

TIBCO BusinessEvents®

Developer's Guide

Software Release 5.0

April 2011

The Power to Predict®

 **TIBCO®**
The Power of Now®

Important Information

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document contains confidential information that is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, The Power of Now, TIBCO ActiveMatrix, TIBCO ActiveMatrix BusinessWorks, TIBCO Administrator, TIBCO ActiveSpaces, TIBCO Designer, TIBCO Enterprise Message Service, TIBCO Hawk, TIBCO Runtime Agent, TIBCO Rendezvous, are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

EJB, Java EE, J2EE, and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

THIS SOFTWARE MAY BE AVAILABLE ON MULTIPLE OPERATING SYSTEMS. HOWEVER, NOT ALL OPERATING SYSTEM PLATFORMS FOR A SPECIFIC SOFTWARE VERSION ARE RELEASED AT THE SAME TIME. SEE THE README.TXT FILE FOR THE AVAILABILITY OF THIS SOFTWARE VERSION ON A SPECIFIC OPERATING SYSTEM PLATFORM.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

Copyright © 2004-2011 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Figures	xvii
Tables	xix
TIBCO BusinessEvents Resource Reference Tables	xxi
Preface	xxiii
Changes from the Previous Release of this Guide	xxiv
TIBCO BusinessEvents Express	xxvii
Related Documentation	xxviii
TIBCO BusinessEvents and Add-On Product Documentation	xxviii
Accessing TIBCO BusinessEvents Functions Reference Documentation	xxxi
Other TIBCO Product Documentation	xxxi
Typographical Conventions	xxxii
Connecting with TIBCO Resources	xxxv
How to Join TIBCOCommunity	xxxv
How to Access All TIBCO Documentation	xxxv
How to Contact TIBCO Support	xxxv
Chapter 1 Introduction to Project Development	1
Chapter 2 Project Tasks	3
Creating a Project	4
Finding a Project Element	6
Exporting (Generating) Concept and Event Schema (XSD) Files	7
Validating a Project or Project Resource	9
Working with External Library and Custom Function Paths	10
Working with Project Libraries	12
Creating (Exporting) a Project Library	12
Adding and Removing Project Libraries in a Project	14
Working with Global Variables	17
Adding and Managing Global Variables	17
Using Global Variables	20
Overriding Global Variables at Deploy Time	20

Storing Trusted Certificates Outside of Your Project	22
Building an Enterprise Archive (EAR File)	24
Deployment Options	24
Building an EAR File in TIBCO BusinessEvents Studio	24
Chapter 3 TIBCO BusinessEvents Studio Tools Utility	27
Building an Enterprise Archive (EAR File) at the Command Line	28
Importing a TIBCO BusinessEvents (4.x) Studio Project at the Command Line	29
Importing a TIBCO BusinessEvents 3.x Project at the Command Line	32
Working With Project Libraries at the Command Line	34
Migrating Core Coherence Functions at the Command Line	37
Importing an Excel File into TIBCO BusinessEvents Decision Manager at the Command Line	38
Validating a Decision Table at the Command Line	40
Generating Decision Table Class Files at the Command Line	41
Committing Project Artifacts to RMS at the Command Line	43
Setting up RMS Projects at the Command Line	44
Chapter 4 Element Refactoring Operations	45
Renaming, Moving, Deleting, and Copy-Pasting Elements	46
Project Level Actions	46
Renaming, Moving, and Deleting Elements	47
Copy-Pasting an Element	50
Migrating Core Coherence Functions	52
Automatic Refactoring Actions and Limitations	55
Refactoring Limitations	55
Refactoring for Move and Rename Operations	55
Refactoring for Delete Operations	57
Chapter 5 Channels and Destinations	59
Overview of Channels and Destinations	60
Types of Channels	60
Selecting a Serializer	61
Deploy-time Configuration	61
Mapping Incoming Messages to Non-default Events	62
Working with Rendezvous Channels	63
Rendezvous Message Header	63
Basic Serializer	63
Serializer For Use with Payloads	63
Avoiding REGISTRATION COLLISION RVCM Advisory Messages	64

Working with Local Channels	66
Event Use Count	66
Using a Local Channel	66
Adding Channels and Destinations	67
Adding a Destination to a Channel	69
Communicating with Other Sources using TCP	70
APIs for TCP Communication	72
Channel Resource Reference	73
Wizard and Configuration Section	73
Destination Resource Reference	76
Chapter 6 JMS Channels	81
Overview of JMS Channels	82
Selecting a JMS Serializer	83
BytesMessageSerializer	83
TextMessageSerializer	84
Creating Unique JMS DurableSubscriber Name Properties	85
Changing the JMS Message Acknowledgment Mode	87
Using CLIENT_ACKNOWLEDGE Mode with Websphere MQ and Cache-Aside	89
Using JMS Header Properties in Incoming and Outgoing Messages	90
Setting Certain Header Properties in Destinations	90
Setting Header Properties Using Header Properties from Incoming JMS Messages	90
Setting JMS Header Properties in Outgoing JMS Messages Using Event Properties	90
How TIBCO BusinessEvents Sets the JMSReplyTo Header in Outgoing Messages	91
JMS Header Field Names	92
Chapter 7 HTTP and SOAP Channels	95
Overview of HTTP and SOAP Channels	96
Selecting the Server Type	96
SOAP Support	96
Working with HTTP Requests	97
Mapping of HTTP Requests to Events	97
Sending Non-ASCII Content to Event Properties	97
Mapping of HTTP Request URI to Destination	98
Configuring TIBCO BusinessEvents to Receive and Send HTTP Requests	98
HTTP Channel Advanced Configuration Settings	102
Defining Event Properties for Standard HTTP Header Properties	109
Using HTTP Functions to Configure HTTP Request Messages	110
Generating a Self-Signed SSL Certificate (Keystore)	110
Getting POST Data	110
Loading Trusted Certificates	111

Sending an Event	111
Sending an Asynchronous Request (Not Secure)	112
Sending a Secure Asynchronous Request	113
Sending a Secure Synchronous Request	114
Configuring TIBCO BusinessEvents as a SOAP Server and Client	116
Overview of SOAP Related Resources	116
Mapping of SOAP Request URI to Destination	117
Manually Creating Resources to Work with SOAP Services	117
Creating Resources Using the WSDL Import Utility	119
Exporting Resources as a WSDL File	120
Parsing and Building SOAP Messages	123
Working with Incoming SOAP Messages (Event Payloads)	123
Working with Outgoing SOAP Messages (Event Payloads)	126
Understanding the WSDL to Project Resource Mapping	128
Example WSDL	128
Example Project Folder Structure	129
How Project Artifacts are Named	131
Chapter 8 Simple Events	133
Overview of Simple Events	134
Using Inheritance	134
Working with Events in Rules	135
Explicitly Assert Events Created in Rules	135
Specifying Default and Non-Default Destinations	135
Scheduling Simple Events	135
Adding a Simple Event	136
Simple Event Reference	137
Wizard and Configuration (Standard Tab)	137
Properties (Standard Tab)	139
Declaration and Expiry Action (Advanced Tab)	139
Payload (Advanced Tab)	140
Simple Event Attributes Reference	143
Chapter 9 Time Events and Scheduler Functions	145
Overview of Time Events	146
Scheduled Time Events	146
Rule Based Time Events	147
Working With Time Events	148
Adding a Time Event	148
Configuring a Rule Based Time Event in a Rule or Rule Function	148
TimeEvent Resource Reference	150

Wizard and Configuration Tab	150
TimeEvent Attributes Reference	152
Rule Based TimeEvent Function Reference	154
Using Scheduler Functions (Requires Cache OM)	155
Chapter 10 Advisory Events	157
Working With Advisory Events	158
Uses of Advisory Events	158
Adding an Advisory Event to a Rule	158
Advisory Event Attributes Reference	160
Chapter 11 Concepts	163
Overview of Concepts	164
Adding Concepts and Concept Relationships	165
Adding a Concept	165
Adding Concept Relationships	165
Concept Resource Reference	167
Wizard and Configuration Tab	167
Properties	168
Metadata	169
Concept Attributes Reference	170
Chapter 12 Scorecards	173
Understanding and Working With Scorecards	174
Adding a Scorecard	174
Using a Scorecard in Rules	175
Scorecard Resource Reference	175
Chapter 13 Domain Models	177
Overview of Domain Models	178
Domain Model Value Descriptions for User Friendly Presentation	178
Adding Domain Models	179
Importing and Exporting Domain Models	183
Export To and Import From Excel	183
Import From a Database Table	184
Associating Domain Models with a Property	186
Validating Data in Domain Models	187

Chapter 14 Shared Resources	189
Adding a Shared Resource	190
Enabling the Test Connection Feature	190
HTTP Connection	192
Wizard and Configuration Tab	192
Configure SSL	193
Identity Resource	195
Wizard and Configuration Tab	195
JDBC Connection	197
Wizard and Configuration Tab	197
Connection Pooling	201
Test Connection Button	202
JMS Application Properties	203
Wizard and Configuration Tab	203
JMS Connection	204
Wizard and Configuration Tab	204
Test Connection Button	207
Advanced Tab	207
Configure SSL	208
JNDI Configuration	211
Wizard and Configuration Tab	211
Advanced Section	212
Rendezvous Transport	214
Wizard and Configuration Tab	214
Configure SSL Button	215
Advanced Section	216
Chapter 15 Rules and Rule Functions	219
Overview of Rules and Rule Functions	220
Form-based and Source Rule Editors	220
Rule Components	220
Effect of Cache Only Cache Mode	221
Adding a Rule	222
Rule Editor Reference	226
Adding a Rule Function	228
Rule Function Resource Reference	231
Using Variables and Functions in the Rule Editor	233
Using Catalog Functions in the Rule Editor	233
Using Global Variables in the Rule Editor	233
Using the Function Argument Mapper	234

Using Priority and Rank to Control Order of Rule Execution	237
Using the Quick Fix Feature in the Rule Editor	239
Tips for Working in the Rule Editor	241
Event Preprocessors	243
Configuring an Event Preprocessor	243
Transaction Error Handler Rule Function	244
Chapter 16 Functions	247
Overview of Catalog Functions	248
Built-in Functions	248
Custom Functions	250
Enabling Extended Functions	251
Function Tooltips and Decorations	252
Tool Tips	252
Decorations Indicating Where Functions can be Called	252
Temporal Functions and Their Parameters	254
VRF Functions	256
Guidelines for Use of Cache Related Functions	260
Guidelines for Use of Oracle Coherence Functions	262
Constants and Extractors Categories	262
Filters Category	262
Query Category	262
Query Optimization	263
Adding Custom Functions	265
Task Summary	265
Restrictions on Use of Custom Functions	267
Static and Non-Static Functions	267
Return Types	267
Name Overloading	267
Editing Custom Functions	267
Structure of a Function Catalog	268
Elements in the Function Catalog	268
Example Function Catalog	271
Chapter 17 Rule Language Grammar	273
Rule Language Basics	274
Whitespace	274
Comments	274
Separators	275
Identifiers (Names)	275
Local Variables	276

Literals	277
Escape Sequences	278
Operators	278
Keywords and Other Reserved Words	280
Attributes	281
Accessing Concept and Event Properties	283
Concept Property Atom	283
Concept Property Array	284
Event Property	285
Exception Handling	286
Syntax	286
Examples	287
Flow Control	289
if/else	289
for	289
while	290
Chapter 18 Rule Language Datatypes	291
Concept Properties to XML Datatype Conversions	292
Compatibility of Operators with Types	293
Correcting Inconsistencies of Type	295
String Operands	295
Arithmetic Expressions	295
Assignment Conversion	296
Function Argument Conversion	296
Chapter 19 Mapping and Transforming Data	297
Overview of Mapping and Transformation	298
Function Section	298
Input Section	298
Mapping and Transforming Data to Function Input	299
Statements, Hints, and Errors	299
Buttons, Menus, and Icons	301
Toolbar and Right-Click Menu on the Input Section	301
Icons for Schema Element Datatypes	304
Qualifier Icons	305
Specifying Constants	307
Date and Datetime Strings in Constants	307
Data Validation	308
Repairing Incorrect Mappings	309
Shortcuts	310

Statement Menu Options	310
Dragging to the Left	310
Cutting and Pasting	311
Automatic Testing (at Runtime)	312
Examples of Mappings	314
Setting an Element Explicitly to Nil	314
Merging Input from Multiple Sources	315
Converting a List Into a Grouped List	319
Merging Two Corresponding Lists	323
Coercions	326
XSLT Statements	330
Attribute	330
Choose	330
Comment	331
Copy	331
Copy-Contents-Of	331
Copy-Of	331
Element	332
For-Each	332
For-Each-Group	332
Generate Comment	332
Generate PI	333
If	333
Value-Of	333
Variable	333
Chapter 20 XPath Formula Builder	335
XPath Basics	336
Addressing Schema Elements	336
Evaluation Context	337
Namespaces	337
Search Predicates	337
Testing for Nil	338
Comments	338
The XPath Formula Builder	339
String Representations of Datatypes	342
Date and Time Functions	343
Chapter 21 ActiveMatrix BusinessWorks Integration	347
Overview of Integration with ActiveMatrix BusinessWorks	348
The Container and the Contained Engine	349
Integration Components	350

ActiveMatrix BusinessWorks Activities Palette	350
TIBCO BusinessEvents Functions	350
Design Considerations	352
Integration Scope	352
Thread Management	352
Design Considerations Related to Container	353
Fault Tolerance With a TIBCO BusinessEvents Container	354
Tips for Working With ActiveMatrix BusinessWorks Containers	354
Configuring the Environment for ActiveMatrix BusinessWorks Containers	355
Configuring the Environment For TIBCO BusinessEvents Containers	357
Configuring a RuleServiceProvider Configuration Resource	362
TIBCO BusinessEvents RuleServiceProvider Configuration Resource Reference	363
Configuration	363
Working With the TIBCO BusinessEvents Activities	365
Receive Event Resource Reference	366
Configuration	366
Misc	367
Output	367
Send Event Resource Reference	368
Configuration	368
Input	368
Wait for Event Resource Reference	369
Configuration	369
Event	370
Input	370
Output	371
Invoking a TIBCO BusinessEvents Rule Function from a Process	372
Specifying Input Arguments	372
Using Synchronous Invocation	372
Using the lockWM Parameter	373
Overriding the Rule Function at Runtime	373
Working With Invoke RuleFunction Activities	374
Invoke RuleFunction Resource Reference	376
Configuration	376
Input	377
Output	377
Working with the BusinessWorks Functions	378
Providing Paths to TIBCO BusinessEvents Project Resources Using Schemas	378
Using invokeProcess()	378
Using startProcess()	380
Using cancelProcess()	382
Using init()	382

Using shutdown()	382
Chapter 22 TIBCO BusinessEvents Performance Profiler	385
Overview of Profiler	386
Changing the Delimiter Character	387
Turning Profiler On and Off	388
To Turn Profiler On and Off Using TIBCO BusinessEvents Monitoring and Management	388
To Turn Profiler On and Off Using Properties	388
To Turn Profiler On and Off Using Functions	391
To Turn Profiler On and Off Using TIBCO Hawk Methods	392
Profiler Reference	394
Chapter 23 Testing and Debugging Projects	399
Overview	400
Debugging	400
Testing Projects	400
Launch Configurations	401
Test Data	401
The Rule Agenda and Variable Views	401
Viewing and Understanding Results	402
Preparing to Run (Test) or Debug a Project	403
Build an EAR File	403
Create Test Data (as Desired)	403
For Remote Debugging Only, Configure Java Debug Interface (JDI)	403
Adding and Working with Launch (Debug or Run) Configurations	405
Launch Configurations Reference	408
For Testing and Local Debugging	408
For Remote Debugging	409
Creating and Working With Test Data	410
Working with Concept and Event Test Data	410
Working with Rule Data	412
Setting Breakpoints and Running Debugger	413
Running Tester	415
Running the Engine	415
Asserting Rule Input Data	415
Viewing the Results	417
Understanding Result Data	417
Viewing and Understanding Working Memory Contents	418
Chapter 24 Diagrams	421
Overview of Diagrams	422

Working with Diagrams	424
Configuring Diagram Preferences and Properties	424
Performing Common Tasks	425
Using Diagram Tools	426
Exporting a Diagram to an Image	428
Printing a Diagram	429
Setting Diagram Preferences	431
Project Analyzer and Selected Entity Project Diagrams	432
Advantages of Project Analyzer and Selected Entity Project Diagrams	433
Working with Project Analyzer and Selected Entity Project Diagrams	433
Dependency Diagrams	436
Sequence Diagrams	438
Concept Model Diagrams	439
Event Model Diagrams	440
Diagram Options and Tools Reference	441
Layout Options	443
Chapter 25 TIBCO BusinessEvents Studio Preferences	445
Setting Preferences	446
TIBCO BusinessEvents	446
Code Generation Shared Archive Ignored Resources Preferences	446
Compare/Merge, Decision Table, and Decision Tree	446
Diagram Preferences	446
Printing Preferences	447
Error/Warning Preferences	447
Rules Preferences and Rule Debug Preferences	447
Rules Management Server Preferences	447
Tester and Tester Appearance Preferences	447
Decision Table Related Preferences	448
Decision Table Analyzer Preferences	449
Decision Table Appearance Preferences	449
Compare/Merge Preferences	450
Diagram Preferences	452
Tester Preferences	457
Appendix A Handling Null Properties	459
Handling Null Concept Property Values	460
Enabling Use of the Nillable Attribute	460
Enabling Null Property Values to Appear When Serializing Concepts to XML	460
Examples of Nillable Attribute and Null Properties Settings	461
Enabling and Setting Special Treatment of Numeric Null Values	462

