

MQSeries® Integrator

# User's Guide

Version 1.0

**Note:** Before using this information and the product it supports, be sure to read the general information under “Notices” on page 137.

**First edition (January 1999)**

This edition applies to IBM® MQSeries Integrator, Version 1.0 and to all subsequent releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of the product.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

At the back of this publication is a page titled “Sending your comments to IBM”. If you want to make comments, but the methods described are not available to you, please address them to:

IBM United Kingdom Laboratories  
Information Development,  
Mail Point 095,  
Hursley Park,  
Winchester,  
Hampshire,  
England,  
SO21 2JN.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright New Era of Networks, Inc., 1998, 1999. All rights reserved.

© Copyright International Business Machines Corporation, 1999. All rights reserved.

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

---

# Contents

---

<b>Chapter 1: Introduction .....</b>	<b>1</b>
Overview .....	2
MQSeries .....	2
NEONFormatter.....	2
NEONRules .....	2
MQSeries Integrator Rules Daemon .....	3
Product Documentation Set .....	4
Documentation Conventions .....	4
Supported Platforms and Compilers .....	5
<b>Chapter 2: NEONFormatter .....</b>	<b>7</b>
Starting Formatter .....	8
The Formatter Window .....	10
Opening a New Formatter Window .....	11
Tree Functionality .....	11
Using Drag and Drop .....	11
Case Sensitivity .....	14
Formatter Menus.....	15
The Formatter Menu Bar.....	15
The File Menu .....	15
The Edit Menu .....	16
The Tools Menu.....	16
The Window Menu.....	16
The Help Menu.....	16
Formatter Popup Menus.....	17
Left Pane Formatter Popup Menus .....	17
Right Pane Popup Menu.....	19
Formatter Toolbar .....	20
New.....	20
Find .....	20
Print Tree.....	21
Help Topics .....	22
Formatter Features.....	22
Used In.....	22
Export .....	23
Import .....	24
Building Formats.....	27
Literals .....	27
Creating a Literal.....	28
Fields.....	30
Creating a Field .....	30
Input Controls .....	31
Creating an Input Control .....	32

Year 2000 Compliance .....	43
Output Controls .....	44
Creating an Output Control .....	44
Output Operations.....	56
Output Operation Collections.....	57
Math Expression.....	58
Operation Substitute Strings .....	60
Formats.....	62
Creating a Format .....	63
Creating a Flat Input Format.....	64
Creating a Compound Input Format .....	65
Creating a Flat Output Format.....	69
Access Modes .....	70
Field Map .....	72
Creating a Compound Output Format .....	73
User-Defined Data Types.....	75
Creating a User-Defined Data Type.....	75
Using a User-Defined Data Type in an Input Control.....	76
The Validation Parameters Property Sheet .....	77
Alternative Input and Output Formats .....	78
Optional Components and Fields.....	78
Alternative Input Formats .....	78
Alternative Input Formats and Parsing.....	79
Alternative Output Formats.....	80
Tagged Input Formats.....	80
Tagged Input with Alternative Component Example .....	80
Alternative Output Format Example .....	82
<b>Chapter 3: NEONRules .....</b>	<b>83</b>
Starting Rules.....	84
The Rules Window .....	86
Opening a New/Additional Rules Window .....	87
Tree Functionality .....	87
Tabbed Property Sheets .....	87
Security .....	88
Case Sensitivity .....	88
Rules Menus .....	89
The Rules Menu Bar .....	89
The File Menu.....	90
The Insert Menu .....	90
The Tools Menu.....	90
The Window Menu.....	91
The Help Menu.....	91
Rules Popup Menus.....	91
Left Pane.....	91

Right Pane .....	92
Rules Toolbar .....	93
Building Rules .....	94
Application Groups .....	94
Adding an Application Group.....	94
Message Types.....	95
Adding a Message Type.....	95
Rules.....	97
Adding a Rule.....	97
Deleting a Rule .....	98
Duplicating a Rule .....	99
Renaming a Rule .....	100
Rule Security.....	100
Expressions .....	102
Expression Components .....	103
Field List.....	103
Rules Operators.....	103
Values .....	106
Functions.....	106
Creating or Modifying an Expression.....	107
Deleting an Expression.....	108
Subscriptions .....	108
Adding a Subscription .....	109
Duplicating a Subscription.....	110
Renaming a Subscription.....	110
Deleting a Subscription.....	111
Adding a Comment to a Subscription .....	111
Subscription Security.....	112
Actions.....	113
Adding an Action.....	113
Adding Multiple putqueue Options .....	115
Deleting an Action .....	117
Conditional Branching .....	118
<b>Appendix A: ASCII Extended Character</b>	
<b>Set.....</b>	<b>121</b>
<b>Appendix B: EBCDIC Character Set .....</b>	<b>127</b>
<b>Appendix C: Notices.....</b>	<b>137</b>
Trademarks and Service Marks .....	139
<b>Glossary .....</b>	<b>141</b>
<b>Index.....</b>	<b>149</b>



---

## Chapter 1

# Introduction

---

This user guide describes MQSeries Integrator. It is organized into the following sections:

Chapter 1, *Introduction*, provides documentation conventions, available documentation, and a brief overview of MQSeries Integrator.

Chapter 2, *NEONFormatter*, describes the Formatter module and provides procedures to create, change, and delete both input and output formats.

Chapter 3, *NEONRules*, describes the Rules Engine and provides the procedures to create, change, and delete a rule.

Appendix A, *ASCII and Extended Character Set*, contains an ASCII and extended character set table.

Appendix B, *EBCDIC Character Set*, contains an EBCDIC character set table.

Appendix C, *Notices*, contains trademark and service mark information.

A Glossary listing terms used in describing MQSeries Integrator.

# Overview

MQSeries Integrator provides the flexibility and scalability that allows true application integration. MQSeries Integrator consists of four components:

- MQSeries
- NEONFormatter
- NEONRules
- MQSeries Integrator Rules daemon

## MQSeries

MQSeries is message-oriented middleware that is ideal for high-value message handling and high-volume applications because it guarantees that each message is delivered only once, and it supports transactional messaging. Messages are grouped into units of work and either all or none of the messages in a unit of work are processed. MQSeries coordinates message work with other transaction work, like database updates, so data integrity is always maintained.

## NEONFormatter

NEONFormatter translates messages from one format to another.

Formatter handles multiple message format types from multiple data value sources with the ability to convert and parse messages. Messages can be converted from any described format to any other described format. If fields in input data formats are missing, you can set up defaults for those fields on output. When a message is provided as input to Formatter, the message is parsed and data values are returned. Formatter can handle virtually any message format, including fixed, for example, COBOL records, delimited, for example, C null delimited strings, and variable, tagged, delimited, repetitive and recursive formats, for example, S.W.I.F.T. messages.

Defining message formats in the Formatter database is done through the graphical user interface (GUI). The GUI leads you through the definitions of format components, for example, tags, delimiters, and patterns, to the building of complete message definitions.

## NEONRules

NEONRules lets you develop rules for managing message destination IDs, receiver locations, expected message formats, and any processes initiated upon message delivery. The creation and dispatch of multiple messages to multiple destinations from a single input message is supported, and different formats and transport methods for each are allowed. The dynamic nature of



the Rules Engine means that rules can be effective immediately, staged over time, or delayed, depending on how the reload messages are timed, which allows flexibility in rapidly changing environments.

Rules can examine the value of any field or group of fields in a message to make its determinations. It can aggregate conditions with the Boolean AND and OR operators without architectural limits as to the number or complexity of the expressions.

For more in-depth descriptions of the Formatter and Rules modules, refer to the overviews in *NEONFormatter* on page 7 and *NEONRules* on page 83.

## **MQSeries Integrator Rules Daemon**

The MQSeries Integrator Rules daemon combines MQSeries, Formatter, and Rules in a generic server process. The MQSeries Integrator Rules daemon processes messages from an MQSeries input queue, uses Formatter to parse messages, uses Rules to determine what transformations to perform and where to route the messages, and then puts the output messages on MQSeries queues for delivery to applications.

# Product Documentation Set

The MQSeries Integrator documentation set includes:

- ***MQSeries Integrator Installation and Configuration Guide*** helps end users and engineers install and configure MQSeries Integrator.
- ***MQSeries Integrator User's Guide*** helps MQSeries Integrator users understand and apply the program through its graphical user interfaces (GUIs).
- ***MQSeries Integrator System Management Guide*** is for system administrators and database administrators who work with MQSeries Integrator on a day-to-day basis.
- ***MQSeries Integrator Application Development Guide*** assists programmers in writing applications that use MQSeries Integrator APIs.
- ***Programming References*** are intended for users who build and maintain the links between MQSeries Integrator and other applications. The documents include:
  - ***MQSeries Integrator Programming Reference for NEONFormatter*** is a reference to Formatter APIs for those who write applications to translate messages from one format to another.
  - ***MQSeries Integrator Programming Reference for NEONRules*** is a reference to Rules APIs for those who write applications to perform actions based on message contents.

## Note

For information on message queuing, refer to the ***IBM MQSeries*** documentation.

---

## Documentation Conventions

### Tip

Tips point out shortcuts or procedures that can help you use MQSeries Integrator more effectively.

---

### Note

Notes point out useful extra information.

---

### WARNING!

Do not ignore anything associated with a warning—it alerts you to something that can cause loss of, or damage to your work.

---

## Supported Platforms and Compilers

<b>Operating System</b>	<b>DBMS</b>	<b>Compiler</b>
Windows NT 4.0	DB2 5.0 Oracle 7.3 Oracle 8 SQL Server 6.5 Sybase Client 11.1.1 Sybase Server 11.03, 11.5	Microsoft Visual C++ version 4.2
Solaris 2.5.1, 2.6	DB2 5.0 Oracle 7.3 Sybase Client 11.1.1 Sybase Server 11.03, 11.5	Sparcworks C++ compiler version 4.0
HP-UX 10.20	DB2 5.0 Oracle 7.3 Oracle 8 Sybase Client 11.1.1 Sybase Server 11.03, 11.5	HP C++ version 10.34
AIX 4.2	DB2 5.0 Oracle 7.3 Sybase Client 11.1.1 Sybase Server 11.03, 11.5	IBM C Set ++ version 3.1.4



---

## Chapter 2

# NEON **Formatter**

---

NEONFormatter translates messages from one format to another, simplifying message exchange between different systems and applications. Formatter standardizes how message formats are described, eliminating the need to custom-code between applications to translate each others' messages.

Formatter breaks complex messages into simple messages for processing, handling anything from formats containing fixed, tagged, or delimited fields to multi-part nested formats.

Through Formatter's use of tables, you only have to describe a message format once. The components of the format can be reused in multiple formats. After that, conversion between formats is dynamic. The Formatter GUI simplifies format definition and configuration.

Formatter can reformat messages at different points in the communication process. For example, a message can be reformatted upon receipt or when it's sent to its next destination. The exact points vary based on your configuration.

### ***Control Tables***

Formatter uses control tables within the MQSeries Integrator database to recognize and parse input and output messages. These tables describe message formats and format components. Formatter uses this information to interpret values from incoming messages and dynamically construct outgoing messages.

Formatter can:

- Interpret incoming messages
- Add, remove, or rearrange message data
- Insert literals into output as headers or trailers
- Perform arithmetic operations to format numeric data
- Execute user-written functions to format fields

### ***Operating Modes***

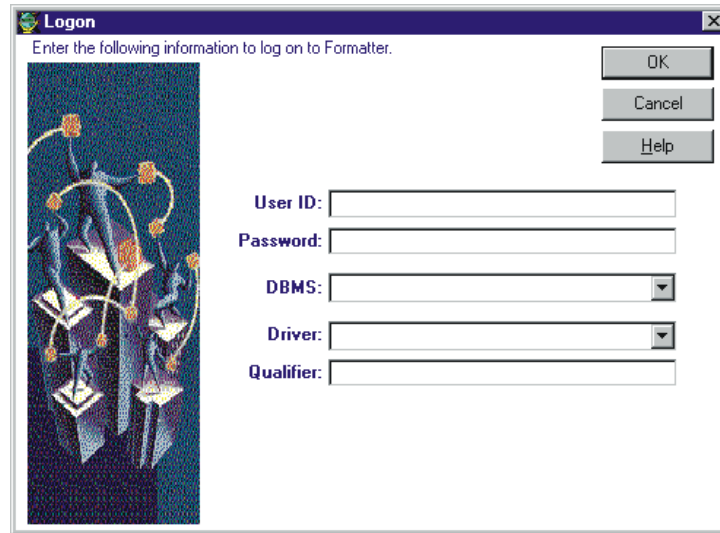
Formatter operates in two modes - parsing and converting.

- When parsing a message, Formatter breaks the message into its component parts for easy access to message data. For example, this feature is used internally when Rules needs data values.
- When converting messages, Formatter parses the input message and formats the output message data based on a different output format.

# Starting Formatter

To access Formatter:

1. Double-click the Formatter icon. The Formatter splash screen appears, quickly followed by the Logon dialog box.



*Formatter Logon Dialog Box*

## Note

If all the defaults are correct, click the Password field, type your password, and either press *Return* or choose the OK button.

2. Click the User ID field and choose one of the following:
  - Press *Tab* to accept the User ID (if one is displayed).
  - Type your User ID and press *Tab*.
3. Type the password associated with the User ID and press *Tab*.
4. Choose one of the following:
  - Press *Tab* to accept the server (if one is displayed).
  - Type the server name you want to connect to and press *Tab*.
5. Type the name of the desired database and press *Tab*.

## Note

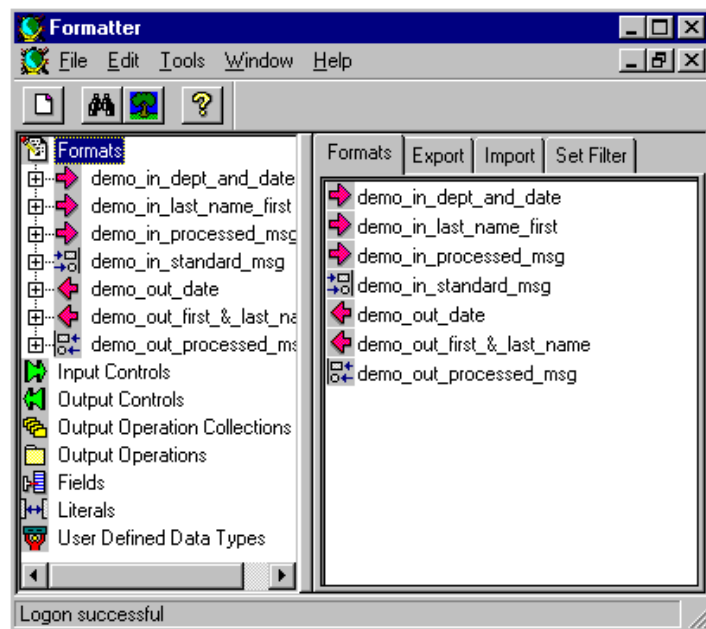
Oracle users should leave the database field blank.

6. Choose one of the following:
  - To accept the DBMS currently displayed, skip to step 7.
  - Click the down arrow to open a drop-down list and select the desired DBMS.

### Note

The DBMS client appropriate for your DBMS must be installed before you can run the GUI. If you do not know the DBMS you need to connect to, ask your System Administrator.

7. Either choose the OK button or press *Return*. The Formatter window appears, displaying Formatter resources.



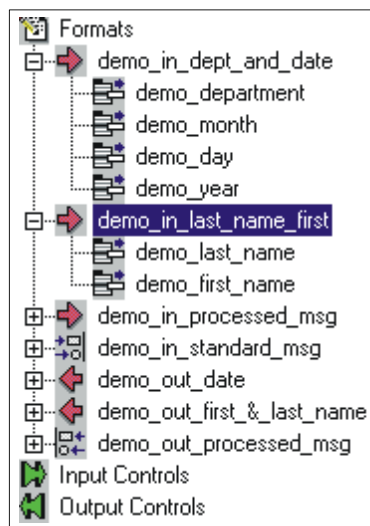
*Formatter Main Window*

## The Formatter Window

The main Formatter window is designed to work like Microsoft Explorer in Microsoft Windows NT and Windows 95.

Format components are displayed in a tree-structured or hierarchical organization in the left pane. The right pane shows the contents of the currently selected root object.

To expand a root object, for example Formats, double-click it. The small boxes with a plus sign (+) next to the objects in the left pane indicate that there may be other items subordinate to the object, but the subordinate items are not displayed. Click the plus sign to expand the tree to display the subordinate items. When the tree is expanded, the plus sign changes to a minus sign (-).



*Expanded Tree*

When you select a component in the left pane, the right pane of the window then displays detailed information about the selected component. The detailed information is kept in property sheets labeled with tabs. A property sheet works like a dialog box; you must fill in information in mandatory fields. You can see the properties of a component and change components using property sheets. To collapse the tree, click the minus sign.



## Opening a New Formatter Window

You can open more than one Formatter window at a time. To open a new window, choose File→New from the menu bar.

### Note

When you create a new window, you do not create a new instance of the database. You are always connected to the same database.

### Tip

#### To Simplify Creating Formatter Components:

- When building Input Controls, you will want to have two windows available. One window should be open to the new Input Control and the other should be open to Fields.
- When building Output Controls, you will want to have four windows available. One for Fields, one for Output Operations, one for Output Operation Collections, and one for the new Output Control.
- When building Flat Input Formats, you will want to have two windows available. One window for the new Format and one for Input Controls to be associated with input Fields.
- When building Flat Output Formats, you'll want to have two windows available. One window for the new Format and one for Output Controls to be associated with output Fields.

## Tree Functionality

Depending on the component or element selected, when you click the right mouse button, a popup menu appears containing commands such as New, Delete, Duplicate, and Add Field Components. For more information on popup menus, refer to the section *Formatter Popup Menus* on page 17.

## Using Drag and Drop

You can use the drag and drop feature to:

- Copy a subordinate component from one window into the related detail of another window
- Create output formats in field mapping (from the Field Map tab in a flat, output format)
- Reorder fields (in flat input and output formats)

Drag and drop works across both windows and panes. To copy an object using drag and drop, drag the object from the first window's tree to the second window's tabs. You cannot drag and drop in the same pane unless you are either reordering fields or using field mapping.

You can only drag and drop at the root level. When you drop an object, it is always added to the end of the list. Note that you cannot drag and drop literals.

When a flat input or output format or output operation collection is changed (by either adding or deleting a field), it is a global change. Any compound format the flat format is contained in also reflects the addition or deletion of the field.

## Note

- For more information on reordering fields, refer to the *Using Drag and Drop to Reorder Fields* section on page 13.
- For more information on field mapping, refer to the *Field Map* section on page 72.

In the following table, the column headings list the format types and the row headings list the format components that you can drag and drop.

An x in an intersecting cell indicates that the component can be dropped into the format type.

	Format Types			
	Input Flat Format	Input Compound Format	Output Flat Format	Output Compound Format
Input Flat Format		X		
Input Compound Format		X		
Output Flat Format				X
Output Compound Format				X
Input Controls	X			
Output Controls			X	
Fields	X		X	
Literals				

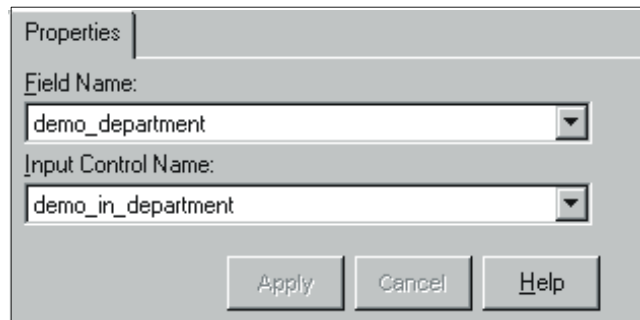
Note that drag and drop cannot be used with literals.

### ***Assigning a Field to an Input Control that was Dragged and Dropped***

When you use drag and drop to add an input control to a flat input format, the input control component that you drop into the format is named NONE.

To name the component:

1. Close and reopen the format and select NONE. The property sheet for the field appears.



*Input Field's Property Sheet*

2. Click in the Field Name field to open the drop-down list and select the field.
3. Click in the Input Control Name field to open the drop-down list and select the input control.

### ***Using Drag and Drop to Field Map***

When you are creating flat output formats using the field mapping tab, you can drag and drop an input field into an output field. For more information on field mapping, refer to the *Field Map* section on page 79.

### ***Using Drag and Drop to Reorder Fields***

From the Fields tab within a flat input or flat output format, you can use drag and drop to change the order of the fields (in the right pane). When you drop an object from another pane, it is always added to the end of the list. If field order is ordinal (you want the fields to appear in the specified order), you may want to change the order in which they appear in the Field list to map to an existing format from either a sending or receiving application.

To change the order of the field names, select the field that you want to move, hold down the left mouse button, move that field on top of the field that you want it to appear above, and release the mouse button. The field is now repositioned in the list.

To reposition a field in a list that is longer than the pane, in the right pane, press the right mouse button to open the popup menu and choose View→Details to rearrange the fields into columns. You can then select the field you want to reposition and drag it on top of the field you want it to be above.

## Case Sensitivity

### **WARNING!**

If you are using a case-insensitive database, you cannot name components the same with only a change in case to identify them. For example, you cannot name one format "f1" and another format "F1". In a case-insensitive environment, make each item unique using something other than case differences.

If importing components exported from a context-sensitive database into a context-insensitive database, these differences will cause NNFie to fail during import if a conflict arises between two components named the same with only case differences. See the *MQSeries Integrator System Management Guide* for information on using NNFie.

See the *MQSeries Integrator System Management Guide* for information on how to change a case-insensitive installation to case sensitive.

---

# Formatter Menus

Formatter features both a menu bar and popup menus. The menu bar contains items that let you open, close, and arrange windows, display and move the toolbar, exit Formatter, and view Formatter version information.

The popup menu lets you create, delete, and duplicate format components.

Formatter menus can also be accessed using your keyboard by pressing *Alt* plus the letter underlined in the menu name. For example, you can open the File menu by pressing *Alt+F*. After the menu is opened, you can access its functions by pressing the underlined letter of the function. For example, from the File menu, to open a new Formats window, type N (the underlined letter in the function, New).

## The Formatter Menu Bar

The Formatter menu bar contains the following menus:

- File
- Edit
- Tools
- Window
- Help

### Note

If only the Formatter window is open, only the File and Help menus appear.

## The File Menu

The File menu contains the following submenus:

- New
- Close
- Print Tree
- Exit

These menu are described in detail in the following sections.

### **New**

The New function opens a new Formatter window. The new window connects to the same database as the previous window. To open a new window, choose File→New.

### **Close**

The Close function closes the active window. To close the active window, choose File→Close.

### ***Print Tree***

The Print Tree function gives you a detailed display of the components of a given format. To print the tree, choose the Print button in the lower right-hand corner of the tree window.

### ***Exit***

The Exit function closes Formatter. To exit Formatter, choose File→Exit.

## **The Edit Menu**

The Edit menu contains the following submenus:

- Cut
- Copy
- Paste
- Clear
- Find

### ***Cut, Copy, Paste***

The Cut, Copy, and Paste functions let you cut text in fields or copy text from one field and paste it into another. For example, you could copy the Tag Value field data from the Input Control property sheet and paste it in the Input Tag Value field of the Output Control property sheet.

### ***Clear***

The Clear function deletes the selected text from a user-entry field.

### ***Find***

The Find function lets you search for format, input control, output control, field, literal names, output operations, and output operation collections. For the procedure to use this function, refer to the *Find* section on page 20.

## **The Tools Menu**

The Tools menu contains the Customize Toolbars function. You can turn the toolbar either off or on, position it in your window and select the button size.

## **The Window Menu**

The Window menu lets you cascade, tile either vertically or horizontally, or layer windows. It also lists the open windows of the application.

## **The Help Menu**

The Help menu contains Help Topics and About submenus. Use the Help Topics submenu to access Formatter online help. The About submenu displays MQSeries Integrator copyright and version information plus database, DBMS, and server information.

# Formatter Popup Menus

Both the right and left window panes contain popup menus. These menus are selected by pressing the right mouse button.

## Left Pane Formatter Popup Menus

Depending on what is selected in the left window pane, when you press the right mouse button, a popup menu appears. The popup may contain any of the following functions:

- New
- Delete
- Duplicate
- Save as Output
- Add Field Components
- Add Output Operations
- Add Substitute Items
- Expand Items
- Collapse Items

<b>Menu Item</b>	<b>Description</b>	<b>Associated Components</b>
New	Creates a new component of the component category type selected. The New function operates on the root level of the associated components.	Formats*, Input Controls, Output Controls, Output Operation Collections, Output Operations, Fields, Literals, or User-Defined Data Types. An output operation is one of the following: Default, Length, Math Expressions, Prefix/Suffix, Substitute, Substring, Trim, User Exit.  *The New menu item initiated on the Formats item leads to two submenus. In the first submenu, choose either Compound or Flat, depending on which type of format you want to construct. In the last menu, choose either Input or Output, further defining your new format.
Delete	Deletes the selected component at the parent level.	Formats, Input Controls, Output controls, Output Operation Collections, Output Operations, Fields, Literals, User- Defined Data Types

<b>Menu Item</b>	<b>Description</b>	<b>Associated Components</b>
Duplicate	Duplicates the selected component at the parent level and enables you to change the name of the duplicate.	Formats, Input Controls, Output Controls, Output Operation Collections, Output Operations, Fields, Literals
Save as Output	Saves an input control as an output control. See <i>Saving an Input Control as an Output Control</i> on page 43 for details.	Input Controls
Add Field Components	Opens the Field window for an easy way to add fields to a new or existing flat input or output format. Select the fields you want added. Choose Accept Selection. See <i>Creating Flat Input Formats</i> (page 64) or <i>Creating Flat Output Formats</i> (page 69) for details. <b>Note:</b> Fields are always added in alphabetical order to the end of any existing list of fields for the format.	Flat Input or Output Formats
Add Output Operations	Opens the Output Operations window for an easy way to add output operations or other collections to a new or existing collection. Select the operations you want to add. Then choose Accept Selection. <b>Note:</b> Operations and other collections are always added in alphabetical order to the end of any existing list of operations for the collection.	Output Operation Collections
Add Substitute Items	Opens the Literal window for an easy way to add literals as values to be substituted. Select the literals you want to add. Then choose Accept Selection. See <i>Operation Substitute Strings</i> on page 60. <b>Note:</b> Literals will always be added in alphabetical order to the end of any existing list of literals for the substitute.	Substitute Operations



<b>Menu Item</b>	<b>Description</b>	<b>Associated Components</b>
Expand Item	Displays all items subordinate to the associated component. Expand Item is associated with components on both the root and parent levels.	Formats, Input Controls, Output Controls, Output Operation Collections, Output Operations, Fields, Literals, and User-Defined Data Types. Includes the following items from Output Operations: Default, Length, Math Expressions, Prefix/Suffix, Substitute, Substring, Trim, and User Exit.
Collapse Item	Hides all items subordinate to the associated component.	Formats, Input Controls, Output Controls, Output Operation Collections, Output Operations, Fields, Literals, and User-Defined Data Types. Includes the following items from Output Operations: Default, Length, Math Expressions, Prefix/Suffix, Substitute, Substring, Trim, and User Exit.

