

APPENDIX G

Features supported in different versions of RSView

This appendix describes:

- which versions of RSView® ME Station™ are supported.
- which features are not supported in previous versions of RSView ME Station.

Which versions are supported

RSView® Studio™ allows you to create runtime (.mer) files for these versions of RSView ME Station:

- RSView ME Station version 4.0
- RSView ME Station version 3.2
- RSView ME Station version 3.1
- RSView ME Station version 3.0

Multiple version support is useful for system designers and others who create and modify applications for different versions of RSView ME Station on an ongoing basis. You can use the latest version of RSView Studio on a single development computer to provide applications for terminals that use previous versions of RSView ME Station.

To check which version of RSView ME Station you are using:

1. In RSView ME Station, click Terminal Settings.
2. Click System Information.
3. Click About RSView ME Station.

Creating runtime application files for previous versions

When you create the runtime application file (with the file extension .mer), you can specify the version of RSView ME Station for which to create the file. For example, if the application will run on a terminal that uses RSView ME Station version 3.2, you can specify that version for the .mer file.

If the application contains features that are not supported by the version you select, RSView displays a validation report that lists the unsupported features. The runtime application file is not created. You must remove or turn off the unsupported features before you can create the runtime application file.

For information about creating runtime application files, see Chapter 25.

The remainder of this appendix lists the features that are not supported in previous versions of RSView ME Station. The tables also show how to remove or replace the unsupported features.

Features that are not supported in version 3.2 or earlier

These version 4.0 features are not supported in version 3.2 of RSView ME Station, nor in earlier versions. The right column describes how to remove or replace the feature.

To remove or replace this feature	Do this
Unsupported RSLinx Enterprise feature or shortcut—warning	Delete or replace the feature or shortcut. A hardware patch might be available that allows you to use the feature. Therefore, the runtime application file will still be created.
Unsupported RSLinx Enterprise feature or shortcut—error	Delete or replace the feature or shortcut.
Global reference object that is not linked to a global base object	Delete the global reference object, or link it to a global base object. All global reference objects that are linked to global base objects will be converted to standard graphic objects in the .mer file.
Language switch button graphic object	Delete the button.
Password button graphic object	Delete the button.
RecipePlus button graphic object	Delete the button.
RecipePlus selector graphic object	Delete the selector.
RecipePlus table graphic object	Delete the table.
Acknowledge all alarms button with a filtered trigger	Clear the Filtered triggers box.
Print alarm history button with a filtered trigger	Clear the Filtered triggers box.
Print alarm status button with a filtered trigger	Clear the Filtered triggers box.

To remove or replace this feature	Do this
Clear alarm history button with a filtered trigger	Clear the Filtered triggers box.
Clear alarm history button with the Reset alarm status option cleared	Select Reset alarm status.

Features that are not supported in version 3.1 or earlier

The features listed in the table on page G-2 are not supported in version 3.1 of RSView ME Station. In addition, these version 3.2 alarm options are not supported in version 3.1, nor in earlier versions. The right column describes how to remove or replace the feature.

To remove or replace this feature	Do this
Alarm list graphic object with an unsupported combination of alarm conditions	Do one of the following: ▪ Select the Display check box for each alarm condition ▪ Select the Display check box for only these alarm conditions: ▪ Active and unacknowledged ▪ Inactive and unacknowledged.
Alarm list graphic object with Blink selected for one or more alarm conditions	For all alarm conditions that you are displaying, clear the Blink check box.
Alarm list graphic object with Use alarm colors turned off for one or more alarm conditions	For all alarm conditions that you are displaying, select the Use alarm colors check box.
Alarm list graphic object with the Acknowledged symbol column turned off	Select the Display column check box for the Acknowledged symbol column.
Alarm list graphic object with an Acknowledged symbol other than *	Change the Acknowledged symbol to *.
Alarm list graphic object with the Active symbol column displayed	Clear the Display column check box for the Active symbol column.
Alarm list graphic object with tags or expressions assigned to one or more connections	Clear the tags or expressions assigned to the connections.

To remove or replace this feature	Do this
Alarm list graphic object with Selected alarm indicator set to Cursor	Change the Selected alarm indicator to Highlight bar.
Alarm list graphic object with Lines per alarm set to >1	Change the Lines per alarm to 1.
Alarm banner graphic object with Use alarm colors selected	Clear the Use alarm colors check box.
Alarm status list graphic object with Use alarm colors selected	Clear the Use alarm colors check box.
Alarm status list graphic object with Lines per alarm set to >1	Change the Lines per alarm to 1.
Alarm status list graphic object with a Fore color other than white	Change the Fore color to white.

Features that are not supported in version 3.0

The features listed in the previous tables are not supported in version 3.0 of RSView ME Station. In addition, these version 3.1 features are not supported in version 3.0. The right column describes how to remove or replace the feature.

To remove or replace this feature	Do this
Piloted control list selector graphic object	Delete the object. Try using a control list selector instead.
Alarm banner graphic object with a filtered trigger	Clear the Filtered triggers box.
Alarm trigger with Message Notification connection assigned	Clear the tag or expression assigned to the connection.
Alarm trigger with Message Handshake connection assigned	Clear the tag or expression assigned to the connection.

Index