Setting Runtime Properties for Special Treatment of Null Values.463

Property Reference for Null Property Handling464

Figures

Figure 1 Serializer and Deserializer Behavior 61

Figure 2 Rule Form Editor 222

Figure 3 Rule Source Editor 223

Figure 4 Temporal Functions Parameters 254

Figure 5 Dragging to the left to change a hint to a statement 311

Figure 6 Creating an XPath formula 341

Tables

Table 1	General Typographical Conventions	xxxii
Table 2	Syntax Typographical Conventions	xxxiii
Table 3	Global Variable Reference	18
Table 4	Build Enterprise Archive Reference	24
Table 5	TIBCO BusinessEvents Studio Tools Options for Building an EAR File	28
Table 6	TIBCO BusinessEvents Studio Tools Options for Importing an Existing Project	30
Table 7	TIBCO BusinessEvents Studio Tools Options for Importing 3.x TIBCO Designer Projects	32
Table 8	TIBCO BusinessEvents Studio Tools Options for Working with Project Libraries	35
Table 9	TIBCO BusinessEvents Studio Tools Options for Importing Excel Files into TIBCO BusinessEvents Decision Manager 38	
Table 10	TIBCO BusinessEvents Studio Tools Options for Validating Decision Tables.	40
Table 11	TIBCO BusinessEvents Studio Tools for Generating Class Files	41
Table 12	TIBCO BusinessEvents Studio Tools Options for RMS Commit	43
Table 13	TIBCO BusinessEvents Studio Tools Options for RMS Project Setup.	44
Table 14	Mapping of Coherence Functions to DataGrid Functions	53
Table 15	Refactoring for Move and Rename Operations	56
Table 16	Which Serializers to Use for JMS Message Types	83
Table 17	Variables for Use with DurableSubscriberName	85
Table 18	JMS Message Acknowledgement Modes	87
Table 19	JMS Header Field Names	92
Table 20	HTTP Channel Advanced Configuration Settings	102
Table 21	Exporting Project Artifacts as WSDL	121
Table 22	Imported WSDL Project Artifacts	129
Table 23	Simple Event Payload Element Parameters	140
Table 24	Attributes Used for Each Type of Advisory Event	160
Table 25	Quick Fix Feature Options	239
Table 26	Tips for Working in the Rule Editor.	242
Table 27	Common Arguments for VRF Functions	258

Table 28	Function Catalog Elements	268
Table 29	Escape Sequences	278
Table 30	Operators in the TIBCO BusinessEvents Rule Language.	278
Table 31	Attributes	281
Table 32	Concept Properties to XML Datatype Conversions.	292
Table 33	Operator Matrix	293
Table 34	Input tab toolbar buttons	301
Table 35	Icons for schema items	304
Table 36	Additional icons for hints	305
Table 37	Datatype validation	308
Table 38	XPath Formula Builder Reference.	339
Table 39	Formatting characters in date or time strings	343
Table 40	Example date and time format patterns.	345
Table 41	Design Considerations Related to Container (Integration with ActiveMatrix BusinessWorks) . . .	353
Table 42	ActiveMatrix BusinessWorks integration Properties for TIBCO BusinessEvents Containers . . .	360
Table 43	Profiler Configuration Properties	388
Table 44	Profiler Column Heading Reference	394
Table 45	Types of diagrams available for each type of project element	423
Table 46	Creating Diagrams	425
Table 47	Context Menu Options for Canvas and Objects	428
Table 48	TIBCO BusinessEvents Diagram Tools Reference.	441
Table 49	Global Diagram Preferences	452
Table 50	Specific Diagram Preferences (Alphabetical)	454
Table 51	Reference to Tester Preferences	457
Table 52	Reference to Tester Appearance Preferences	457
Table 53	Properties for Null Property Handling	464

TIBCO BusinessEvents Resource Reference Tables

Channel Resource Reference	73
Destination Resource Reference	76
Simple Event Reference	137
TimeEvent Resource Reference	150
Concept Resource Reference	167
Scorecard Resource Reference	175
Rule Editor Reference	226
Rule Function Resource Reference	231
TIBCO BusinessEvents RuleServiceProvider Configuration Resource Reference	363
Receive Event Resource Reference	366
Send Event Resource Reference	368
Wait for Event Resource Reference	369
Invoke RuleFunction Resource Reference	376

Preface

TIBCO BusinessEvents® allows you to abstract and correlate meaningful business information from the events and data flowing through your information systems, and take appropriate actions using business rules. By detecting patterns within the real-time flow of events, TIBCO BusinessEvents can help you to detect and understand unusual activities as well as recognize trends, problems, and opportunities. TIBCO BusinessEvents publishes this business-critical information in real time to your critical enterprise systems or dashboards. With TIBCO BusinessEvents you can predict the needs of your customers, make faster decisions, and take faster action.

TIBCO BusinessEvents
The Power to Predict®

Topics

- [Changes from the Previous Release of this Guide, page xxiv](#)
- [TIBCO BusinessEvents Express, page xxvii](#)
- [Related Documentation, page xxviii](#)
- [Typographical Conventions, page xxxii](#)
- [Connecting with TIBCO Resources, page xxxv](#)

Changes from the Previous Release of this Guide

This section itemizes the major changes from the previous release of this guide.

Quick Fix Feature

In the rule editor you can now create code that references entities and properties that do not yet exist. Clicking a new lightbulb icon on the left displays one or more links to appropriate wizards or dialogs where you can create the entity or rule function or property. See [Using the Quick Fix Feature in the Rule Editor on page 239](#).

In addition a Quick Fix feature works in the problems view. See [Validating a Project or Project Resource on page 9](#).

Metadata Properties for Backing Store Now in CDD File

Documentation of metadata properties is updated. Metadata properties in entity resources are now disabled except for database concepts (in the TIBCO BusinessEvents Data Modeling add-on). Backing store metadata properties are now configured in the CDD file, as documented in *TIBCO BusinessEvents Administration*. Notes stating that metadata properties can be added for customization only are present in [Adding a Simple Event on page 136](#), [Adding Concepts and Concept Relationships on page 165](#), and [Understanding and Working With Scorecards on page 174](#). Reference topics also mention that metadata properties are not to be used ([Simple Event Reference on page 137](#) and [Concept Resource Reference on page 167](#)).

Adding External Library and Custom Function Paths

When you work with external libraries or custom functions in your project at design time, you must ensure that the engine can find all the relevant libraries. The section [Working with External Library and Custom Function Paths on page 10](#) explains how to use new features to do this.

The same functionality is also available from the Launch Configuration Editor Environment tab, as noted in [Chapter 23, Testing and Debugging Projects, on page 399](#).

Mention of this requirement is added to [Adding a Shared Resource on page 190](#).

Enabling and Using Query Optimization (Oracle Coherence Only)

A new subsection within the section [Guidelines for Use of Oracle Coherence Functions](#) called [Query Optimization on page 263](#) explains how to enable and use property indexes.

Miscellaneous

- Validation can now be done at the project, folder, or individual resource level. See [Validating a Project or Project Resource on page 9](#) for details.
- HTTP Channel Properties Now in HTTP Channel Advanced Tab. See [HTTP Channel Advanced Configuration Settings on page 102](#).
- You can now store trusted certificates for SSL in a folder external to the project. You can use a special global variable to reference this folder. See [Storing Trusted Certificates Outside of Your Project on page 22](#). Also see [Using HTTP Functions to Configure HTTP Request Messages on page 110](#). Also see [HTTP Connection on page 192](#), and JMS Connection [JMS Connection on page 204](#) > [Configure SSL on page 208](#)
- The section [Exporting Resources as a WSDL File on page 120](#) has been updated to make clear which project resources are used by the WSDL export utility. [Understanding the WSDL to Project Resource Mapping on page 128](#)
- A new section explains how to import domain models from Microsoft Excel and from databases. The section also explains how to export domain models to Excel See [Importing and Exporting Domain Models on page 183](#).
- Support for array types is now available for rule functions. The form editor pop-up dialog for selecting scope variables now has a checkbox, `isArray`. You can specify a variable array in the source editor in the usual way, for example, `int [] myArr`. See [Adding a Rule Function on page 228](#) for a new step in the procedure reflecting this change.
- The section [Guidelines for Use of Cache Related Functions on page 260](#) is updated for changes in the names of certain functions.
- A new section describes the Rule Agenda view in Debugger. See [The Rule Agenda and Variable Views on page 401](#).
- Project Analyzer now includes shared resources (HTTP Connection, Identity Resource, JDBC Connection, JMS Connection, JMS Application Properties, JNDI Configuration, Rendezvous Transport). This is noted in [Project Analyzer and Selected Entity Project Diagrams on page 432](#).
- Keywords and other reserved word list is updated and corrected. See [Keywords and Other Reserved Words on page 280](#).

- Two diagram properties are now documented. See [Diagram Properties on page 424](#).
- Shared resources and Metrics are added to [Table 45, Types of diagrams available for each type of project element, on page 423](#).
- Preferences are now documented in [Chapter 25, TIBCO BusinessEvents Studio Preferences, on page 445](#).

TIBCO BusinessEvents Express

The TIBCO BusinessEvents Express edition provides more limited functionality than the TIBCO BusinessEvents standard edition. Some content in this documentation is not relevant to users of TIBCO BusinessEvents Express. Such content includes but is not limited to any chapters and major sections that contain a note indicating that the content does not apply to TIBCO BusinessEvents Express.

Minor references to unsupported features may not be called out in the text. Use the following general guidelines to understand what is and is not supported in these cases:

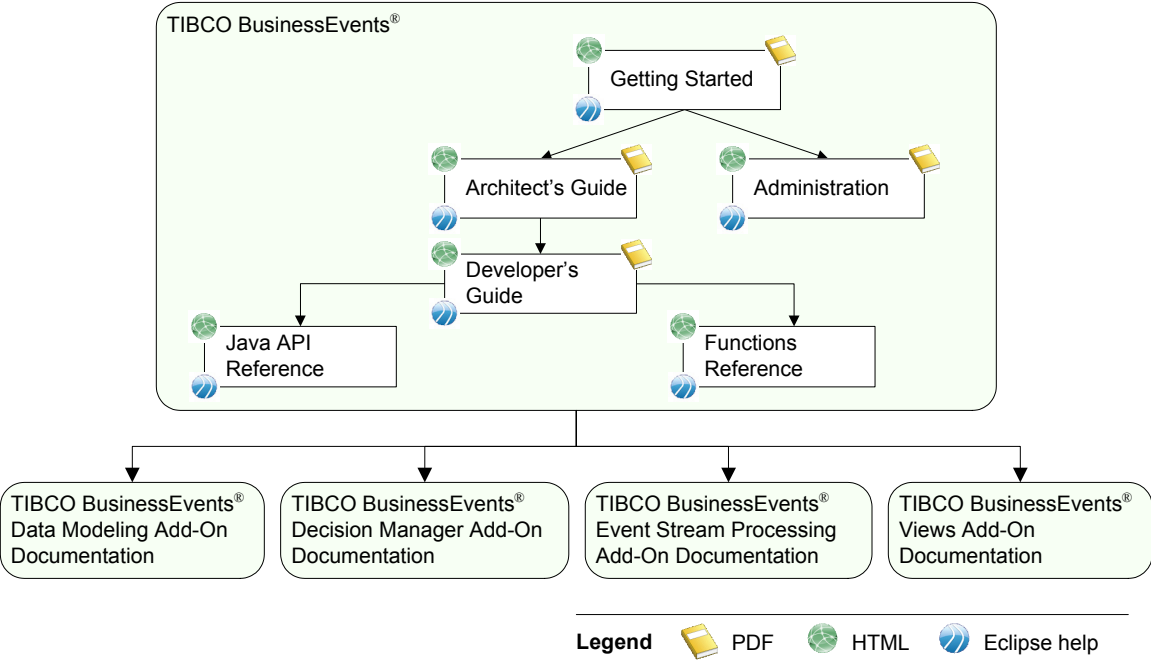
- Only In Memory object management (OM) is supported. Therefore all functionality that requires Cache OM, such as use of a backing store, is not available. Berkeley DB OM is also not supported with the TIBCO BusinessEvents Express edition.
- The Monitoring and Management component and the related Site Topology editor are not supported with the TIBCO BusinessEvents Express edition. Instead you can use TIBCO Administrator and JMX for deployment, monitoring, and management.
- Only the TIBCO BusinessEvents Decision Manager add-on is supported with the TIBCO BusinessEvents Express edition in this release. Other add-on products are not supported.

Related Documentation

This section lists documentation resources you may find useful.

TIBCO BusinessEvents and Add-On Product Documentation

The following diagram shows the main documents in the TIBCO BusinessEvents documentation set, and the documentation sets for the optional add-on products.



Each set also contains an installation guide, release notes, and a readme file.

TIBCO BusinessEvents Documentation

TIBCO BusinessEvents Studio, the design-time UI, is supported on Windows and Linux. The documentation set for TIBCO BusinessEvents is as follows.

- *TIBCO BusinessEvents Installation:* Read this manual for instructions on site preparation, installation, upgrading from an earlier release, and project migration.
- *TIBCO BusinessEvents Getting Started:* After the product is installed, use this manual to learn the basics of TIBCO BusinessEvents: project design, cache

OM, and backing store. This guide explains the main ideas so you gain understanding as well as practical knowledge.

- *TIBCO BusinessEvents Architect's Guide*: If you are architecting an application using TIBCO BusinessEvents, read this guide for overview and detailed technical information to guide your work.
- *TIBCO BusinessEvents Developer's Guide*: Use this guide when you implement a project design in TIBCO BusinessEvents Studio. It covers topics such as project-level tasks, resource-level tasks, debugging, and integration with TIBCO ActiveMatrix BusinessWorks.
- *TIBCO BusinessEvents Administration*: This book explains how to configure, deploy, monitor, and manage a TIBCO BusinessEvents application and the data it generates. It explains how to configure the CDD file for different object management options, set up a backing store, and how to work with TIBCO BusinessEvents Monitoring and Management component, and with TIBCO Administrator.
- Online References:
 - *TIBCO BusinessEvents Java API Reference*: This online reference is available from the HTML documentation interface. It provides the Javadoc-based documentation for the TIBCO BusinessEvents API.
 - *TIBCO BusinessEvents Functions Reference*: This reference is available from the HTML documentation interface. It provides a listing of all functions provided with TIBCO BusinessEvents, showing the same details as the tooltips available in TIBCO BusinessEvents Studio.
- *TIBCO BusinessEvents Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

TIBCO BusinessEvents Event Stream Processing

This TIBCO BusinessEvents add-on is available separately, and includes the TIBCO BusinessEvents Query Language features and the Pattern Matcher Service.

- *TIBCO BusinessEvents Event Stream Processing Installation*: Read this brief manual for installation instructions. A compatible version of TIBCO BusinessEvents must be installed before you install any add-on.
- *TIBCO BusinessEvents Query Developer's Guide*: This manual explains how to use the object query language to query various aspects of the running system. For details on configuring and deploying query agents, see *TIBCO BusinessEvents Administration*.

- *TIBCO BusinessEvents Event Stream Processing Pattern Matcher Developer's Guide*: This manual explains how to use the pattern matcher language and engine to correlate event patterns in a running system.
- *TIBCO BusinessEvents Event Stream Processing Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

TIBCO BusinessEvents Decision Manager

This TIBCO BusinessEvents add-on is available separately. It incorporates the Decision Manager decision modeling business user interface (supported on Windows and Linux), and the Rules Management Server (supported on all platforms supported by TIBCO BusinessEvents).

- *TIBCO BusinessEvents Decision Manager Installation*: Read this brief manual for installation instructions. A compatible version of TIBCO BusinessEvents must be installed before you install any add-on.
- *TIBCO BusinessEvents Decision Manager User's Guide*: This manual explains how business users can use decision tables and other decision artifacts to create business rules. It also covers configuration and administration of Rules Management Server, which is used for authentication, authorization, and approval processes.
- *TIBCO BusinessEvents Decision Manager Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

TIBCO BusinessEvents Data Modeling

This TIBCO BusinessEvents add-on is available separately. It contains state models and database concept features.

- *TIBCO BusinessEvents Data Modeling Installation*: Read this brief manual for installation instructions. A compatible version of TIBCO BusinessEvents must be installed before you install any add-on.
- *TIBCO BusinessEvents Data Modeling Developer's Guide*: This manual explains data modeling add-on features for TIBCO BusinessEvents. The database concepts feature enables you to model TIBCO BusinessEvents concepts on Database tables. The state modeler feature enables you to create state machines.
- *TIBCO BusinessEvents Data Modeling Release Notes*: Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

TIBCO BusinessEvents Views

This TIBCO BusinessEvents add-on is available separately. It includes graphical dashboard components for run-time event monitoring.

- *TIBCO BusinessEvents Views Installation:* Read this manual for instructions on site preparation and installation. A compatible version of TIBCO BusinessEvents must be installed before you install any add-on.
- *TIBCO BusinessEvents Views Developer's Guide:* This guide explains how to use TIBCO BusinessEvents Views to create meaningful metrics that are presented to business users in real-time for proactive decision making.
- *TIBCO BusinessEvents Views User's Guide:* This book explains how to monitor metrics in TIBCO BusinessEvents TIBCO BusinessEvents Views and how to represent the business processes graphically.
- *TIBCO BusinessEvents Views Release Notes:* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Accessing TIBCO BusinessEvents Functions Reference Documentation

Reference documentation for functions, including those used in add-ons, is available in the HTML documentation interface for the TIBCO BusinessEvents documentation set, and as tooltips in TIBCO BusinessEvents Studio. To use the HTML-based functions reference from the file system do the following:

1. Browse to `BE_HOME/doc/standard/html` and click `index.htm`. The HTML documentation interface appears.
2. In the left panel, browse to Online References and in the right panel choose TIBCO BusinessEvents Functions Reference. The reference opens in a new tab.
3. Click the navigation links to browse to the functions as desired.

Other TIBCO Product Documentation

You may find it useful to refer to the documentation for the following TIBCO products:

- TIBCO ActiveSpaces[®]
- TIBCO Enterprise Message Service[™]
- TIBCO Hawk[®]
- TIBCO ActiveMatrix BusinessWorks[™]
- TIBCO Rendezvous[®]

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
<i>TIBCO_HOME</i> <i>ENV_HOME</i> <i>BE_HOME</i>	<p>Many TIBCO products must be installed within the same home directory. This directory is referenced in documentation as <i>TIBCO_HOME</i>. The value of <i>TIBCO_HOME</i> depends on the operating system. For example, on Windows systems, the default value is <code>C:\tibco</code>.</p> <p>Other TIBCO products are installed into an <i>installation environment</i>. Incompatible products and multiple instances of the same product are installed into different installation environments.</p> <p>An environment home directory is referenced in documentation as <i>ENV_HOME</i>. The value of <i>ENV_HOME</i> depends on the operating system. For example, on Windows systems the default value is <code>C:\tibco</code>.</p> <p>TIBCO BusinessEvents installs into a directory within an <i>ENV_HOME</i>. This directory is referenced in documentation as <i>BE_HOME</i>. The value of <i>BE_HOME</i> depends on the operating system. For example on Windows systems, the default value is <code>C:\tibco\be\5.0</code>.</p>
<code>code font</code>	<p>Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example:</p> <p>Use <code>MyCommand</code> to start the foo process.</p>
<code>bold code font</code>	<p>Bold code font is used in the following ways:</p> <ul style="list-style-type: none">• In procedures, to indicate what a user types. For example: Type <code>admin</code>.• In large code samples, to indicate the parts of the sample that are of particular interest.• In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, <code>MyCommand</code> is enabled: <code>MyCommand [enable disable]</code>

Table 1 General Typographical Conventions (Cont'd)




Convention	Use
<i>italic font</i>	<p>Italic font is used in the following ways:</p> <ul style="list-style-type: none"> To indicate a document title. For example: See <i>TIBCO BusinessWorks Concepts</i>. To introduce new terms. For example: A portal page may contain several <i>portlets</i>. Portlets are mini-applications that run in a portal. To indicate a variable in a command or code syntax that you must replace. For example: <code>MyCommand <i>pathname</i></code>
Key combinations	<p>Key name separated by a plus sign indicate keys pressed simultaneously. For example: <code>Ctrl+C</code>.</p> <p>Key names separated by a comma and space indicate keys pressed one after the other. For example: <code>Esc, Ctrl+Q</code>.</p>
	The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.
	The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.
	The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.