## Right Pane Popup Menu

The right window pane contains a popup menu when a component list is displayed. This popup menu contains the following functions:

- View
- Arrange Icons

### ***View***

The View function lets you view either large or small (the default) icons. You can also select to arrange the icons in a list (the default).

If you select to view large icons, they are displayed from left to right, rather than in a list, and the title of the property sheet is no longer displayed. To see the icons in a list and once again display the property sheet's name, select the Details function.

### ***Arrange Icons***

The Arrange Icons function lets you change the order of the icons. By default, the icons are alphabetically displayed from a-z. When you select Arrange Icons→by (root-level name), the icons are then arranged alphabetically, from z-a. You can switch between these two alphabetic displays by reselecting Arrange Icons→by (root-level name).

## Formatter Toolbar

The Formatter toolbar contains the New, Find, Print Tree, and Help Topics functions.

### New

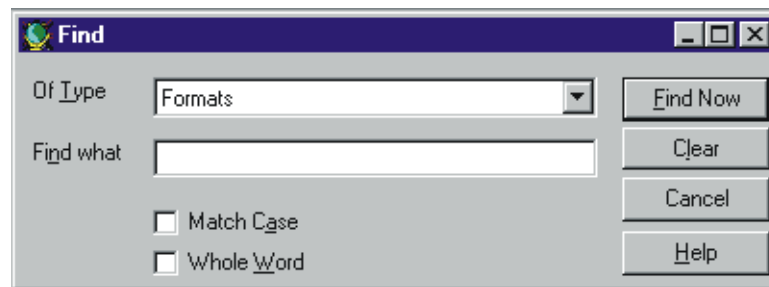
The New function opens a new Formatter window. The new window connects to the same database as the previous window.

### Find

The Find function lets you search for format, input control, output control, field, literal names, output operations, and output operation collections.

To perform a search:

1. Choose either the Find button or Edit→Find. The Find dialog box appears.



*Find Dialog Box*

2. Click the Of Type field to open the drop-down list and select the item type.
3. Click the Find What field and type either the name or partial name of what you want to search for.

#### Note

An asterisk can be used as a wildcard in your search.

4. To match the case of the name in the Find what field, select the Match Case checkbox.

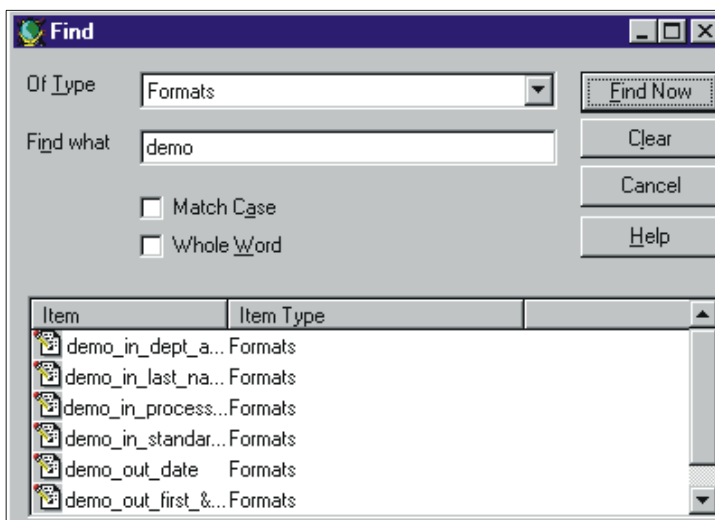
#### Note

If you are using a case-insensitive database, this option may not work, depending on how you named your data.

5. To search for all names that exact match the name in the Find What field, select the Whole Word checkbox.

If you select both Match Case and Whole Name, the literal value of the Find What field is searched on.

6. To begin the search, choose the Find Now button. The search is performed and the results are displayed at the bottom of the dialog box.



*Find Dialog Box with the Search Results Displayed*

7. To go to one of the items (in the example window above, a format) found by the search, double-click the name in the Item Name column. A new window appears displaying the item's placement in the tree. The item is highlighted.
8. To clear the fields to conduct another search, choose the Clear button.
9. To close the dialog box, choose the Cancel button.

## Print Tree

Use the Print Tree function to view the subordinate items of a selected component within the Print Tree window.

To use Print Tree:

1. Select a component from the left pane.
2. Choose either the Print Tree button or File→Print Tree. The Print Tree window appears, displaying the tree structure of the component.
3. To print the component's hierarchy, choose the Print button.
4. To view another format, select the Close button in the lower right-hand corner of the tree window to close the Print Tree window, select a different format, and reopen the Print Tree window.

## Help Topics

Use the Help Topics window to access on-line help from the Formatter toolbar. For on-line information on how to build formats, select the Help Topics icon and then choose the How Do I... button. To use context-sensitive help, select the item you want to view help for and press F1.

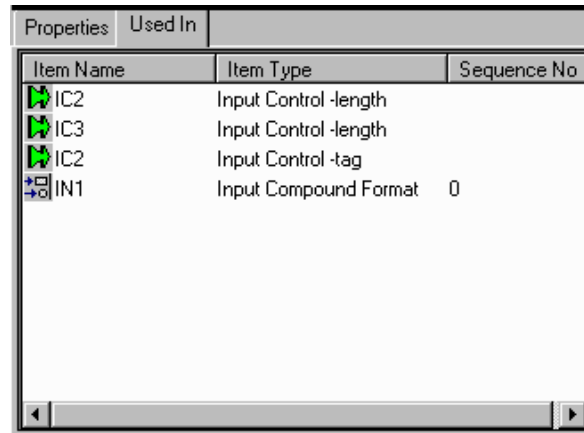
## Formatter Features

### Used In

The Used In function enables you to display a list of places the selected literal, input control, output control, field, format, operation, operation collection, or user-defined data type is used.

To see where a selected value is used:

1. Open the property sheet for the component that you want to look at.
2. Select the Used In tab. The Used In list appears.



Item Name	Item Type	Sequence No
IC2	Input Control -length	
IC3	Input Control -length	
IC2	Input Control -tag	
IN1	Input Compound Format	0

*Used In Property Sheet*

### Note

A literal can be used in many different ways, for example, as a pad character, suffix, or prefix. Therefore, the Used In list for literals may be extensive.

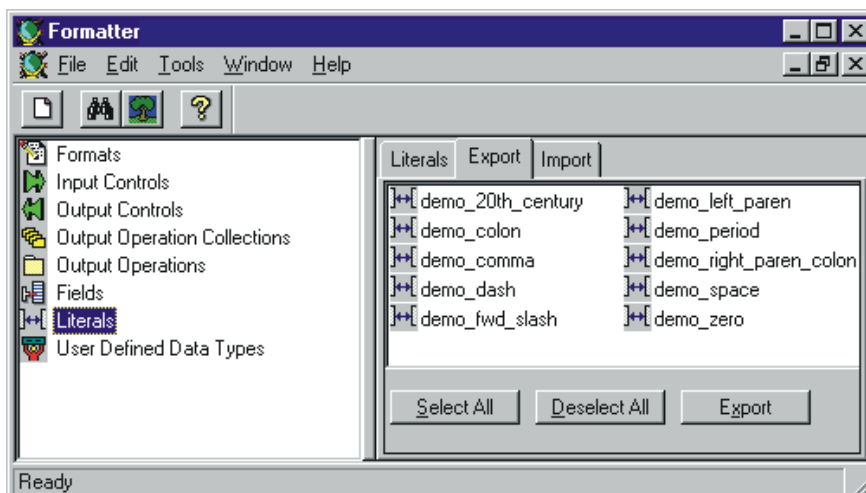
## Export

The Export function enables you to export one or more format objects. You can only export one type at a time. For example, you could export input controls and then separately export the delimiters of a format. Export is available from each root object, excluding Output Operations Collections.

The components of a format can be exported in any order. To export a complete format, be sure to export any contained Literals, Output Operations, Output Operation Collections, User-Defined Data Types, Fields, Input Controls, Output Controls, Flat Input or Output Formats, and Input or Output Compound Formats.

To export components:

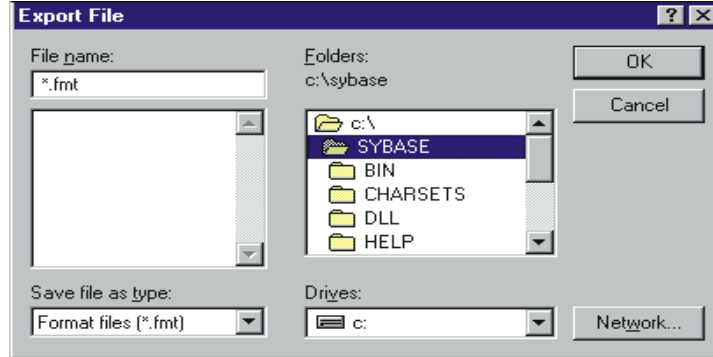
1. With the property sheet open for the component that you want to export, select the Export tab. The Export dialog box appears. (The example dialog box below displays literals; all components have a similar dialog box and are exported in the same way.)



*Export Dialog Box*

2. Choose one of the following:
  - Click each component that you want to export.
  - To select all of the available components, choose the Select All button.
  - To deselect previously selected components, choose the Deselect All button.

3. Choose the Export button. The Export File dialog box appears.



*Export File Dialog Box*

4. Select the drive and directory that you want to export the components to and choose the OK button. The selected components are exported and the dialog box closes.

## **WARNING!**

GUI exported files and NNFile exported files cannot be used interchangeably.

## **Import**

### **WARNING!**

If you are importing components exported from a context-sensitive database into a context-insensitive database, these differences will cause the import to fail if a conflict arises between two components named the same with only case differences.

The Import function enables you to import components to a database that you previously exported from another database. When you use the GUI export function to import a complete format, you must import each of its components separately. The Import function is available from each root object.

A compound component that contains another compound component must have the compound component it contains imported first. For example, the compound A12 within the compound B1 must be defined before compound B1 can be imported. Also, import flat formats before you import compound formats.

To import a complete format, import the components in the following order:

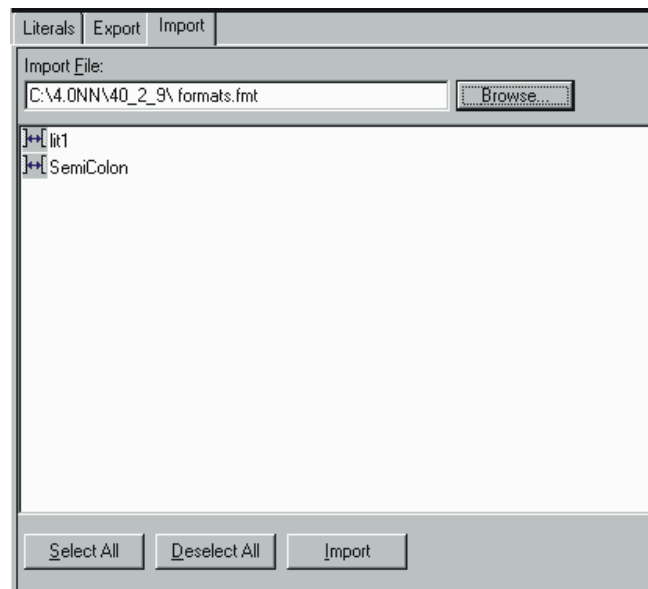
1. Literals
2. Output Operations
3. Output Operation Collections
4. User-Defined Data Types
5. Fields
6. Input (parse) Controls
7. Output Format Controls
8. Flat Input Formats
9. Flat Output formats
10. Compound Input Formats
11. Compound Output Formats

## WARNING!

GUI exported files and NNFile exported files cannot be used interchangeably.

To import formats:

1. With the property sheets open for the component that you want to import, select the Import tab.



*Import Property Sheet Window*

2. Click the Browse button. The Import File window appears.
3. Select the drive and directory. Then select the file that you want to import and choose the OK button.

4. Choose one of the following:
  - Click each component that you want to import.
  - To select all components, choose the Select All button.
  - To deselect previously selected components, choose the Deselect All button.
5. To import the components, choose the Import button.



# Building Formats

The easiest way to build formats is to create the root elements first. Define the literals, input controls, output controls, fields, and formats. The root elements are all created essentially the same way.

To create a flat input format, first define:

- Literals (for use as delimiters, tags)
- Fields
- Input Controls
- Flat Input Format

To create a flat output format, first define:

- Literals
- Fields
- Output Controls
- Flat Output Format
- Output Operations and/or Output Operation Collections

The following sections explain each root object and provide the procedures to create them.

## Literals

A literal is a string of characters that Formatter processes as:

- delimiters
- format terminators
- repeating sequence terminators
- default values
- trim characters
- prefixes and suffixes
- pad characters
- substitute values
- comparison values
- tag values

### Note

When literals are used as pad characters, only the first character is used.

Formatter uses literals in input controls, output controls, and operations, as format terminators, and repeating sequence terminators.

## Creating a Literal

To create a new literal:

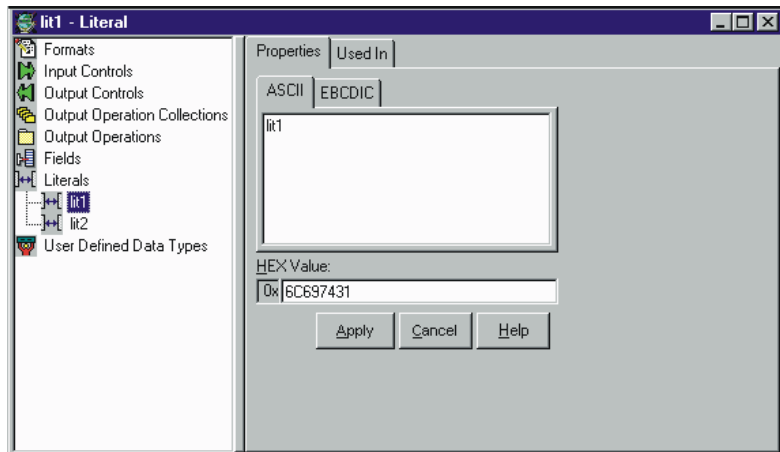
1. In the left pane of the Formatter window, select the Literals component category and right-click. A popup menu appears with New as a menu item.
2. Select New. A new literal component is added to the left pane and the Literal property sheet is displayed in the right pane. The cursor is positioned in the text box where you must type a new, unique name (maximum of 32 characters).

### Note

Single quotes cannot be used in names.

3. Click any mouse button outside the name text box or press *Return* to finish defining the literal. The name is highlighted and moves to its alphabetical location in the list.

- In the property sheet, enter the value for the literal. Both the ASCII and Hex values are automatically filled in when you type either the ASCII or Hex value. Note that the ASCII and EBCDIC fields have a 127-character maximum (which equals a 254-character hex value).



*Literal Property Sheet*

### Note

Refer to Appendix A for ASCII extended character set information.

### Tip

You can change the name of a literal by selecting the name of the literal, then click on the name again. A box appears around the selected literal name and the name is highlighted. Type the new name and press *Return* to enter the name and position it in the list.

### Note

You cannot drag and drop literals.

## Fields

Fields are named items of message data and are the smallest possible container for information. In Formatter, the field name is the link between the input and output data. Each field can be used in multiple formats and associated with different input and output controls. To identify a field in a message, a combination of the field name and a control is used.

### Note

Enter a unique name for each format component. Component names can be no greater than 32 characters.

## Creating a Field

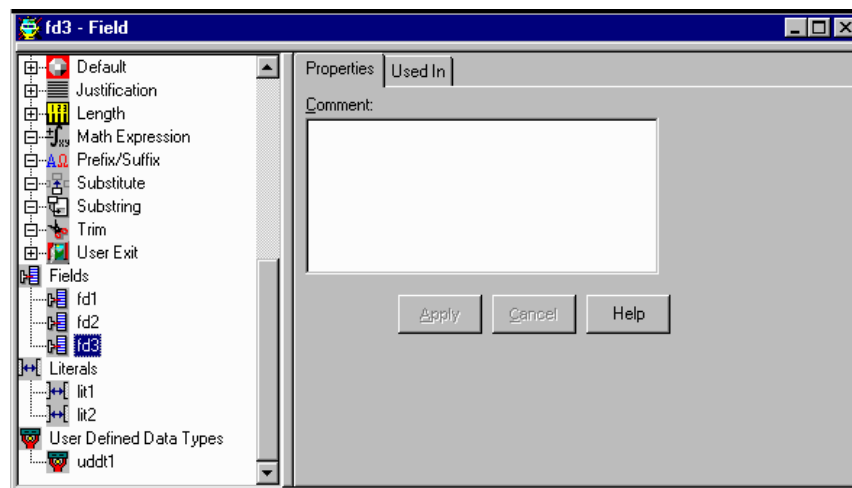
To create a field:

1. In the left pane of the Formatter window, select the Fields component category and right-click. A popup menu appears with New as a menu item.
2. Select New. A new field component is added to the left pane and the fields property sheet is displayed in the right pane. The cursor is positioned in the text box where you must type a new, unique field name (maximum of 32 characters).

### Note

Names with quotes may only have either single or double quotes, not both.

Field names with spaces or underscores must be surrounded by single or double quotes. A field with the name Field\_1 could be used as "Field\_1" or 'Field\_1' (but not "Field\_1" or 'Field\_').



*Fields Property Sheet*

3. Click any mouse button outside the name text box or press *Return* to finish defining the field. The field name moves to its alphabetical location in the list and is highlighted.
4. To add a descriptive comment for the field, click the Comment field of the property sheet and type the comment.

### Note

For information on the Used In tab, refer to the *Used In* section on page 22.

### Tip

You can change the name of a field by selecting the field name, then clicking again on the field name. A box appears around the selected field name and the name is highlighted. Type the new name and press *Return* to enter the name and position it in the list.

## Input Controls

To build input formats, you need to build input controls. Input (parse) controls are used to parse input data. Formatter uses input controls to determine how to find the beginning and end of the data in a field. Input controls have various ways of looking for data including: tag search, fixed length positional, delimiters, and embedded length fields.

An input control also determines whether the data for a field is mandatory or optional. Mandatory means that Formatter considers the parse successful if the parse start and end strings are found, even if the associated data is NULL. If the parse is successful, the field passes the mandatory test and the parse continues. If the parse parameters are not found, the field fails the mandatory test and the parse fails for the rest of the format. Optional means that the parse continues even if parse parameters for the field are not found.

Input controls can use tags to separate data. Tags are sets of bits or characters explicitly defining a string of data. For example, <DATE> and </DATE> could mark the beginning and end of a date field in a message.

After an input control is created, it can be saved, via a popup menu, as an output control. For more information, refer to the section *Saving an Input Control as an Output Control* page 43.

## Creating an Input Control

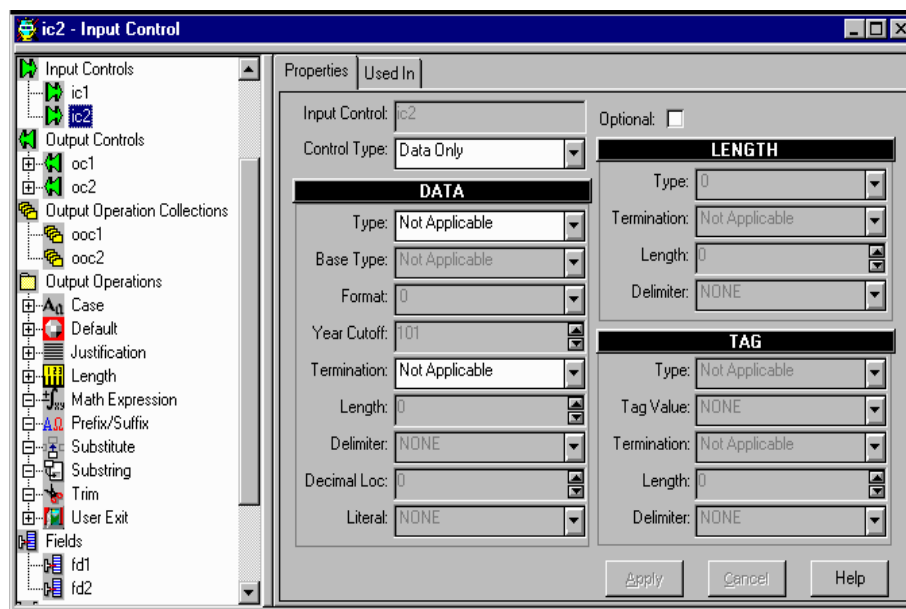
To create an input control:

1. In the left pane of the Formatter window, select the Input Controls component category and right-click. A popup menu appears.
2. Select New. A new input control component is added to the left pane and the input control property sheet is displayed in the right pane. The cursor is positioned in the text box where you type the new, unique input control name (maximum 32 characters). Input control names should be descriptive.

### Note

Single quotes cannot be used in names.

3. Click any mouse button outside the name text box or press *Return* to finish defining the input control. The new input control moves to its alphabetical location in the list and becomes highlighted.



*Input Control Dialog Box*

4. On the property sheet, click the Input Control Type drop-down list and select an input control type. Input Control Types describe the type of field parsed by Formatter.

Input Control types are described in the following table:

### Input Control Types

Value	Description
Data Only	Field has a data component only. Data is the value of the field.
Tag & Data	Field has a tag and data component (in this order). A tag label or identifier for the data.
Tag, Length & Data	Field has a tag, length, and data component (in this order). The length in bytes of the data portion of the field.
Length & Data	Field has a length and data component (in this order).
Repetition Count	Field contains the count of a repeating component.
Literal	Field value is a literal.
Length, Tag & Data	Field has a length, tag and data component (in this order).
Regular Expression	Compares input data to the value of the literal parse control using pattern matching. A Regular Expression requires an associated ASCII data type and a literal value.

#### Note

Once an input control is created, you can save it as an output control using the Save As Output popup menu.

#### Note

The Input Control Type determines which sections and fields require information. A section not requiring data is disabled. For example, if the Control Type = Data, then only the Data section of the tab is enabled. Also, you do not need to enter a value in every field of an enabled section. A field containing Not Applicable does not require a value.

### Data Section

The Data section is almost always enabled, however, the data type selected in this section determines which fields are enabled. All fields are discussed below.

1. In the Data section, click the Type field and select the type of parse control from the drop-down list.

If you select Regular Expression as the Control Type, the Type field contains the String data type.

If you select a user-defined format as the data type, refer to *User Defined Formats* on page 84.

The data type defines the type of the field in the input format. Data Types are described in the following table.

### Data Type Field Values

Data Type Field Value	Description
Not Applicable	No data type is assumed.
String	A string of standard ASCII characters. Note that non-printable characters are valid as long as they are in the ASCII character set. (EBCDIC characters outside the valid ASCII String range are not valid ASCII String characters. During a reformat from ASCII to EBCDIC if a character being converted is not in the EBCDIC character set the conversion results in a EBCDIC space (hexadecimal 40)).
Numeric	A string of standard ASCII numeric characters.
Binary Data	The Binary data type is used to parse any value and transform that value to an ASCII representation of the value internally in the Formatter. The internal representation takes each byte of the input value and converts it to a readable form. An example of this is parsing a byte whose value is (hexadecimal) 0x9C and transforming that to the internal ASCII representation of 9C, which is the hexadecimal value 0x3943. If this value is used in an output format with the output control's data type set to String, the value placed in the message is ASCII 0x9C. If this value is again placed in an output message with the data type Binary, the ASCII value is not printable and occupies one byte with the value of (hexadecimal) 0x9C. Conversely, an input value of ASCII 3B7A parsed with the String data type can be output using the Binary data type. The output value is (hexadecimal) 0x37BA and occupies 2 bytes in the output message. Valid characters that can be converted to Binary from the String data type are 0 through 9 and A through F. All other characters are invalid.



Data Type Field Value	Description
EBCDIC Data	A string of characters encoded using the EBCDIC (Extended Binary Coded Decimal Interchange Code) encoding used on larger IBM computers. During a reformat from EBCDIC to ASCII, if a character being converted is not in the EBCDIC character set, the conversion results in a space (hexadecimal 20).
IBM Packed Integer	Data type on larger IBM computers used to represent integers in compact form. Each byte represents two decimal digits, one in each nibble of the byte. The final nibble is always a hexadecimal F. For example, the number 1234 is stored as a 3-byte value: 01 23 4F (the number pairs show the hexadecimal values of the nibbles of each byte). The number 12345 is stored as a 3-byte value: 12 34 5F. There is no accounting for the sign of a number, so all numbers are assumed to be positive.
IBM Signed Packed Integer	<p>Data type on larger IBM computers used to represent integers in compact form. This data type takes into account the sign (positive or negative) of a number. Each byte represents two decimal digits, one in each nibble of the byte. The final nibble is a hexadecimal C if the number is positive, and a hexadecimal D if the number is negative.</p> <p>An example of how to generate a default value for an IBM Packed Integer is:</p> <p>Data Type: IBM Signed Packed Decimal  Default Value: -12345 (default value in ASCII)  Data Length: (Null - use the numbers in this field.)  The control is optional and there is no corresponding field in the input message, so Formatter uses the default value, converts it to IBM Signed Packed Decimal, and generates the following output: 12 34 5D. Each pair of numbers represents the two nibbles of a byte. The result is three bytes long.</p>
IBM Zoned Integer	Data type on larger IBM computers used to represent integers. Each decimal digit is represented by a byte. The left nibble of the byte is a hexadecimal F. The right nibble is the hexadecimal value of the digit. For example, 1234 is represented as F1 F2 F3 F4 (the number pairs show the hexadecimal values of the nibbles of each byte).
IBM Signed Zoned Integer	Data type on larger IBM computers used to represent integers. Each decimal digit is represented by a byte. The left nibble of each byte, <i>except</i> the last byte, is a hexadecimal F. The left nibble of the last byte is a hexadecimal C if the number is positive, and a hexadecimal D if the number is negative. The right nibble of each byte is the hexadecimal value of the digit. For example, 1234 is represented as F1 F2 F3 C4 (the number pairs show the hexadecimal values of the nibbles of each byte). -1234 is represented as F1 F2 F3 D4.

<b>Data Type Field Value</b>	<b>Description</b>
Little Endian 2	Two-byte integer where the bytes are ordered with the rightmost byte being the high order or most significant byte. For example, the hexadecimal number 0x0102 is stored as 02 01 (where the number pairs show the hexadecimal values of the nibbles of a byte).
Little Swap Endian 2	Two-byte integer where the two bytes are swapped with respect to a Little Endian 2 value. For example, the hexadecimal number 0x0102 is stored as 01 02.
Little Endian 4	Four-byte integer where the bytes are ordered with the rightmost byte being the high order or most significant byte. For example, the hexadecimal number 0x01020304 is stored as 04 03 02 01 (where the number pairs show the hexadecimal values of the nibbles of a byte).
Little Swap Endian 4	Four-byte integer where the two bytes of each word are swapped with respect to a Little Endian 4 value. For example, the hexadecimal number 0x01020304 is stored as 03 04 01 02.
Big Endian 2	Two-byte integer where the bytes are ordered with the leftmost byte being the high order or most significant byte. For example, the hexadecimal number 0x0102 is stored as 01 02 (where the number pairs show the hexadecimal values of the nibbles of a byte).
Big Swap Endian 2	Two-byte integer where the two bytes are swapped with respect to a Big Endian 2 value. For example, the hexadecimal number 0x0102 is stored as 02 01.
Big Endian 4	Four-byte integer where the bytes are ordered with the leftmost byte being the high order or most significant byte. For example, the hexadecimal number 0x01020304 is stored as 01 02 03 04 (where the number pairs show the hexadecimal values of the nibbles of a byte).
Big Swap Endian 4	Four-byte integer where the two bytes of each word are swapped with respect to a Big Endian 4 value. For example, the hexadecimal number 0x01020304 is stored as 02 01 04 03.
Decimal, International	Data type where every third number left of the decimal point is preceded by a period. The decimal point is represented by a comma. Numbers right of the decimal point represent a fraction of one unit. For example, the number 12345.678 is represented as 12.345,678. Decimal international data types can contain negative values.

<b>Data Type Field Value</b>	<b>Description</b>
Decimal, U.S.	Data type where every third number left of the decimal point is preceded by a comma. The decimal point is represented by a period. Numbers right of the decimal point represent a fraction of one unit. For example, the number 12345.678 is represented as 12,345.678. Decimal US data types can contain negative values.
Unsigned Little Endian 2	Like Little Endian 2, except that the value is interpreted as an unsigned value.
Unsigned Little Swap Endian 2	Like Little Swap Endian 2, except that the value is interpreted as an unsigned value.
Unsigned Little Endian 4	Like Little Endian 4, except that the value is interpreted as an unsigned value.
Unsigned Little Swap Endian 4	Like Little Swap Endian 4, except that the value is interpreted as an unsigned value.
Unsigned Big Endian 2	Like Big Endian 2, except that the value is interpreted as an unsigned value.
Unsigned Big Swap Endian 2	Like Big Swap Endian 2, except that the value is interpreted as an unsigned value.
Unsigned Big Endian 4	Like Big Endian 4, except that the value is interpreted as an unsigned value.
Unsigned Big Swap Endian 4	Like Big Swap Endian 4, except that the value is interpreted as an unsigned value.
Date and Time*	Based on the international ISO-8601:1988 standard datetime notation: YYYYMMDDhhmmss. See the first paragraph of each of the Date and Time type descriptions for details on representing Date and Time components. Combined dates and times may be represented in any of the following list of base data types. For some data types, a minimum of 8 bytes is required. The list includes: Numeric, String, and EBCDIC.
Time*	Based on the international ISO-8601:1988 standard time notation: hhmmss where hh represents the number of complete hours that have passed since midnight (between 00 and 23), mm is the number of minutes passed since the start of the hour (between 00 and 59), and ss is the number of seconds since the start of the minute (between 00 and 59). Times are represented in 24-hour format. Times may be represented in any of the following list of base data types. For some data types, a minimum of 4 bytes is required. The list includes: Numeric, String, and EBCDIC.

Data Type Field Value	Description
Date*	Based on the international ISO-8601:1988 standard date notation: YYYYMMDD where YYYY represents the year in the usual Gregorian calendar, MM is the month between 01 (January) and 12 (December), and DD is the day of the month with a value between 01 and 31. Dates may be represented in any of the following list of base data types. For some data types, a minimum of 4 bytes is required. The list includes: Numeric, String and EBCDIC.
Custom Date and Time*	<p>Custom Date and Time enables users to specify different formats of dates, times, and combined dates and times.</p> <p>Date/Time formats may include:</p> <ol style="list-style-type: none"> <li>1) Variations in year (2- or 4-digit year representation: YY or YYYY).</li> <li>2) Variations in month – use of a month number (01-12) or three-letter abbreviation (JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC).</li> <li>3) Variations in the day of the month – use of a day of the month number (01-31).</li> <li>4) Variations in hour – 12-hour or 24-hour representation, with or without a meridian indicator (AM or PM.)</li> <li>5) Custom date/time formats are available in the Format drop-down list. Custom date/time formats must have a base data type of Numeric, String, or EBCDIC.</li> </ol> <p>For information on how to set the Year Cutoff value, refer to the section <i>Year 2000 Compliance</i> on page 43.</p>

\* If you use Date and Time, Date, Time, or Custom Date Time as the Data Type, select a Base Data Type of ASCII String, Numeric, or EBCDIC. If you select Custom Date Time, also select a format string from the Format drop-down list and a Year Cutoff. For more information, refer to the section *Year 2000 Compliance* on page 43.

### Note

A 16-byte length limit applies to IBM platforms using packed data types (IBM Packed Integer and IBM Signed Packed Integer). However, the 16-byte limit has been removed for the IBM zoned data types (IBM Zoned Integer and IBM Signed Zoned Integer) to reflect the way the data is actually handled by IBM platform assemblers.

### Note

The data types listed above may be followed by user-defined data types.

2. If the data in a field is constant, select a control type of Literal. You must define a literal containing the regular expression and assign it.

If the Input Control Type is Regular Expression, the data type field contains an ASCII data type, and you must enter a regular expression in the Regular Expression field (maximum of 32 bytes). You must define a literal containing the regular expression and assign it.

Regular expressions are strings that express rules for string pattern matching. Regular expression parse controls function like literal parse controls, but instead of a direct character-by-character match, the parse control value is interpreted as a regular expression to match to the input. The string matching capabilities of this feature comply with the POSIX 1003.2 standard for regular expressions. Examples of regular expressions are shown in the following table.

### Regular Expressions

Value	Description
a	match a
\\	match \
^a	match a, only if it begins the string
a\$	match a, only if it ends the string
^ a\$	match a, only if it consists entirely of the one character a
.	match any one character
[abc]	match one of the characters a, b, or c
[^abc]	match any one character that is not a, b, or c
a +	match a string of one or more of the character a
ab*	match the character a optionally followed by any number of the character b
a{3}	match 3 or more concatenated characters a. Equivalent to aaa+ or aaa*
(a b \\ )	match either the character a or b or

3. To specify a year cutoff number for year 2000 compliance, in the Year Cutoff field, type a year cutoff number from 0 to 99. The default is 101, an invalid number. You must enter a valid number to continue.

To specify a year cutoff, the (Data) Type field must be Custom Date and Time. The Format must include a two-digit year notation (for example, MM/DD/YY). For more information, refer to *Year 2000 Compliance* on page 43.

4. Click the Termination field and select the termination type for the data from the drop-down list.

The termination type tells Formatter when to stop parsing for this field. Data terminators are described in the following table.

### Data Terminators

Data Terminator Value	Description
Not Applicable	No data termination. Read to end of message.
Delimiter	The field is terminated by a delimiter.
White Space Delimited	The field is terminated by a white space.
Exact Length	The field has a fixed length.
Minimum Length and Delimiter	Parse a minimum number of characters and then look for a delimiter.
Minimum Length and White Space	Parse a minimum number of characters and then look for a white space.

5. If you specified a data termination type of either exact length or minimum length, choose one of the following:
  - If you specified exact length, type the exact length of the data, used for fixed-format messages in the Length field.
  - If you specified minimum length, type the number of bytes the Formatter skips before it starts looking for the termination control in the Length field.
6. If the termination type is delimited, click the Delimiter field and choose a literal from the drop-down list.
7. If the data type requires a decimal, click the Decimal Loc field and either type a decimal location or click on the arrows until you reach the desired decimal location. Only a positive value is allowed, and it must not be more than the number of digits the field can hold.

Note that decimal location pertains only to IBM packed and zoned decimal data. Only a positive value is allowed, and the value cannot be more than the number of digits the field can hold. The value determines how many digits are to the right of the decimal location when Formatter interprets the message data. For example, an input value of 12345 with a decimal location of 2 is interpreted as 123.45.

### **Optional/Mandatory**

Check the Optional checkbox to make the input field parse optional. Optional means the parse continues even if parse parameters for the field are not found. By default, input controls are mandatory.

If the parse is successful, the field passes the mandatory test and the parse continues. If the parse parameters are not found, the field fails the mandatory test and the parse fails for the rest of the format.

### **Length Section**

The Length section is enabled if you select one of the following as your Input Control Type:

- Tag, Length & Data
- Length & Data
- Length, Tag & Data

When you select any of the above as the Input Control Type, enter the length in the Length section. The Length field in the Data section is disabled.

If a length is required and you do not specify one, the default length is zero.

The following is an example of how the data length is used:

Data Type: ASCII

Data Length: 10 (the final field width, in bytes)

Padding Character: \*

The control is optional and there is no corresponding field in the input message, so Formatter uses the default value, pads it with \* and generates the following output:

xxx\*\*\*\*\*

Numeric fields of all types are right justified when the default length is less than the field length.

1. In the Length section, click the Type field and select the parse control length data type from the drop-down list.

### **Note**

For a description of the data types, refer to the table on page 34.

2. Click the Termination field and select the parse control length termination type from the drop-down list.

The parse control length termination type tells Formatter when to stop parsing for the length field value. When the length delimiter is reached, the value that was parsed is the length value.

For a description of termination types, refer to the table *Data Terminators* on page 40.

3. If the termination type is exact length, click the Length field and type the exact length of the data.
4. If the length termination type is delimited, click the Delimiter field and select a literal from the drop-down list.

### **Tag Section**

The Tag section is enabled if you select one of the following as your Input Control Type:

- Tag & Data
  - Tag, Length & Data
  - Length, Tag & Data
1. In the Tag section, click the Type field and select the data type of the tag data from the drop-down list.  
For a description of the data types, refer to the table on page 34.
  2. Click in the Tag Value field and select the value from the list.
  3. Click in the Termination field and select the termination type for the tag from the drop-down list.

This tag tells Formatter when to stop parsing the tag value. When the tag delimiter is reached, the value that was parsed is compared to the value in the tag value field to determine if this is the data being searched for.

For a description of termination types, refer to the table *Data Terminators* on page 40.

4. If the tag termination is fixed length, click the Length field and type the exact tag length.
5. If the tag termination type is delimited, click the Delimiter field and select a literal from the drop-down list.

#### **Note**

For information on the Used In tab, refer to the *Used In* section on page 22.

#### **Tip**

You can change the name of an input control by double-clicking the input control name (without moving the mouse). A box appears around the selected input control name and the name is highlighted. Type the new, unique name and press *Return* to enter the name and position it in the list.



### ***Saving an Input Control as an Output Control***

Input controls can be saved as output controls. The output controls mirror (as closely as possible) the contents of the input control. The output control may have to be modified, since certain properties of input controls do not have exact matches on the output control side.

To save an input control as an output control:

1. In the left pane of the Formatter window, select the input control you want to save as an output control and right-click. A popup menu appears.
2. Select Save as Output. The input control is saved as an output control with the prefix "OFC\_#\_" and appears in the Output Controls list.

For example, the first time the input control "Input\_Control" is saved as an output control, you will see "OFC\_1\_Input\_Control," the second one will be "OFC\_2\_Input\_Control," and so on.

3. To either change or add information for the output control, select it and modify the details on the Output Control property sheet.

For more information on the Output Control property sheet, refer to the section *Output Controls* on page 44.

## **Year 2000 Compliance**

To address Year 2000 compliance, indicate a Year Cutoff number for Input Controls using a (Data) Type of Custom Date and Time, and a Date Format that includes a two-digit year notation (such as MM/DD/YY). The year cutoff number must be from 0 to 99 (inclusive).

Once you specify a cutoff number, Formatter converts a two-digit year (YY) to a four-digit year (YYYY).

The Formatter GUI defaults the year cutoff to 101, an invalid number. For a two-digit year input format, you **MUST** enter a valid number to continue.

Here are some examples of how Formatter interprets the year cutoff number:

- If the year cutoff is 50, then all two-digit input dates from 50 to 99 are designated as 1950 to 1999 output dates; all two-digit input dates from 00 to 49 are designated as 2000 to 2049 output dates.
- If the year cutoff is 75, all two-digit input dates from 75 to 99 are designated as 1975 to 1999 output dates; all two-digit input dates from 00 to 74 are designated as 2000 to 2074 output dates.

## Output Controls

Output controls are used to format output data. Formatter uses output format controls to determine how to justify and trim data, whether to add prefixes or suffixes, whether to perform an arithmetic expression, to use a user exit, conditional field, existence check, or transformed field. Using the output control types of either Input Field Value = or Input Field Exists allows alternative output formatting.

For more information on alternative output formatting, refer to the section *Alternative Input and Output Formats* on page 78.

## Creating an Output Control

To add an output control:

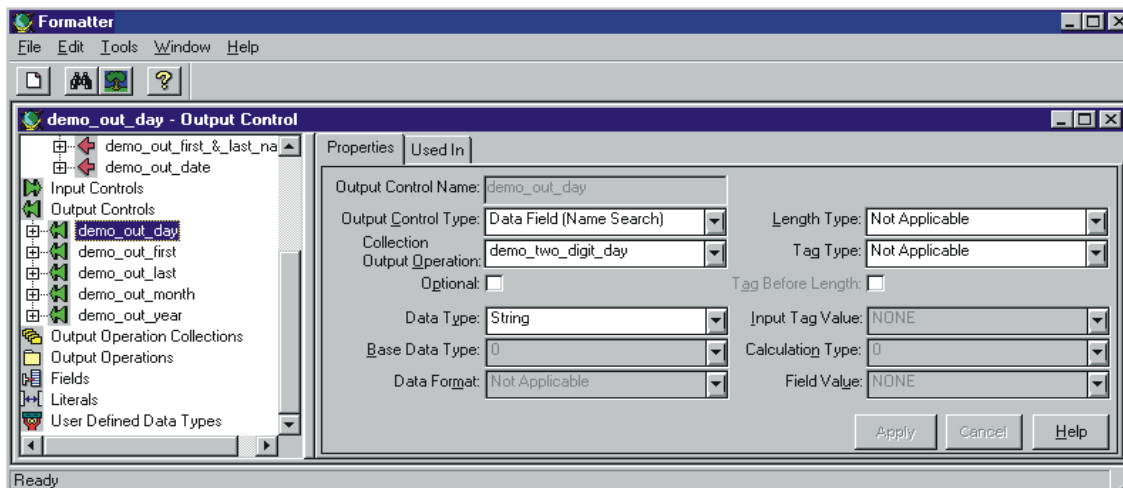
1. From the Formatter window tree, select Output Controls and click the right mouse button. A popup menu appears with the New menu item.
2. Select New to create a new output control entry in the left pane and open the Output Control property sheet in the right pane. The cursor is positioned in the output control text box where you can type the new, unique output control name (maximum of 32 characters).

### Note

Single quotes cannot be used in names.

3. Either click any mouse button outside the output controls text box or press *Return* to add the output control to the list of output controls.

The output control is highlighted and alphabetically positioned in the output control list and the Output Control property sheet appears.



*Output Control Property Sheet*

- In the Output Control property sheet, click the Output Control Type field and select the type of control from the drop-down list. An output control defines how you want Formatter to output a message field. To do this, define how that data should appear in the output message.

Output Control types identify how to map to the input data and the type of reformatting to use. The Data Field (Name Search) and Data Field (Tag Search) values map the output field to the input field by either name or tag. The other output control types define details concerning data mapping.

Output Control types are described in the following table.

Output Control Types	Description
Data Field (Name Search)	Map the output field to the input field by field name.
Data Field (Tag Search)	Map the output field to the input field by tag value.
Literal	Field value is a literal.
Left Operand Field	Mark field as a left operand.
Right Operand Field	Mark field as a right operand.
Calculated Field	Perform a calculation using the left and right operand fields.
Conditional Field	Mark field as output only if existence check field exists.
Existence Check Field	Mark field as an existence check field.
Rules Field	Output format control is chosen based on Boolean logic.
Input Field Exists	An input field exists.
Input Field Value =	The input field's value equals a particular value.

**Optional/Mandatory**

Use the Optional check box to specify whether the output control is either optional or mandatory.

Optional means Formatter should continue with the output message if the field was not found in the input message or the input field was NULL.

By default, the output control is mandatory, Mandatory means Formatter should fail with an error message if the field was not found in the input message or the input field was NULL, and there was no data-producing operation associated with the output control.

A data-producing operation is capable of generating data in the absence of input data. The following operations are data producing:

- Default
- Prefix/Suffix (if Null Action is checked)
- User Exit
- Math Expression

**Note**

User Exit or Math Expression may still fail if they are written to depend on the existence or value of an input field and the input field is missing or NULL.

**Data Field (Name Search)**

Data Field (Name Search) output controls map the value of an input field directly to an output field. Formatter searches the parsed input message for the input field value by field name specified.

For example, if the contents of an output field map directly to the contents of an input field, you could define a Data Field (Name Search) output control. The output control could look like:

- Output Control Name: FieldNameWithComma
- Output Control Type drop-down list: Data Field (Name Search)
- Data Type drop-down list: String
- Tag Before Length checkbox: (not checked)
- Optional checkbox: Checked

**Note**

To add a comma suffix to the end of the field data, create a Suffix operation, then select it in the Output Operation drop-down list.

**To define a Data Field (Name Search) output control:**

1. Select the Data Field (Name Search) in the Output Control Type drop-down list.
2. Select a data type for the output data from the Data Type drop-down list.
3. If the selected data type is Date and Time, Date, Time, or Custom Date and Time, select the Base Data Type (String, EBCDIC, or Numeric) to output the data.
4. If the selected data type is Custom Date and Time, select a date/time format from the Data Format drop-down list.
5. To perform an operation on the data, select one from the Output Operation drop-down list.
6. If the output control parse is optional, check the Optional checkbox.
7. If the output data is to have an embedded length, specify the Length data type in the Length Type drop-down list.
8. If the output data is to have an associated tag, specify the data type for the tag in the Tag Type drop-down list.
9. If the tag should be output before the embedded length, check the Tag Before Length box.

**Data Field (Tag Search)**

Data Field (Tag Search) output controls map the value of an input field directly to an output field. Formatter searches for the input field value by tag, looking for the specified literal tag value in the parsed input message.

For example, in a random-ordered flat format, you are searching for a specific tagged field in the parsed input data. If you are looking for an input Tag and Data field with <FirstName> as the tag and want to output the data minus the tag, the output control might look like:

- Output Control Name: InTagDataOutDataComma
- Output Control Type drop-down list: Data Field (Tag Search)
- Data Type drop-down list: Numeric
- Input Tag Value field: FirstName
- Tag Before Length checkbox: (not checked)
- Optional checkbox: (not checked)

**Note**

To add a comma suffix to the end of the field data, create a Suffix operation, then select it in the Output Operation drop-down list. For more information, see *Output Operations* on page 56.

**To define a Data Field (Tag Search) output control:**

1. Select the Data Field (Tag Search) in the Output Control Type drop-down list.
2. Select a data type for the output data from the Data Type drop-down list.
3. If the selected data type is Date and Time, Date, Time, or Custom Date and Time, select the Base Data Type (String, EBCDIC, or Numeric) to output the data.
4. If the selected data type is Custom Date and Time, select a date/time format from the Data Format drop-down list.
5. To perform an operation on the data, select one from the Output Operation drop-down list.
6. If the output control parse is optional, check the Optional checkbox.
7. If the output data is to have an embedded length, specify the Length data type in the Length Type drop-down list.
8. If the output data is to have an associated tag, specify the data type for the tag in the Tag Type drop-down list.
9. After selecting a Tag Type, choose a Literal from the Input Tag Value drop-down list as the tag.
10. If the tag should be output before the embedded length, check the Tag Before Length box.

***Literal***

Literal output controls insert a literal (static) value into the output message.

For example, there may be a static value separating the header for a message from the body. So you would want to add <BODY>. An output control might look like:

- Tag Before Length checkbox: (Not Checked)
- Optional checkbox: (Not Checked)
- Data Type (Data Types) drop-down list: String
- Output Control Type drop-down list: Literal
- Output Control Name: Out<FirstName>Literal

**To define a Literal output control:**

1. Select Literal in the Output Control Type drop-down list.
2. To perform an operation on the data, select one from the Output Operation drop-down list.
3. If the output control parse is optional, check the Optional checkbox.
4. If the output data is to have an embedded length, specify the Length data type in the Length Type drop-down list.
5. If the output data is to have an associated tag, specify the data type for the tag in the Tag Type drop-down list.
6. If the tag should be output before the embedded length, check the Tag Before Length box.
7. Select a literal from the Field Value drop-down list.

**Calculated Field****Note**

The Math Expression output control offers many more options than Calculated Field. We recommend you use Math Expression instead of Calculated Field, since Calculated Field will be phased out in a future release.

Calculated Field output controls perform basic arithmetic operations (Add, Subtract, Multiply, and Divide) on the data defined by the "Left Operand Field" and "Right Operand Field" output controls.

For example, to use the Add, Subtract, Multiply, or Divide operations with values from two fields in the input format and output the result in an output field, use one Left Operand Field, one Right Operand Field, and Calculated Field output controls. The Calculated Field output control performs the specified operation on the fields specified as Left and Right Operands in the flat output format.

A sample output control might look like:

- Output Control Name: AddLeftRightOutDataComma
- Output Control Type drop-down list: Calculated Field
- Data Type (Data Types) drop-down list: Numeric
- Calculation drop-down list: Add
- Tag Before Length checkbox: (Not Checked)
- Optional checkbox: (Not Checked)

**Note**

A Left Operand Field output control and Right Operand Field output control **MUST** be defined in an output format for a Calculated Field output control to work properly.

**To define a Calculated Field output control:**

1. Select Calculated Field in the Output Control Type drop-down list.
2. Select a data type for the output data from the Data Type drop-down list.
3. If the selected data type is Date and Time, Date, Time, or Custom Date and Time, select the Base Data Type (String or Numeric) the data will be output as.
4. If the selected data type is Custom Date and Time, select a date/time format from the Data Format drop-down list.
5. To perform an operation on the data, select it from the Output Operation drop-down list.
6. If the output control parse is optional, check the Optional checkbox.
7. If the output data is to have an embedded length, specify the Length data type in the Length Type drop-down list.
8. To perform a calculation on the data, select a type of calculation from the Calculation drop-down list. Add, Subtract, Multiply, and Divide are available for Calculated Fields.
9. If the output data is to have an associated tag, specify the data type for the tag in the Tag Type drop-down list.
10. If the tag should be output before the embedded length, check the Tag Before Length box.

***Left Operand Field*****Note**

The Math Expression output control offers many more options than Calculated Field. We recommend you use Math Expression instead of Calculated Field, since Calculated Field will be phased out in a future release.

Left Operand Field output controls mark an input field as "left operand." Once a Right Operand Field has also been defined, the two fields can be used in a Calculated Field output control. See *Calculated Field* on page 49 for details.

For example, if you want to use the Add, Subtract, Multiply, or Divide calculations with values from two fields in the input format and output the result in an output field, use one Left Operand Field, one Right Operand Field, and one Calculated Field output control. The Calculated Field output control performs the specified calculation on the fields specified as Left and Right Operands in the flat output format.

**Note**

Both a Left Operand Field output control and Right Operand Field output control **MUST** be defined in an output format for a Calculated Field output control to work properly.



**To define a Left Operand Field output control:**

1. Select the Left Operand Field in the Output Control Type drop-down list.
2. If the output control parse is optional, check the Optional checkbox.

***Right Operand Field*****Note**

The Math Expression output control offers many more options than Calculated Field. We recommend you use Math Expression instead of Calculated Field because Calculated Field will be phased out in a future release.

---

Right Operand Field output controls mark an input field as "right operand." Once a Left Operand Field has also been defined, the two fields can be used in a Calculated Field output control. See *Calculated Field* on page 49 for details.

For example, to use the Add, Subtract, Multiply, or Divide calculations with values from two fields in the input format and output the result in an output field, use one Left Operand Field, one Right Operand Field, and one Calculated Field output control. The Calculated Field output control performs the specified calculation on the fields specified as Left and Right Operands in the flat output format.

**Note**

A Left Operand Field output control and Right Operand Field output control **MUST** be defined in an output format for a Calculated Field output control to work properly.

---

**To define a Right Operand Field output control:**

1. Select the Right Operand Field in the Output Control Type drop-down list.
2. If the output control parse is optional, check the Optional checkbox.