Table 2 Syntax Typographical Conventions

Convention	Use
[]	<p>An optional item in a command or code syntax.</p> <p>For example:</p> <pre>MyCommand [optional_parameter] required_parameter</pre>
	<p>A logical OR that separates multiple items of which only one may be chosen.</p> <p>For example, you can select only one of the following parameters:</p> <pre>MyCommand param1 param2 param3</pre>

Table 2 *Syntax Typographical Conventions*

Convention	Use
{ }	<p>A logical group of items in a command. Other syntax notations may appear within each logical group.</p> <p>For example, the following command requires two parameters, which can be either the pair <code>param1</code> and <code>param2</code>, or the pair <code>param3</code> and <code>param4</code>.</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command requires two parameters. The first parameter can be either <code>param1</code> or <code>param2</code> and the second can be either <code>param3</code> or <code>param4</code>:</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command can accept either two or three parameters. The first parameter must be <code>param1</code>. You can optionally include <code>param2</code> as the second parameter. And the last parameter is either <code>param3</code> or <code>param4</code>.</p> <pre>MyCommand param1 [param2] {param3 param4}</pre>

Connecting with TIBCO Resources

This section provides links to helpful TIBCO resources.

How to Join TIBCOCommunity

TIBCOCommunity is an online destination for TIBCO customers, partners, and resident experts, a place to share and access the collective experience of the TIBCO community. TIBCOCommunity offers forums, blogs, and access to a variety of resources. To register, go to <http://www.tibcommunity.com>.

How to Access All TIBCO Documentation

After you join TIBCOCommunity, you can access the documentation for all supported product versions here:

<http://docs.tibco.com/TibcoDoc>

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, please contact TIBCO Support as follows.

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:
<http://www.tibco.com/services/support>
- If you already have a valid maintenance or support contract, visit this site:
<https://support.tibco.com>

Entry to this site requires a user name and password. If you do not have a user name, you can request one.

Introduction to Project Development

TIBCO BusinessEvents Developer's Guide explains how to build a TIBCO BusinessEvents project using the TIBCO BusinessEvents core project resources such as channels, events, concepts, rules, and so on. This brief introductory chapter places this guide in the context of other documentation so you can understand its role in the overall documentation set.

TIBCO BusinessEvents Studio is an Eclipse-based UI used to design, build, and maintain TIBCO BusinessEvents projects. It is integrated into the standard Eclipse menus where appropriate, and works with many established Eclipse UI methodologies and plug-ins.

A TIBCO BusinessEvents Studio project contains resources used to build, test, and view a design-time TIBCO BusinessEvents project.

Before you begin to use *TIBCO BusinessEvents Developer's Guide*, gain a basic familiarity with the product by completing the tutorials in *TIBCO BusinessEvents Getting Started*.

TIBCO BusinessEvents Developer's Guide provides practical details on using TIBCO BusinessEvents Studio to build a project. Each chapter focuses on a different aspect of developing a project.

For in-depth explanations about designing a TIBCO BusinessEvents project and other topics, read *TIBCO BusinessEvents Architect's Guide*. References to relevant topics in that guide are provided in *TIBCO BusinessEvents Developer's Guide*.

When it is time to configure your application for deployment, deploy, and manage it, refer to *TIBCO BusinessEvents Administration*.

Chapter 2 **Project Tasks**

This chapter covers various project-level tasks for TIBCO BusinessEvents Studio. For information about analyzing or viewing projects, see [Chapter 24, Diagrams, on page 421](#).

Topics

- [Creating a Project, page 4](#)
- [Finding a Project Element, page 6](#)
- [Exporting \(Generating\) Concept and Event Schema \(XSD\) Files, page 7](#)
- [Validating a Project or Project Resource, page 9](#)
- [Working with External Library and Custom Function Paths, page 10](#)
- [Working with Project Libraries, page 12](#)
- [Working with Global Variables, page 17](#)
- [Storing Trusted Certificates Outside of Your Project, page 22](#)
- [Building an Enterprise Archive \(EAR File\), page 24](#)

Creating a Project

This section explains the basic procedure for creating a new project. See *TIBCO BusinessEvents Getting Started* for tutorials on building and populating a basic TIBCO BusinessEvents project.



To import a TIBCO BusinessEvents Studio project into your workspace, select File > Import > General > Existing projects into Workspace. Do this for current release projects only.

To import a project from a prior release, see Chapter 4, Migrating Projects from Earlier Versions in *TIBCO BusinessEvents Installation*.

To Create a Project

1. Start TIBCO BusinessEvents Studio. In Windows, click Start > All Programs > TIBCO > *YourEnvironment* > TIBCO BusinessEvents 5.0 > TIBCO BusinessEvents Studio.
2. If you are prompted, select or create the Eclipse workspace directory where your project files will be stored. (If you check the option to use this workspace as a default, you are not prompted again.)
3. You may be prompted to upload usage data collected by the Eclipse Usage Data Collector. Select an option as desired. (This is an Eclipse activity, and not part of using TIBCO BusinessEvents Studio.)
4. You see the TIBCO BusinessEvents Studio UI. From the File menu select **New > Project**. You see the New Project — Select a Wizard dialog.
5. Select **TIBCO BusinessEvents > Studio Project** and click **Next**.
6. In the New Studio Project field, enter a project name.
Names follow Java variable naming restrictions. Do not use any reserved words. See [Identifiers \(Names\) on page 275](#).
7. Accept the default location or uncheck the Use default location checkbox and specify the desired location.
8. Click **Finish**.

A new folder tree appears in TIBCO BusinessEvents Studio Explorer with a default set of folders you can use as desired. (The `defaultVars` folder, however, is required.)



Source Control If the project is under source control using Perforce, editing a resource automatically checks out the resource and makes it writable.

If the file is not under Perforce, but is read only, you are prompted to make the file writable when a change is made.

Finding a Project Element

You can find a project element in various ways and then take some action relating to that element such as opening its editor, or creating a dependency diagram. Depending on the size and complexity of the project, you may find some of the below methods of finding elements more convenient than others.

Note that an element can be a part of a resource (such as a concept property):

- Expand the project tree in TIBCO BusinessEvents Studio Explorer and double-click the element name.
- Select **Navigate > Open Studio Element** and select an element from the alphabetical list of elements. You can search using wild cards.
- Select the **Resource** perspective, then select **Navigate > Open Resource** (or press **Ctrl+Shift+R**). A dialog allowing you to search through all resources in your workspace appears. To display the whole list, enter ******. You can also use the asterisk (*****) as a wild card. For example to display all concept resources, enter ***.concept**.
- Open the **Project View** diagram and navigate to the element in the diagram. You can search using filters. Double click the element on the diagram to open it. See [Chapter 24, Diagrams, on page 421](#).

Exporting (Generating) Concept and Event Schema (XSD) Files

The Generate Schema utility lets you export concepts and events to XML Schema Definition (XSD) files, one per entity, in a specified location. You can generate schema for all concepts, all events, or both. (You can't generate schema for one selected entity type.) The files use the same folder structure as the project from which they are exported. In addition, `_BaseConcept.xsd` and `_BaseEvent.xsd` are generated in the root of the selected directory.

XML schemas are used for interoperability between TIBCO BusinessEvents and third party tools or SOA platforms that use well-defined XML for message communication, transformation, and validation.

In the XSD files, concepts are represented as follows:

- Each concept is exported to its own complex type using its own namespace.
- Referenced concepts have a corresponding `ref` attribute in the parent's complex type.
- Contained concepts have a corresponding `type` attribute in the parent's complex type.

To Generate an XML Schema (XSD File)

1. Select the project whose schema you want to generate.
2. From the File menu select **Export**. In the Export wizard, expand **TIBCO BusinessEvents** and select **Generate Schema**.

You can also reach this utility from the option (right-click) menu anywhere in Studio Explorer.

3. In the Schemas Folder field, browse to and select the folder where you want to put the schema files.
4. Select the **Override TIBCO BusinessEvents Namespace** checkbox to specify a different namespace.

If you do not select a different namespace, an informational message displays. Click **Yes** to continue or **No** to return to the dialog and provide a namespace.

Provide a different namespace to avoid conflicts with the source concepts and events. If you do not provide a namespace, the default TIBCO BusinessEvents namespace is used. (TIBCO BusinessEvents events and concepts have a hidden schema.) If the source entities and generated schema files are in the same folders, use of the default TIBCO BusinessEvents namespace results in a namespace conflict. In this case, you must provide a namespace.

5. In the Select Resources panel, select Concepts or Events or both. Schemas for all concepts or all events in the project (or both) are generated accordingly.
6. Click **Finish**.

The XSD files for the selected resources (all concepts, all events, or both) are generated in subdirectories of the selected directory. Subdirectory names match the project folders. The `_BaseConcept.xsd` and `_BaseEvent.xsd` files are generated in the root of the selected directory.

Validating a Project or Project Resource

When you save a resource, TIBCO BusinessEvents performs validation checks to ensure that all resource requirements are met. For example, it checks that required fields are completed, names are valid, syntax in rules is correct and no unknown functions are called.

You can also perform validation explicitly for an entire project or selected project resource

To Validate a Project, Project Folder, or Project Resource

In TIBCO BusinessEvents Studio Explorer, do one of the following:

- Right click a project name, folder name, or a project resource name, and select **Validate**.
- Select a project name, folder name, or a project resource name and select **Project > Validate**.

A pop-up message displays the message, "Validation was successful" or summary information about any problems. Details about problems display in the Problems view.

To Fix Validation Errors

Many validation issues can be fixed using the Quick Fix feature. In the Problems view, right click on a problem and select Quick Fix.

See also [Using the Quick Fix Feature in the Rule Editor on page 239](#) for a related use of this feature.

Project Validation and Project Analysis

In addition to validating for internal consistency, you can run a project analyzer feature. It helps you to understand your project better, and find ways to improve it. For example, it identifies rule functions that are not used in any rule, and rules that will never be triggered at runtime. See [Project Analyzer and Selected Entity Project Diagrams on page 432](#).

Working with External Library and Custom Function Paths

When you work with external libraries or custom functions in your project at design time, and when you run, test, or debug such projects in TIBCO BusinessEvents Studio, you must ensure that the engine can find all libraries (including dependencies on third-party libraries) and custom functions, if the project requires any.

If a project uses Rendezvous channels, for example, Java allows the user to set a property, `java.library.path`, to ensure that the environment path contains a reference to the directory containing Rendezvous DLLs (or on UNIX the SO files).

Similarly TIBCO BusinessEvents captures the native library path along with the build path information to pass to engines running inside TIBCO BusinessEvents Studio. You can enter this information as described below, or when you are configuring a run configuration or debug configuration.



Test Connection Feature To make the Test Connection feature work for JMS Connection and JDBC Shared Connection shared resources, see [Enabling the Test Connection Feature on page 190](#).

To Add an External Library or Custom Function Path



This functionality is also available in the Debug Configurations > ClassPath tab, and the Run Configurations > ClassPath tab, for your convenience. See [Chapter 23, Testing and Debugging Projects, on page 399](#).

1. Open the project in TIBCO BusinessEvents Studio.
2. In TIBCO BusinessEvents Studio Explorer, right-click the project and click **Properties** (or press Alt-Enter). You see the properties dialog for the project.
3. In the left panel, select **Build Path** and then select one of the following:
 - The **Custom Functions** tab, to add custom functions paths
 - The **Java Libraries** tab, to add third-party libraries paths
4. Click **Add Library** and browse to and select the JAR file.

5. If a library or custom function depends on additional third-party Java archives, do the following:
 - a. Expand the library or custom function entry and double-click the **Native library location** entry.
 - b. At the Native Library Folder Configuration dialog, select **External Folder** or **Workspace** as appropriate, and browse to and select the folder containing the native libraries.
 - c. Click **OK** to dismiss this dialog.
6. Click **OK**. One of the following occurs:
 - If a custom function depends on third-party Java archives and you did not do [step 5](#), a dialog displays when you click OK. It lists the name of the class or classes that could not be loaded. To resolve the problem, do [step 5](#).
 - You see the message: "It is recommended that you rebuild your project after changing the build path. Would you like to rebuild now?" Click Yes to validate and fix any references already in code to the JARs you have added.
7. Save the resource.



Deployment Action Required To make custom functions or external libraries available at runtime, do the following on all machines where TIBCO BusinessEvents is installed:

- Either copy the JAR or JARs to the `lib/ext/tpcl` directory, or other directory in the classpath; or update the classpath in the TRA file to point to the location of the JAR or JARs.
- If a JAR has dependencies on native libraries, edit `BE_HOME/bin/be-engine.tra` and as needed, update `PATH_LD_LIBRARY_PATH`, `SHLIB_PATH`, and `LIBPATH` as needed, depending on the operating system.

To Remove an External Library or Custom Functions JAR Path

Follow the instructions in [To Add an External Library or Custom Function Path](#), but instead of clicking Add Library, select the JAR file you want to remove and click Remove Library. The entry for any native libraries it depends on is also removed.

Working with Project Libraries

This section explains how to work with project libraries in TIBCO BusinessEvents Studio. You can also work with project libraries at the command line (see [Working With Project Libraries at the Command Line on page 34](#) for details).

Project libraries (design-time libraries) are archives that enable you to create project resources once, and share them with other projects.

Project libraries can contain any resources from a TIBCO BusinessEvents Studio project. For example, a project library might contain some concepts that are standard across projects (such as Person), so that multiple projects do not need to redefine these concepts.

Project libraries have the file extension `.projlib`. TIBCO BusinessEvents Studio includes the following features to allow the refactoring of project resources into project libraries.



Avoid Conflicting Elements Ensure that the project library does not use elements already in use in the local project. If elements in an imported project library conflict with the local project elements, those in the local project have priority. The project library element is ignored in this case for build purposes and is not included in the EAR file.

If multiple project libraries have the same element, then the first one on the build path wins, and the other ones are ignored.

Creating (Exporting) a Project Library

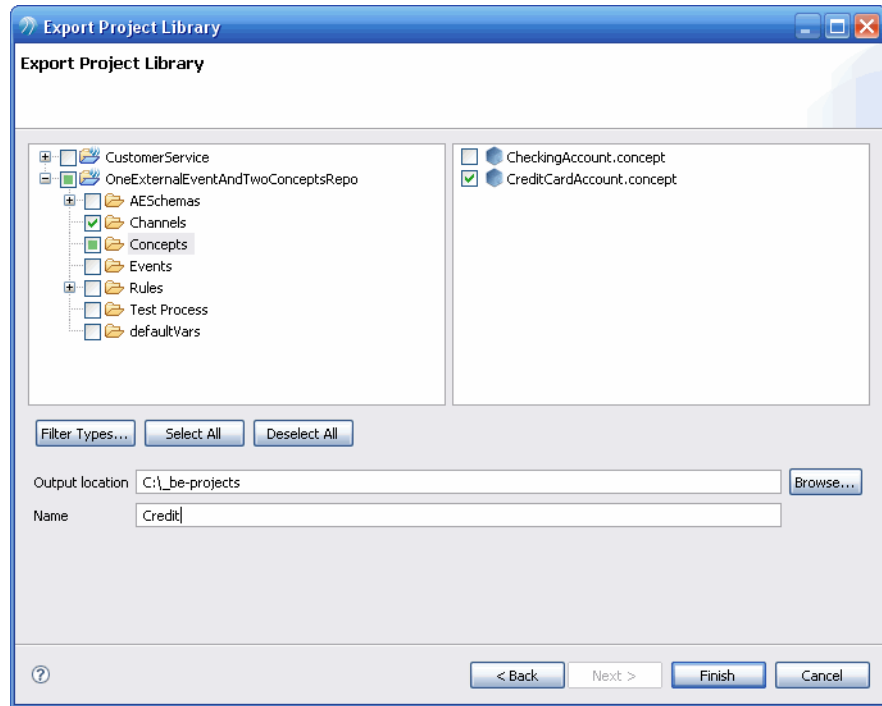
To create a project library you export selected resources to an archive.

To Create a Project Library

1. Optionally, select a project or any of its resources in TIBCO BusinessEvents Studio.
2. Do one of the following:
 - From the File menu select **Export**.
 - Right-click anywhere in Studio Explorer and select **Export**.

In the Select dialog, select **TIBCO BusinessEvents > Project Library** and click **Next**. You see the Export Studio Project Library dialog.

3. In the Export Studio Project Library dialog, select the resources you want to export. (If you selected resources in Studio Explorer they are selected automatically.)



The left panel displays folders. The right panel displays resources. Select what to export in any of the following ways:

- Select a folder to select all resources within it.
 - Highlight a folder to display its resources in the right panel. Select individual resources as desired.
 - To reduce the set of selected resources, first select resources then click the **Filter Types** button. Select one or more resource types, then click **OK**. The Export Studio Project Library dialog selection is now reduced to only resources of the selected type or types.
4. Enter an output location and give the project library a name, then click **Finish**.



Remember to include the library you just exported, to avoid validation issues.

Adding and Removing Project Libraries in a Project

Adding a project library does not physically import the files. Instead a pointer to the files is maintained in the project and the resources appear in TIBCO BusinessEvents Studio Explorer as if they are in the project. The physical location can be in the project folders or external to them, as long as they are available to the project at design time.