### **Conditional Field**

Conditional Field output controls mark a field as output only if a specific field exists. One other field in the output format must be associated with an Existence Check Field output control.

For example, to output an account number only if a person's name is in the input message. The output control might look like:

- Output Control Name: IfPersonNameOutAccountComma
- Output Control Type drop-down list: Conditional Field
- Data Type (Data Types) drop-down list: String
- Tag Before Length checkbox: (Not Checked)
- Optional checkbox: Checked

### **Note**

Another field (in this example, the "PersonName" field) in the output format **MUST** be associated with an Existence Check Field output control for the Conditional Field output control to work correctly.

#### **To define a Conditional Field output control:**

1. Select Conditional Field in the Output Control Type drop-down list.
2. Select a data type for the output data from the Data Type drop-down list.
3. If the selected data type is Date and Time, Date, Time, or Custom Date and Time, select the Base Data Type (String, EBCDIC, or Numeric) the data will be output as.
4. If the selected data type is Custom Date and Time, select a date/time format from the Data Format drop-down list.
5. To perform an operation on the data, select one from the Output Operation drop-down list.
6. If the output data is to have an embedded length, specify the Length data type in the Length Type drop-down list.
7. If the output data is to have an associated tag, specify the data type for the tag in the Tag Type drop-down list.
8. If the tag should be output before the embedded length, check the Tag Before Length box.

### ***Existence Check Field***

Existence Check output controls are used to define the field that a Conditional Field output control depends on. See *Conditional Field* on page 52 for details.

#### **To define a Existence Check Field output control:**

1. Select Existence Check Field in the Output Control Type drop-down list.
2. If the output control parse is optional, check the Optional checkbox.

Optional means that Formatter should continue with the output message if the input field was null and there was no data-producing operation associated with the output control.

By default, output controls are mandatory. Mandatory means if the associated input field is either NULL or not referenced in the input format, and there is no default value, the mandatory test fails and Formatter will not output data.

### ***Rules Field***

Use the Rules Field (formerly called conditional branching) to create several different output controls for a single output field by integrating Rules with Formatter. Formatter can then use the boolean logic capabilities of the Rules Engine to express and evaluate the conditions for formatting a field.

Based on the fields defined for the input format, you build different output controls for the same output field. This eliminates the need to create several output formats for a single input format.

If no rule hits, output data is formatted to the other settings in the Rules Field output control. There will be one subscription or output control for each boolean field in the output format, unless the rule did not evaluate a *true* for that rule. The data is formatted according to the output control specified by the rule.

### **Note**

When you define rules for the output control, you cannot delete the fields used in the rules without first deleting the rules that use the fields.

**To define a Rules Field output control:**

1. Select Rules Field in the Output Control Type drop-down list.
2. To perform an operation on the data, select one from the Output Operation drop-down list.
3. If the output control parse is optional, check the Optional checkbox.
4. Select a data type for the output data from the Data Type drop-down list.
5. If the selected data type is Date and Time, Date, Time, or Custom Date and Time, select the Base Data Type (String, EBCDIC, or Numeric) the data will be output as.
6. If the selected data type is Custom Date and Time, select a date/time format from the Data Format drop-down list.
7. If the output data is to have an embedded length, specify the Length data type in the Length Type drop-down list.
8. If the output data is to have an associated tag, specify the data type for the tag in the Tag Type drop-down list.
9. If the tag should be output before the embedded length, check the Tag Before Length box.

**To add the rules for the field:**

1. Choose the Jump to Rules button. The Rules Login dialog box appears.
2. Log in to Rules. The Rules GUI appears.
3. Enter the rules for the field using the instructions in Chapter 3, *NEONRules* on page 83.
4. Exit the Rules GUI. You are returned to the Formatter GUI.

***Input Field Exists***

Input Field Exists output controls indicate that the output format is part of the output message if it exists in the input message.

For example, if a last name input field is in the input message, you may want to output the format that includes first, middle, and last names. You would have four fields in this format:

<b>Field</b>	<b>Output Control Type</b>
LastName	Input Field Exists
FirstName	Data Field (Name Search)
MiddleName	Data Field (Name Search)
LastName	Data Field (Name Search)

**The output control for Last Name might look like:**

- Output Control Name: IfMiddleNameOutData
- Output Control Type drop-down list: Input Field Exists
- Tag Before Length checkbox: (Not Checked)
- Optional checkbox: Checked

**To define an Input Field Exists output control:**

1. Select Input Field Exists output control.
2. If the output control parse is optional, check the Optional checkbox.

***Input Field Value =***

Input Field Value = output controls indicate that the output format the control is part of should be output if the value of the field has a specific value.

For example, if an input field contains security information, you may want to limit the data in the output message based on that information. The output control might look like:

Output Control Name: IfSecurity=HighThenOut

Output Control Type drop-down list: Input Field Value =

Input Field Value drop-down list: High

Tag Before Length checkbox: (Not Checked)

Optional checkbox: Checked

<b>Note</b>
-------------

The Output Control type determines which fields require information. Fields containing "Not Applicable" or "NONE" do not require a value.

---

**To define an Input Field Value = output control:**

1. Select Input Field Value = as the output control type
2. Select a Literal from the Field Value drop-down list to compare the field contents to.
3. If the output control parse is optional, check the Optional checkbox.

## Output Operations

Output operations provide the different actions that can be performed on an output field. Using operations, you can, for example, change the case of output data, perform mathematical expressions based on input field contents, and extract substrings.

Through the use of output operation collections, you can collect operations to perform them sequentially. For example, you could left justify and right trim a substring of the contents of an input field. The order in which these operations are defined in the collection is the order in which they are performed.

Available operations are described in the following table:

Output Operation Type	Description
Case	Case operations affect the case of the field data. The two defined case operations are LOWER_CASE and UPPER_CASE.
Default	Default operations provide a default value for a field if an input field either does not exist in the input message or has a length of zero. Note that default operation names have a 32-character maximum length.
Prefix/Suffix	Prefix/Suffix operations enable you to attach literals to the beginning or end of the field data.
Justify	Justify operations justify field data to be left, center, or right within the length of the field. If either center or right justify operations are used, specify a pad character to fill the space on the left side of the data. If no pad character is specified, a space is used.
Length	Length operations ensure that an output string is given a length. If the data length is longer than the specified length, the data is truncated. If the data length is shorter than the specified length, pad characters are used. Non-numeric data types are padded on the right, numeric data types are padded on the left. Note that length operation names have a 32-character maximum length.
Math Expression	Math Expression operations output a value resulting from an arithmetic expression. The expression can be built using arithmetic operators, constants, and input field values. For more information, see <i>Math Expression</i> on page 58.

Output Operation Type	Description
Substitute	Substitute operations enable you to define a list of input strings to substitute and the output strings to replace them. For each substitute item within a substitute operation, define a literal to look for as the input value, a literal to replace it with, and the data type in which to output the new data. Note that substitute operation names have a 32-character maximum. For more information, see <i>Operation Substitute Strings</i> on page 60.
Substring	Substring operations enable you to extract a portion of an input string, defined by byte position and length, and place it in the output field. Note that substring operation names have a 32-character maximum.
Trim	Trim operations remove a defined trim character to the right and/or left of the output data. Note that trim operation names have a 32-character maximum.
User Exit	User Exit operations enable you to run predefined user exit functions. User Exits are external, user-created routines used to compute the value of an output field.

## Output Operation Collections

Use Output Operations Collections to group and sequence a series of output operations and/or other output operation collections. Operations are executed in the order in which they appear. For example, to have the contents of an input field take a substring of the left-justified, right-trimmed contents of an input field, LEFT\_JUSTIFY, RIGHT\_TRIM, and then SUBSTRINGXXXX.

### Note

You cannot create collections that contain themselves. This is recursive. If you see the recursion icon, it is probable that something has corrupted the data in your Formatter database.

**To define an output operation collection:**

1. In the left pane of the Formatter window, select Output Operation Collection and right-click. A popup menu appears with New as a menu item.
2. Select New. A new collection is added to the left pane.
3. The cursor is positioned in the text box where you can type the new, unique Output Operation Collection name. Names should be descriptive.
4. Click any mouse button outside the name text box or press *Return* to finish defining the Output Operation Collection. The new entry moves to its alphabetical location in the list and is highlighted.
5. Right-click the collection and choose Add Output Operations. The Output Operations window opens.
6. Select the operations you want to add, then choose the Accept Selection button.

**Tip**

In the collection's pane, to reorder component output operations or output operation collections, click and drag the component to be moved to the highest level of the collection. It is inserted as the first component in the collection. You can rearrange the other components to fit the new order in this manner.

**Math Expression**

Using Math Expression operations, you can output a value resulting from an arithmetic expression. The expression can be built using arithmetic operators, constants, and input field values.

Operators	Expression Components
Available Operators	+ - * / ( ) and Unary -
Order of Operator Precedence	( ) * / + -
Available Operands	Numeric Constants and Input Field Names

**Note**

Math Expression ignores any previous operations.



For example, if an input message is defined with fields **InF1**, **InF2**, and **InF3** and an output message is defined with field **OutF1**. You could define a math expression as part of an output control associated with output message field **OutF1** as:

**InF1 + InF2 \* -InF3**

This expression will be evaluated as **InF1 + (InF2 \* (- InF3))** based on the precedence rules.

Other expression examples include:

InF1 + -InF2

InF1 \* 8

InF1 \* 9.3

InF1 \* -8

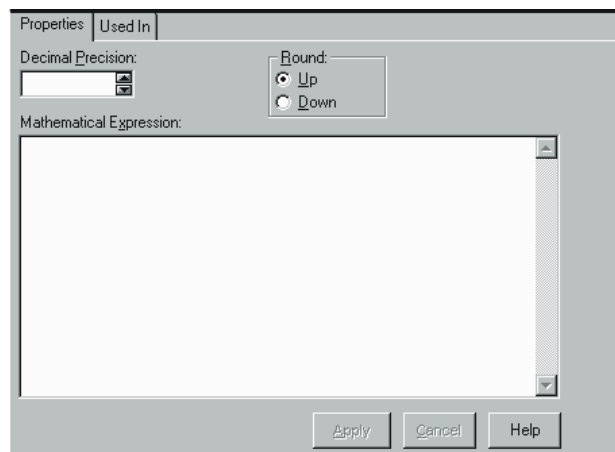
InF1 \* -9.3

(InF1 + InF2) \* 3/InF3

(InF1 \* (InF2 + InF3) \* 4

**To define a Math Expression operation:**

1. In the left pane of the Formatter window, select the Math Expression output operation category and right-click. A popup menu appears with New as a menu item.
2. Select New. A new operation is added to the left pane.
3. The cursor is positioned in the text box where you can type the new, unique Math Expression operation name. Names should be descriptive.
4. Either click any mouse button outside the name text box or press *Return* to finish defining the Math Expression operation. The new entry moves to its alphabetical location in the list and is highlighted.



*Mathematical Expression Dialog Box*

5. Select the Properties tab in the right pane.
6. Click the Decimal Precision field and either use the up/down arrows or type the number of decimal points the expression result should be rounded to.
7. Select whether the expression result should be rounded up or down using the appropriate option button.
8. Type the expression in the Mathematical Expression field. Fields defined in mathematical expressions must be in quotes (for example, "field1" = "field2"). A field name cannot contain a dash (for example, if the field name is abc-def, type the name as "abc\_def").
9. Choose Apply.

### Note

When multiplying fields in math expressions, multiply by 1.0 to avoid erroneous characters after the decimal.

### Note

To set up an output control to handle a 6 digit ASCII number and insert a decimal, multiply the number by 1.000 where the number of zeroes after the decimal equals the amount of decimal precision required in the final number.

## Operation Substitute Strings

Substitute operations enable you to define a list of input strings to substitute and the output strings to replace them.

For example, each time Formatter receives the value x as input, you want to output the value as xx. You can change the value from the input of x to the output of xx.

### To define a Substitute operation:

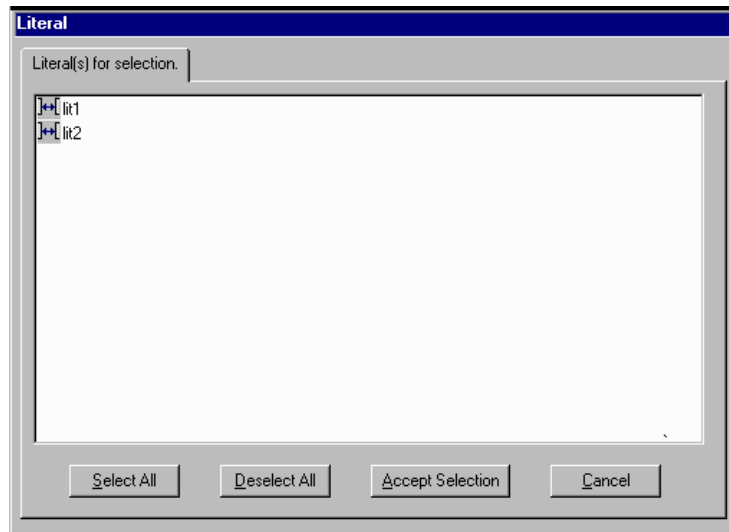
1. In the left pane of the Formatter window, select the Substitute operation and right-click. A popup menu appears with New as a menu item.
2. Select New. A new operation is added to the left pane.
3. The cursor is positioned in the text box, where you can type the new, unique Substitute operation name. Names should be descriptive.
4. Either click any mouse button outside the name text box or press *Return* to finish defining the Substitute operation. The new entry moves to its alphabetical location in the list and is highlighted.

### Note

The Substitute operation itself has no properties. You must add Substitute Items.

**To assign Substitute Items:**

1. In the left pane of the Formatter window, select the substitute operation you want to add items to and right-click. A popup menu appears with Add Substitute Items as a menu item.
2. Select Add Substitute Items. The Literal window appears.

*Literal Window*

3. Select the literals you want to use as input strings and choose Accept Selection.
4. Click the new substitute operation in the left pane.
5. For each substitute item, select:
  - A data type from the Output Data Type drop-down list.
  - A literal from the Output Value drop-down list. This value will be substituted for the value specified in the Input Value drop-down list.
6. Choose Apply.
7. Repeat steps 5 and 6 for each Substitute Item.

## Formats

Use the Formats function to build both input and output formats. The formats can be either flat or compound.

The component formats of a compound format are either optional or mandatory. An optional component can have missing components and the parse or reformat will still succeed. A mandatory component must exist, and all its mandatory components must also exist, or the parse or reformat will fail. The component formats are mandatory by default.

Before creating flat or compound formats, define their component parts.

Format	Components
flat input format	literals, fields, input (parse) controls, and user-defined types (if used)
flat output format	literals, fields, and output controls, output operations, and output operation collections
compound input format	flat input formats or other compound input formats
compound output format	flat output formats or other compound output formats

### Tip

#### To simplify creating Formatter components:

- When building Input Controls, open two windows. One window should be open to the new Input Control and the other should be open to Fields.
- When building Output Controls, open four windows. One for Fields, one for Output Operations, one for Output Operation Collections, and one for the new Output Control.
- When building Flat Input Formats, open two windows. One for the new Format and one for Input Controls to be associated with input Fields.
- When building Flat Output Formats, open two windows. One for the new Format and one for Output Controls to be associated with output Fields.
- When building Compound Input or Output Formats, open two windows. One for the new Compound Format and one for other Formats to be placed within it.

## Creating a Format

To create a new format:

1. From the Formatter window tree, select **Formats** and click the right mouse button. The **New popup** menu appears.
2. Hold down the right mouse button and move to the right to open the submenu. Choose one of the following:
  - To create a flat format, highlight **Flat**.
  - To create a compound format, highlight **Compound**.
3. Continue to hold down the right mouse button and move to the right to open the next submenu. Choose one of the following:
  - To create an input format, highlight **Input**.
  - To create an output format, highlight **Output**.

Release the right mouse button. A new icon appears followed by a blank text box.

4. Type a unique format name and press *Return*. The name is alphabetically inserted into the list of formats and the associated property sheets are displayed in the right pane.

The property sheets vary, depending on the type of format you are creating. The different formats and their associated property sheets are described in the following sections.

### Tip

You can change the name of a format by selecting the format name, then click again on the name. A box appears around the selected format name and the name is highlighted. Type the new name (maximum of 32 characters) and press *Return* to enter the name and position it in the list. Note that names cannot contain single quotes.

## Creating a Flat Input Format

Use the Input Flat Format property sheet to build a list of fields and associated parse controls for an input message. The information in this dialog box matches the input message string sent to Formatter.

To add a flat input format:

1. Follow the steps in the *Creating a Format* section on page 63, selecting flat and input from the popup menu. The flat input format dialog box appears.

*Flat Input Format Dialog Box*

2. To specify the component format order as Random (fields can appear in any order), select the Random checkbox. The default order is ordinal (the fields appear in the specified order in the input message).
3. Click in the Format Termination field to open the drop-down list and select the termination type. Format Termination types are described in the following table.

Format Termination Value	Description
Not Applicable	No data termination. Read to end of message.
Delimiter	The format is terminated by a delimiter.
Exact Length	The format has a fixed length.
White Space Delimited	The format is terminated by a white space.
Minimum Length + Delimiter	Parse a minimum number of characters and then look for delimiter.
Minimum Length + White Space	Parse a minimum number of characters and then look for a white space.

4. If the termination type is Delimiter or Minimum Length + Delimiter, select a literal from the Delimiter drop-down list.

If the termination type is Exact Length, Minimum Length + Delimiter, or Minimum Length + White Space, enter the fixed length of the field or the minimum number of characters to parse before looking for a delimiter or white space in the Length field

5. To add fields, select the format in the tree and right click. The popup menu appears.
6. From the popup menu, select Add Field Components. The Field window appears, displaying a list of the fields.
7. Select the fields that you want to add, then choose the Accept Selection button. The fields are added to the format tree.
8. When selecting each component for the format, make sure each has associated input controls and fields on their Property sheets. Select fields and input controls from the Field Name and Input Control Name drop-down lists.

### Note

When a flat input format is changed by either adding or deleting a field, it is a global change. Any compound format the flat format is contained in also reflects the addition or deletion of the field.

## Creating a Compound Input Format

Compound Input Formats are composed of other flat and compound input formats. You can break down complex formats into many individual formats to simplify parsing message data.

You create compound input formats by adding flat or other compound formats to the format. For example, you might have three flat input formats: Header, Detail, Trailer. A compound input format could look like:

Format Name	Component Format	Characteristics
CompleteFormat	Header	N/A
N/A	Detail	Repeating, Ordinal
N/A	Trailer	N/A

The component formats of a compound format are either optional or mandatory. An optional component can have missing mandatory components and the parse or reformat will still succeed. A mandatory component must exist, and all its mandatory components must also exist, or the parse or reformat will fail.

The order of component formats in an input compound format are ordinal, tagged ordinal, or alternative. Ordinal specifies that component formats must appear in the specified order in the input message. Tagged ordinal format is a flat ordinal format where the first field is a literal. Alternative means that only one of the component formats can appear in the message.

### Note

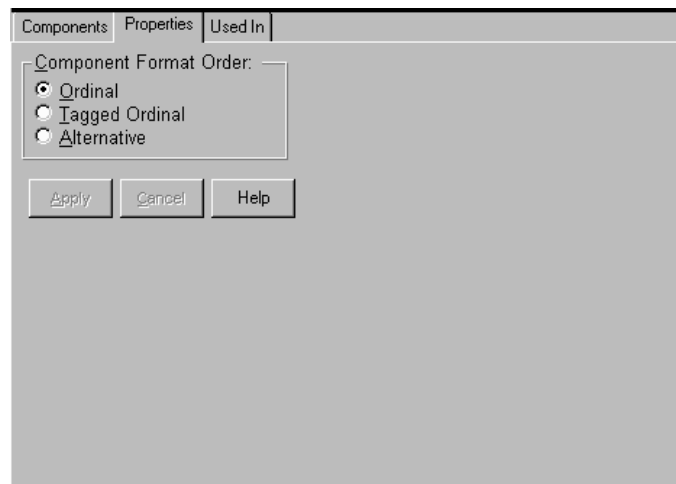
We strongly suggest you not use apostrophes or quotes in format names.

### Note

You cannot create compound input formats that contain themselves. These are known as recursive. If you see a recursion icon (two arrows creating a circle), it is most likely that something has corrupted the data in your Formatter database.

#### To add compound input format:

1. Follow the steps in the *Creating a Format* section on page 63, selecting compound and input from the popup menu. The compound input format property sheet appears.



*Compound Input Format Property Sheet*

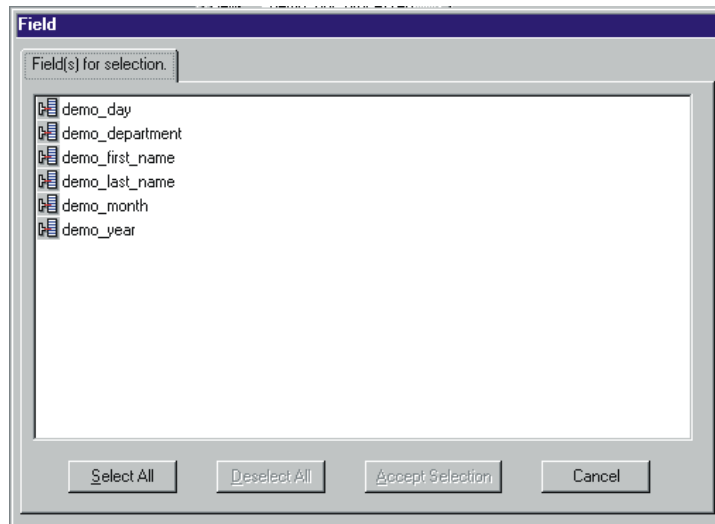


2. In the Component Format Order section, choose one of the following:
  - To have compound formats appear in the specified order in the input message, select ordinal.
  - To have the first field of the format be defined as a literal, select Tagged Ordinal.
  - To have only one component format appear in the message, select Alternative.

### Note

For each parse or reformat of an alternative format, a different component of the alternative format can apply.

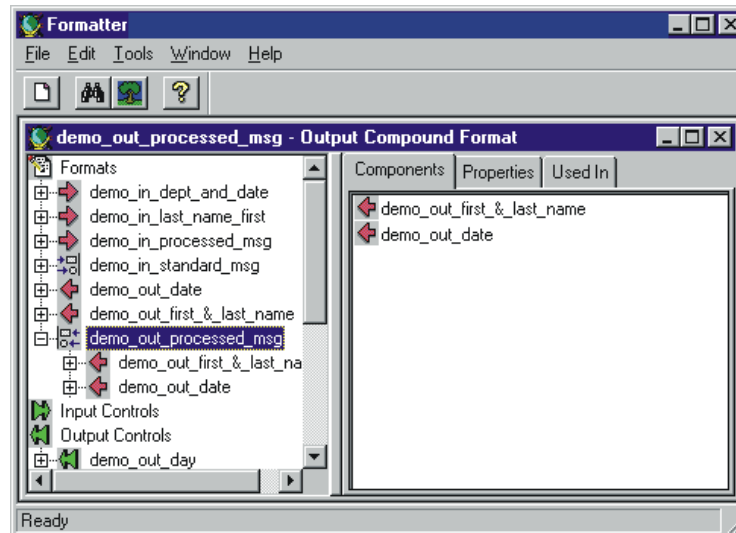
3. To add components, select the format in the tree and right click. The popup menu appears.
4. From the popup menu, select Add Component Formats. The Components window appears displaying a list of the available components.



*Components Window*

5. Select the format components that you want to add, then choose the Accept Selection button.

6. To change the properties of the component you just added, select the component in the left pane. The associated property sheet appears in the right pane.



*Component's Property Sheet*

7. Choose one of the following from the properties tag:
  - To define the format as optional, select the Optional checkbox. If the format is optional, skip the rest of this procedure.
  - To define the format as repeating, select the Repeating checkbox. The Repeat Terminators field is enabled.
8. Click in the Repeat Termination field to open the drop-down list and select a repeat terminator. Repeat terminators are described in the following table.

### Repeat Terminators

Repeat Termination Value	Description
Not Applicable	No format termination. Read to end of message.
Delimiter	Repeating group is terminated by a delimiter.
White Space	Repeating group is terminated by a white space.
Exact Count	There is an exact count of repeating formats. When you choose Exact Count, you must then indicate the number in the Repeat Count field.

Repeat Termination Value	Description
Field contains repeat count	The field has a repeat count. The field containing the repeat count is defined in the Repeat Field. The defined repeat field should also be dragged and dropped into the format. The location of the field in the format's field hierarchy does not matter (the field must exist in the message someplace prior to the repeating component).

9. Choose one of the following:
- If you select Delimiter as a termination type, select one from the Delimiter drop-down list.
  - If you select Exact Count, indicate the repeat count in the Repeat Count field.
  - If you select Field Contains Repeat Count, specify the field in the Repeat Field drop-down list. After specifying the field, drag and drop the field into the compound format Components tab.

### Note

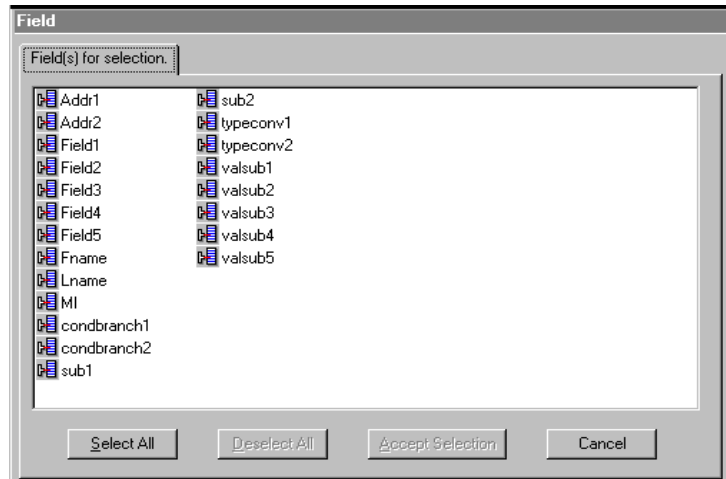
For information on the Used In tab, refer to the *Used In* section on page 22.

## Creating a Flat Output Format

Use the Flat Output Format property sheet to build a list of fields and associated parse controls for an input message. The information in this dialog box matches the input message string sent to Formatter.

1. Follow the steps in the *Creating a Format* section on page 63, selecting flat and output from the popup menu. The flat output format Fields property sheet appears.
2. To add fields, select the format in the tree and right click. The popup menu appears.

- From the popup window, select Add Field Components. The Fields window appears.



*Fields Window*

- Select the fields that you want to add, then choose the Accept Selection button. The fields are added to the format tree.

### Note

When a flat output format is changed by either adding or deleting a field, it is a global change. Any compound format the flat format is contained in also reflects the addition or deletion of the field.

#### To change the properties for fields:

- To be sure each field has an associated output control and input field name (if appropriate), select fields and output controls from the Input Field Name and Output Control Name drop-down lists.
- If the access mode you want is other than Not Applicable, select the appropriate access mode from the Access Mode drop-down list.

## Access Modes

Each output field has an associated Access Mode. Access Modes define how Formatter accesses fields in the input message to generate fields in the output message. You select output field access modes and associated input field names to tell Formatter how to map fields from the input message to fields in the output message.

The following table provides descriptions the access modes supported by Formatter.

### Access Modes

Access Mode	Description
Not Applicable	Do not access any field instance.
Normal Access	Access the instance in the same repeating component as the current controlling field instance. If there is no controlling field, access the first instance. This behaves just like Access sibling instance.
Access with Increment	A field with this access mode is the controlling field for the repeating component (see <i>Controlling field</i> ).
Access Using Relative Index	The first field in a repeating component that Formatter encounters with this access mode is the controlling field for the repeating component (see <i>Controlling field</i> ). Any other field in the repeating component with this access mode behaves as if it has access mode Access sibling instance or Normal access (access the sibling of the controlling field).
Access nth instance of field	Access the nth instance (n = 0 means get the first instance) of the field in the input message.
Controlling field	This field is the controlling field for the repeating component. On each repetition, access the next field instance that is still a child of the current controlling field instance of the parent format. If there is no parent controlling field, the repetitions end with the last field instance from the input message.
Access current instance	Access the same field instance as on the previous access (the first access will get the first instance of the field).
Access next instance	Access the next field instance relative to the previous access.
Access parent instance	Access the instance that is the first ancestor of the current controlling field instance.
Access sibling instance	Access the instance in the same repeating component as the current controlling field instance. If there is no controlling field, access the first instance.

## Field Map

When building a flat output format, you have the option of mapping a field in the input format to a field with a different name in the output format by selecting the Field Map tab. If you do not want to specify a mapped name, the input fields are mapped to output fields by matching names.

### To map an input format field to an output format field:

1. With the output format selected, choose the Field Map tab. The Field Map property sheet appears.

### Note

The output format must have defined and associated output fields before you can map to input fields.

*Field Map Property Sheet*

2. Click the Format Name field to open the drop-down list and select the desired input format. The fields for the selected format are displayed in the Input Fields list box.
3. Select the desired field from the list box. Holding down the mouse button, drag the field to the appropriate Mapped From cell.

The input field is now mapped to an output field. In the example Field Map property sheet above, output field 2 maps to input field 1. The other fields map to identical fields.

You can map as many fields as you want by repeating the drag-and-drop process.

## Creating a Compound Output Format

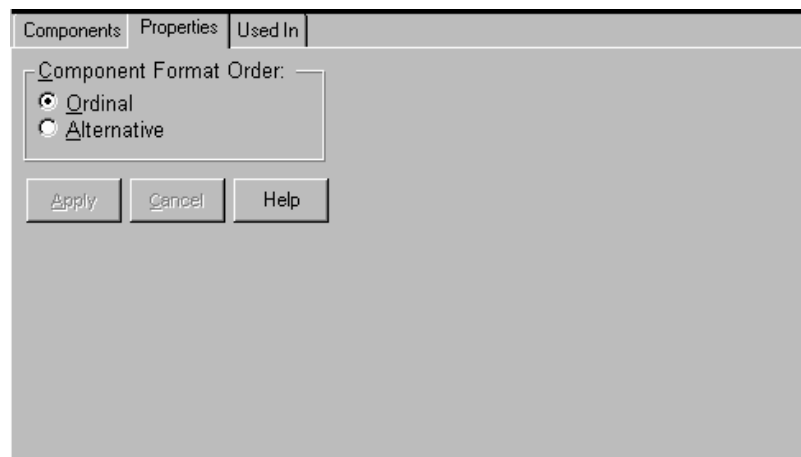
Use compound output formats to combine several flat and compound output formats into a single output message format. Compound output formats are composed of lists of the component flat and compound formats describing the output message.

You create compound output formats by adding flat or other compound formats to the format. For example, you might have three flat output formats: Header, Detail, Trailer. A compound output format could look like:

Format Name	Component Format
CompleteFormat	Header
	Detail
	Trailer

### To add a compound output format:

1. Follow the steps in the *Creating a Format* section on page 80, selecting compound and output.
2. A new compound output format component is added to the left pane and the Components and Properties sheets are displayed in the right pane. The cursor is positioned in the text box where you type the new, unique format name. Format names should be descriptive.



*Compound Output Dialog Box*

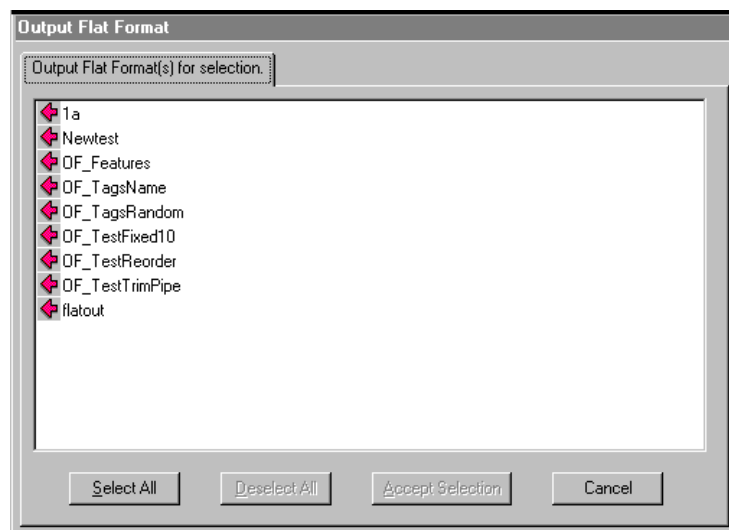
3. Click any mouse button outside the name text box or press *Return* to finish defining the format. The new format is highlighted and moves to its alphabetical location in the list.

4. To select the order for the component formats in the compound, select one of the following:
  - To have component formats appear in the specified order in the input message, select Ordinal.
  - To have only one component format appear in a message, select Alternative. See *Alternative Output Formats* on page 80 for more information.

### Note

For each parse or reformat of an alternative format, a different component of the alternative format can apply.

5. To begin adding flat and compound format components, select the Components tab. The Components sheet appears.
6. Select the format in the tree and right click. The popup menu appears.
7. From the popup menu, select Add Component Formats. The Components window appears.



*Components Window*

8. Select the components that you want to add, then choose the Accept Selection button. the components are added to the format tree.

#### To change the properties for component formats:

1. Select that component in the left pane. The associated property sheets appear in the right pane.
2. Select the Properties tab.



3. Check the boxes for one or both of the following:
  - To define the format as optional, select the Optional checkbox. (If the format is optional, skip the rest of this procedure.)
  - To define the format as repeating, select the Repeating checkbox. The Repeat Termination list becomes active.
4. Select a termination type from the Repeat Termination drop-down list.
5. If you select Delimiter as a termination type, select a literal from the Delimiter drop-down list.

## User-Defined Data Types

Use a user-defined data type to define your own data types (based on native data types) and associate your own validation logic with these types. The validation logic is executed for input fields defined as user-defined types. User validation logic is done in addition to native type validation. User-defined types can be assigned to both input and output fields, but user validation is done on input fields only.

For a list of valid native data types, refer to the table on page 34.

## Creating a User-Defined Data Type

**To create a user-defined data type:**

1. From the Formatter tree, select User Defined Data Types and press the right mouse button. A popup menu appears displaying the New menu item.
2. Select New. A new user-defined type component is added to the left pane and the user-defined data types property sheet is displayed in the right pane. The cursor is positioned in the text box where you must type a new, unique name.
3. Click any mouse button outside the name text box or press *Return* to finish defining the user-defined data type. The user-defined data type name is highlighted and moves to its alphabetical location in the list.

4. In the property sheet, click the Native Data Type field to open the drop-down list and select the desired data type. Data types are described in *Input Controls* on page 31.

*User-Defined Property Sheet*

5. Type the name of the validation routine in the Validation Routine field. See *User-defined Data Type Input Field Validation* in the **MQSeries Integrator Programming Reference for NEONFormatter** for details about validation routines.

**To change the name of a user-defined data type:**

1. Select the user-defined data type name.
2. Click on the name again. A text box appears around the selected name and the name is highlighted.
3. Type the new name.
4. Either press *Return* or click the mouse outside the name text box. The user-defined type name is highlighted and moves to its alphabetical location in the list.

## Using a User-Defined Data Type in an Input Control

When creating an input control, you can select a user-defined data type. An input control with a user-defined data type can have both an associated length and tag, but only the data itself can be user defined. The length and tag data cannot be user defined.

## The Validation Parameters Property Sheet

When you select a user-defined data type in an input control, the Validation Parameters property sheet appears. You can define the name of the argument for the runtime data lookup function. This function is one of the methods of the validation callback object. For more information on user callback, refer to the *User Callback API Function* section in the *MQSeries Integrator Programming Reference for NEONFormatter*.

The Validation Parameters property sheet also enables you to define name-value pairs. The name identifies the meaning of the value. For example, if the name is phone number, the value would be the actual number.

*Validation Parameters Property Sheet*

### **Creating a Validation Parameter**

To add a validation parameter:

1. From the Validation Parameter property sheet, click the Named Parameter Name field and type the name of the lookup function.
2. To add name value pairs, select the Add button. A text entry field appears.
3. Click the Name field and type the name of the function.
4. Either press *Tab* or click the Value field and type the value of the function.
5. Repeat steps 2-4 until you enter all the name value pairs.

### **Deleting a Name Value Pair**

To delete a name value pair, click the Name Value Pair field that you want to delete and select the Delete button. Both the name and value are deleted.

# Alternative Input and Output Formats

Alternative formats are a special form of compound format in which one format in a set of alternatives will apply to a message. For example, if an alternative format is named A, it may contain component formats B, C, and D. A message of format A may actually be of variation B, C, or D.

Exactly one of the alternatives must apply or the entire alternative compound format does not apply.

An alternative format can be used anywhere a format can be used, and each component format can be any kind of its respective input or output format.

## Optional Components and Fields

Optional components and fields are each characterized as optional within the context of a compound object.

For example, a compound format composed of component formats may have a mix of mandatory and optional component formats. And a format may have a mix of mandatory and optional fields.

If a mandatory component (of a format or field) is not present in a message, the compound or flat format does not apply. However, if an optional component is not present, Formatter continues to the next component.

## Alternative Input Formats

If an alternative input format is applied to an input message, the first component of the alternative format is compared to the message. If it parses with the first component format, parsing is finished. If parsing fails, Formatter tries the second component format. If that fails, it tries the next component format. If all components fail to parse, then the parse for the entire alternative input format fails.

## Alternative Input Formats and Parsing

If a format has optional fields and the fields are delimited and not tagged, it may be impossible to determine which of the optional fields occur in an input message. For example, if a simple space delimited format is:

```
[F1] F2 [F3] [F4]
```

The first, third, and fourth fields are optional. If an input message value1 value2 value3 is received, there is no way, without some rules to remove ambiguity, to determine if the message is actually F1 F2 F3, F2 F3 F4, or F1 F2 F4.

Alternative input formats enable you to specify all possible configurations of mandatory and optional fields in a format. You must explicitly define all possible combinations to avoid possible parsing errors because Formatter does not recursively try all combinations by itself.

As an example, if you have the following input format:

Field 1: optional, comma delimited

Field 2: mandatory, colon delimited

Field 3: optional, comma delimited

Field 4: optional, forward-slash delimited

The input message "field1,field2:field3,field4/" parses into four fields correctly.

However, "field1 field2:field3,field4/" fails. The first field of the format is comma delimited and parsed as "field1 field2:field3". The second field is colon delimited, so Formatter looks for a colon in "field4/", detects the end of the message, and fails to parse.

It may seem simple to figure out that since the first field is optional, ignore it and try to parse the second field, but Formatter does not see that. You have to explicitly define all possible combinations (F1 F2 F3, F2 F3 F4, F1 F2 F4). In this case, the second alternative format would have three fields, starting with the mandatory colon-delimited field.

For alternative formats, you determine the order alternative components are parsed in. Remember that components are taken on a first-parsed, only-parsed basis. With that in mind, component order is critical.

For example, if you have a message "abcde," you could specify two alternatives (or more). It could be parsed into a 5-byte field or two separate fields, one 2-bytes long and one 3-bytes long. If the 5-byte field format is the first alternative, you will never parse using the second format. If two parses are valid for the same input, only the first occurs.

## Alternative Output Formats

To format an alternative output format, Formatter attempts to create the first component of the alternative format. If creating the first component is successful, formatting is finished. If it fails, Formatter tries to create the second component, and so on. If all components fail to be created, formatting the alternative output format fails.

For information on using alternative formats in Formatter, refer to the section *Output Controls* on page 44.

### Note

A component is not created if a component or field mandatory for the output format is not present in the incoming message.

## Tagged Input Formats

A compound format can have a property of “Tagged Ordinal.” This means that the first field in each component format is a literal. The component format can be flat or compound.

### Tagged Input with Alternative Component Example

Tagged input formats can be very useful when used in conjunction with alternative formats. The following is an example of what might come in the data segment of a SWIFT message.

```
“:10:f1 f2 f3<CRLF>:20:C/1234/<CRLF>first description<CRLF>second
description<CRLF>:30:f4,f5<CRLF>”
```

The following is a loose definition of the format:

```
:10: field1 field2 field3 <CRLF>
```

```
:20: [C/acctnum/< CRLF>] (optional)
```

```
desc1 <CRLF>
```

```
[desc2 <CRLF>] (optional)
```

```
:30:first, second <CRLF>
```

Parse segment :20: using the following rules:

- If there are two slash-delimited fields before the <CRLF>, the credit code and account number exist.
- If not, continue with the mandatory desc1 field, followed by an optional desc2 field.
- Do not run into the :30: segment by parsing “:30:first, second” into the <CRLF>-delimited desc2 field when the desc2 field is actually not present, since it is optional.

With tagged formats, we now have a way to ensure that we do not overrun the boundaries of the :20: segment. Any trailing optional fields in a tagged flat format can be parsed or determined to be absent by parsing only up to the component boundary, instead of looking for a field delimiter (or other termination) beyond the component boundary.

Define a compound tagged format with three components.

Segment10 :

First Field Literal “:10:”  
 space-delim field  
 space-delim field  
 <CRLF> delim field

Segment20 : Alternative format with two components

Segment20\_1 (this is the first alternative component)

First Field Literal “:20:”  
 Credit Code : slash delim, mandatory  
 AcctNum : /<CRLF> delim, mandatory  
 desc1 : <CRLF> delim, mandatory  
 desc2 : <CRLF> delim, optional

Segment20\_2 (this is the second alternative component)

First Field Literal “:20:”  
 desc1 : <CRLF> delim, mandatory  
 desc2 : <CRLF> delim, optional

Segment30 :

First Field Literal “:30:”  
 comma delim field  
 <CRLF> delim field

When parsing a Segment20, the parser first attempts to parse Segment20\_1. If it fails, (the credit code and account number are not part of this particular :20: segment), it parses the second alternative.

You could have different first field literals for 20\_1 and 20\_2, for example, :20A: and:20B:, like SWIFT sometimes does.

A tagged input format can be included as a component of a compound, where the other components can be any other kind of input format. For SWIFT, you might define a SWIFT 570 message as a compound ordinal of three components.

Basic Header : Ordinal Flat Format  
 570 Data Segment : Tagged Compound Format  
 Trailer : Random Tagged Flat

Note that tagged formats do not apply to output formats. An output format can certainly have a first field literal, but calling it a tagged format doesn't gain us anything. It is only when we need an additional way to determine message boundaries on parsing and input message where first field literals become more explicitly necessary.

## Alternative Output Format Example

Creating an output format is perhaps conceptually clearer. You define a set of alternatives, then define an alternative compound having all of the alternatives as components, sequenced in order of desirability. As with input formats, it may be appropriate to order alternatives from specific to general. (This is not always the case. Many times order is irrelevant on output because of mutually exclusive mandatory fields, that is, there is only one format that applies.)

Using the previous description of a :20: segment, assume we need to create a :20: segment instead of parse it. We do not know what kind of input message we had, whether it was a Segment20\_1, Segment20\_2, or some other input format that allowed a parse of some or all of the fields required for a segment 20.

You cannot just say you have some optional leading fields CreditCode and AcctNum, since you need to put in a <CRLF> if the fields exist, but not if they don't. While not the only way, alternative output formats make this easy.

Create a set of (two) alternatives.

Seg20\_1 1literal :20:

- 2CreditCode Mandatory with / delim
- 3AcctNum Mandatory with /<CRLF> delim
- 4desc1 Mandatory with <CRLF> delim
- 5desc2 Optional with <CRLF> delim

Seg20\_2 1literal :20:

- 2desc1 Mandatory with <CRLF> delim
- 3desc2 Optional with <CRLF> delim

Notice the similarity between the two. It is easy to go from most specific to more general by using the GUI Save As and Field Delete features, given this kind of situation. By having several mandatory fields in a format, you are ANDing their existence together. All must be present to create the given format. If you sequenced Seg20\_2 before Seg20\_1 in the alternative output (compound) format, Seg20\_1 would never be applied.



---

## Chapter 3

# NEON Rules

---

NEONRules evaluates the contents of a message (which is a string of data) and uses the evaluation results to perform actions on the message. The following describes what constitutes a rule and a method to define rules through the GUI. Management APIs can also be used to define rules.

Each rule has a list of evaluation criteria (called an expression) made up of fields from the message and associated Rules operators linked together with Boolean operators. Field names are defined by the user through the Formatter definition mechanism. Fields can be compared either against constant data or other fields within a message.

Rules are defined within an application group/message type pair. Rules are uniquely identified by the application group/message type/rule name triplet. Rule names are defined by the user.

Rules application groups allow you to easily maintain rules associated with business needs. An application group is a logical grouping used to organize rules. For example, a company may split rules into groups by projects or a company may split projects into logical sub-groups.

Each application group can contain several message types and a message type can be in more than one application group. Message types are defined by the user through the Rules definition mechanism. A message type defines the layout of a string of data. When using the Formatter, the message type is the same as the input format name.

When a rule evaluates true, Rules provides a mechanism to retrieve associated actions that should be processed by the application. These actions can be thought of as computer commands and the associated parameters (options) needed to execute them.

A rule can have multiple subscriptions and each subscription can have multiple actions. A subscription is created in the Subscription list, then assigned to one or more rules within an application group/message type. A subscription is defined by an application group/message type in the same way as a rule. Subscriptions are groupings of actions processed when a rule evaluates true. Without being assigned to a rule, an action can not be executed. If a subscription is shared by more than one rule, Rules only retrieves the subscription once during evaluation, even if more than one of its rules evaluates to true.

The user defines actions and option name/option value pairs, although MQSeries Integrator provides two predefined actions with associated options: *putqueue* and *reformat*.

# Starting Rules

To access Rules:

1. Double-click the Rules icon. The Rules Logon dialog box appears.

*Rules Logon Dialog Box*

## Note

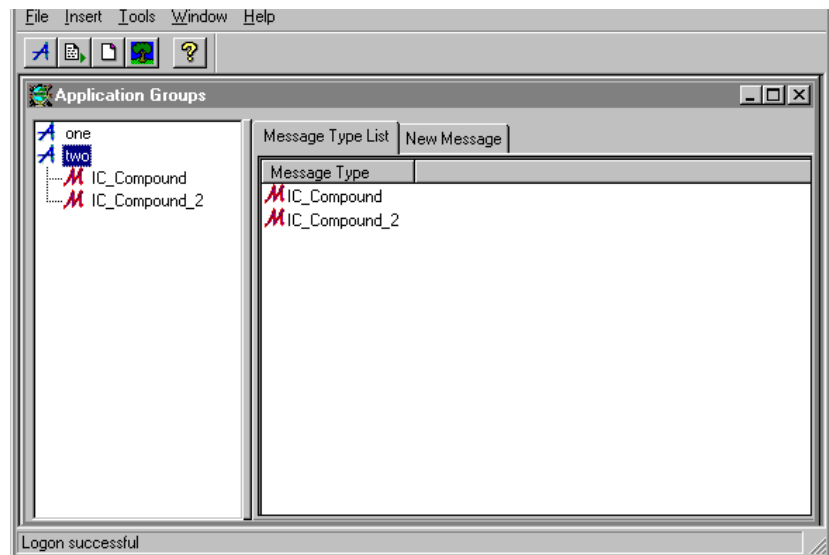
If all defaults are correct, click the Password field, type your password, and choose the OK button.

2. Click in the User ID field and choose one of the following:
  - Press *Tab* to accept the User ID displayed.
  - Type your User ID and press *Tab*.
3. Type the password associated with the User ID and press *Tab*.
4. Choose one of the following:
  - Press *Tab* to accept the server currently displayed.
  - Type the server name you want to connect to and press *Tab*.
5. Type the name of the desired database and press *Tab*.

## Note

Oracle users should leave the database field blank.

6. Choose one of the following:
7. To accept the DBMS currently displayed, skip to step 7.
8. Click on the arrow to open a drop-down list, select the desired DBMS.
9. The DBMS client appropriate for your DBMS must be installed before you can run the GUI. If you do not know the DBMS you need to connect to, ask your System Administrator.
10. Choose the OK button. The Rules main window appears with the Rules tree window open. (When you have built the hierarchy within application groups, your Rules tree window will resemble the following screen.)

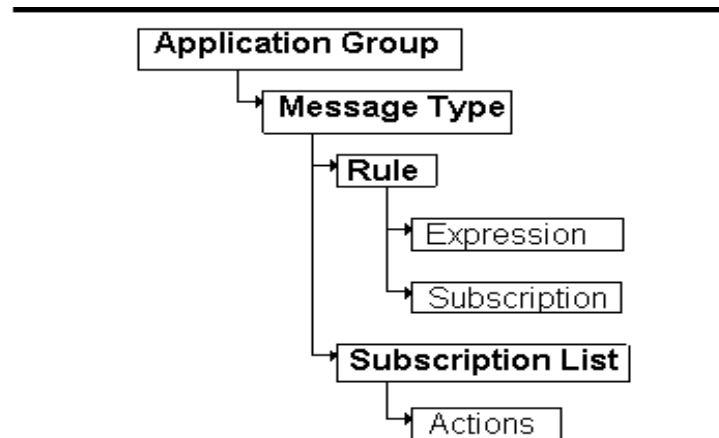


*Rules Window*

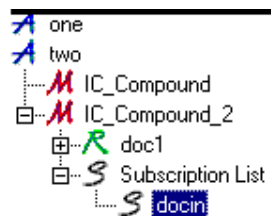
## The Rules Window

The Rules window is designed to work like the Microsoft Explorer. Rules components are displayed in an hierarchical or tree-structured organization in the left pane, with Application Groups at the top level of the hierarchy. The right pane contains tabbed property sheets that are associated with the selected object.

The Rules hierarchy is:



To expand the root object Application Groups, double-click it. The small boxes with a plus sign (+) next to the objects in the left pane indicate that there may be other items subordinate to the object, but the subordinate items are not displayed. Click the plus sign to expand the tree to display the subordinate items. When the tree is expanded, the plus sign changes to a minus sign (-). To collapse the tree, click the minus sign.



*Expanded Tree*

Detailed information about the selected component is kept in tabbed property sheets. A property sheet is similar to a dialog box; you must enter information in mandatory fields. If you have Update permission, you can add, change, and delete components through property sheets.

## Opening a New/Additional Rules Window

You can open more than one Rules window at a time. To open a new window, choose File→New from the menu bar.

### Note

When you create a new window, you do not create a new connection to the database. You are always connected to the same database.

## Tree Functionality

Depending on the component or element selected in the tree, when you click the right mouse button, a popup menu appears containing commands such as New, Delete, or Duplicate. For more information on popup menus, refer to the section *Rules Popup Menus* on page 91. Using the popup menu, you can add, delete, or duplicate a new rule or subscription.

## Tabbed Property Sheets

For each object in the tree, there are tabbed property sheets. The following table describes the property sheets available for each object.

Object	Tabs	Tab Description
Application Groups	Message Type List	A list of message types included in this application group.
	New Message	A property sheet containing a list of the input message types available to be added to the application group.
Message Types	Rule List	List of rules included in the selected message type.
Rules	Subscription List	A list of subscriptions for the selected rule.
	Security	A property sheet that enables you administer users' and PUBLIC permissions for the selected rule.
	Expression	A property sheet that enables you to create, change, delete, and insert arguments for the selected rule.
Subscriptions List	Subscription List	Contains subscriptions that can be applied to rules.
Subscriptions	Used In	A list of rules corresponding to the selected subscription.
	Security	A property sheet that enables you administer users' and PUBLIC permissions for the selected subscription.

Object	Tabs	Tab Description
	Actions	A property sheet that enables you to add, insert, and delete actions for the selected subscription.
	Misc	A property sheet containing a comment box where you can add, modify, or delete comments for the subscription.

## Security

When you add either a rule or subscription, you are assigned ownership and update permissions. You must set permission for PUBLIC users to update the rule or subscription (Read permission is the default). You can also change the ownership of a rule or subscription that you own to another user. For more information, see the *Security* section for rules on page 100 and for subscriptions on page 112.

## Case Sensitivity

### WARNING!

If you are using a case-insensitive database, you cannot name components the same with only a change in case to identify them. For example, you cannot name one rule "r1" and another rule "R1". In a case-insensitive environment, make each item unique using something other than case differences.