You can add a project library to a project in either of these two ways:

- By including a project library
- By importing a project library

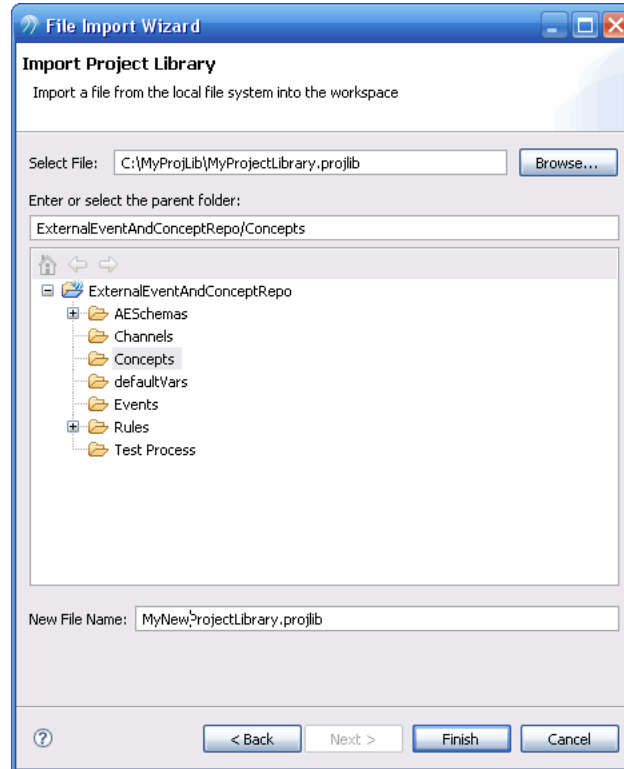
To Include a Project Library (Add it to the Build Path)

1. Open the project in TIBCO BusinessEvents Studio.
2. In TIBCO BusinessEvents Studio Explorer, right-click the project name and click **Properties** (or press Alt-Enter, or select Project > Properties). You see the properties dialog for the project.
3. In the left panel, select **Build Path** and then select the **Project Libraries** tab.
4. Click **Add Library** and browse to and select the desired project library (`.projlib`) file.
5. Click **OK** and then save the resource. The project library appears at the root of the project tree as a Project Dependencies node.

To Import a Project Library

1. Do one of the following:
 - From the File menu select **Import**.
 - Right-click anywhere in TIBCO BusinessEvents Studio Explorer and select **Import**.

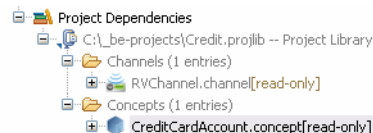
2. In the Import wizard Select dialog, select **TIBCO BusinessEvents > Project Library** and click **Next**. You see the Import Project Library dialog.



3. In the Import Project Library dialog Select File field, browse to and select the project library (.projlib) file.
4. Select a project from the project tree or type the name in the field above the project tree area. You can only import at the root of a project.

Ignore the New File Name field. It is not used by TIBCO BusinessEvents.

5. Click **Finish**. The project library appears at the root of the project tree as a Project Dependencies node.



To Remove a Project Library

To remove a project library, follow the instructions in [To Include a Project Library \(Add it to the Build Path\)](#), page 14, but instead of clicking Add Library, select the project library you want to remove and click Remove Library.

Working with Global Variables

Global variables provide an easy way to set defaults for use throughout your project. When the project is deployed all occurrences of the global variable name are replaced with the provided global variable value, or a deploy-time override.

For example, you could assign the value 7474 to the global variable `RvDaemon`. You can then use the variable in a Rendezvous Transport resource. At deploy time you can override the default value as needed.



- The datatype of the global variable must match the datatype accepted in the field where you use it. If the global variable is of a different type, runtime errors result.
- An exception to the above allows flexibility in numeric fields: global variables used in numeric fields can be of any type, as long as the substituted value of the field is numeric.



- A project folder called `defaultVars` is available but not exposed in Studio Explorer, so that you can share the global variables using source control software. It is not used for other purposes.


Adding and Managing Global Variables

In a new Studio project no global variables are predefined. When you import a project from an earlier version of TIBCO BusinessEvents, however, you see predefined global variables such as the following (as well as any others defined in the project).

◆ Global Variables: FDNew

Global Variables and Groups	Configuration
Define the global variables and groups that will be used in this project	
<div> <div>ABC Deployment</div> <div>ABC DirLedge</div> <div>ABC DirTrace</div> <div>ABC Domain</div> <div>ABC HawkEnabled</div> <div>ABC JmsProviderUrl</div> <div>ABC JmsSslProviderUrl</div> <div>ABC RemoteRvDaemon</div> <div>ABC RvDaemon</div> <div>ABC RvNetwork</div> <div>ABC RvService</div> <div>ABC RvaHost</div> <div>ABC RvaPort</div> <div>ABC TIBHawkDaemon</div> <div>ABC TIBHawkNetwork</div> <div>ABC TIBHawkService</div> <div>ABC New_Variable_1</div> </div> <div> <div>Add Variable</div> <div>Add Group</div> <div>Remove</div> </div>	<div> <div>Name:</div> <div>New_Variable_1</div> </div> <div> <div>Value:</div> <div></div> </div> <div> <div>Type:</div> <div>String</div> </div> <div> <div>Deployment settable:</div> <div><input type="checkbox"/></div> </div> <div> <div>Service settable:</div> <div><input type="checkbox"/></div> </div> <div> <div>Description:</div> <div>NEW VAR</div> </div> <div> <div>Constraint:</div> <div></div> </div> <div> <div>Last Modified:</div> <div>2010/01/15 17:35:16</div> </div>

To Add and Manage Global Variables

1. Open the project in TIBCO BusinessEvents Studio.
2. Select **Project > Edit Global Variables**.
You see the Global Variables view listing the variables available, if any.
3. Do one of the following:
 - Right click inside the Global Variables view and click Edit Global Variables.
 - In the Global Variables view toolbar, click the Edit Global Variables  button.
4. Do any of the following. (See [Table 3, Global Variable Reference, on page 18](#) for a guide to the fields):
 - To add a variable, click **Add Variable** and complete the fields.
 - To edit a variable, select the variable and update the fields.
 - To add a variable group, click **Add Group**.
 - To add a variable to a group, first select a group, then click **Add Variable**.
 - To remove a variable or a group, highlight it and click **Remove**.



Using Groups Groups are used for organizing variables. Variable groups are especially useful if multiple developers share a project using a version control system. When referencing a variable that is in a group, use the complete path, for example `%%mygroup/mysubgroup/myvariable%%`.

You must add at least one variable to a group, or the group will not be saved. If you delete all global variables in a global variable group, the group itself is also automatically deleted.

5. Save the resource. Groups and references to the `defaultVars.substvar` file appear in the `defaultVars` project folder.

Table 3 Global Variable Reference

Field	Description
Name	The variable name.
Value	The variable value. Varies according to type.
Type	One of String, Integer, Boolean, Long, Password. The type must match the type of the field where the global variable is used, or errors result.

Table 3 Global Variable Reference

Field	Description
Deployment settable	<p>For use when deploying with TIBCO Administrator.</p> <p>If checked, the variable is visible and settable when deploying using TIBCO Administrator. The values set at that time are saved in the project that TIBCO Administrator creates from the provided EAR file.</p> <p>If the checkbox is not checked, the variable is not visible in TIBCO Administrator. It is checked by default.</p>
Service Settable	<p>For deployment using TIBCO Administrator. If this box is checked the value of the global variable can be set differently for each deployable instance.</p> <p>Used only when Deployment Settable is also checked (ignored otherwise).</p> <p>If checked, the variable is included when the Include all service level global variables option is selected when building the enterprise archive file.</p> <p>Values set at the service or service instance level are passed to the engine at runtime in the format <code>tibco.clientVar.VariableName=value</code>.</p> <p>Not checked by default.</p>
Description	A helpful description, as needed.
Constraint	<p>Optional. For String and Integer types, allows you to provide a range of allowed values. The constraint field for Strings is an enumeration, for example, <code>one, two, three</code>. The constraint field for Integers is for a range, for example, <code>1-100</code>.</p> <p>Note that constraints are currently not implemented in TIBCO Administrator.</p>
Last Modified	Non-editable field records date and time this variable was last modified.

Using Global Variables

To Use Global Variables in TIBCO BusinessEvents Studio Project Fields

To use a global variable as the value for a project setting, drag it from the list of variables into the text box for the setting, or enter it manually using the syntax `%% Variable_Group/Variable_Name%%`. You must include the global variable group hierarchy, if one exists. For example, to use a global variable in a File Path field, you might enter `%%myvars/myFileLoc%%`.

To Use Global Variables in the Rule Editor

To use a global variable in the rule editor, use one of the `System.getGlobalVariableAs*` functions. For example:

```
System.getGlobalVariableAsString("myvars/Hostname", "Localhost")
```

Where `myvars/Hostname` is the name of the variable group and variable, and `Localhost` is an optional literal value to use if the variable is not found.

Overriding Global Variables at Deploy Time

You can override default global variable values in one of these ways, depending on how you will deploy:

- For starting at the command line:
 - In the design time CDD file. See *Overriding Global Variables in the CDD File (for Command Line Startup)* in *TIBCO BusinessEvents Administration*.
 - When starting at the command line, using the `--propVar` option, or using the `--propFile` option to specify a property file where the override properties are defined. See *Starting a TIBCO BusinessEvents Engine at the Command Line* in *TIBCO BusinessEvents Administration*.
- For deployment using TIBCO BusinessEvents Monitoring and Management component, set overrides in the MM Console. See *Overriding Global Variables at Deploy Time* in *TIBCO BusinessEvents Administration*.
- For deployment using TIBCO Administrator, set overrides in the TIBCO Administrator UI. See *Deploying a Project in a TIBCO Administrator Domain* in *TIBCO BusinessEvents Administration*.

To Override Global Variables when Deploying at the Command Line

You can also override variable values when starting an engine directly at the command line. You can pass the overrides directly, using the `--propVar` option, or in a file, referenced using the `-p` option.

To Override Global Variables when Deploying with TIBCO Administrator

See Deploying a Project in a TIBCO Administrator Domain in *TIBCO BusinessEvents Administration* for the complete procedure.

You can override variables at the deployment level or at the service level. Values set at the deployment level are used in all deployed engines. Values set at the service level are used for the specific engine you are deploying.

Service settable global variables are only available if the "Include All Service Level global variables" checkbox in the Build Enterprise Archive dialog is checked (see [Building an Enterprise Archive \(EAR File\) on page 24](#)).

Storing Trusted Certificates Outside of Your Project

Trusted certificates are used when you configure SSL, such as in an [HTTP Connection](#) or [JMS Connection](#).

Trusted certificates can be used to ensure that remote servers are who they claim to be and to ensure that TIBCO BusinessEvents can identify itself as a valid client when connecting to a server.

You can store the certificates within a project folder, or you can use a special global variable, `BE_GLOBAL_TRUSTED_CA_STORE`, to specify the location of an external directory that contains all the certificates known to TIBCO BusinessEvents.

When you store the certificates within a project folder, then when a certificate changes or expires, you must import any new certificates or certificate chains into the project, rebuild the EAR file, and re-deploy your project.

Using the global variable, however, avoids this problem. When you use the global variable to specify the external location of certificates, then when certificates change or expire, you just replace certificates or add new certificates and then restart the engine to load the changes.

You can set the global variable value and then use the variable in the usual ways, as described in this chapter. For example you could use the global variable in the CDD at the PU level as follows:

```
tibco.clientVar.BE_GLOBAL_TRUSTED_CA_STORE=file:///somepath/myGTCAFolder
```

To Store Trusted Certificates Outside of the Project

1. Create a directory where you want to store the trusted certificates. You must copy this directory to each machine where engines are deployed. Alternatively, the location can be a shared network area accessible by all process engines.
2. Create a global variable named `BE_GLOBAL_TRUSTED_CA_STORE`. See [Working with Global Variables on page 17](#) for more information.
3. Set the value of `BE_GLOBAL_TRUSTED_CA_STORE` to the location of the trusted certificates folder on your file system. The value must be a file URL, for example, `file:///c:/tibco/certs`.

The location can be the same for all deployed engines (that is, you copied it to the same location on each machine or it is a shared network drive).

Alternatively you can change the value of the global variable as needed when you deploy the project

4. Specify a value in the Trusted Certificates field in the SSL Configuration dialog. When the project runs, the value of BE_GLOBAL_CA_STORE is used, and not the value you specify in the Trusted Certificates field.
5. Save the resource.

Building an Enterprise Archive (EAR File)

You can build an enterprise archive file using a TIBCO BusinessEvents Studio dialog, and also using a command-line utility (see [Building an Enterprise Archive \(EAR File\) at the Command Line on page 28](#) for details).



Do not store the EAR file in a project folder Doing so includes the prior EAR file when you build the EAR file again, needlessly increasing the size.

Deployment Options

For deployment options see Chapter 8, Deploying and Managing Engines with MM and Chapter 10, Deployment Using TIBCO Administrator and at the Command Line in *TIBCO BusinessEvents Administration*. For testing purposes, you may find it useful to refer to the section Starting a TIBCO BusinessEvents Engine at the Command Line in that guide.

Building an EAR File in TIBCO BusinessEvents Studio

When testing project before development, you may want to build the EAR as explained here, and run the project at the command line.

1. In Studio Explorer, highlight the project name, then from the top menus select **Project > Build Enterprise Archive**.

If you see a message asking you to save all project resources, click Yes. It means an unsaved resource editor is open.
2. At the Build Enterprise Archive dialog, complete values according to guidelines provided in [Table 4, Build Enterprise Archive Reference](#).
3. Click **Apply** to save the configuration details.
(To revert to the version already saved, click Revert.)
4. Click **OK** to build the archive.

Table 4 Build Enterprise Archive Reference

Name	Name of this EAR configuration. (Not the EAR filename.) Default value is the project name

Table 4 Build Enterprise Archive Reference

Author	<p>Person responsible for the EAR file.</p> <p>Default value is the currently logged-on user name.</p>
Description	Optional description.
Archive Version	Increments on each build of the EAR.
Generate Debug Info	<p>Check this checkbox if you want to use the debugger. (See Chapter 23, Testing and Debugging Projects, on page 399.)</p> <p>Default setting is checked.</p>
Include all service level global variables	Check to include service level global variables.
File Location	Browse to the directory in which you want to store the EAR and enter an EAR filename.
Delete Temporary Files	<p>Before TIBCO BusinessEvents packages the EAR file, it generates the Java code in a temporary directory. After the files are packaged in the EAR file, then the temporary files and directory are deleted.</p> <p>You can choose to keep the generated Java files, for example to troubleshoot some problem with an EAR file. To do so, uncheck the Delete Temporary Files checkbox, and specify where to store the Java files in the Compilation Directory field.</p> <p>Default setting is checked, meaning that temporary files are not saved.</p>
Compilation Directory	If you uncheck the Delete Temporary Files checkbox, specify the directory where you want to save the Java files generated during the process of building the EAR file.

TIBCO BusinessEvents Studio Tools is a command-line utility with various operations (tools) you can use to automate common procedures. Some tools pertain only to the TIBCO BusinessEvents Decision Manager add-on and are documented in TIBCO BusinessEvents Decision Manager documentation. Others are also documented in other books in the TIBCO BusinessEvents documentation set, such as in the migration chapter of *TIBCO BusinessEvents Installation*.

However the complete set of tools is also documented here for your convenience and reference.



All arguments of a command-line utility must use only ASCII characters.

Topics

- [Building an Enterprise Archive \(EAR File\) at the Command Line, page 28](#)
- [Importing a TIBCO BusinessEvents \(4.x\) Studio Project at the Command Line, page 29](#)
- [Importing a TIBCO BusinessEvents 3.x Project at the Command Line, page 32](#)
- [Working With Project Libraries at the Command Line, page 34](#)
- [Migrating Core Coherence Functions at the Command Line, page 37](#)
- [Importing an Excel File into TIBCO BusinessEvents Decision Manager at the Command Line, page 38](#)
- [Validating a Decision Table at the Command Line, page 40](#)
- [Generating Decision Table Class Files at the Command Line, page 41](#)
- [Committing Project Artifacts to RMS at the Command Line, page 43](#)
- [Setting up RMS Projects at the Command Line, page 44](#)

Building an Enterprise Archive (EAR File) at the Command Line

The `buildear` operation is useful for automation purposes, for example, in testing environments. For full details on deployment, see *TIBCO BusinessEvents Administration*.

To Build an EAR File at the Command Line

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -op buildEar [-h] [-x] [-o outputEarFile>] -p studioProjectDir
```

For example:

```
studio-tools -op buildEar -o c:\FD.ear -p D:\Workspace\FraudDetection
```

[Table 5, TIBCO BusinessEvents Studio Tools Options for Building an EAR File](#), provides detailed information about the options.

Table 5 TIBCO BusinessEvents Studio Tools Options for Building an EAR File

Option	Description
-op buildEar	Specifies the <code>buildear</code> operation for building EAR files.
-h	Optional. Displays help.
-x	Optional. Overwrites the specified output file if it exists.
-o	Optional. Specifies the filename for the output EAR file. If not specified the EAR file is the same as the final (leaf) directory name in the <i>projectDir</i> path.
-p	Absolute path to the TIBCO BusinessEvents Studio project directory. The EAR file is built using this project.

Importing a TIBCO BusinessEvents (4.x) Studio Project at the Command Line

This option is equivalent to the following menu option in TIBCO BusinessEvents Studio: File > Import > TIBCO BusinessEvents > Existing TIBCO BusinessEvents 4.0 Project. It is used to migrate 4.x projects to 5.0. When you run this command-line utility, various migration actions are done, as explained in *Migration from 4.x to 5.0 — Automatic Migration Changes in TIBCO BusinessEvents Installation*.

Before you can use a 4.x project imported at the command-line in TIBCO BusinessEvents Studio you must do another procedure, explained in [To Open a Project Imported at the Command Line in TIBCO BusinessEvents Studio on page 30](#).

To Import an Existing 4.x Project at the Command Line

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -op importExistingProject [-h] -p studioProjDir [-o targetProjDir] [-c CDDprojectPath] [-u PUNameFromCDD]
```

For example:

```
studio-tools -op importExistingProject -p C:\FT\SomeProj -o c:\MyWorkspace\SomeProj -c COM.cdd -u Invproc
```

If HTTP channel properties are migrated (from a specified CDD and processing unit to all HTTP channel resources' Advanced tab) you see a message like this:

```
Migrating HTTP properties of Processing Unit "PUName" from CDD "CDDprojectPath" to HTTPChannel(s) present in the project
```

When the import has completed successfully, you see a message in the command window like the following:

```
The existing 4.0 TIBCO BusinessEvents project has been successfully imported to c:\MyWorkspace\SomeProj.
```

Table 6, TIBCO BusinessEvents Studio Tools Options for Importing an Existing Project, provides detailed information about the options.