If importing components exported from a context-sensitive database into a context-insensitive database, these differences will cause NNRie to fail during import if a conflict arises between two components named the same with only case differences. See the *MQSeries Integrator System Management Guide* for information on NNRie.

Also, case-sensitive operators (see *Rules Operators* on page 103) may not work correctly on case-insensitive databases.

See the *MQSeries Integrator System Management Guide* for information on how to change a current case-insensitive installation to be case sensitive.

# Rules Menus

The Rules GUI features a menu bar, toolbar, and popup menus. The menu bar contains items that let you open, close, and arrange windows, display and move the toolbar, exit Rules, and view Rules version information.

The tree popup menu lets you create, delete, and duplicate a rule and its components.

The tab popup menus contain a variety of options such as View and Arrange from the Rules tab and Insert, Update, Delete from various other tabs.

Rules menus can also be accessed using your keyboard by pressing Alt plus the letter underlined in the menu name. For example, you can open the File menu by pressing *Alt+f*. After the menu is opened, you can access its functions by pressing the underlined letter of the function. For example, to open a new Rules window from the File menu, type N (the underlined letter in the function New).

## Note

Windows has a Cut, Copy, and Paste menu that appears when editing text. This is not a Rules menu. Refer to your Windows documentation for the procedures to use this menu.

---

## The Rules Menu Bar

The Rules menu bar contains the following menus:

## Note

If only the Rules window is open (the tree window is closed), only the File and Help menus appear.

---

- File
- Insert
- Tools
- Window
- Help

## The File Menu

The File menu contains the following submenus:

- New
- Close
- Print Setup
- Print Tree
- Exit

These menu are described in detail in the following sections.

### ***New***

The New function opens a new Rules window. The new window connects to the same database as the previous window. To open a new window, choose File→New.

### ***Close***

The close function closes the active window. To close the active window, choose File→Close.

### ***Print Setup***

Use the Print Setup function to select the desired printer for output. You can also designate the print orientation and the number of copies.

### ***Print Tree***

Use the Print Tree function to display and print the application groups with their properties. To print the tree, choose the Print Tree icon on the toolbar, File→Print Tree or the Print button in the Print Tree window.

### ***Exit***

The Exit function closes Rules. To exit Rules, choose File→Exit.

## The Insert Menu

The Insert menu contains the New Application function. This function inserts an application group text box at the top of the application group tree where you can enter a new application group name.

## The Tools Menu

The Tools menu contains the following submenus:

- Security Summary Report
- Customize Toolbars



### ***Security Summary Report***

The Security Summary Report function prints a report, showing who has what permission for all of the rules in use. The permissions are grouped by application group and message type and then each rule is listed in the group. Each rule lists the owner's name, user names, and their permission(s).

The report is sent to your designated printer as soon as you choose this function. The report is not displayed on your screen.

### ***Customize Toolbars***

The Customize Toolbars function lets you turn the toolbar either off or on, position it in your window and select the button size.

## **The Window Menu**

The Window menu lets you cascade, tile either vertically or horizontally, or layer windows. For more information, refer to your Windows documentation.

## **The Help Menu**

The Help menu contains Help Topics and About submenus. Use the Help Topics submenu to access Rules online help. The About submenu displays MQSeries Integrator copyright and version information plus database, DBMS, and server information.

## **Rules Popup Menus**

Both the right and left window panes, under certain conditions, contain popup menus. These menus are selected by pressing your right mouse button.

### **Left Pane**

Depending on what is selected in the left window pane, when you press the right mouse button, a popup menu appears. The popup may contain any of the following functions:

- New
- Delete
- Duplicate
- Print Tree

**New**

Use the New function to create a new rule or subscription.

The popup menu allowing you to create a new rule appears when you select a message type and hold down the right mouse button.

The popup menu allowing you to create a new subscription appears when you select a rule and hold down the right mouse button.

**Delete**

Use the Delete function to delete the selected rule or subscription.

If you have either a rule or subscription selected and press the right mouse button, the popup menu containing the Delete and Duplicate functions appears.

If you are not the owner or do not have Update permission for a rule or subscription, the Delete function is inactive, preventing you from deleting the rule or subscription.

**Duplicate**

Use the Duplicate function to duplicate rules and subscriptions. When you select Duplicate, a text entry box appears at the top of the component list. You must enter a unique name. The component's property sheet displays the properties of the rule or subscription that you duplicated. You can change any of these properties.

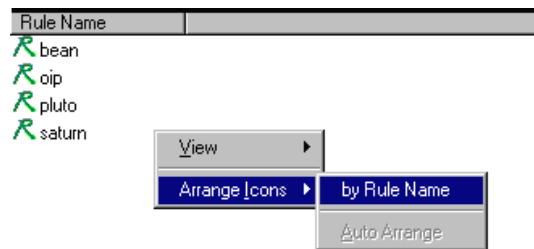
**Print Tree**

Use the Print Tree function to display and print the application groups with their properties. To print the tree, choose the Print Tree icon on the toolbar, File→Print Tree or the Print button in the Print Tree window.

**Right Pane**

The right window pane contains a popup menu when a component list is displayed. This popup menu contains the following functions:

- View
- Arrange Icons



*Right Window Pane Popup Menu*

### **View**

The View function lets you view icons either large or small (the default). You can also select to arrange the icons in a list (the default).






If you select to view large icons, they are displayed from left to right, rather than in a list. To see the icons in a list, select the List function.

### **Arrange Icons**

The Arrange Icons function lets you change the order of the icons. By default, the icons are alphabetically displayed from a-z and numerically displayed in ascending order. When you select Arrange Icons by (for example, Rule Name), the icons are then arranged alphabetically from z-a and numerically in descending order. The sort order is case sensitive.

## **Rules Toolbar**

Use the Rules toolbar to quickly execute the functions shown in the following table.

<b>Toolbar Icon</b>	<b>Description</b>
	Inserts a text box at the top of the tree for you to enter a new Application Group name.
	Prints the Security Summary Report. For more information, see the <i>Security Summary Report</i> on page 91.
	Opens a new Rules window. For more information, see <i>Opening a New/Additional Rules Window</i> on page 87.
	Opens the Print Tree window. For more information, see <i>Print Tree</i> on page 21.
	Opens the online Help Topics. For more information, see <i>The Help Menu</i> on page 91.

# Building Rules

## Application Groups

An application group is a method that lets you logically organize rules associated with a particular subject. For example, an application group could be the Accounting Department of a company.

### Adding an Application Group

To add an application group:

1. Choose one of the following:
  - Select the New Application icon from the toolbar.
  - Select Insert→New Application from the menu bar.

A text box appears at the top of the application group list. The cursor is positioned in the application group text box where you can type the new, unique application group name.

#### **WARNING!**

If you are using a case-insensitive database, you cannot use the same name with a change in case to identify components. For example, you cannot name one application group D1 and another d1. In a case-insensitive environment, make each item unique using something other than case differences.

2. Press *Return* to add the application group.

#### **Note**

If you click any mouse button outside the application group text box (and you have not pressed *Return*), the application group is added.

The application group is highlighted and alphabetically positioned in the list in the left pane and the New Message property sheet appears in the right pane.

An application group should contain at least one message type, which corresponds to your input formats. For the procedure to add a message type, refer to the section *Adding a Message Type* on page 95.

### **Renaming an Application Group**

To change an application group name, select the name that you want to change and click on the name. A box appears around the name and the name is highlighted (see the figure below). Type the new, unique name and press *Return*.



*An Application Group Selected for Renaming*

## **Message Types**

The message type is the input format name from Formatter. A rule is evaluated based on the message type.

An application group should contain at least one message type. For the procedure to add an Application Group, see *Adding an Application Group* on page 94.

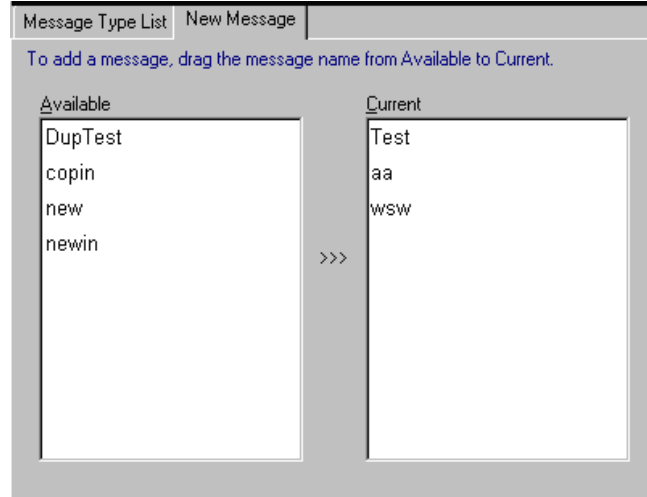
### **Adding a Message Type**

Use the New Message property tab to add a message type to the selected application group.

**To add a message type:**

1. Select the application group that you want to add a message type to. The list of current message types and the New Message tab is displayed in the right pane.

2. Select the New Message property sheet.



*New Message Property Sheet*

The Available box on the left displays available message types. This list contains the input format names defined in Formatter.

The Current box on the right displays the message types currently associated with the application group.

3. To add a message type to the selected application group, select the message type from the Available list box, hold down the left mouse button and drag the message type to the Current list box. The message will appear both in the current list box and in the left pan below the Application Group.

### **Note**

Each message type should contain at least one rule. For the procedure to add a rule, see the section *Adding a Rule* on page 97.

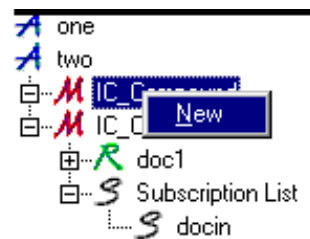
# Rules

A rule contains subscriptions which allow you to define message destination IDs, receiver locations and message formats, and any processes initiated upon message delivery.

## Adding a Rule

To add a rule:

1. From the Rules window tree, select the message type that you want to define a rule for and click the right mouse button. A popup menu appears with the New menu item.



*Add New Rule*

2. Select New. The cursor is positioned in the rule text box where you can type the new, unique rule name.

### WARNING!

If you are using a case-insensitive database, you cannot use the same name with a change in case to identify components. For example, you cannot name one rule R1 and another r1. In a case-insensitive environment, make each item unique using something other than case differences.

3. Press *Return* to add the rule.

### Note

If you click any mouse button outside the rule text box (and you have not pressed *Return*), the rule is added.

The rule is highlighted and alphabetically positioned in the list to create a new rule in the left pane. The Expressions property sheet appears in the right pane. In addition to the Expression property sheet, you can access the Subscription List and the Security property sheets via their tabs.

For the Rules engine to correctly process a rule, you must define the rule's expression, subscription, and user permissions. The procedure to add an expression to a rule is described in the *Expressions* section on page 102. The procedure to add a subscription to a rule is described in *Subscriptions* on page 108. The procedure to add user permissions to a rule is described in the *Security* section on page 100.

## Deleting a Rule

When you delete a rule, the expression and links to subscriptions belonging to the rule are also deleted. The subscriptions are not deleted.

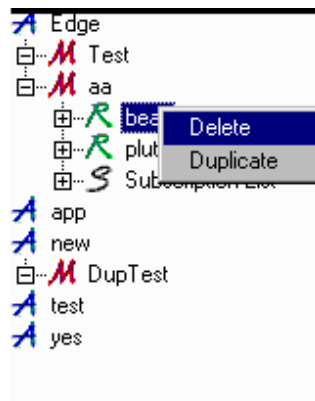
You must be the Rule owner and have Update permission to delete a rule.

### Note

If you do not have Update permission, the rule cannot be deleted. If you own the rule, the rule can be deleted. If you do not own the rule, the rule is disabled. For more information on enabling and disabling rules, refer to the *Security* section on page 100.

#### To delete a rule:

1. Select the rule that you want to delete.
2. Hold down the right mouse button to activate the popup menu and select Delete. The Delete confirmation box appears.



*Deletion of a Rule*

3. To delete the rule, choose the OK button. The Delete box closes and the rule is deleted.



## Duplicating a Rule

The Duplicate function lets you duplicate the selected rule, which includes the rule's associated expression and its links to subscriptions. The new rule is owned by the current user who has Update permission.

### To duplicate a rule:

1. Select the rule that you want to duplicate.
2. Hold down the right mouse button to activate the popup menu and select Duplicate. A text entry box appears at the top of the rules list.



*Duplicate Rule Text Entry Box*

3. Type a unique rule name in the text box and press *Return* to save the name.

### WARNING!

If you are using a case-insensitive database, you cannot use the same name with a change in case to identify components. For example, you cannot name one rule R1 and another r1. In a case-insensitive environment, make each item unique using something other than case differences.

### Note

If you click any mouse button outside the rule text box (and you have not pressed *Return*), the new name is saved.

4. To change the user permissions for the new rule, select the Security tab. For more information, refer to the *Security* section on page 100.
5. To change the rule's expressions, select the Expression tab and make the desired changes. For more information, refer to the *Expressions* section on page 102.
6. To change a rule's subscriptions, select the subscription and make the desired changes. For more information, refer to the *Subscriptions* section on page 108.

## Renaming a Rule

Use the Rename Rule function to change the name of a rule. You must have Update permission to change a rule's name.

To change a rule name, select the name that you want to change and click on the name. A box appears around the rule name and the name is highlighted (see the figure below). Type the new, unique name and press *Return* to save the name.

### WARNING!

If you are using a case-insensitive database, you cannot use the same name with a change in case to identify components. For example, you cannot name one rule R1 and another r1. In a case-insensitive environment, make each item unique using something other than case differences.



*Rule Selected for Renaming*

## Rule Security

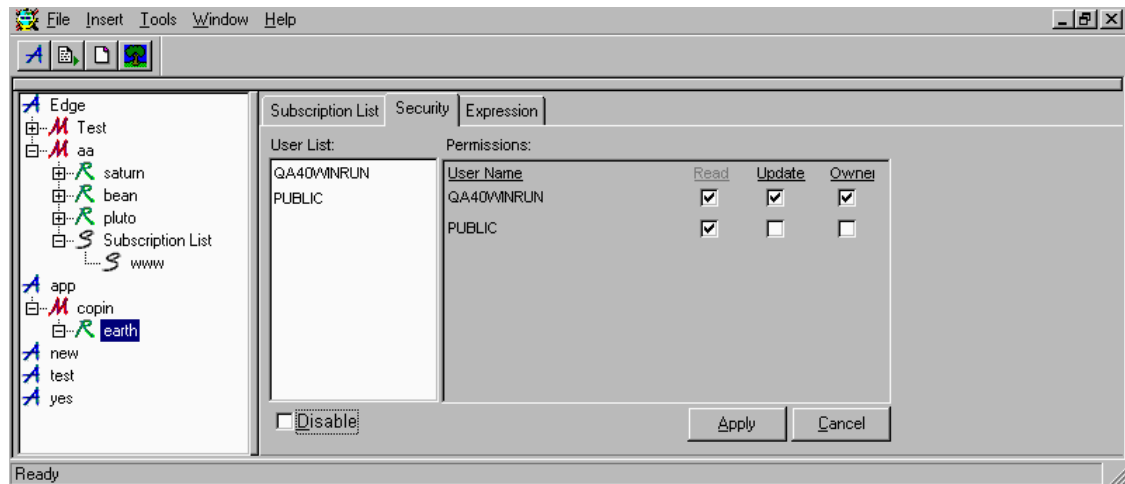
Use the Security property sheet to enter permissions for an individual rule or subscription. When you add a rule, you are assigned ownership, and must set the permissions for the rule. By default, Rules designates the creator of a rule as the owner, who is given read and update privileges. To give other users the ability to change your rule, give PUBLIC user Update permission. PUBLIC defaults to read only. Read permission is the minimum security currently allowed. The Read checkbox is marked by default and cannot be changed.

When a user deletes a rule and has Update permission, but does not own the rule, the rule is deactivated instead of deleted. The rule appears in the tree, but cannot be changed because it has been deactivated. Users can see that a rule has been deactivated by opening the Security tab and viewing the Disable checkbox at the bottom of the window. A checked box indicates that the rule has been deactivated.

## Adding or Changing Rule Security

To add or change user permissions:

1. Select the rule that you want to add or change the user permission for.
2. Choose the Security tab. The Security property sheet appears displaying a User List containing the names of all users in the same user group plus the PUBLIC user name. The User Name column in the Permissions box displays users who are assigned permission to the selected rule.



*Rules Permissions Dialog Box*

3. To add a user to the User Name list in the Permissions Dialog Box, select the user from the User List, hold down the left mouse button and drag the name to the Permissions box.

### Note

Currently, you cannot deny Read permission.

4. To allow the user to update the rule, select the Update checkbox.
5. To have the user own the rule, select the Owner checkbox. Only one user can own a rule, so when you select this checkbox, the Owner checkbox associated with your username is automatically cleared.
6. To disable the rule, the owner can select the Disable checkbox. Only the owner of a rule can enable/disable the rule using this checkbox.

### Note

If a user with Update permission has “deleted” a rule that you own, the rule is really disabled, not deleted. You can uncheck the Disable checkbox to enable the rule.

7. When you are done adding (or changing) users permissions, choose one of the following:
  - To add the user permission, choose the Apply button.
  - To delete the user permissions you added and return the property sheet to its original configuration, choose the Cancel button.

### Note

If you move the cursor to the left pane and click the mouse button after you add user permissions, the permissions are automatically saved.

---

## Expressions

The Expression property sheet enables you to view, create, modify, or delete an expression for a selected rule.

The following tabs appear within the Expression property sheet:

- Expression Components
- Field List
- Operators
- Values
- Function

The evaluation criteria for a rule consists of a Boolean expression containing the Boolean operators "&" (AND) and "|" (OR), expressions, and parentheses to control the order of evaluation. You can use OR to explicitly direct what Boolean operations do together.

There must be at least one space between the field name and the Rules operator as well as between the Rules operator and the comparison value. The EXIST and NOT\_EXIST operators must be followed by a least one space before a parenthesis or a Boolean operator. If the field name or static comparison value contains spaces, quotes, or parentheses, the item must be enclosed in quotes (either single or double --whatever the value does NOT have). A value may not have both single and double quotes.

## Expression Components

The Expression Components property sheet contains expression templates that allow you to create an expression for a rule. An expression contains arguments (operation expressions, combined with Boolean operators (Functions) and parentheses. An argument is the smallest component of a rule that can be evaluated. It consists of a field name, Rules operator, and either another field name (field-to-field comparisons), a static value (static comparisons), or nothing (existence operators). Boolean operators (Functions) include the AND and OR operators to combine arguments into an expression. Boolean algebra's precedence rules are followed when evaluating an expression, with innermost parenthesis evaluated first, and ANDs evaluated before ORs if no parentheses are present.

## Field List

Field List contains the field names associated with the selected message type. The field names are defined when an input format is defined. A rule's message type is a flat input format that contains the field or a compound input format that contains a flat input format, containing the field. Rules field names are the same as the field names in Formatter.

If the field name contains spaces ( ' '), quotes ( ' ' or " " ), or parentheses ( ' ( ' ) ' ), the name must be enclosed in quotes (either single or double--whatever the name does not have). A field name can not have both single and double quotes.

## Rules Operators

A Rules operator is a data type comparison with a field. Rules operators are field existence, field non-existence, and the following operators: <, <=, >, >=, <>, = for INT (whole number), FLOAT (decimal number), DATE, TIME, DATETIME, and STRING fields, field-to-field comparisons (for example, field1 compares against field2), and case-sensitive string comparisons (for example, where "a" does not equal "A").

Specific operators only apply to a certain data type. For example, there are no operators that compare integer and float data types. Operators are executed against a field name and a value, or between two field names (each of the two field names, as defined by the message type, must be of the same data type). Existence operators enable you to determine if a field exists in a message. Integer, string and float operators evaluate a message field against a static value using the operator symbol. Field-to-field operators compare two groups of data (fields) within the message.

### Data Types and Operators

Operator	Description
<b>Existence Operators</b>	
EXIST	Field is present
NOT_EXIST	Field is not present

### String and Field-to-Field String Operators

<b>Operator</b>	<b>Description</b>
STRING=	String is equal to
STRING<>	String is not equal to
STRING <	String is less than
STRING >	String is greater than
STRING >=	String is greater than or equal to
STRING <=	String is less than or equal to
F2FSTRING=	Field-to-field string equal to
F2FSTRING>=	Field-to-field string greater than or equal to
F2FSTRING<=	Field-to-field string less than or equal to
F2FSTRING<>	Field-to-field string not equal to
F2FSTRING >	Field-to-field string greater than
F2FSTRING<	Field-to-field string less than
<b>Case Sensitive Operators</b>	
CSSTRING =	Case sensitive field equals
CSSTRING<>	Case sensitive field not equal to
CSSTRING>	Case sensitive field greater than
CSSTRING<	Case sensitive field less than
CSSTRING >=	Case sensitive field greater than or equal to
CSSTRING<=	Case sensitive field less than or equal to
F2FCSSTRING<	Case sensitive field-to-field less than
F2FCSSTRING>=	Case sensitive field-to-field greater than or equal to
F2FCSSTRING<=	Case sensitive field-to-field less than or equal to
F2FCSSTRING =	Case sensitive field-to-field equal to
F2FCSSTRING>	Case sensitive field-to-field greater than
F2FCSSTRING<>	Case sensitive field-to-field not equal to
<b>Integer and Field-to-Field Integer Operators</b>	
INT =	Integer equals
INT<>	Integer not equal to
INT>	Integer greater than
INT<	Integer less than
INT>=	Integer greater than or equal to
INT<=	Integer less than or equal to
F2FINT<	Field-to-field integer less than

<b>Operator</b>	<b>Description</b>
F2FINT>=	Field-to-field integer greater than or equal to
F2FINT<=	Field-to-field integer less than or equal to
F2FINT =	Field-to-field integer equal to
F2FINT>	Field-to-field integer greater than
F2FINT<>	Field-to-field integer not equal to
<b>Float (Decimal) Operators</b>	
FLOAT =	Float equals
FLOAT<>	Float not equal to
FLOAT>	Float greater than
FLOAT<	Float less than
FLOAT>=	Float greater than or equal to
FLOAT<=	Float less than or equal to
<b>Date, Time, DateTime and Field-to-Field Date, Time, and DateTime Operators</b>	
DATE=	Date Equal To
DATE>	Date Greater Than
DATE<	Date Less Than
DATE<=	Date Greater Than Or Equal To
DATE>=	Date Less Than Or Equal To
DATE<>	Date Not Equal To
F2FDATE=	Field To Field Time Equal To
F2FDATE>	Field To Field Time Greater Than
F2FDATE<	Field To Field Time Less Than
F2FDATE<=	Field To Field Time Less Than Or Equal To
F2FDATE>=	Field To Field Date Greater Than Or Equal To
F2FDATE<>	Field To Field Date Not Equal To
TIME=	Time Equal To
TIME>	Time Greater Than
TIME<	Time Less Than
TIME>=	Time Greater Than Or Equal To
TIME<=	Time Less Than Or Equal To
TIME<>	Time Not Equal To
F2FTIME=	Field To Field Time Equal To
F2FTIME>	Field To Field Time Greater Than

<b>Operator</b>	<b>Description</b>
F2FTIME<	Field To Field Time Greater Than
F2FTIME>=	Field To Field Time Greater Than Or Equal To
F2FTIME<=	Field To Field Time Less Than Or Equal To
F2FTIME<>	Field To Field Time Not Equal To
DATETIME=	DateTime Equal To
DATETIME>	DateTime Greater Than
DATETIME<	DateTime Less Than
DATETIME>=	DateTime Greater Than Or Equal To
DATETIME<=	DateTime Less Than Or Equal To
DATETIME<>	DateTime Not Equal To
F2FDATETIME=	Field To Field DateTime Equal To
F2FDATETIME>	Field To Field DateTime Greater Than
F2FDATETIME<	Field To Field DateTime Less Than
F2FDATETIME>=	Field To Field DateTime Greater Than Or Equal To
F2FDATETIME<=	Field To Field DateTime Less Than Or Equal To
F2FDATETIME<>	Field To Field DateTime Not Equal To

## Values

Values contains the matching data types of the MQSeries Integrator operator in the operation expression field.

## Functions

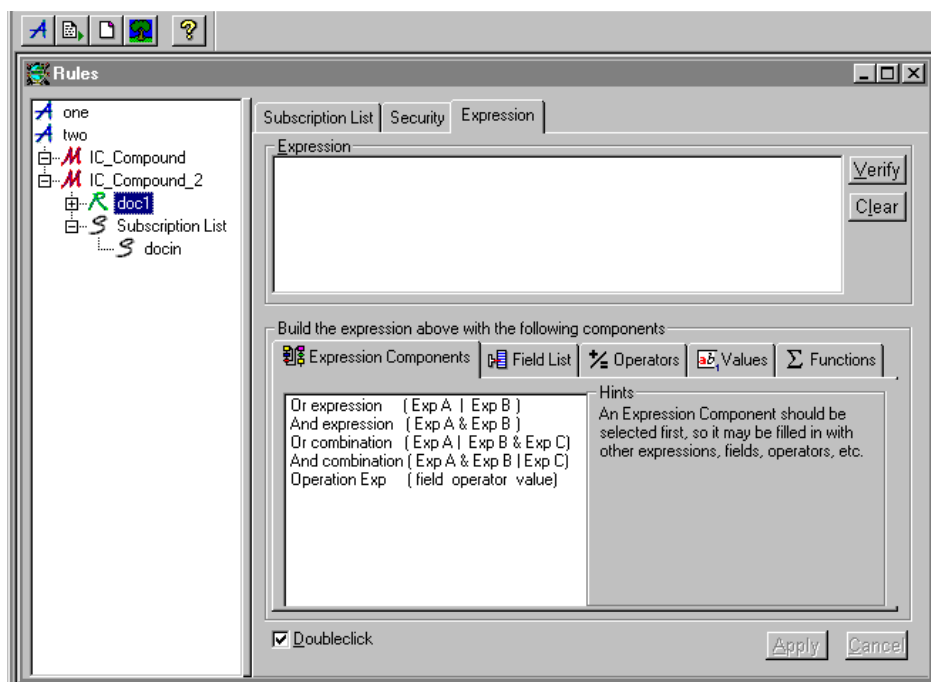
Functions contain the Boolean operators AND and OR and parentheses for use either with a single expression or more than one expression connected with Boolean operators with optional parentheses.