Table 6 TIBCO BusinessEvents Studio Tools Options for Importing an Existing Project

Option	Description
-op importExistingProject	Specifies the <code>importExistingProject</code> operation for importing a TIBCO BusinessEvents Studio project into the workspace.
-h	Optional. Displays help.
-p	Source project: absolute path to the project directory of the TIBCO BusinessEvents Studio project to be imported.
-o	<p>Optional. Absolute path to the target project directory, where the project is imported to.</p> <p>If you specify the source project directory name as the last element in the path, it is used as the target project directory. If you specify a different directory as the last element in the path, the directory is created if it does not exist, and the source project directory is imported within the specified target directory.</p> <p>If you do not specify a target project directory, the original project contents are updated. If the project to be imported is a TIBCO BusinessEvents version 4 project, it is no longer compatible with version 4 after the import.</p> <p>If the target location points to an existing project, the import does not proceed and this message displays:</p> <p>The specified target location already exists and cannot be used.</p>
-c	Optional. The CDD to use for migration actions. Project path of the CDD (path relative to the root directory of the source project).
-u	<p>Optional but if specified <code>-c</code> must also be specified. Specifies the name of the PU (within the specified CDD) that contains settings to be migrated.</p> <p>HTTP channel settings from this PU are migrated to all HTTP channel resources in the project.</p>

To Open a Project Imported at the Command Line in TIBCO BusinessEvents Studio

To open a project imported at the command line, you must add it as a new project.

1. Start TIBCO BusinessEvents Studio. In Windows, click Start > All Programs > TIBCO > *YourEnvironment* > TIBCO BusinessEvents 5.0 > TIBCO BusinessEvents Studio.
2. From the File menu select **New > Project**. You see the New Project — Select a Wizard dialog.
3. Select **TIBCO BusinessEvents > Studio Project** and click **Next**.
4. In the Project Name field, enter the directory name where the imported project is located. (This is used as the project name.)
5. (If you imported the project to a directory in your default workspace, skip this step.) If the project directory is located outside the default workspace, uncheck the Use default location checkbox and browse to the *parent* directory of the project imported at the command line.
6. Click **Finish**. The project folders appear in the Studio Explorer view.

Importing a TIBCO BusinessEvents 3.x Project at the Command Line

Importing TIBCO BusinessEvents 3.x projects is done in the context of migration. You can also import 3.x projects using a TIBCO BusinessEvents Studio option. See Chapter 4, Migrating Projects from Earlier Versions in *TIBCO BusinessEvents Installation* for details.

This utility imports a TIBCO BusinessEvents 3.x (TIBCO Designer) project at the command line. Projects imported using this command-line utility do not run in the context of Eclipse (TIBCO BusinessEvents Studio). An additional step is required if you want to open the project in TIBCO BusinessEvents Studio, as explained below.

To Import a 3.x Project at the Command Line

1. Navigate to *BE_HOME*/studio/bin/ and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -op importDesigner -d designerProjDir [-n studioProjName] -p studioProjDir
```

For example:

```
studio-tools -op importDesigner -d C:\FT\FT_Project -n FTImport -p c:\myWorkspace\FTImported
```

When the import has completed successfully, you see a message in the command window:

```
Imported the Designer Project successfully.
```

[Table 7, TIBCO BusinessEvents Studio Tools Options for Importing 3.x TIBCO Designer Projects](#), provides detailed information about the options.

See [To Open a Project Imported at the Command Line in TIBCO BusinessEvents Studio, page 33](#) for a procedure you must do if you want to open the project in TIBCO BusinessEvents Studio.

Table 7 TIBCO BusinessEvents Studio Tools Options for Importing 3.x TIBCO Designer Projects

Option	Description
-op importDesigner	Specifies the importDesigner operation for importing a TIBCO BusinessEvents 3.x (TIBCO Designer) project.

Table 7 TIBCO BusinessEvents Studio Tools Options for Importing 3.x TIBCO Designer Projects (Cont'd)

Option	Description
-d	Absolute path to the TIBCO Designer project directory (the <code>.dat</code> file can be included but is not required). Import of release 3.x projects is supported.
-n	Optional. Specifies the name of the TIBCO BusinessEvents 4.x project (TIBCO BusinessEvents Studio project). If not specified, the TIBCO BusinessEvents Studio 3.x project name is used.
-p	Absolute path to the TIBCO BusinessEvents 4.x project directory (the TIBCO BusinessEvents Studio project). The directories in the path are created if they do not exist.

To Open a Project Imported at the Command Line in TIBCO BusinessEvents Studio

To open a project imported at the command line, you must add it as a new project.

1. Start TIBCO BusinessEvents Studio. In Windows, click **Start > All Programs > TIBCO > YourEnvironment > TIBCO BusinessEvents 5.0 > TIBCO BusinessEvents Studio**.
2. From the File menu select **New > Project**. You see the New Project — Select a Wizard dialog.
3. Select **TIBCO BusinessEvents > Studio Project** and click **Next**.
4. In the Project Name field, enter the directory name where the imported project is located. (This is used as the project name.)
5. (If you imported the project to a directory in your default workspace, skip this step.) If the project directory is located outside the default workspace, uncheck the Use default location checkbox and browse to the directory of the project imported at the command line.
6. Click **Finish**. The project folders appear in the Studio Explorer view.

Working With Project Libraries at the Command Line

See [Working with Project Libraries on page 12](#) for an introduction to project libraries and procedures for working with them in TIBCO BusinessEvents Studio.

This section explains how to use a command-line utility to do the following:

- Create a project library (and optionally overwrite any existing library at the same location with the same name). This action creates the `.projlib` file. The corresponding action in TIBCO BusinessEvents Studio is exporting a project library.
- Add an existing project library to a TIBCO BusinessEvents Studio project. The corresponding action in TIBCO BusinessEvents Studio is importing a project library (or adding it to the build path property page).
- Remove an existing project library from a TIBCO BusinessEvents Studio project. The corresponding TIBCO BusinessEvents Studio action is removing the library from the Build Path property page.

To Work with Project Libraries at the Command Line

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt. Arguments for creating a project library are shown separately for clarity:

To create a project library:

```
studio-tools -op buildLibrary [-h] -p studioProjDir [-f resources] [-x] -n projectLib
```

To add or remove an existing project directory in a TIBCO BusinessEvents Studio project:

```
studio-tools -op buildLibrary [-h] -p studioProjDir [-a] [-r] -n projectLib
```

Examples for Creating a Project Library

Example 1 The following command creates a project library using the specified project's contents:

```
studio-tools -op buildLibrary -p C:\workspace\MyProj -n C:\test\myproj.projlib
```

Example 2 The following command creates a project library at the location specified, and overwrites any project library already at that location. From the specified project, the project library uses only the resources specified in the `-f` argument: the `Concepts` folder, the `MyRule.rule` and all dependent resources referred to by these resources.

```
studio-tools -op buildLibrary -x -p C:\workspace\MyProj -n C:\test\myproj.projlib
-f Concepts,Rules\MyRule.rule
```

Examples for Adding and Removing a Project Library

When a project library exists, built either using TIBCO BusinessEvents Studio or using the command-line options shown in [Examples for Creating a Project Library, page 34](#), you can use the `buildLibrary` operation to add a project library to or remove a project library from a TIBCO BusinessEvents Studio project.

Example 3 The following command adds the `myproj.projlib` project library to the TIBCO BusinessEvents Studio project at `C:\workspace\MyProj`.

```
studio-tools -op buildLibrary -a -p C:\workspace\MyProj -n C:\test\myproj.projlib
```

Example 4 The following example removes the `myproj.projlib` project library from the TIBCO BusinessEvents Studio project at `C:\workspace\MyProj`.

```
studio-tools -op buildLibrary -r -p C:\workspace\MyProj -n C:\test\myproj.projlib
```

[Table 8](#) provides detailed information about the options.

Table 8 TIBCO BusinessEvents Studio Tools Options for Working with Project Libraries

Option	Description
<code>-op buildLibrary</code>	Specifies the <code>buildLibrary</code> tool for adding, removing, and creating (overwriting as needed) project libraries.
<code>-h</code>	Optional. Displays help.
<code>-p</code>	File path to the TIBCO BusinessEvents Studio project. Resources from this project are used to create the project library. This filepath is also used to identify the project to which you want to add a project library.
<code>[-r -a]</code>	Use one of these arguments as needed. <code>-r</code> : removes the specified project library from the project specified by <code>-p</code> . <code>-a</code> : adds the specified project library to the specified project. If

Table 8 TIBCO BusinessEvents Studio Tools Options for Working with Project Libraries (Cont'd)

Option	Description
- x	Optional. Overwrites any existing project library at the location specified by -n.
- n	The absolute path of the project library to be created or added or removed
- f	Optional. A comma-separated list of resources to include in the project library, using a path relative to the <code>projectDir</code> . If not specified all resources of the specified project are used.

Migrating Core Coherence Functions at the Command Line

The TIBCO BusinessEvents DataGrid component is the default cache provider for TIBCO BusinessEvents. Use of Oracle Coherence is also supported for those who want to use their own copy of this software.

A set of core Coherence functions has been renamed in TIBCO BusinessEvents 5.0 (and additional internal changes have been made) so that these functions can be used with either the Coherence or the TIBCO BusinessEvents DataGrid cache provider.

Migrating existing projects to use the new function names is one of the actions done automatically when you import a project from a previous release, as explained in Chapter 4, *Migrating Projects from Earlier Versions in TIBCO BusinessEvents Installation*.

Separate use of the `migrateCoherenceCalls` operation may be required in some cases. For example, you create a project in the current version with Coherence as the cache provider, and enable and use the Coherence-specific catalog functions. Then you switch to using TIBCO BusinessEvents DataGrid as the cache provider. You would then use the `migrateCoherenceCalls` operation to migrate those functions.

To Migrate Core Coherence Functions at the Command Line

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools.exe -op migrateCoherenceCalls -projectDir studioProjectDir
```

When the migration has completed successfully, you see a message in the command window:

```
All Coherence function calls have been migrated successfully.
```

Importing an Excel File into TIBCO BusinessEvents Decision Manager at the Command Line

You can import a correctly formatted Microsoft Excel file as a decision table using the TIBCO BusinessEvents Decision Manager user interface, or at the command line. This section explains how to do this task at the command line.

To Import an Excel File into TIBCO BusinessEvents Decision Manager at the Command Line

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -op importExcel [-h] -studioProjPath studioProjectDir -projWsPath projectWorkspaceDir -excelPath excelFilePath -dtName decisionTableName -folderPath decisionTableProjectFolderPath -vrfPath vrfProjectFolderPath
```

For example:

```
studio-tools -op importExcel -studioProjPath D:\Workspace\CreditCardApplication -projWsPath D:\Workspace -excelPath c:\BankUser.xls -dtName BankUser -folderPath \Virtual_RF -vrfPath \Virtual_RF\BankUser_VirtualRuleFunction
```

Table 9 provides detailed information about the options.

Table 9 TIBCO BusinessEvents Studio Tools Options for Importing Excel Files into TIBCO BusinessEvents Decision Manager

Option	Description
-op importExcel	Specifies the <code>importExcel</code> tool for importing Excel files containing decision table definitions.
-h	Optional. Displays help.
-studioProjPath	File path to the TIBCO BusinessEvents Studio project into which the decision table will be added.
-projWsPath	File path to the Eclipse Workspace used for the TIBCO BusinessEvents Studio project.
-excelPath	File path to the Excel file to be imported as a decision table.
-dtName	Name of the decision table that the Excel file imports as.

Table 9 TIBCO BusinessEvents Studio Tools Options for Importing Excel Files into TIBCO BusinessEvents Decision Manager (Cont'd)

Option	Description
<code>-folderPath</code>	TIBCO BusinessEvents Studio project path to the decision table (but not including the decision table name).
<code>-vrfPath</code>	TIBCO BusinessEvents Studio path to the virtual rule function under which to create the decision table. The decision table is an implementation of this virtual rule function.

Validating a Decision Table at the Command Line

You can validate a decision table using the TIBCO BusinessEvents Decision Manager user interface, or at the command line. This section explains how to do this task at the command line.

To Validate a Decision Table at the Command Line

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -op validateTable [-h] -studioProjPath studioProjectDir -projWsPath projectWorkspaceDir -dtPath decisionTableProjectFolderPath
```

For example:

```
studio-tools.exe -op validateTable -studioProjPath
D:\Workspace\CreditCardApplication -projWsPath D:\Workspace -dtPath
\Virtual_RF\BankUser
```

[Table 10, TIBCO BusinessEvents Studio Tools Options for Validating Decision Tables](#), provides detailed information about the options.

Table 10 TIBCO BusinessEvents Studio Tools Options for Validating Decision Tables

Option	Description
-op validateTable	Specifies the validateTable tool for building validating decision tables.
-h	Optional. Displays help.
-studioProjPath	File path to the TIBCO BusinessEvents Studio project containing the decision tables.
-projWsPath	File path to the Eclipse Workspace used for the TIBCO BusinessEvents Studio project.
-dtPath	TIBCO BusinessEvents Studio project path to the decision table (including the decision table name.

Generating Decision Table Class Files at the Command Line

You can generate class files for a decision table using the TIBCO BusinessEvents Decision Manager user interface, or at the command line. The class files are used for deployment of decision tables. This section explains how to do this task at the command line.

To Generate Class Files at the Command Line

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -op generateClass [-h] -p studioProjectDir [-n studioProjectName] -o outputPath
[-x {true | false}] [-cp extendedClasspath]
```

For example:

```
studio-tools -op generateClass -p D:\Workspace\FraudDetection -o c:\temp -x true
-cp c:\tibco\be\5.0\lib\myjar.jar
```

* We need to give separate jar file path for each jar required for project compilation.

[Table 11, TIBCO BusinessEvents Studio Tools for Generating Class Files](#), provides detailed information about the options.

Table 11 *TIBCO BusinessEvents Studio Tools for Generating Class Files*

Option	Description
<code>-op generateClass</code>	Specifies the <code>generateClass</code> operation for generating class files for deployment of decision tables.
<code>-h</code>	Optional. Displays help.
<code>-p</code>	Absolute path to the TIBCO BusinessEvents Studio project directory. The EAR file is built using this project.
<code>-n</code>	Optional. Specifies the name of the TIBCO BusinessEvents Studio project containing the decision table. If not specified, the final (leaf) directory name in the <i>projectDir</i> path is used as the project name.
<code>-o</code>	Specifies the output directory for generated classes.
<code>-x</code>	Optional. If <code>true</code> , overwrites any existing class file with the same name.

Table 11 TIBCO BusinessEvents Studio Tools for Generating Class Files (Cont'd)

Option	Description
-cp	<p>Optional. Extended classpath. Use as needed. For example, additional classpath information is needed if the decision table uses custom functions or third-party JAR files. Separate entries by the appropriate path separator. For example if the separator is semicolon (;):</p> <pre>C:\customjars\custom.jar;C:\customjars\custom2.jar</pre>

Committing Project Artifacts to RMS at the Command Line

You can commit various project artifacts to RMS so that TIBCO BusinessEvents Decision Manager users can check them out to work locally on them.

To Commit Project Artifacts to RMS at the Command Line

1. Navigate to `BE_HOME/studio/bin/` and open a command prompt.
2. Execute a command with this format (all one line) at a command prompt:

```
studio-tools -op commit [-h] -studioProjPath <Studio Project Workspace Path> -rmsBaseURL
baseURLForRMS -rmsUsername RMSLoginUsername [-rmsPassword RMSLoginPassword]
-artifactPath projectPathtoArtifact -artifactType {decisiontable | domain | channel |
concept | event | timeevent | metric}
```

For example:

```
studio-tools -op commit -studioProjPath D:\Workspace\CreditCardApplication
-rmsBaseURL http://sesame.acme.com:5000/Channels/MyChannel/ -rmsUsername admin
-rmsPassword admin -artifactPath \Virtual_RF\BankUser -artifactType decisiontable
```

[Table 12, TIBCO BusinessEvents Studio Tools Options for RMS Commit](#), provides detailed information about the options.

Table 12 TIBCO BusinessEvents Studio Tools Options for RMS Commit

Option	Description
<code>-op commit</code>	Specifies the <code>commit</code> tool for committing project artifacts to RMS.
<code>-h</code>	Optional. Displays help.
<code>-studioProjPath</code>	File path to the TIBCO BusinessEvents Studio project that contains the artifacts to be committed.
<code>-rmsBaseURL</code>	Base URL of the RMS server.
<code>-rmsUsername</code>	Username required to log into the RMS server
<code>-rmsPassword</code>	Password, if required to log into the RMS server.
<code>-artifactPath</code>	TIBCO BusinessEvents Studio project path to the artifacts to be imported. Valid values are <code>decisiontable</code> , <code>domain</code> , <code>channel</code> , <code>concept</code> , <code>event</code> , <code>timeevent</code> , and <code>metric</code> . When another user checks out the project, the committed artifacts are copied to that user's project.

Setting up RMS Projects at the Command Line

You can set up RMS projects in TIBCO BusinessEvents Decision Manager. This section explains how to do this task at the command line.

To Set up RMS Projects at the Command Line

1. Navigate to *BE_HOME*/studio/bin/ and open a command prompt.
2. Execute a command with the following format (all on one line) at a command prompt:

```
studio-tools -op setup [-h] -projName RMSProjName [-baseLocation RMSBaseLoc]  
[-sourceLocation StudioProjectDir] [-aclPath ACLDir]
```

For example:

```
studio-tools -op setup -projName CreditCardApplication -baseLocation  
C:\tibco\be\5.0\rms\examples -sourceLocation D:\Workspace -acl c:\ACL\  
CreditCardApplication.acl
```

Table 13, TIBCO BusinessEvents Studio Tools Options for RMS Project Setup, provides detailed information about the options.

Table 13 TIBCO BusinessEvents Studio Tools Options for RMS Project Setup

Option	Description
-op setup	Specifies the <code>setup</code> tool for setting up an RMS project.
-h	Optional. Displays help.
-projName	The name of the RMS project.
-baseLocation	Optional. Specifies the base location of the RMS project. If not specified, the current directory is used.
-sourceLocation	Optional. Specifies the source location of the studio project. If specified, the contents are copied to the <code>decisiondata</code> directory.
-aclPath	Optional. File path to the ACL file used for this project. If not specified, the default ACL file is used. For file-based authentication, it is located under the default base location for RMS: < <i>RMSBaseLoc</i> > / < <i>projName</i> > / config / < <i>projName</i> > .acl

Chapter 4

Element Refactoring Operations

When you make changes to a project element, all references to that element must be updated accordingly. When such changes affect only the structure and not the behavior of the project, this operation is known as *project refactoring*. This chapter explains how to use the refactoring features, and some related features to do with copy-paste operations.

Topics

- [Renaming, Moving, Deleting, and Copy-Pasting Elements, page 46](#)
- [Migrating Core Coherence Functions, page 52](#)
- [Automatic Refactoring Actions and Limitations, page 55](#)

Renaming, Moving, Deleting, and Copy-Pasting Elements

Changes that affect the structure of a project, but not its behavior, are known as *project refactoring* changes. Refactoring ensures that the project structure remains self-consistent.

Copy-paste operations are not strictly speaking refactoring operations. However some refactoring is also done to support these operations, so they are included here.



You can copy items from one project to other projects in the workspace. Ensure that the items are suitable for their destination projects.

Moving, renaming, deleting, or copy-pasting project elements are changes that often affect other parts of a TIBCO BusinessEvents Studio project. Names of elements, element properties, and element locations, are referenced in various parts of a project such as rules, rule functions, and concept relationship properties.

When you make changes to a project element, all references to that element must be updated accordingly. The refactoring wizard has a preview page that enables you to review all these changes (see [Working with the Preview Page on page 49](#)).

This section explains the refactoring (and related) procedures. See [Automatic Refactoring Actions and Limitations on page 55](#) to understand what TIBCO BusinessEvents does for each type of refactoring operation.

Updating All References is Strongly Recommended

To ensure the integrity of the project, it is strongly recommended that you make all changes to all locations where a renamed or moved items is referenced. Only disable such changes if you are certain there are no references to the element, or there are unusual circumstances that justify such action.



To Undo Changes You can undo any change you make to a project. Click **Edit > Undo** or press **Ctrl+Z**.

Project Level Actions

You can rename a project (use File > Rename) and you can copy and paste a project. However, you can't move a project.

Renaming, Moving, and Deleting Elements

Renaming, moving, and deleting are refactoring actions that can have an effect on other parts of the project where element names and locations are referenced.

To Rename a Project Element

1. Rename the element using one of the following methods:
 - Right-click a project element in Studio Explorer and select **Refactor > Rename**.
 - Highlight the element, then select **File > Rename**.
 - Highlight the element, then press **F2**.

The first page of the Rename Element wizard appears.
2. In the New Name field, type the new name.
3. If you don't want to rename the element in places where it is referenced, uncheck the Update References checkbox (but see [Updating All References is Strongly Recommended on page 46](#)).
4. Click **Preview**. One of the following occurs:
 - The preview page displays so you can examine the effect of this change. See [Working with the Preview Page on page 49](#) for details.
 - The problems page displays if the rename can't be done, for example, because a new element name is used by an existing element. Click **Back** to fix the problem, or **Cancel** to cancel the rename.
5. In the preview page, clear checkboxes if you don't want the change to be made in some referenced locations. This is not generally recommended (see [Updating All References is Strongly Recommended on page 46](#)).
6. To complete the change, click **OK**.

To Move an Element to a Different Project Folder

1. Move the element using one of the following methods:
 - Right-click a project element name in Studio Explorer and select **Refactor > Move**.
 - Highlight the element, then select **File > Move**.
 - Drag the element to the target folder (go to [step 3](#)).
2. If you opened the Move Element wizard using menus, navigate the project tree in the first page of the wizard to select a destination folder, then click **Preview**.

3. One of the following occurs:
 - The preview page displays so you can examine the effect of this change. See [Working with the Preview Page on page 49](#) for details.
 - The problem page displays if the move can't be made. Click **Back** to fix the problem, or **Cancel** to cancel the move.
4. In the preview page, clear checkboxes if you don't want the change to be made in some referenced locations. This is not generally recommended (see [Updating All References is Strongly Recommended on page 46](#)).
5. To complete the change, click **OK**.

To Delete an Element or Folder

When you delete a folder, all elements within that folder are also deleted.

1. Do one of the following:
 - Right-click a project element or folder name in Studio Explorer and select **Delete**.
 - Highlight the element, then select **Edit > Delete**.
 - Highlight the element, then press the **Delete** key.
2. At the Delete Resources page, click **OK** to delete without previewing, or click **Preview** to preview the effect of the deletion.
3. If you click Preview, one of the following occurs:
 - The preview page displays so you can examine the effect of this change. (For deletions, there is generally no information.) Click **OK** to delete.
 - The problems page displays if there is a problem with the deletion. Click **Back** to fix the problem, or **Continue** to force the deletion (or **Cancel** to cancel the deletion).

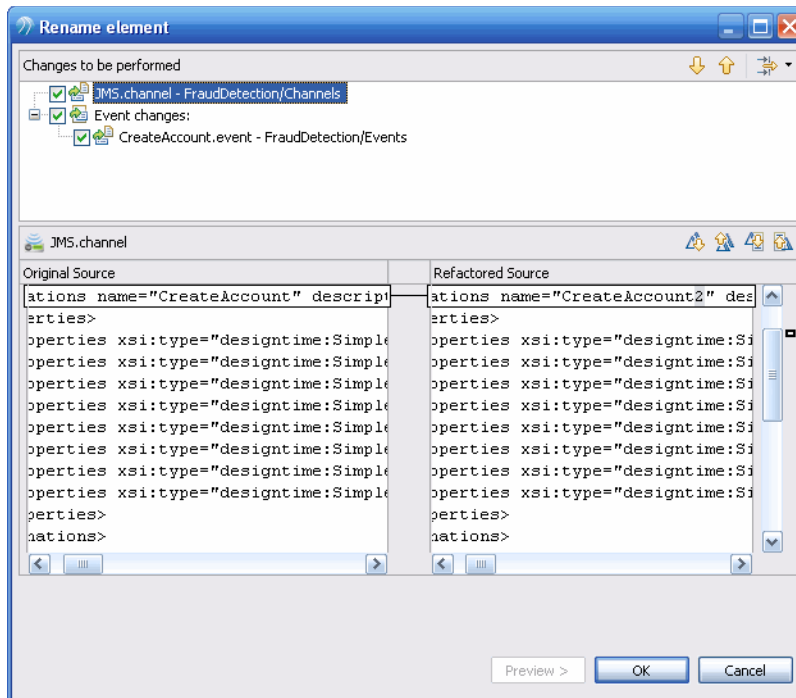
To Delete a Project

1. Do one of the following:
 - Right-click the project name in Studio Explorer and select **Delete**.
 - Highlight the project name, then select **Edit > Delete**.
 - Highlight the project name, then press the **Delete** key.
2. If you want to remove the project contents on disk, check the Delete Project Contents on Disk check box. If you do not check this checkbox, then the project is removed from Studio Explorer, but the project contents remain on disk.

3. Click **OK** to delete without previewing, or click **Preview** to preview the effect of the deletion.
4. If you click Preview, one of the following occurs:
 - The preview page displays so you can examine the effect of this change. (For deletions, there is generally no information.) Click **OK** to delete.
 - The problems page displays if there is a problem with the deletion. Click **Back** to fix the problem, or **Continue** to force the deletion (or **Cancel** to cancel the deletion).

Working with the Preview Page

During refactoring operations, you can click Preview to examine the effects of the operation on the project. (If a pre-check finds issues that may prevent the operation from completing successfully, a problem page appears instead.) This section explains how to use the preview page.



Checks in the upper panel indicate elements that will change as a result of a refactoring operation. Expand to take a closer look at individual folders and elements. All elements that appear are affected by the change. When you highlight an element, details of the change to be made are shown in the lower panel.

The Original Source panel on the left and the Refactored Source on the right use the persisted format of the element. Change bars indicate changed areas.

Do any of the following as needed to examine the changes and select a subset of the changes to be done on clicking OK:

- In the upper panel, use the arrows to navigate up and down a long list of changes.
- Click the Filter Changes button and select Hide Derived Resources. For example, a diagram is a derived resource. Diagrams are not persisted. You can easily recreate them. So you may not be interested in seeing those changes.
- If you want to apply the change to only some of the project elements, uncheck the checkboxes next to the elements as desired. For example, you may wish to replace or delete an element after you have completed the refactoring operation, so you don't need to apply the change to that element.



It is recommended that you accept all changes to be performed Your project can become corrupted if you do not make the changes throughout the project. Only deselect changes if you have a specific, valid reason to do so.

Copy-Pasting an Element

Copy-pasting is not a true refactoring operation, because it does change the behavior of the TIBCO BusinessEvents Studio project. You would generally make manual changes to your project to use the newly created project element. However, some limited refactoring is done for your convenience. For example, the definition of a rule or rule function begins with its fully qualified name, such as the following:

```
rule SomeRules.Application_Rule
```

If you copy and paste the above rule into the folder `OtherRules`, the definition of the rule automatically changes to:

```
rule OtherRules.Application_Rule
```

To Copy-Paste an Element

1. In Studio Explorer, highlight the element (or folder) and do one of the following:
 - Right-click the element and select **Copy**.
 - Press `Ctrl+C`.
 - Select **Edit > Copy**.

2. Highlight the project folder where you want to paste the element, and do one of the following:
 - Right-click the folder and select **Paste**.
 - Press `Ctrl+V`.
 - Select **Edit > Paste**.

If you are pasting an element to the same folder that you copied it from, a dialog enables you to provide a different name. The default value is *CopyOfoldname*.

Migrating Core Coherence Functions

The TIBCO BusinessEvents DataGrid component is used as the default cache provider. Use of Oracle Coherence is also supported for those who want to use their own copy of this software.

A set of core functions has been renamed (and additional internal changes have been made) so that these functions can be used with either the Coherence or the TIBCO BusinessEvents DataGrid cache provider.

The renamed functions are in the Standard catalog `cluster.DataGrid` category. You must use these functions going forward, and not the prior versions. See [Table 14, Mapping of Coherence Functions to DataGrid Functions, on page 53](#) for details.

For all projects created prior to version 5.0.0, use the 4.x project import wizard, which migrates the core Coherence function calls automatically.

For occasional use cases where the function calls are migrated manually, you can use the manual refactoring operation. You can also use a command-line option as explained in [Migrating Core Coherence Functions at the Command Line on page 37](#).

To Migrate Coherence Function Calls in TIBCO BusinessEvents Studio

See [Table 14, Mapping of Coherence Functions to DataGrid Functions, on page 53](#) for details of the functions that are mapped using this feature.

1. Open the project in TIBCO BusinessEvents Studio (3.x projects must be migrated first). Migrate the function calls using one of the following methods:
 - Right-click the project name in Studio Explorer and select **Refactor > Migrate Coherence Function Calls**.

The first page of the Migrate Coherence Function Calls wizard appears. It displays an informational list of the functions that the utility renames.

2. Click **OK** to make all necessary changes without previewing, or click **Preview** to preview the effects of the change.
3. If you click Preview, one of the following occurs:
 - The preview page displays so you can see the list of affected elements (which include rules, rule functions, expiry actions and state modeler transition, exit, and entry rules). You can scroll the list and view the original and refactored sources for each affected element. You can uncheck the checkboxes of elements you wish to remain unchanged (and later make

changes manually as needed). Click **OK** to make the changes to all or selected elements.

- The problems page displays if there is a problem with the change. Click **Back** to fix the problem, or **Continue** to force the change (or **Cancel** to cancel the change).
- The preview page displays this message: The refactoring does not change any source code. Click **OK** and complete the migration anyway.



Complete the refactoring operation even if no project code is affected This refactoring operation makes additional changes to your projects so that your projects will work correctly with the new function names.

Mapping of Coherence Functions to TIBCO BusinessEvents DataGrid Functions

Table 14 Mapping of Coherence Functions to DataGrid Functions

4.x Coherence Category Function Name	DataGrid Category Function Name
<code>C_CacheGetEntityById()</code>	<code>CacheGetEntityById()</code>
<code>C_CacheLoadConceptByExtId()</code>	<code>CacheLoadConceptByExtId()</code>
<code>C_CacheLoadConceptById()</code>	<code>CacheLoadConceptById()</code>
<code>C_CacheLoadConceptIndexedByExtId()</code>	<code>CacheLoadConceptIndexedByExtId()</code>
<code>C_CacheLoadConceptsByExtId()</code>	<code>CacheLoadConceptsByExtId()</code>
<code>C_CacheLoadEntity()</code>	<code>CacheLoadEntity()</code>
<code>C_CacheLoadEventByExtId()</code>	<code>CacheLoadEventByExtId()</code>
<code>C_CacheLoadEventById()</code>	<code>CacheLoadEventById()</code>
<code>C_CacheLoadParent()</code>	<code>CacheLoadParent()</code>
<code>C_CacheName()</code>	<code>CacheName()</code>
<code>C_CacheReevaluate()</code>	<code>CacheReevaluate()</code>
<code>C_ClassName()</code>	<code>ClassName()</code>
<code>C_EnableCacheUpdate()</code>	<code>EnableCacheUpdate()</code>
<code>C_Index()</code>	<code>Index()</code>
<code>C_Lock()</code>	<code>Lock()</code>
<code>C_TransactionProperties</code>	<code>TransactionProperties()</code>

Table 14 Mapping of Coherence Functions to DataGrid Functions (Cont'd)

4.x Coherence Category Function Name	DataGrid Category Function Name
C_UnLock()	UnLock()

Automatic Refactoring Actions and Limitations

This section explains what is done for each kind of refactoring operation. It also lists some limitations.

Refactoring Limitations

Changes made to certain items are not refactored in this release. You must handle reference updates manually.

References to moved or changed folders in strings References to changed or moved folders are not updated in strings, including CDD and XSLT strings. CDD strings are used in the Cluster Definition Descriptor editor to point to project resources using their project path. XSLT strings are used in mapper functions, which are completed using the Function Argument Mapper. (Such references are, however, updated for entity refactoring operations).

Copy-paste of folders Elements inside the pasted folder are not updated.

References to global variables and shared resources Refactoring does not handle changes to global variables and shared resources. You must manually update references to global variables and shared resources that you change.



TIBCO BusinessEvents destinations and property definitions cannot be moved.

Refactoring for Move and Rename Operations

Move and rename refactoring operations change only the structure of a project. For example, when you change a concept name, that name must change everywhere the concept is referenced in the project. If the element has its own file, the file must also be renamed.

References to the changed or moved element are handled as shown in [Table 15](#). Projects can be complex; this list covers the main cases.

Table 15 Refactoring for Move and Rename Operations

Renaming this...	Updates references in these places...
Concept	Concepts that inherit from this one Event expiry actions Property definitions for contained or referenced concepts State models Rules and rule functions
Event	Events that inherit from this one Event expiry actions Destination (Default Event) State models Rules and rule functions
Property of a concept or an event	Event expiry actions State models Rules and rule functions
Domain Model	Domain models that inherit from this one Associated properties
Channel	Event default destination paths
Destination	Event default destination paths
Rule	The rule source Rules and rule functions
Rule Function	Event expiry actions State models The rule function source Rules and rule functions

Table 15 Refactoring for Move and Rename Operations (Cont'd)

Renaming this...	Updates references in these places...
Folder	All location that this folder is used in a path, for example path to a default destination in an event, property definitions for contained or referenced concepts, and in rules and rule functions. For folder refactoring limitations, See Refactoring Limitations on page 55 .

Refactoring for Delete Operations

Element deletion can affect project behavior. You must take care to ensure that your project behavior is as desired after the deletion refactoring.

Deletion removes all references to the deleted object only in specific cases:

- References to a deleted domain model are removed from concepts that refer to it.
- Reference to a deleted state model are removed from the owning concept

Chapter 5 Channels and Destinations

This chapter explains how to configure channels and destinations, and includes information about Rendezvous and local channels. Additional information about JMS and HTTP channels is provided in [Chapter 6, JMS Channels, on page 81](#) and [Chapter 7, HTTP and SOAP Channels, on page 95](#).

Topics

- [Overview of Channels and Destinations, page 60](#)
- [Selecting a Serializer, page 61](#)
- [Mapping Incoming Messages to Non-default Events, page 62](#)
- [Working with Rendezvous Channels, page 63](#)
- [Working with Local Channels, page 66](#)
- [Adding Channels and Destinations, page 67](#)
- [Communicating with Other Sources using TCP, page 70](#)
- [Channel Resource Reference, page 73](#)
- [Destination Resource Reference, page 76](#)

Overview of Channels and Destinations

One project can have multiple channels of different types with multiple destinations as needed.

See Also

- Channels and Events in *TIBCO BusinessEvents Architect's Guide* to understand more about the role of channels in a project.
- [Chapter 24, Diagrams, page 421](#) for information on using channel dependency diagrams.
- The section, For TIBCO Enterprise Message Service and TIBCO Rendezvous Channels, in *TIBCO BusinessEvents Administration*, for TRA file updates required for Rendezvous and JMS channels that use TIBCO Enterprise Message Service.

Types of Channels

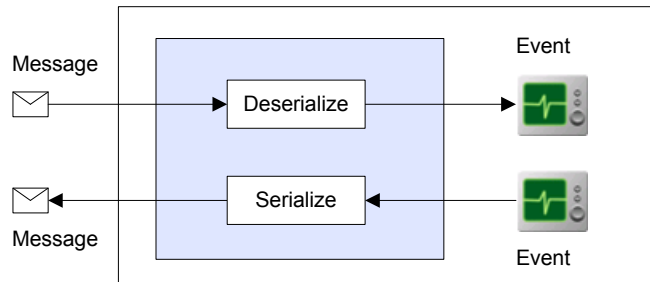
You can choose from the following types of channels:

- **TIBCO Rendezvous channels** Connect TIBCO BusinessEvents to TIBCO Rendezvous sources and sinks. See [Working with Rendezvous Channels on page 63](#).
- **Local channels** Connect co-located agents at runtime. See [Working with Local Channels on page 66](#).
- **JMS channels** Connect TIBCO BusinessEvents to TIBCO Enterprise Message Service provider sources and sinks. See [Chapter 6, JMS Channels, on page 81](#).
- **HTTP channels, including SOAP support** An HTTP channel acts as an HTTP server at runtime. This enables TIBCO BusinessEvents to serve requests from clients, and to act as a client of other servers. Support for SOAP protocol is provided by a specialized event type and a set of functions for extracting information from SOAP request messages and constructing response messages. See [Chapter 7, HTTP and SOAP Channels, on page 95](#).

Selecting a Serializer

For each type of channel (except local channels), TIBCO BusinessEvents uses a serializer to convert events to messages and a deserializer to convert incoming messages to events. Local channels do not require serializers.

Figure 1 Serializer and Deserializer Behavior



When you configure a destination, you select the appropriate serializer. (It actually includes the serializer and deserializer). For JMS and Rendezvous messages, different serializers are available for different needs. For details see [Selecting a JMS Serializer on page 83](#) and [Working with Rendezvous Channels on page 63](#).

Deploy-time Configuration

The following channel-related configuration tasks are performed at deploy time.

- Selecting event preprocessors (rule functions) for a destination and defining worker thread options for each preprocessor. (See [Event Preprocessors on page 243](#) for more details.)
- Specifying destination listeners. You select the destinations that an agent will listen to at runtime when you configure the Cluster Deployment Descriptor. See the section Configuring the Agent Classes Tab (All OM Types) in *TIBCO BusinessEvents Administration*.

See *TIBCO BusinessEvents Administration* for all deploy-time configuration topics.

Mapping Incoming Messages to Non-default Events

Incoming messages can be mapped to a default event that you specify when you configure the destination. (See Default Destinations and Default Events in *TIBCO BusinessEvents Architect's Guide* for more details). You can also map incoming messages to specified event types.

Two fields in a message header instruct TIBCO BusinessEvents to map the incoming message to a specified event type:

- The field named `_ns_` takes a namespace as a value. The namespace points to the event type, for example, `www.tibco.com/be/ontology/Events/MyEvent`
- The field named `_nm_` takes the name of the event, for example, `NewMyEvent`

These fields are added and filled automatically for events created using TIBCO BusinessEvents rules. You can also add these fields to messages coming in from other sources, if you have control of those messages.

See also:

[Mapping of HTTP Request URI to Destination on page 98](#)

[Mapping of SOAP Request URI to Destination on page 117](#)

[Specifying Default and Non-Default Destinations on page 135](#)

Working with Rendezvous Channels

This section explains the behavior of the provided Rendezvous serializer classes and other related information.

Note that the Rendezvous serializers use UTF8 encoding for XML payloads.

The type of the event from which the outbound message is serialized is added to the Rendezvous message header using `_nm_` and `_ns_` fields so that if the message is used in TIBCO BusinessEvents again, the correct event type is used to deserialize the message (ignoring the default event specified for the destination).

See [Adding Channels and Destinations on page 67](#) for details on adding Rendezvous channels.

See the section, For TIBCO Enterprise Message Service and TIBCO Rendezvous Channels, in *TIBCO BusinessEvents Administration*, for TRA file updates required for Rendezvous.

Rendezvous Message Header

For Rendezvous messages, the only header that TIBCO BusinessEvents interprets is `_sendsubject_` which is of type String. It is a read-only property. The event has to define this property to receive the value. The value is the actual Subject on which the message was sent.

Basic Serializer

The basic serializer, `TibRvMsgSerializer` serializer, is for efficient handling of events and messages that do not have payloads. It ignores payloads in messages and in events if any exist.

First level Rendezvous property values are used as values for matching event properties. Any additional (non-matching) Rendezvous properties are ignored.

Serializer For Use with Payloads

To include a payload in a Rendezvous message, ensure that the message has a `_payload_` field. To pass contents between the `Rendezvous message _payload_` field and an event payload, use the `TibRvMsgSerializerWithXMLPayload` serializer.

Deserializing from Rendezvous Message to Event

First level Rendezvous property values are used as values for matching event properties. Any additional (non-matching) Rendezvous properties are ignored.

The `_payload_` field contents are passed into the event payload. Supported `_payload_` field datatypes and Rendezvous wire format types are as follows:

Data Type	Wire Format Type
String	TibrvMsg.STRING
TibrvXML	TibrvMsg.XML
byte[]	TibrvMsg.OPAQUE TibrvMsg.I8ARRAY

If the event defines a payload, but the incoming Rendezvous message does not have a `_payload_` field, TIBCO BusinessEvents attempts to map the entire message as the event payload.

Serializing from Event to Rendezvous Message

Event properties are transformed to first level Rendezvous message properties.

The event payload is passed to the Rendezvous message `_payload_` field.

If the `_payload_` field is of an unsupported type, or is missing, or if the event has not been configured for a payload, the payload is ignored.

Avoiding REGISTRATION COLLISION RVCM Advisory Messages

This section applies to configuring RVCM when using a Rendezvous transport shared resource (see [Rendezvous Transport on page 214](#)).

REGISTRATION.COLLISION RVCM advisory messages can be thrown when when the same CM name (or CMQ name) is used on all engines. These name collisions can result in thrashing.

To prevent this issue, you must ensure that the CM names (and CMQ names) are different on all engines. To do so, add global variables to the CM Name or CMQ Name, and to the ledger file name if a ledger file is used for RVCM, to ensure the uniqueness of these names. Add one or more of the following variables, depending on need (as explained below):