## Creating or Modifying an Expression

To create or modify an expression:

1. Select the appropriate rule.
2. Choose the Expression tab. The Expression property sheet appears.



*Expression Property Sheet*

3. Select the Expression Components tab. The Expression Component appears in the Expression window.

### Note

Use the Doubleclick box in the lower-left corner of the Expression property sheet to click or double-click when selecting components of expressions.

4. Build the rest of the expression by selecting the Field List, Operators, Values, and Functions tabs, clicking their components as needed.

### Note

To enter your expression directly into the Expression window without using the tabs within the Expression property sheet, type the expression into the Expression box.

Once you have entered components into the Expression window, you can modify the expression by right clicking within the Expression window. A text editing menu appears where you can modify the text in the Expression window. The following editing functions are

available: Undo your last change, Cut, Copy, Paste, Delete, and Select All of the text within the Expression window.

5. Click the Verify button to see if the final expression is valid. The verify button tests the expression, but does not save it to the database.
6. To clear the window and start over, click the Clear button.
7. To either create or update an expression, choose the Apply button.

### Note

If you move the cursor to the left pane and click a mouse button after adding an expression, the expression is automatically saved.

### Examples of Expressions

Description	Expression Layout	Expression
Single Argument	A	F1 STRING= INTEGRATOR
Arguments Anded Together	A & B & C	F1 STRING= INTEGRATOR & F2 INT= 100 & F3 INT= 150
Arguments Ored Together	A   B   C	F1 STRING= INTEGRATOR   F2 INT= 100   F3 INT= 150
Precedence	A   B & C	F1 STRING= INTEGRATOR   F2 INT= 100 & F3 INT= 150
Nested Prens	(A   ((B & (C))   D))	(F1 STRING= INTEGRATOR   ((F2 INT= 100 & (F3 INT= 150))   F4 INT= 200))

### Deleting an Expression

You can delete the expression for a selected rule. You must have Update permission for the rule to delete an expression.

#### To delete an expression:

1. Select the rule that contains the expression that you want to delete. The expression appears in the property sheet.
2. Click the Clear button. The expression is removed for the selected rule.

### Subscriptions

After a rule is created, subscriptions that contain actions should be added to the rule's Subscription List. Each application/message group has its own Subscription List. You can drag and drop a subscription to any rule from the Subscription List or drag and drop into the Subscription List tab of a rule.

A subscription describes the actions that are performed if the rule's expressions pass evaluation. A subscription allows you to define message destination IDs, receiver locations, and message formats, and any processes initiated upon message delivery. Subscriptions have permissions similar to Rules permissions.

A subscription can only be assigned to a rule within the same application group/message type.

### Note

Subscription Sharing: A subscription can be assigned to several rules if the rules are in the same application group and message type.

### Note

You must have Update permission for the rule to add a subscription to a rule.

## Adding a Subscription

To add a subscription:

1. From the Rules window tree, select the Subscription List for the Application Group/Message Type that you want to add a subscription to and click the right mouse button. A popup menu appears.
2. Select New. The cursor is positioned in the Subscription List text box where you can type the new, unique subscription name.

### WARNING!

If you are using a case-insensitive database, you cannot use the same name with a change in case to identify components. For example, you cannot name one subscription S1 and another s1. In a case-insensitive environment, make each item unique using something other than case differences.

3. Press *Return* to add the subscription to the Subscription List.

### Note

If you click either mouse button outside the subscription text box (and you have not pressed *Return*), the subscription is added.

If there is no duplicate name for this subscription, the subscription is added and the Actions property sheet appears.

Each subscription should have at least one action. For the procedure to add an action, see *Adding an Action* on page 113.

## Duplicating a Subscription

Use the Duplicate function to duplicate the selected subscription. The new subscription is owned by the current user who has Update permission.

**To duplicate a subscription:**

1. Select the subscription that you want to duplicate.
2. Hold down the right mouse button to activate the popup menu and select Duplicate. A text entry box appears at the top of the rules list.
3. Type a unique subscription name in the text box and press *Return* to save the name.

### WARNING!

If you are using a case-insensitive database, you cannot use the same name with a change in case to identify components. For example, you cannot name one rule R1 and another r1. In a case-insensitive environment, make each item unique using something other than case differences.

### Note

If you click any mouse button outside the subscription text box (and you have not pressed *Return*), the new name is saved.

4. To change the user permissions for the new subscription, select the Security tab.

### Note

For more information on security, refer to *Subscription Security* on page 112.

## Renaming a Subscription

Use the Rename Subscription function to change the name of a subscription. You must have Update permission to change a subscription's name.

To change a subscription name, select the name that you want to change and click on the name. A box appears around the subscription name and the name is highlighted (see the figure below). Type the new, unique name and press *Return* to save the name.

### WARNING!

If you are using a case-insensitive database, you cannot use the same name with a change in case to identify components. For example, you cannot name one subscription S1 and another s1. In a case-insensitive environment, make each item unique using something other than case differences.

## Deleting a Subscription

A subscription can only be deleted if it is not linked to any Rules. You must be the subscription owner and have Update permission to delete a subscription.

If you own the subscription and the subscription is not linked to any rules, the subscription is deleted. If you do not own the subscription and if the subscription is not linked to any rules, the subscription is disabled. For more information on enabling and disabling subscriptions, see *Subscription Security* on page 112.

### To delete a subscription:

1. Select the subscription that you want to delete.
2. Click the right mouse button to activate the popup menu and select Delete. The Delete confirmation box appears.
3. To delete the subscription, choose the OK button. The delete box closes and the subscription is deleted.

## Adding a Comment to a Subscription

You can associate a comment with a subscription.

### To add a comment:

1. Select the subscription for which you want to add a comment.
2. Choose the Misc tab. A Comments text entry box appears containing the word "None" (if no comments were previously added).



*Misc. Tab's Comments Text Box*

3. Select and delete the word None and type your comment; then choose the Apply button.

## Subscription Security

Use the subscription's Security property to enter permissions for an individual subscription. When you add a subscription, you are assigned ownership and you must set the permissions for the subscription. By default, Rules designates the creator of a subscription as the owner, who is given Read and Update privileges. Owners should give Update permission if they want others to be able to change the subscription. PUBLIC defaults to read only. Read permission is the minimum security currently allowed. The Read checkbox is marked by default and cannot be changed.

Owners should give Update permission if they want others to be able to change the subscription. When a user deletes a subscription and has Update permission, but does not own the subscription, the subscription is deactivated instead of deleted. The subscription appears in the tree, but does not allow changes because it has been deactivated. Users can see that a subscription has been deactivated by opening the Security tab and viewing the Disable checkbox at the bottom of the window. A checked box indicates that the subscription is deactivated.

### ***Adding or Changing Subscription Security***

To add or change user permissions:

1. Select the subscription that you want to add (or change) the user permission for.
2. Choose the Security tab. The Permissions property sheet appears displaying a User List containing the names of all users in the same user group plus the PUBLIC user name. The User Name column in the Permissions box displays users that are assigned permission to the selected subscription.
3. To add a user to the User Name list, select the user from the User List, hold down the left mouse button, and drag the name to the Permissions box.

#### **Note**

Currently, you cannot deny Read permission.

4. To allow the user to update the subscription, select the Update checkbox.
5. To have the user own the subscription, select the Owner checkbox. Only one user can own a subscription, so when this checkbox is selected, the Owner checkbox associated with your username automatically loses its check mark.
6. To disable the subscription, the owner can select the Disable checkbox.

#### **Note**

If a user with Update permission has “deleted” a subscription that you own, the subscription is really disabled, not deleted. As the

owner, you can uncheck the Disable checkbox to enable the subscription.

---

7. When you are done adding (or changing) permissions, choose one of the following:
  - To add the user permissions, choose the Apply button.
  - To delete the user permissions you added and return the property sheet to its original configuration, choose the Cancel button.

### Note

If you move the cursor to the left pane and click the mouse button after you add user permissions, the permissions are automatically saved.

---

## Actions

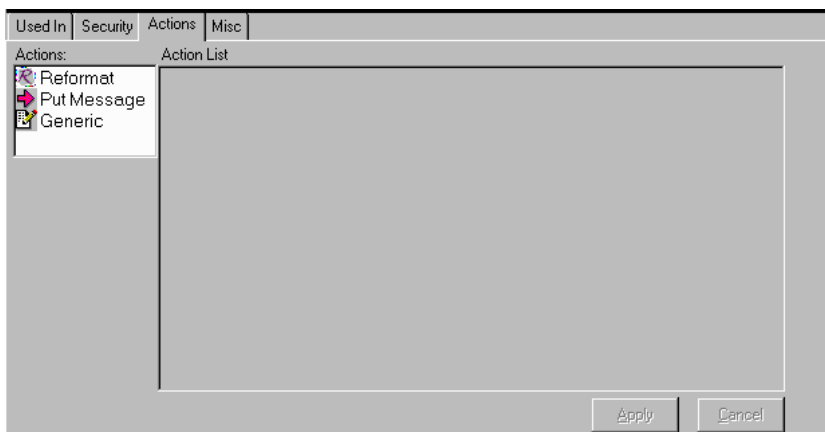
Actions hold subscription instructions. The three Actions affecting subscriptions are called Generic, Reformat, and Put Message. From the Actions property sheet, you can add (to the bottom of the list), insert, and delete actions.

You must have Update permission for the subscription to add an action.

## Adding an Action

**To add an action to a subscription:**

1. Select the subscription in which you want to add an action. The Actions property sheet appears.



*Subscription Selected - Actions Property Sheet Displayed*

The Actions box displays the following actions:

Action	Description
Reformat	Reformats a message. Using Formatter, the message format is defined both at input and at output. The Rule Engine daemon uses an input format (what the message looks like coming into Rules) and an output format (what the message looks like after the reformat) to reformat the message contents.
Put Message	Instructs Rules to route (deliver) a message to a specific queue. You define the queue name and any other message options to be set using Rules.
Generic	Allows you to create a user-defined action. Generic actions can have one or more options and the option name and option values can be changed. You can rename a generic action.

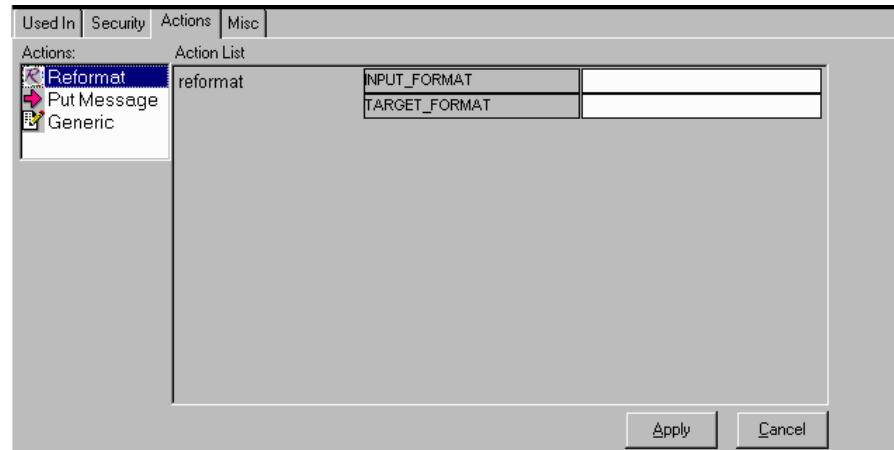
2. Select the action that you want to add, hold down the left mouse button, and drag the action to the Action List box. Then do one of the following:
  - To add the action to the bottom of the list, drop the action onto the gray space below the last action in the list.
  - To insert the action above an existing action, drop the action on top of the position you want the action to be above.

### Note

The parameters you enter into in the Action Values box should be based on the type of action selected.

- If you select the Reformat action, when it is added to the Action List box, the options INPUT\_FORMAT and TARGET\_FORMAT appear. These options each have a mandatory associated value. You can either enter the value (format name), or select it from its drop-down list. The INPUT\_FORMAT drop-down list displays only input formats and the TARGET\_FORMAT drop-down list displays only output formats. Multiple Reformat actions can be added.





*Reformat Action Added*

- If you select the Put Message action, it is added to the Action List box and labeled putqueue. Multiple Put Message actions can be added. See *Adding Multiple putqueue Options* on page 115.
  - If you select the Generic action, when it is added to the Action List box, you can change the action name and option name and value. Multiple Generic actions can be added.
3. Choose one of the following:
- To save the actions, choose the Apply button.
  - To delete the new actions and return the property sheet to its initial configuration, choose the Cancel button.

## Note

The Rules Engine daemon does not support generic actions. Users will have to develop their own Rules Engine daemon to process generic actions.

## Adding Multiple putqueue Options

Multiple options can be added. To add more than one option, right click on an Action line to display the popup menu, then select Add Option. A blank option field appears under the existing option field. Type the option.

### Additional putqueue Option Names and Values

Option Name	Option Value	Default Value	Description
user input	user input	N/A	Data source is user input.

Option Name	Option Value	Default Value	Description
MQS_FORMAT	user input NONE MQADMIN MQCHCOM MQCMD1 MQCMD2 MQDEAD MQHDIST MQEVENT MQIMS MQIMSVS MQHMDE MQPCF MQHREF MQSTR MQTRIG	N/A	Data source is section [putqueue] in the NEONet.ini file. Specifies the MQSeries message format on an outbound message.
MQS_PROPAGATE	PROPAGATE NO_PROPAGATE	NO_PROPAGATE	Data source is section [MQS_PROPAGATE] in the NEONet.ini file. On a per-message basis, specifies whether to include the MQSeries Integrator header in the user data of the outbound message. For more information, refer to the section, See <i>MQS_PROPAGATE</i> on page 116. .
MQS_PERSIST	PERSIST NO_PERSIST	Propagate the expiration setting of the input message to the output message.	Data source is section [MQS_PERSIST] in the NEONet.ini file. Indicates whether the outbound message will be persistent regardless of whether the inbound message was persistent. The MQS_PERSIST option is not set for a given putqueue action. The MQSeries Integrator daemon sets the persistence field of the outbound message's message descriptor to the value of the persistence field of the inbound messages's message descriptor.
MQS_EXPIRY	PROPAGATE NO_PROPAGATE	NO_PROPAGATE The expiry field of the outbound message descriptor is set to unlimited.	Data source is section [MQS_EXPIRY] in the NEONet.ini file. .
OPT_MSG_TYPE		N/A	Formatter application

### **MQS\_PROPAGATE**

**MQS\_PROPAGATE=PROPAGATE**

If you set MQS\_PROPAGATE=PROPAGATE, if the MQHRF header is present on the inbound message, the format field of the outbound message's MQHRF header will be the same as that of the inbound message.

If the MQHRF header is *not* present on the inbound message, the MQSeries Integrator daemon rejects the message and puts it on the Failure queue.

#### **MQS\_PROPAGATE=NO\_PROPAGATE**

If you set MQS\_PROPAGATE=NO\_PROPAGATE, if the MQHRF header is present on the inbound message, the format field of the outbound message's message descriptor is set to the value contained in the format field of the inbound message's MQHRF header.

If the MQHRF header was *not* present on the inbound message, the format field of the outbound message's message descriptor is set to the value contained in the inbound messages's message descriptor.

#### **MQS\_PROPAGATE=PROPAGATE and MQS\_FORMAT=SOME\_FORMAT**

If you set MQS\_PROPAGATE=PROPAGATE and MQS\_FORMAT=SOME\_FORMAT, if the MQHRF header is present on the inbound message, the format field of the outbound message's MQHRF header is set to the value SOME\_FORMAT.

If the MQHRF header was *not* present on the inbound message, the MQSeries Integrator daemon rejects the message and puts it on the Failure queue.

#### **MQS\_PROPAGATE=NO\_PROPAGATE and MQS\_FORMAT=SOME\_FORMAT**

If you set MQS\_PROPAGATE=NO\_PROPAGATE and MQS\_FORMAT=SOME\_FORMAT, if the MQHRF header is present or not on the inbound message, the format field of the outbound message's descriptor is set to the value SOME\_FORMAT.

### ***MQS\_EXPIRY***

#### **MQS\_EXPIRY=PROPAGATE**

If MQS\_EXPIRY=PROPAGATE, the Expiry field of the outbound message's message descriptor is set to the value of the Expiry field of the inbound message's message descriptor.

#### **MQS\_EXPIRY=NO\_PROPAGATE**

If MQS\_EXPIRY=NO\_PROPAGATE, the Expiry field of the outbound message's message descriptor is set to MQEI\_UNLIMITED.

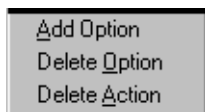
## **Deleting an Action**

Use the Delete Action function to delete an action. This function is available from the popup menu.

You must have Update permission to delete an action.

To delete an action:

1. In the Action List, select the action that you want to delete.
2. Position the cursor on the action, and press the right mouse button to open the popup menu.



### Note

After you add actions, if you move the cursor to the left pane and click the mouse button, the actions are automatically saved.

3. From the Popup menu, choose Delete Action. The action is deleted from the Actions property sheet.
4. Choose one of the following:
  - To delete the action, choose the Apply button.
  - To cancel the delete, choose the Cancel button. The Actions property sheet is returned to its original configuration.

### Note

Reformat and Put Message must have both options defined to work properly. Only Generic actions are allowed to have user-defined options added and deleted.

## Conditional Branching

If the Rules GUI is started by the Formatter GUI, the Rules GUI opens in Conditional Branching mode. The only application group that appears in the Rules tree is NEONET\_FORMATTER.

You can add message types, rules, arguments, subscriptions, and actions to the NEONET\_FORMATTER application group using the procedures described earlier in this chapter. However, there is a difference in the Actions property sheet.

The Actions property sheet contains only the Rules Field action. When adding or inserting Rules Field actions, three options are needed:

- FIELD\_NAME
- OUTPUT\_FORMAT
- FORMAT\_CONTROL

See *Adding an Action* on page 113 for details on adding and changing actions and options.

### Note

When you reformat from a compound input format to a compound output format, use the compound input format (not a component flat format) as the

message type and the compound output format (not a component flat format)  
as the OUTPUT\_FORMAT in the subscription.

---



## Appendix A

# ASCII Extended Character Set

Decimal Value	Hex Value	Extended Character Set	Decimal Value	Hex Value	Extended Character Set
000	00	NUL	026	1A	SUB
001	01	SCH	027	1B	ESCAPE
002	02	STX	028	1C	FS
003	03	ETX	029	1D	GS
004	04	EOT	030	1E	RS
005	05	ENO	031	1F	US
006	06	ACK	032	20	SPACE
007	07	BEL	033	21	!
008	08	BS	034	22	“
009	09	HT	035	23	#
010	0A	LF	036	24	\$
011	0B	VT	037	25	%
012	0C	FF	038	26	&
013	0D	CR	039	27	‘
014	0E	SO	040	28	(
015	0F	SI	041	29	)
016	10	DLE	042	2A	*
017	11	DC1	043	2B	+
018	12	DC2	044	2C	,
019	13	DC3	045	2D	-
020	14	DC4	046	2E	.
021	15	NAK	047	2F	/
022	16	SYN	048	30	0
023	17	ETB	049	31	1
024	18	CAN	050	32	2
025	19	EM	051	33	3

<b>Decimal Value</b>	<b>Hex Value</b>	<b>Extended Character Set</b>		<b>Decimal Value</b>	<b>Hex Value</b>	<b>Extended Character Set</b>
052	34	4		084	54	T
053	35	5		085	55	U
054	36	6		086	56	V
055	37	7		087	57	W
056	38	8		088	58	X
057	39	9		089	59	Y
058	3A	:		090	5A	Z
059	3B	;		091	5B	[
060	3C	<		092	5C	\
061	3D	=		093	5D	]
062	3E	>		094	5E	^
063	3F	?		095	5F	_
064	40	@		096	60	'
065	41	A		097	61	a
066	42	B		098	62	b
067	43	C		099	63	c
068	44	D		100	64	d
069	45	E		101	65	e
070	46	F		102	66	f
071	47	G		103	67	g
072	48	H		104	68	h
073	49	I		105	69	i
074	4A	J		106	6A	j
075	4B	K		107	6B	k
076	4C	L		108	6C	l
077	4D	M		109	6D	m
078	4E	N		110	6E	n
079	4F	O		111	6F	o
080	50	P		112	70	p
081	51	Q		113	71	q
082	52	R		114	72	r
083	53	S		115	73	s



Decimal Value	Hex Value	Extended Character Set	Decimal Value	Hex Value	Extended Character Set
116	74	t	148	94	”
117	75	u	149	95	•
118	76	v	150	96	–
119	77	w	151	97	—
120	78	x	152	98	~
121	79	y	153	99	™
122	7A	z	154	9A	š
123	7B	{	155	9B	›
124	7C		156	9C	œ
125	7D	}	157	9D	unused
126	7E	~	158	9E	unused
127	7F	DEL	159	9F	ÿ
128	80	unused	160	A0	nonbreaking space
129	81	unused	161	A1	ı
130	82	,	162	A2	ç
131	83	f	163	A3	£
132	84	„	164	A4	¤
133	85	...	165	A5	¥
134	86	†	166	A6	ı
135	87	‡	167	A7	§
136	88	^	168	A8	¨
137	89	‰	169	A9	©
138	8A	Š	170	AA	ª
139	8B	‹	171	AB	«
140	8C	Œ	172	AC	¬
141	8D	unused	173	AD	-
142	8E	unused	174	AE	®
143	8F	unused	175	AF	-
144	90	unused	176	B0	°
145	91	‘	177	B1	±
146	92	’	178	B2	²
147	93	“	179	B3	³

Decimal Value	Hex Value	Extended Character Set	Decimal Value	Hex Value	Extended Character Set
180	B4	´	212	D4	Ô
181	B5	µ	213	D5	Õ
182	B6	¶	214	D6	Ö
183	B7	·	215	D7	×
184	B8	,	216	D8	Ø
185	B9	¹	217	D9	Û
186	BA	º	218	DA	Ú
187	BB	»	219	DB	Û
188	BC	¼	220	DC	Ü
189	BD	½	221	DD	Ý
190	BE	¾	222	DE	Þ
191	BF	¿	223	DF	ß
192	C0	À	224	E0	à
193	C1	Á	225	E1	á
194	C2	Â	226	E2	â
195	C3	Ã	227	E3	ã
196	C4	Ä	228	E4	ä
197	C5	Å	229	E5	å
198	C6	Æ	230	E6	æ
199	C7	Ç	231	E7	ç
200	C8	È	232	E8	è
201	C9	É	233	E9	é
202	CA	Ê	234	EA	ê
203	CB	Ë	235	EB	ë
204	CC	Ì	236	EC	ì
205	CD	Í	237	ED	í
206	CE	Î	238	EE	î
207	CF	Ï	239	EF	ï
208	D0	Ð	240	F0	ð
209	D1	Ñ	241	F1	ñ
210	D2	Ò	242	F2	ò
211	D3	Ó	243	F3	ó

<b>Decimal Value</b>	<b>Hex Value</b>	<b>Extended Character Set</b>		<b>Decimal Value</b>	<b>Hex Value</b>	<b>Extended Character Set</b>
244	F4	ô		250	FA	ú
245	F5	õ		251	FB	û
246	F6	ö		252	FC	ü
247	F7	÷		253	FD	ý
248	F8	ø		254	FE	þ
249	F9	ù		255	FF	ÿ



---

## Appendix B

# EBCDIC Character Set

---

Decimal Value	Hex Value	EBCDIC Value*	Description	Binary
000	00	NUL	Null	0000 0000
001	01	SOH	Start of Heading	0000 0001
002	02	STX	Start of Text	0000 0010
003	03	ETX	End of Text	0000 0011
004	04	SEL	Select	0000 0100
005	05	HT	Horizontal Tab	0000 0101
006	06	RNL	Required New Line	0000 0110
007	07	DEL	Delete	0000 0111
008	08	GE	Graphic Escape	0000 1000
009	09	SPS	Superscript	0000 1001
010	0A	RPT	Repeat	0000 1010
011	0B	VT	Vertical Tab	0000 1011
012	0C	FF	Form Feed	0000 1100
013	0D	CR	Carriage Return	0000 1101
014	0E	SO	Shift Out	0000 1110
015	0F	SI	Shift In	0000 1111
016	10	DLE	Data Link Escape	0001 0000
017	11	DC1	Device Control 1	0001 0001
018	12	DC2	Device Control 2	0001 0010
019	13	DC3	Device Control 3	0001 0011
020	14	RES/ENP	Restore/Enable Presentation	0001 0100
021	15	NL	New Line	0001 0101
022	16	BS	Backspace	0001 0110
023	17	POC	Program-Operator Communication	0001 0111
024	18	CAN	Cancel	0001 1000
025	19	EM	End of Medium	0001 1001

<b>Decimal Value</b>	<b>Hex Value</b>	<b>EBCDIC Value*</b>	<b>Description</b>	<b>Binary</b>
026	1A	UBS	Unit Backspace	0001 1010
027	1B	CU1	Customer Use 1	0001 1011
028	1C	IFS	Interchange File Separator	0001 1100
029	1D	IGS	Interchange Group Separator	0001 1101
030	1E	IRS	Interchange Record Separator	0001 1110
031	1F	IBT/IUS	Intermediate Transmission Block/Interchange Unit Separator	0001 1111
032	20	DS	Digit Select	0010 0000
033	21	SOS	Start of Significance	0010 0001
034	22	FS	Field Separator	0010 0010
035	23	WUS	Word Underscore	0010 0011
036	24	BYP/INP	Bypass/Inhibit Presentation	0010 0100
037	25	LF	Line Feed	0010 0101
038	26	ETB	End of Transmission Block	0010 0110
039	27	ESC	Escape	0010 0111
040	28	SA	Set Attribute	0010 1000
041	29	SFE	Start Field Extended	0010 1001
042	2A	SM/SW	Set Mode/Switch	0010 1010
043	2B	CSP	Control Sequence Prefix	0010 1011
044	2C	MFA	Modify Field Attribute	0010 1100
045	2D	ENQ	Enquiry	0010 1101
046	2E	ACK	Acknowledge	0010 1110
047	2F	BEL	Bell	0010 1111
048	30			0011 0000
049	31			0011 0001
050	32	SYN	Synchronous Idle	0011 0010
051	33	IR	Index Return	0011 0011
052	34	PP	Presentation Position	0011 0100
053	35	TRN	Transparent	0011 0101
054	36	NBS	Numeric Backspace	0011 0110
055	37	EOT	End of Transmission	0011 0111