```
%%EngineName%%  
%%ChannelName%%  
%%ChannelURI%%
```

The `%%EngineName%%` variable is generally required for all names. Note that you must start engines using unique names so that the value of each engine's `%%EngineName%%` variable be different at runtime.

In addition, if different channels use the same RVCM shared resource, you also need to add `%%ChannelName%%` or `%%ChannelURI%%`.

Use `%%ChannelURI%%` in cases where channels using the same RVCM shared resource have the same name but are in different folders.

You must define any of the above String type global variables you use. They are not predefined. However, TIBCO BusinessEvents provides the value at runtime, so you can use any string value or use an empty string as the value when you define the variables.

Working with Local Channels

Local channels are used in rules or rule functions to route events to an appropriate agent running in the same engine (processing unit). Local channels are useful in two cases:

- For applications using In Memory object management (generally used only for testing)
- For certain scenarios where an inference agent is co-deployed with a query agent. See *TIBCO BusinessEvents Event Stream Processing Query Developer's Guide*.

See [Adding Channels and Destinations on page 67](#) for details on adding local channels.

Event Use Count

Local channels pass the same event object between the agents. Consuming the event in one agent does not affect other agents that also received the event over a local channel. A use count is maintained for each event to track how many agents have received the event but not consumed it. The use count of the event is incremented depending on the number of agents it is routed to. When an event is consumed in one agent, TIBCO BusinessEvents deletes the reference to the event in that agent and it decrements the use count of the original event instance.

Using a Local Channel

To route an event to a local channel, use the `Event.routeTo()` function. (You can use this function for other purposes too.)

In the provided example,

`BE_HOME/Examples/MultipleSessionsAndLocalChannel`, events containing small orders are sent to the agent that deals with small orders as follows:

```
Event.routeTo(order, "/Channels/Local/toSmall", "");
```

The signature of this function is as follows:

```
SimpleEvent routeTo(SimpleEvent event, String destinationPath, String properties)
```

Adding Channels and Destinations

The general procedure for creating channels and destinations of all types is the same, though the configuration options are different. A TIBCO Rendezvous channel is shown as an example.

Channel: RV

Configuration

Description:

Driver:

Methods of Configuration:

Extended Configuration

Properties

Service:

Network:

Daemon:

Destinations

+ Add - Remove

input

Name

Description

Default Event

Serializer/Deserializer

Subject

RVCM Pre Registration

LimitPolicy

MaxEvents

DiscardAmount

To Add a Channel

1. In Studio Explorer, right click the folder where you want to store the channel and select **New > Channel**. You see the New Channel Wizard.
2. In the Channel name field, type a name for the channel. In the Description field, type a description.



You cannot change the name in the editor. To change the name of any project element, right-click the element in Studio Explorer and select **Refactor > Rename**. See [Chapter 4, Element Refactoring Operations, on page 45](#) for more details.

3. In the Driver type field, select the appropriate driver:
 - Rendezvous
 - Local
 - JMS
 - HTTP

If you select Local, channel configuration is complete.

4. Click **Finish**. You see the Channel editor.

To Edit a Channel

1. Edit the Description as needed.
2. In the Driver type field, the driver you selected in the wizard is selected. If you want to change to a different type of channel you can do so now. Select the appropriate driver:
 - Rendezvous
 - Local
 - JMS
 - HTTP
3. For all channel types (except Local and HTTP), from the Method of Configuration drop-down list, select one of the following:
 - **Resource** Select Resource if you have a shared resource in your project whose properties you want to reuse for this channel. For HTTP channels, Resource is preselected and cannot be changed.
 - **Properties** Select Properties to configure this channel resource using properties. When you select Properties, a Properties section displays appropriate fields for the type of channel. See [Channel Resource Reference on page 73](#) for help in entering the correct values.



Resource names and directory names The path to the resource and the resource name cannot contain any of the keywords or other reserved words listed in [Keywords and Other Reserved Words on page 280](#).

4. For HTTP channels only, in the serverType field of the Extended Configuration section, select a server type from the drop-down list. See [Configuring TIBCO BusinessEvents to Receive and Send HTTP Requests on page 98](#) for a more detailed procedure.

5. If you selected Resource in the Method of Configuration field, in the Resource field click Browse and select the shared resource you want to use.
6. If you are configuring an HTTP channel, click Advanced and configure tuning properties. See [Configuring TIBCO BusinessEvents to Receive and Send HTTP Requests on page 98](#) for a more detailed procedure. and [HTTP Channel Advanced Configuration Settings on page 102](#) for details.
7. Continue to the section [Adding a Destination to a Channel on page 69](#) and add destinations to the channel as needed.

Adding a Destination to a Channel

One channel can have multiple destinations. Each is shown in the Destinations section of the Channel editor.

1. If it is not already open, open the editor for the channel to which you want to add a destination. To open the channel editor, double-click the channel name in the TIBCO BusinessEvents Studio Explorer view.
2. In the Destinations section, click **Add**.

Common Fields

3. Enter a Name and Description for the destination.
4. In the Default Event field, browse to and select the event to be created (by default) from incoming messages received by this destination. If you have not yet created the event, you can select the default event later.
5. In the Serializer/Deserializer field, select the appropriate class. See the following sections for more details:
 - For JMS destinations: [Selecting a JMS Serializer on page 83](#)
 - For Rendezvous destinations: [Working with Rendezvous Channels on page 63](#)
 - For HTTP and SOAP destinations: [Task D, Add a Destination, on page 100](#), in the section [Configuring TIBCO BusinessEvents to Receive and Send HTTP Requests on page 98](#). (Destination configuration is the same for SOAP and HTTP destinations. See [Manually Creating Resources to Work with SOAP Services on page 117](#) and [Creating Resources Using the WSDL Import Utility on page 119](#) for more information about creating SOAP channels and destinations)

Various fields appear, depending on your selection.

Channel-Specific Fields

6. Complete the rest of the properties for the type of destination you are creating. See [Destination Resource Reference on page 76](#) for details.
7. Save the project.

Communicating with Other Sources using TCP

In addition to Channels, TIBCO BusinessEvents can also communicate with other data sources using TCP. You can create a local TCP server and a TCP client so that TIBCO BusinessEvents can communicate with data sources not otherwise available through channels, using TCP.

TCP communication is available as a Communication Built-in Function in the Catalog Functions view. Using this set of functions you do the following to communicate with TCP servers:

- Create a local TCP server in a startup function using `TCP.createLocalServer()`
- Connect to a Remote TCP server as a client using `TCP.connectToRemoteServer()`
- Register the session listener using `TCP.registerSessionListener()`
- Start the local server using `TCP.startLocalServer()`
- Create callback rule functions and register them as callbacks to the TCP listeners. These callback rule functions create events that are sent to an appropriate destination.

```
TCP.readIntoPayload(SessionName)
```

Example 1 TCP Rule Function to Start a Local TCP Server

Here is a sample rule function to start a local TCP server:

```
void rulefunction RuleFunctions.InitTCPServers {
    attribute {
        validity = ACTION;
    }
    scope {

    }
    body {
        System.debugOut( "Initializing TCP servers" ) ;

        try {

            TCP.createLocalServer("MyTCPServer", "localhost",
            System.getGlobalVariableAsInt("NSN/SocketAdaptor/Port",
            8055));

            TCP.registerSessionListener("MyTCPServer",
            "/RuleFunctions/RawCDRCallback");

            TCP.startLocalServer("MyTCPServer");
```

```

    } catch (Exception ex) {
        System.debugOut("Exception occurred while initializing
        TCP server: " + ex@message);
    }
    System.debugOut("TCP server initialization done");
}
}

```

Example 2 TCP Rule Function to Connect to a Remote TCP Server

Here is a sample rule function to connect to a remote TCP server as a client:

```

Events.RemoteMsgResponseEvent rulefunction
RuleFunctions.RemoteTCPSender {
    attribute {
        validity = ACTION;
    }
    scope {
        String host;
        int port;
        String message;
    }
    body {
        String tcpNickName = "TCP-" + host + "-" + port + "-"
            + uri + "-" + closure + "-" + System.nanoTime();

        Events.RemoteMsgRequestEvent requestEvent =
            Events.RemoteMsgRequestEvent.RemoteMsgRequestEvent(null,
                message);

        TCP.connectToRemoteServer(tcpNickName, host, port);
        TCP.write(tcpNickName, requestEvent);
        TCP.endWrite(tcpNickName);

        Events.RemoteMsgResponseEvent responseEvent =
            TCP.readIntoPayloadFully(tcpNickName,
                "/Events/RemoteMsgResponseEvent");