<b>Decimal Value</b>	<b>Hex Value</b>	<b>EBCDIC Value*</b>	<b>Description</b>	<b>Binary</b>
056	38	SBS	Subscript	0011 1000
057	39	IT	Indent Tab	0011 1001
058	3A	RFF	Required Form Feed	0011 1010
059	3B	CU3	Customer Use 3	0011 1011
060	3C	DC4	Device Control 4	0011 1100
061	3D	NAK	Negative Acknowledge	0011 1101
062	3E			0011 1110
063	3F	SUB	Substitute	0011 1111
064	40	SP	Space	0100 0000
065	41	RSP		0100 0001
066	42			0100 0010
067	43			0100 0011
068	44			0100 0100
069	45			0100 0101
070	46			0100 0110
071	47			0100 0111
072	48			0100 1000
073	49			0100 1001
074	4A	ç		0100 1010
075	4B	.		0100 1011
076	4C	<		0100 1100
077	4D	(		0100 1101
078	4E	+		0100 1110
079	4F			0100 1111
080	50	&		0101 0000
081	51			0101 0001
082	52			0101 0010
083	53			0101 0011
084	54			0101 0100
085	55			0101 0101
086	56			0101 0110
087	57			0101 0111

<b>Decimal Value</b>	<b>Hex Value</b>	<b>EBCDIC Value*</b>	<b>Description</b>	<b>Binary</b>
088	58			0101 1000
089	59			0101 1001
090	5A	!		0101 1010
091	5B	\$		0101 1011
092	5C	*		0101 1100
093	5D	)		0101 1101
094	5E	;		0101 1110
095	5F	-		0110 1111
096	60	-		0110 0000
097	61	/		0110 0001
098	62			0110 0010
099	63			0110 0011
100	64			0110 0100
101	65			0110 0101
102	66			0110 0110
103	67			0110 0111
104	68			0110 1000
105	69			0110 1001
106	6A			0110 1010
107	6B	,		0110 1011
108	6C	%		0110 1100
109	6D	_		0110 1101
110	6E	>		0110 1110
111	6F	?		0110 1111
112	70			0111 0000
113	71			0111 0001
114	72			0111 0010
115	73			0111 0011
116	74			0111 0100
117	75			0111 0101
118	76			0111 0110
119	77			0111 0111



<b>Decimal Value</b>	<b>Hex Value</b>	<b>EBCDIC Value*</b>	<b>Description</b>	<b>Binary</b>
120	78			0111 1000
121	79			0111 1001
122	7A	:		0111 1010
123	7B	#		0111 1011
124	7C	@		0111 1100
125	7D	'		0111 1101
126	7E	=		0111 1110
127	7F	"		0111 1111
128	80			1000 0000
129	81	a		1000 0001
130	82	b		1000 0010
131	83	c		1000 0011
132	84	d		1000 0100
133	85	e		1000 0101
134	86	f		1000 0110
135	87	g		1000 0111
136	88	h		1000 1000
137	89	i		1000 1001
138	8A			1000 1010
139	8B			1000 1011
140	8C			1000 1100
141	8D			1000 1101
142	8E			1000 1110
143	8F			1000 1111
144	90			1001 0000
145	91	j		1001 0001
146	92	k		1001 0010
147	93	l		1001 0011
148	94	m		1001 0100
149	95	n		1001 0101
150	96	o		1001 0110
151	97	p		1001 0111

<b>Decimal Value</b>	<b>Hex Value</b>	<b>EBCDIC Value*</b>	<b>Description</b>	<b>Binary</b>
152	98	q		1001 1000
153	99	r		1001 1001
154	9A			1001 1010
155	9B			1001 1011
156	9C			1001 1100
157	9D			1001 1101
158	9E			1001 1110
159	9F			1001 1111
160	A0			1010 0000
161	A1	~		1010 0001
162	A2	s		1010 0010
163	A3	t		1010 0011
164	A4	u		1010 0100
165	A5	v		1010 0101
166	A6	w		1010 0110
167	A7	x		1010 0111
168	A8	y		1010 1000
169	A9	z		1010 1001
170	AA			1010 1010
171	AB			1010 1011
172	AC			1010 1100
173	AD			1010 1101
174	AE			1010 1110
175	AF			1010 1111
176	B0			1011 0000
177	B1			1011 0001
178	B2			1011 0010
179	B3			1011 0011
180	B4			1011 0100
181	B5			1011 0101
182	B6			1011 0110
183	B7			1011 0111

<b>Decimal Value</b>	<b>Hex Value</b>	<b>EBCDIC Value*</b>	<b>Description</b>	<b>Binary</b>
184	B8			1011 1000
185	B9			1011 1001
186	BA			1011 1010
187	BB			1011 1011
188	BC			1011 1100
189	BD			1011 1101
190	BE			1011 1110
191	BF			1011 1111
192	C0	{		1100 0000
193	C1	A		1100 0001
194	C2	B		1100 0010
195	C3	C		1100 0011
196	C4	D		1100 0100
197	C5	E		1100 0101
198	C6	F		1100 0110
199	C7	G		1100 0111
200	C8	H		1100 1000
201	C9	I		1100 1001
202	CA	SHY		1100 1010
203	CB			1100 1011
204	CC			1100 1100
205	CD			1100 1101
206	CE			1100 1110
207	CF			1100 1111
208	D0	}		1101 0000
209	D1	J		1101 0001
210	D2	K		1101 0010
211	D3	L		1101 0011
212	D4	M		1101 0100
213	D5	N		1101 0101
214	D6	O		1101 0110
215	D7	P		1101 0111

<b>Decimal Value</b>	<b>Hex Value</b>	<b>EBCDIC Value*</b>	<b>Description</b>	<b>Binary</b>
216	D8	Q		1101 1000
217	D9	R		1101 1001
218	DA			1101 1010
219	DB			1101 1011
220	DC			1101 1100
221	DD			1101 1101
222	DE			1101 1110
223	DF			1101 1111
224	E0	\		1110 0000
225	E1			1110 0001
226	E2	S		1110 0010
227	E3	T		1110 0011
228	E4	U		1110 0100
229	E5	V		1110 0101
230	E6	W		1110 0110
231	E7	X		1110 0111
232	E8	Y		1110 1000
233	E9	Z		1110 1001
234	EA			1110 1010
235	EB			1110 1011
236	EC			1110 1100
237	ED			1110 1101
238	EE			1110 1110
239	EF			1110 1111
240	F0	0		1111 0000
241	F1	1		1111 0001
242	F2	2		1111 0010
243	F3	3		1111 0011
244	F4	4		1111 0100
245	F5	5		1111 0101
246	F6	6		1111 0110
247	F7	7		1111 0111

<b>Decimal Value</b>	<b>Hex Value</b>	<b>EBCDIC Value*</b>	<b>Description</b>	<b>Binary</b>
248	F8	8		1111 1000
249	F9	9		1111 1001
250	FA			1111 1010
251	FB			1111 1011
252	FC			1111 1100
253	FD			1111 1101
254	FE			1111 1110
255	FF	EO	Eight Ones	1111 1111

\* In the IBM-DOS Character Set, the nonprinting characters may be displayed as figures, for example, (x03) ETX is shown as a heart, and (x0D) CR is shown as a musical note.



---

## Appendix C

# Notices

---

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing  
2-31 Roppongi 3-chome, Minato-ku  
Tokyo 106, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this document to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those

Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM United Kingdom Laboratories,  
Mail Point 151,  
Hursley Park,  
Winchester,  
Hampshire,  
England,  
SO21 2JN.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Programming License Agreement, or any equivalent agreement between us.

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.



# Trademarks and Service Marks

The following, which appear in this book or other MQSeries Integrator books, are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

MQSeries  
AIX  
DB2  
IBM

NEONFormatter and NEONRules are trademarks of New Era of Networks, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

Other company, product, or service names may be the trademarks or service marks of others.



---

# Glossary

---

**abend**

Acronym for abnormal end. More commonly known as a crash.

**accelerator keys**

A combination of keystrokes that bypasses the menu system to carry out an action.

**action**

An action defines a function to perform in a subscription. The Reformat and Put Message actions are predefined and the user can provide generic actions as well. An action must contain at least one option name-value pair.

**administration workstation**

A form of “client” from which a user carries out MQSeries Integrator management duties.

**alternative format**

A special form of compound format where one format in a set of alternatives applies to a message. For example, if the alternative format is named A, it may contain component formats B, C, and D. A message of format A may actually be of variation of only B, C, or D.

**application group**

A logical grouping of applications used to organize rules.

**Application Program Interface (API)**

The interface (calling conventions) by which an application program accesses services. An API is defined at source code level and provides a level of abstraction between the application and the kernel (or other privileged utilities) to ensure portability of the code. A MQSeries Integrator API refers to a set of object classes, structures, and functions that can be used by an application program to invoke MQSeries Integrator services. Plural: APIs.

**argument**

In Rules, an argument is evaluation criteria made up of fields from a message and associated operators.

## **ASCII**

Abbreviation for American Standard Code for Information Interchange.

## **asynchronous**

In electronic messaging, a method of operation in which receiving applications are loosely coupled and independent. The receiver need not respond immediately to a message, and the sender does not have to wait for a response before proceeding with the next operation. Compare to synchronous.

## **Boolean operators**

Logical operators, including AND and OR.

## **client**

A system or process that requests a service from another system or process.

## **client-server**

A paradigm for distributed computing under which the system is split between one or more server tasks, which accept requests according to some protocol, and client tasks, which request information or actions. Clients and servers can be placed independently on network nodes.

## **cross-platform**

Used to describe programs that can execute in dissimilar computing environments.

## **data type**

Describes how to interpret the data.

## **DBMS**

Abbreviation for Database Management System.

## **delimiter**

One or more characters marking either the end or beginning of a piece of data.

## **daemon**

In UNIX, a program that executes in the background. Also known as agents, daemons only activate when defined system conditions become true. For example, a daemon may activate when the system clock reaches 2:00 a.m. every other Saturday. Another daemon may activate when the system senses the arrival of e-mail.

## **field**

The smallest possible container for information. You can use a field in more than one table. If the `first_name` field exists in one table, for example, you can use the same field in other tables.

## **get**

A request for the next message in a queue. Compare to `put`.

## **GUI**

Abbreviation for graphical user interface.

## **heterogeneous**

Composed of different parts of different kinds. Having dissimilar constituents.

## **hex**

Hexadecimal means 16. The base 16 numbering system is used to represent binary numbers.

## **literal**

One or more symbols or letters in data that represents itself.

## **massive mirroring**

Massive mirroring is a means of boosting processing power by replicating entire databases on different computers and running them in parallel (keeping the databases synchronized).

## **message type**

A message type defines the layout of a string of data. The message type name in Rules is the same as the input format name in Formatter.

## **middleware**

Software intended to work between software applications and the underlying computer operating system.

## **option**

An option consists of a name-value pair of data related to an action. An option name can be predefined (for Reformat and Put Message) or user-defined.

## **parse**

To analyze a message by breaking down into its component fields.

## **permission**

Rules and subscriptions are associated with user permissions that determine whether a user can change the item definition or any part of it. The PUBLIC user includes any user of the system. Read permission is always granted. The user creating the rule or subscription is the Owner. The Owner can change ownership to another user. Update permission is needed to modify component definition.

## **persistence**

The ability of a computerized system to remember the state of data or objects between runs.

## **popup menu**

A floating list of actions. To open a popup menu, click an object and hold down the right mouse button; the available actions depend on the object

## **protocol**

A set of rules that govern the transmission and reception of data.

## **put**

A request to store a message in a queue. Compare to get.

## **regular expression**

Strings which express rules for string pattern matching.

## **rule**

A rule is uniquely defined by its application group, message type, and rule name. It contains evaluation criteria (a rules expression) and is associated with subscriptions to perform if the rule evaluates to true. Rules also have permissions that determine user access.

## **scalable, scalability**

Changeable in size and configuration. The ability to expand, implying minimal procedural change to accommodate growth.

## **server**

A computer that, through its network connections, carries out parts of computing tasks for other networked computers.

## **shared subscription**

A subscription that is associated with more than one rule. This subscription will only be retrieved once by the Rules APIs even if multiple associated rules evaluate true.

**subscription**

A subscription is uniquely identified by its application group, message type, and subscription name. It contains actions with options and can be associated with one or more rules. Subscriptions also have permissions that determine user access to change the subscription definition.

**synchronous**

In electronic messaging, a method of operation in which sender and receiver applications are tightly coupled and dependent. The receiver must answer the sender's message immediately with a well-defined response; the sender must wait for the receiver's response before proceeding to the next operation. Compare to asynchronous.

**table**

A database remembers relationships between pieces of information by storing the information in tables. The columns and rows in each table define the relationships in a highly structured way. We can classify tables by function into two types: support tables and data tables. Most tables fit into only one category, but some can serve as both support and data tables.

A support table stores information that changes infrequently and functions as a list from which you make selections.

**tag**

A set of bits or characters that identify various conditions about data in a file. In Formatter, a standard value indicating the field's name.

**transaction**

An activity or request. Additions, changes, and deletions are typical transactions stored in a computer.

**transaction management**

A method of handling electronic messaging such that only committed messages are sent and only messages received and committed are considered delivered.

**transaction processing**

A method of handling computer operations in which the operations take place immediately upon receipt of the processing request. Also called realtime or online system. Compare to batch processing.









---

# Index

---

## A

- access modes 70
- actions
  - deleting 117
  - Generic action 113
  - Put Message action 113
  - Reformat action 113
- Actions property sheet 118
- adding actions to subscriptions 113
- adding application groups 94
- adding comments to subscriptions 111
- adding input controls to flat input formats 13
- adding parse controls 44
- adding Rules 97
- adding subscriptions 109
- adding user permissions 112
- alternative input formats 78
- alternative output formats 78
- AND operator 106
- application groups 83, 94
- Arrange Icons function 19, 93
- arranging icons 19, 93
- ASCII Extended Character Set 121
- assigning fields to input controls 13
- assigning update permissions 88

## B

- Boolean operators
  - AND operator 106
  - OR operator 106
- building formats 27

## C

- Calculated Field output control 49
- cascading windows 16
- case sensitivity 14, 88
- case-sensitive string comparisons 103
- changing user permissions 112
- Clear function 16
- clearing text 16
- Close function 15, 90
- closing a window 15
- closing Formatter 16
- closing Rules 90
- closing Rules window 90
- comments 111
- comparison values 27
- conditional branching 53, 118
- Conditional Branching mode 118
- Conditional output control 52
- context-sensitive help 22
- control tables 7
- converting messages 7
- Copy function 16
- copying objects 11

- copying text 16
- creating 64
  - compound input formats 65
  - compound output formats 73
  - expressions 103
  - flat input formats 64
  - flat output formats 13, 69
  - formats 63
  - input controls 32
  - literals 28
  - output controls 44
  - output formats 11
  - user-defined data types 75
  - validation parameters 77
- creating new rules 92
- creating new subscriptions 92
- Customize Toolbars function 16
- Customize toolbars function 91
- Cut function 16
- cutting text 16

## D

- Data Field (Name Search) output control 46
- Data Field (Tag Search) output control 47
- data type comparisons 103
- data type values 106
- data types
  - creating user-defined 75
  - user-defined 75
- data-producing operations 46
- DATE field 103
- DATETIME field 103
- default values 27
- Delete Action function 117
- Delete function 92
- deleting a name value pair 77
- deleting actions 117
- deleting Rules 92, 98
- deleting subscriptions 92
- delimiters 27
- displaying format components 16
- displaying tree structure 21
- documentation set 4
- drag-and-drop feature 11
  - adding input controls to flat input formats 13
  - assigning fields to input controls 13
  - creating output formats 11
  - reordering fields 11, 13
- drap-and-drop feature
  - creating flat output formats 13
  - field mapping 13
- Duplicate function 92, 110
- duplicating Rules 92, 99
- duplicating subscriptions 92, 110

## E

- EBCDIC Character Set 127
- Edit menu 16
  - Clear function 16
  - Copy function 16
  - Cut function 16
  - Find function 16
  - Paste function 16
- Existence Check Field output control 53
- Exit function 16, 90
- exiting Formatter 16
- exiting Rules 90
- Export function 23
- exporting components 23
- exporting formats 23
- Expression Components property sheet 103
- Expression property sheet 102, 107
- expressions 83, 102, 103
  - creating 107
  - deleting 108
  - modifying 107

## F

- field existence operator 103
- field mapping 13
- field names 30
- field non-existence operator 103
- fields 30
  - creating 30
  - reordering 13
  - searching 20
- field-to-field comparisons 103
- File menu 15, 90
  - Close function 15, 90
  - Exit function 16, 90
  - New function 15, 90
  - Print Setup function 90
  - Print Tree function 16, 90, 92
- Filed List 103
- Find function 16, 20
- finding text 16
- flat input formats 27
- flat output formats 27
- FLOAT operator 103
- format terminators 27
- formats
  - building 27
  - compound output formats 73
  - creating
    - compound input formats 65
    - flat input formats 27
    - flat output formats 27
    - creating flat output formats 69
- Formats function 62
- Formatter 2, 7, 70
  - access modes 70
  - alternative formats 78
    - optional components 78
    - optional fields 78
    - output formats 80
    - parsing 79
  - alternative input formats 78
  - alternative output formats 78

- arranging icons 19
- building formats 27
- case sensitivity 14
- clearing text 16
- closing a window 15
- conditional branching 53
- control tables 7
- copying text 16
- creating
  - fields 30
  - formats 63
  - literals 28
  - user-defined data types 75
- creating formats
  - compound input formats 65
  - compound output formats 73
  - flat input formats 64
  - flat output formats 69
- creating input controls
  - user-defined data types 76
- cutting text 16
- displaying format components 16
- exiting 16
- Export 23
- Export function 23
- fields 30
- finding text 16
- Formats function 62
- Formatter indow
  - tree functionality 11
- Formatter toolbar
  - Help Topics button 22
  - searching fields 20
  - viewing items in the Print Tree window 21
- Formatter window 10
  - copying objects 11
  - drag-and-drop feature 11
  - opening new windows 11, 20
- Import function 24
- input controls 31, 77
  - creating 32
  - saving as output controls 43
  - Year 2000 compliance 43
- input formats
  - alternative formats 78
  - tagged 80
- literals 27
- mapping fields 72
- menu bar 15
- menus 15
  - Edit menu 16
  - File menu 15
  - Help menu 16
  - Left Pane popup menus 17
  - popup menus 17
  - Right Pane popup menu 19
  - Tools menu 16
  - Window menu 16
- opening a new window 15
- operating modes 7
- optional components 78
- optional fields 78
- output controls 44
  - Calculated Field 49
  - Conditional 52

- creating 44
- Data Field (Name Search) 46
- Data Field (Tag Search) 47
- data-producing operations 46
- Existence Check Field 53
- Input Field Exists 54
- Input Field Value = 55
- Left Operand Field 50
- Literal 48
- Math Expression operations 58
- operation substitute strings 60
- output operation collections 57
- output operations 56
- Right Operand Field 51
- Rules Field 53
- output formats 78
- pasting text 16
- popup menus 15
- regular expressions 39
- starting Formatter 8
- Used In function 22
- user-defined data types 75, 77
- Validation Parameters property sheet 77
- viewing components 21
- viewing icons 19
- Year 2000 compliance 43
- Formatter icon 8
- Formatter toolbar
  - Find function 20
  - New function 20
  - opening new windows 20
  - Print Tree function 21

**G**

- Generic action 113
- Glossary 141

**H**

- Help menu 16, 91
- Help Topics button 22

**I**

- Import function 24
- importing components 24
- importing formats 24
- input controls 31
  - mandatory data 32
  - optional data 32
  - parsing data 32
  - regular expressions 39
  - user-defined data types 76
- Input Field Exists output control 54
- Input Field Value = output control 55
- input formats 80
- Insert menu 90
- INT operator 103

**L**

- layering windows 16
- Left Operand Field output control 50
- Left Pane popup menus 17, 91

- Literal output control 48
- literals 28

**M**

- mandatory data 32, 44
- mapping fields 70, 72
- Math Expression operations 58
- menu bar 15
- menus 15
- message types 83, 95
  - adding message types 95
  - Field List 103
- modifying expressions 107
- MQSeries Integrator Overview 2

**N**

- NEONFormatter 2
- NEONRules 2
- New Application function 90
- New function 15, 20, 90, 92

**O**

- opening a new Rules window 90
- opening a new window 15
- opening new Formatter windows 11, 20
- operating modes
  - converting messages 7
  - parsing messages 7
- operation substitute strings 60
- operators
  - 103
  - <= operator> 103
  - <> operator 103
  - = operator 103
  - > operator 103
  - >= operator 103
  - AND operator 106
  - Boolean operators 106
  - case-sensitive string comparisons 103
  - data type values 106
  - DATE field 103
  - DATETIME field 103
  - field existence operator 103
  - field non-existence operator 103
  - field-to-field comparisons 103
  - FLOAT operator 103
  - INT operator 103
  - OR operator 106
  - STRING field 103
  - TIME field 103
- optional components and fields 78
- optional data 32, 44
- OR operator 106
- output controls
  - formatting output data 44
  - mandatory data 44
  - Math Expression operations 58
  - optional data 44
  - output operation collections 57
  - output operations 56
  - saving input controls 43
- output formats 80

- output operation collections 57
- output operationcollections 57
- output operations 56
- Overview 2
- ownership of rules 88

## P

- pad character 27
- parsing 79
- parsing input data 32
- parsing messages 7
- Paste function 16
- pasting text 16
- permissions 88
  - adding 101
  - changing 101
- popup menus 15, 17
- prefixes 27
- Print button 90, 92
- Print Setup function 90
- Print Tree function 16, 21, 90, 92
- Print Tree icon 90, 92
- printing application groups 90, 92
- printing output 90
- processing literals 27
- property sheets 87
- Put Message action 113
- putqueue 83

## R

- reformat 83
- Reformat action 113
- regular expressions 39
- Rename Subscription function 110
- renaming application groups 95
- renaming Rules 100
- renaming subscriptions 110
- reordering fields 11, 13
- repeating sequence terminators 27
- Right Operand Field output control 51
- Right Pane popup menu 19
- Right Pane popup menus 92
- rule security 101
- Rules 2
  - adding application groups 94
  - adding Rules 97
  - application groups 83, 94
  - Arrange Icons function 93
  - assigning ownership 88
  - building Rules 94
  - case sensitivity 88
  - conditional branching 118
  - creating expressions 107
  - Delete function 92
  - deleting expressions 108
  - deleting Rules 98
  - deleting subscriptions 111
  - Duplicate function 92
  - duplicating Rules 99
  - Expression Components property sheet 103
  - Expression property sheet 102
  - expressions 83, 102
  - Field List 103

- functions 106
- menus
  - Left Pane popup menus 91
  - Right Pane popup menus 92
- message types 83, 95, 103
- New function 92
- opening a new Rules window 87
- ownership 88
- property sheets 87
- renaming application groups 95
- renaming Rules 100
- rule security
  - adding user permissions 101
  - Security property sheet 100
- Rules menu bar 89
  - File menu 90, 92
  - Help menu 91
  - Insert menu 90
  - Tools menu 91
  - Window menu 91
- Rules menus 89
- Rules operators 103
- Rules popup menus 91
- Rules toolbar 93
- Rules window 86
  - security 88
  - starting Rules 84
  - subscriptions 83, 108, 109, 110, 111, 112, 113, 117
  - tree functionality 87
  - update permissions 88
  - View function 93
- Rules Field output control 53
- Rules hierarchy 86
- Rules operators 103
  - data type comparisons 103
- Rules toolbar 93
- Rules window 86, 87
- RuntimeDataLookup function 77

## S

- saving input controls as output controls 43
- searching fields 20
- security 88, 112
- Security property 112
- Security property sheet 100
- Security Summary Report function 91
- starting Formatter 8
- starting Rules 84
- STRING field 103
- Subscription List 108
- subscriptions 108
  - adding 109
  - adding actions 113
  - adding comments 111
  - adding security 112
  - changing security 112
  - deleting 111
  - deleting actions 117
  - duplicating 110
  - renaming 110
  - security 112
- substitute operations 60
- substitute values 27
- suffixes 27

**T**

- tag values 27
- tagged input format 80
- tiling windows 16
- TIME field 103
- Tools menu
  - Customize Toolbars function 16
  - Customize toolbars function 91
  - Security Summary Report function 91
- tree functionality 11, 87
- tree structure, displaying 21
- trim characters 27

**U**

- Update permission 108
- Used In function 22
- user permissions 112
- user-defined data types 75

**V**

- validation callback object 77
- validation parameters 77
- Validation Parameters property sheet 77
- View function 19, 93
- viewing components 21
- viewing icons 19, 93
- viewing items in Print Tree window 21

**W**

- Window menu 16
  - cascading windows 16, 91
  - layering windows 16, 91
  - tiling windows 16, 91

**Y**

- Year 2000 compliance 43





## **Sending your comments to IBM**

### **MQSeries Integrator**

#### **User's Guide**

#### **GC34-5504-00**

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM.

Feel free to comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this book. Please limit your comments to the information in this book only and the way in which the information is presented.

To request additional publications or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

You can send your comments to IBM in any of the following ways:

- By fax:
  - From outside the U.K., use your international access code followed by 44 1962 870229
  - From within the U.K., use 01962 870229

Electronically, use the appropriate network ID:

- IBM Mail Exchange: GBIBM2Q9 at IBMMAIL
- IBMLink: HURSLEY(IDRCF)
- Internet: idrcf@hursley.ibm.com

Whichever you use, ensure that you include:

- The publication number and title
- The page number or topic number to which your comment applies
- Your name/address/telephone number/fax number/network ID



GC34-5504-00