        TCP.disconnectFromRemoteServer(tcpNickName);
        return responseEvent;
    }
}

```

APIs for TCP Communication

The Catalog Functions view lists the following functions (APIs) for TCP communication:

- `connectToRemoteServer()`
- `createLocalServer()`
- `disconnectFromRemoteServer()`
- `disconnectLocalSession()`
- `endRead()`
- `endWrite()`
- `readIntoPayload()`
- `readIntoPayloadFully()`
- `registerSessionListener()`
- `startLocalServer()`
- `stopLocalServer()`
- `write()`
- **Advanced**
 - `getReaderInputStream()`
 - `readIntoByteArray()`

Documentation for functions is provided in the tooltips you can see when browsing the functions in TIBCO BusinessEvents Studio. You can also see this documentation in the *TIBCO BusinessEvents Functions Reference*, available in the HTML product documentation.

Channel Resource Reference



Channels allow TIBCO BusinessEvents to listen to and send out messages. Channels contain destinations.

You can configure channels of different types, using the appropriate driver. See [Overview of Channels and Destinations on page 60](#).



Local channels are in memory; information in a local channel could be lost if the TIBCO BusinessEvents engine fails.

Wizard and Configuration Section

The Wizard and the Configuration section have the following fields.

Field	Global Var?	Description
Name	No	(Shown in the Wizard and then in the editor title only.) The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. See Identifiers (Names) on page 275 .
Description	No	Short description of the resource.
Driver	No	Select the driver for the type of channel you are configuring: <ul style="list-style-type: none">• TIBCO Rendezvous (see Working with Rendezvous Channels on page 63)• Local (see Working with Local Channels on page 66)• JMS (see Chapter 6, JMS Channels, on page 81)• HTTP (See Chapter 7, HTTP and SOAP Channels, on page 95)

Field	Global Var?	Description
Method of Configuration	No	<p>Resource Select Resource if you have a shared resource in your project whose properties you want to reuse for this channel.</p> <p>Note The path to the resource and the resource name cannot contain any of the words listed in Keywords and Other Reserved Words on page 280.</p> <p>Properties Select Properties to configure this channel resource using properties. See:</p> <p>Configuration for TIBCO Rendezvous Channels on page 74</p> <p>Configuration for JMS Channels on page 75</p> <p>Chapter 7, HTTP and SOAP Channels, on page 95</p>
Resource	No	<p>If you choose Resource as the method of configuration, the Resource field appears. Browse to and select the resource you want to use. As a convenience, you can open the selected resource by clicking the underlined label.</p>
Configuration for TIBCO Rendezvous Channels		
Service	Yes	<p>The name of the service or port number through which Rendezvous sends messages. In most cases you can leave this field empty, accepting the default value.</p> <p>For more information about the Rendezvous service parameter, see <i>TIBCO Rendezvous Concepts</i> or <i>TIBCO Rendezvous Administration</i>.</p> <p>Default is 7500 (defined in Global Variables).</p>
Network	Yes	<p>The network over which Rendezvous sends messages. In most cases you can leave this field empty. For more information about the network parameter, see <i>TIBCO Rendezvous Administration</i>.</p> <p>Default is an empty string (defined in Global Variables).</p>

Field	Global Var?	Description
Daemon	Yes	<p>The location of the Rendezvous daemon, which is usually expressed as a client socket number, for example, "6555." In most cases, you can leave this field empty, accepting the default value. For more information about the daemon parameter, see <i>TIBCO Rendezvous Concepts</i>.</p> <p>Default is <code>tcp:7500</code> (defined in Global Variables).</p>

Configuration for JMS Channels

ProviderURL	Yes	<p>The URL at which TIBCO BusinessEvents can contact the Enterprise Message Service server.</p> <p>Example: <code>tcp://localhost:7222</code></p>
UserName	Yes	A valid username for the Enterprise Message Service server.
Password	Yes	The password assigned to the username, above, for the purpose of accessing the Enterprise Message Service server.
IsTransacted	Yes	Accepts <code>true</code> or <code>false</code> . Specify <code>true</code> if the session has transaction semantics. Specify <code>false</code> if it has non-transaction semantics. For more information about the <code>IsTransacted</code> property, see TIBCO Enterprise Message Service documentation.
ClientID	Yes	The unique client ID of the connection.

Configuration for HTTP Channels

HTTP Server Type	No	<p>One of the following:</p> <ul style="list-style-type: none"> TOMCAT is a J2EE servlet container. Use TOMCAT if you need to host servlets. BUILT-IN is a good choice if the need is to send and receive many small requests. <p>See Selecting the Server Type on page 96.</p>
------------------	----	---

Destination Resource Reference



Within each channel, destinations direct incoming and outgoing information. A channel resource is not ready to use until it has at least one destination.

The Destinations section of a channel has the following fields.

Field	Global Var?	Description
Name	No	The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. See Identifiers (Names) on page 275 .
Description	No	Short description of the resource.
Default Event	No	<p>The event to be created from incoming messages unless otherwise specified. As a convenience, you can open the selected event resource by clicking the underlined label.</p> <p>Optional, but only if you always specify an event type in the incoming message.</p> <p>Not used for local channel.</p>
Serializer/ Deserializer	No	<p>Specify a serializer class to convert messages to simple events and simple events to messages. See the following sections for details:</p> <ul style="list-style-type: none">• Selecting a JMS Serializer on page 83• Working with Rendezvous Channels on page 63• Task D, Add a Destination, on page 100
Configuration for TIBCO Rendezvous Destinations		
See TIBCO Rendezvous documentation for more details on these settings.		
Subject	Yes	The TIBCO Rendezvous subject for incoming and outgoing messages.

Field	Global Var?	Description (Cont'd)
RVCM Pre Registration	Yes	For TIBCO Rendezvous certified message publishers, specify pre-registered listener names as a comma separated list.
LimitPolicy	Yes	<p>How you want the Rendezvous listener to behave when it receives more messages than the MaxEvents limit. Choose one of:</p> <p><code>Discard_None</code> (default)</p> <p><code>Discard_First</code></p> <p><code>Discard_Last</code></p> <p><code>Discard_New</code></p> <p>When MaxEvents or DiscardAmount are zero (unlimited), the LimitPolicy must be <code>Discard_None</code>.</p>
MaxEvents	No	<p>Maximum number of message events that the queue can hold.</p> <p>The default value, zero (0) means an unlimited number of events.</p>
DiscardAmount	No	<p>The number of events to discard when the queue exceeds its maximum event limit.</p> <p>The default value, zero (0) means events are never discarded.</p>
Configuration for JMS Destinations		
See TIBCO Enterprise Message Service documentation for more detail on these settings.		
Queue	No	Specifies whether the destination is a queue or a topic. Check the checkbox if the destination is a queue. If the destination is a topic, do not check it.

Field	Global Var?	Description (Cont'd)
Name	Yes	<p>Required. The name of the queue or topic.</p> <p>(TIBCO BusinessEvents ignores JMS destinations with null or empty-string queue or topic names. It logs an error message for the ignored destinations. If a JMS message is sent out through an ignored destination, TIBCO BusinessEvents throws an exception and the message is not sent out. TIBCO BusinessEvents also does not receive JMS messages (events) through these ignored destinations.)</p>
Selector	Yes	<p>Specifies a filter to pick up messages from the destination. This is a standard JMS selector based on SQL92 semantics.</p>
DeliveryMode	No	<p>The delivery mode property instructs the server concerning persistent storage for the message. Select one of the following:</p> <p>PERSISTENT (default)—In JMS message headers this is represented by the code 2.</p> <p>NON-PERSISTENT—In JMS message headers this is represented by the code 1.</p> <p>RELIABLE— This value is an extension to the standard, used in TIBCO Enterprise Message Service. In message headers this is represented by the code 22.</p> <p>You can also set a delivery mode in an event. See Using JMS Header Properties in Incoming and Outgoing Messages on page 90.</p>
AckMode	No	<p>The acknowledgement mode. See Changing the JMS Message Acknowledgment Mode on page 87 for a table explaining the various modes.</p> <p>You can also set an acknowledgement mode in a node's engine properties. The setting in the destination overrides the engine property setting.</p> <p>Default is <code>EXPLICIT_CLIENT_ACKNOWLEDGE</code></p>

Field	Global Var?	Description (Cont'd)
Priority	No	<p>The message priority. Takes a numerical value between 0 and 9. Larger numbers represent higher priority.</p> <p>You can also set a priority in an event. See Using JMS Header Properties in Incoming and Outgoing Messages on page 90.</p> <p>Default is 4</p>
TTL	No	<p>The length of time that the message will live (in milliseconds) before expiration. If set to 0, the message does not expire.</p> <p>You can also set a TTL (<code>JMSExpiration</code>) in an event. See Using JMS Header Properties in Incoming and Outgoing Messages on page 90.</p> <p>Default is 0</p>
Durable Subscriber Name	Yes	<p>For destinations that are JMS Topics, if you provide a DurableSubscriber Name, the destination becomes a JMS durable topic subscriber with the specified name. If you do not provide a value, the destination becomes a non-durable topic subscriber.</p> <p>The value of this property can be any unique string and can include any global variables. TIBCO BusinessEvents provides a set of case-sensitive variables that produce a unique DurableSubscriberName string. See Creating Unique JMS DurableSubscriber Name Properties on page 85 for details.</p> <p>Default is:</p> <pre>%%EngineName%%:%%SessionName%%:%%Channel URI%%:%%DestinationName%%</pre>

Configuration for Local Destinations

Local destinations don't use serializers, deserializers, or default events.

Size	No	The maximum number of events to be held in the queue. The default is zero (0), which allows unlimited events in the queue.
------	----	--

Field	Global Var?	Description (Cont'd)
TimeOut	No	Time to wait when sending an event to this local destination: -1 Waits indefinitely <ul style="list-style-type: none">Does not wait >0 Waits for the number of milliseconds specified Default is -1.

Configuration for HTTP Destinations

There are no specific fields for HTTP Destinations apart from the common ones.

Chapter 6 JMS Channels

See [Chapter 5, Channels and Destinations, on page 59](#) for basic information and procedures that apply to all types of channels. This chapter provides additional information about working with JMS channels.

Topics

- [Overview of JMS Channels, page 82](#)
- [Selecting a JMS Serializer, page 83](#)
- [Creating Unique JMS DurableSubscriber Name Properties, page 85](#)
- [Changing the JMS Message Acknowledgment Mode, page 87](#)
- [Using JMS Header Properties in Incoming and Outgoing Messages, page 90](#)
- [JMS Header Field Names, page 92](#)

Overview of JMS Channels

This chapter addresses tasks you may have to complete when configuring a JMS channel.

To understand how to choose a serializer to handle the JMS message types that will be sent to the destination, see [Selecting a JMS Serializer on page 83](#)

To understand how you can create unique DurableSubscriber Name property values for JMS Topic destinations, see [Creating Unique JMS DurableSubscriber Name Properties on page 85](#).

To learn how to change the JMS message acknowledgement type from the default type, see [Changing the JMS Message Acknowledgment Mode on page 87](#).

To understand how TIBCO BusinessEvents handles certain JMS header properties, and how you can work with these header properties, see [Using JMS Header Properties in Incoming and Outgoing Messages, page 90](#) and the related reference section, [JMS Header Field Names on page 92](#).

See Also

- Processing Units Tab — JMS Server Connection Properties in *TIBCO BusinessEvents Administration*, for details about run-time properties used to reconnect to a JMS server.
- The section, For TIBCO Enterprise Message Service and TIBCO Rendezvous Channels, in *TIBCO BusinessEvents Administration*, for TRA file updates required for JMS channels that use TIBCO Enterprise Message Service.

Selecting a JMS Serializer

This section explains the purpose of the provided JMS serializer classes. Choose the serializer that handles the JMS message types that will be sent to the destination you are configuring.

Table 16 Which Serializers to Use for JMS Message Types

Message Type	BytesMessageSerializer	TextMessageSerializer
Message	Supported	Supported
BytesMessage	Supported	(N/A)
MapMessage	Supported	Supported
ObjectMessage	Not Supported	Not Supported
StreamMessage	Not Supported	Not Supported
TextMessage	(N/A)	Supported

For `MapMessage` messages, you create properties whose names match the message keys.

Both serializers support reading and writing application header properties and JMS header properties. The difference between the serializers is in how they handle payloads. `BytesMessageSerializer` decodes the body as a sequence of bytes and parses them to create an XML structure according to the payload definition in the event. With `TextMessageSerializer`, the text from the message is decoded as an XML string.

See [Using JMS Header Properties in Incoming and Outgoing Messages on page 90](#).

BytesMessageSerializer

For incoming messages of type JMS `BytesMessage`, the serializer converts the message bodies to event payloads. The payloads are XML String type, but are not human-readable. However, you can access them using XPath functions.

For outgoing events, the serializer converts XML payloads to JMS `BytesMessage` message bodies.

The `BytesMessageSerializer` class is the default serializer.

TextMessageSerializer

For incoming messages, the `TextMessageSerializer` serializer converts JMS `TextMessage` messages to event payloads. The payloads are XML String type, and are human-readable. You can access them using XPath functions.

For outgoing events, the serializer converts XML payloads to JMS `TextMessage` messages.

Creating Unique JMS DurableSubscriber Name Properties

For destinations that are JMS Topics, if you provide a DurableSubscriber Name when you configure the destination resource, the destination becomes a JMS durable topic subscriber with the specified name. This section explains how you can ensure that the DurableSubscriber Name value is unique.



When using topic destination with a durable name in applications using In Memory OM and fault tolerance, do not provide a value for the Client ID setting and do not check the Auto-generate Client ID check box in the JMS shared resource.

Do not use durable topic destinations for multi-agent applications where more than one agent instance is active at a time. Instead, use queue destinations.

The value of the DurableSubscriber Name property can be any unique string and can include any global variables. TIBCO BusinessEvents provides a set of case-variables that produce a unique DurableSubscriberName string:

```
%%Deployment%%:%%EngineName%%:%%SessionName%%:%%ChannelURI%%:%%DestinationName%%
```

The first variable, %%Deployment%%, is a standard TIBCO global variable. The other three are only for use with the DurableSubscriberName property within TIBCO BusinessEvents. See [Table 17, Variables for Use with DurableSubscriberName](#), for details.



Do not attempt to use %%EngineName%%, %%SessionName%%, %%ChannelURI%%, or %%DestinationURI%% in any area of TIBCO BusinessEvents software except the DurableSubscriberName property.

Table 17 Variables for Use with DurableSubscriberName

Variable	Description
%%EngineName%%	The name of the TIBCO BusinessEvents engine. The name used is established using a series of checks. See Appendix A, Determining the Engine Name in <i>TIBCO BusinessEvents Administration</i> for details.
%%SessionName%%	The name of the agent class that is associated with the durable subscriber. Agent classes are defined in the CDD resource. See <i>TIBCO BusinessEvents Administration</i> for details about CDD configuration.
%%ChannelURI%%	The path to the channel within the TIBCO BusinessEvents project: <i>/ folder/ . . . / channel_name</i>

Table 17 Variables for Use with DurableSubscriberName (Cont'd)

Variable	Description
<code>%%DestinationName%%</code>	The name of the TIBCO BusinessEvents destination, within the channel specified in <code>%%ChannelURI%%</code> .

Changing the JMS Message Acknowledgment Mode

JMS channels support connection to TIBCO Enterprise Message Service destinations. The default acknowledgment mode is EXPLICIT.

You can set the acknowledgement mode either of these two ways:

- In the AckMode field of a destination resource.
- Using a configuration property.

The destination setting overrides the property setting. To define the acknowledgment mode using a configuration property, use one of the following, depending on whether you are using topics or queues:

```
be.channel.tibjms.topic.ack.mode acknowledgment_mode_number
```

```
be.channel.tibjms.queue.ack.mode acknowledgment_mode_number
```

Where *acknowledgment_mode_number* is one of the numbers that represent an acknowledgement mode, as shown in [Table 18](#).

Note that in TIBCO Enterprise Message Service, mode names are slightly different. They are prefixed with `TIBEMS-`. See *TIBCO Enterprise Message Service User's Guide* for more details.

Table 18 JMS Message Acknowledgement Modes

No.	Mode	Description
1	AUTO-ACKNOWLEDGE	Specifies that the session is to automatically acknowledge consumer receipt of messages when message processing has finished.
2	CLIENT_ACKNOWLEDGE	Specifies that the consumer is to acknowledge all messages that have been delivered so far by the session. When using this mode, it is possible for a consumer to fall behind in its message processing and build up a large number of unacknowledged messages. See Using CLIENT_ACKNOWLEDGE Mode with Websphere MQ and Cache-Aside on page 89 for required configuration for Websphere MQ when cache-aside database write strategy is used.

Table 18 JMS Message Acknowledgement Modes (Cont'd)

No.	Mode	Description
3	DUPS_OK_ACKNOWLEDGE	<p>Specifies that the session is to "lazily" acknowledge the delivery of messages to the consumer. "Lazy" means that the consumer can delay acknowledgement of messages to the server until a convenient time; meanwhile the server might redeliver messages. This mode reduces session overhead. However, should JMS fail, the consumer may receive duplicate messages.</p>
23	EXPLICIT_CLIENT_ACKNOWLEDGE (TIBCO Proprietary)	<p>TIBCO Enterprise Message Service extension to JMS acknowledge modes.</p> <p>This is the default.</p> <p>EXPLICIT_CLIENT_ACKNOWLEDGE is like CLIENT_ACKNOWLEDGE except it acknowledges only the individual message, rather than all messages received so far on the session.</p> <p>One example of when EXPLICIT_CLIENT_ACKNOWLEDGE would be used is when receiving messages and putting the information in a database. If the database insert operation is slow, you may want to use multiple application threads all doing simultaneous inserts. As each thread finishes its insert, it can use EXPLICIT_CLIENT_ACKNOWLEDGE to acknowledge only the message that it is currently working on.</p>
24	EXPLICIT_CLIENT_DUPS_OK_ACKNOWLEDGE (TIBCO Proprietary)	<p>TIBCO Enterprise Message Service extension to JMS acknowledge modes.</p> <p>EXPLICIT_CLIENT_DUPS_OK_ACKNOWLEDGE mode is like TIBEMS-DUPS-OK-ACKNOWLEDGE except it "lazily" acknowledges only the individual message, rather than all messages received so far on the session.</p>

Table 18 JMS Message Acknowledgement Modes (Cont'd)

No.	Mode	Description
22	NO-ACKNOWLEDGE (TIBCO Proprietary)	<p>TIBCO Enterprise Message Service extension to JMS acknowledge modes.</p> <p>Suppresses the acknowledgement of received messages. After the server sends a message to the client, all information regarding that message for that consumer is eliminated from the server. Therefore, there is no need for the client application to send an acknowledgement to the server about the received message. Not sending acknowledgements decreases the message traffic and saves time for the receiver, therefore allowing better utilization of system resources.</p> <p>Note Sessions created in no-acknowledge receipt mode cannot be used to create durable subscribers.</p> <p>Note Also, queue receivers on a queue that is routed from another server are not permitted to specify NO-ACKNOWLEDGE mode.</p>

Using CLIENT_ACKNOWLEDGE Mode with Websphere MQ and Cache-Aside

The cache-aside database write strategy is multi-threaded. However, when Websphere MQ messages are sent using `CLIENT_ACKNOWLEDGE_MODE`, each message must be handled from start to finish using a single thread. To address this issue do the following:

1. Define the destination using Caller in the Threading Model setting (in the CDD Collections tab or Agent Classes tab).
See Collections Tab — Destinations Settings and Properties in *TIBCO BusinessEvents Administration*.
2. To override the cache-aside write behavior, set the following property in the CDD file at the appropriate level:

```
Agent.agentClassName.enableParallelOps=false
```

Setting this property to false means that all post-RTC operations are done on a single thread.

Using JMS Header Properties in Incoming and Outgoing Messages

Information in this section assumes you are familiar with JMS and its header properties. Consult your JMS provider documentation for information. This section explains only how TIBCO BusinessEvents supports use of these properties.

Setting Certain Header Properties in Destinations

In the JMS destination Configuration section, you configure the following three header properties:

- `DeliveryMode` (`JMSDeliveryMode`)
- `Priority` (`JMSPriority`)
- `TTL` (`JMSExpiration`)



JMS header properties defined in events take precedence over properties defined in destinations.

Setting Header Properties Using Header Properties from Incoming JMS Messages

You can configure events created from incoming JMS messages to have properties that match the JMS header properties. You can then use those event properties to set JMS header properties in outgoing messages.

These event properties must match the JMS header fields. You must use the names as shown in [Table 19, JMS Header Field Names, on page 92](#). You only have to configure event properties for those fields that you want to use. Incoming JMS message header properties will then populate the corresponding TIBCO BusinessEvents event properties.

Setting JMS Header Properties in Outgoing JMS Messages Using Event Properties

Similarly outgoing JMS message header properties will be populated by the corresponding TIBCO BusinessEvents event properties.

Note that the `JMSMessageID` and `JMSTimeStamp` properties are generated when the message is sent. You can't set these properties manually.

See [How TIBCO BusinessEvents Sets the JMSReplyTo Header in Outgoing Messages on page 91](#) for special handling of the `JMSReplyTo` header.

JMS header properties defined in events take precedence over properties defined in destinations.



You can add the JMS properties to the Base event in your project so that the properties are inherited by all other events.

See [Table 19, JMS Header Field Names, on page 92](#) for details on all properties.

How TIBCO BusinessEvents Sets the JMSReplyTo Header in Outgoing Messages



TIBCO BusinessEvents cannot act as a client in a JMS request-response scenario because TIBCO BusinessEvents currently cannot dynamically create a destination to listen for JMS messages.

If an event has a string type property named `JMSReplyTo` (case sensitive), TIBCO BusinessEvents reads this event property value as a JMS queue or topic name, according to the event's default destination type. TIBCO BusinessEvents looks up the `javax.jms.Destination` on the connected JMS server using this queue or topic name. If TIBCO BusinessEvents cannot find one, it creates a new `javax.jms.Destination` using the given queue or topic name. TIBCO BusinessEvents then sets the `JMSReplyTo` header property of the outgoing JMS message using this destination



If you use the catalog function `Event.replyEvent(requestEvent, replyEvent)` during the RTC in which the *requestevent* is received, then the *replyevent* is sent to the destination in the `JMSReplyTo` header property of the JMS message associated with the *requestevent*.

JMS Header Field Names

The table below shows the names you must use to define event properties corresponding to JMS header field names, as well as some details about the purpose of each property. The property names are not case sensitive.

Table 19 JMS Header Field Names

Field name	Type	Description
JMSDestination	Object	The destination (queue or topic) to which the message is sent.
JMSDeliveryMode	Integer	<p>The delivery mode specified by the sender. This property instructs the server concerning persistent storage for the message. Value can be:</p> <p>2—interpreted as PERSISTENT (default).</p> <p>1—interpreted as NON-PERSISTENT.</p> <p>22—interpreted as RELIABLE. This value is an extension to the standard used in TIBCO Enterprise Message Service.</p> <p>The integer values are interpreted as the text names of delivery modes.</p> <p>You can also set a delivery mode for a destination. See Setting Certain Header Properties in Destinations on page 90.</p>
JMSExpiration	Long	<p>The length of time that the message will live (in milliseconds) before expiration. If set to 0, the message does not expire.</p> <p>You can also set an expiration (TTL) for a destination. See Setting Certain Header Properties in Destinations on page 90.</p>
JMSPriority	Integer	<p>The message priority, a numerical value between 0 and 9. Larger numbers represent higher priority.</p> <p>You can also set a priority for a destination. See Setting Certain Header Properties in Destinations on page 90.</p>

Table 19 JMS Header Field Names (Cont'd)

Field name	Type	Description
JMSMessageID	String	<p>An ID that uniquely identifies each message sent by a provider.</p> <p>A generated value overrides any value set in the corresponding event property.</p>
JMSTimestamp	Long	<p>The time when the message was handed off to a provider to be sent. The message may be sent later than this timestamp value.</p> <p>A generated value overrides any value set in the corresponding event property.</p>
JMSCorrelationID	String	<p>A correlation ID that can be used to link messages. For example, you can link a response message to a request message. Optional.</p>
JMSReplyTo	String	<p>Name of the JMS destination (queue or topic) to send the message reply to. If null, TIBCO BusinessEvents does not set the outgoing message's property.</p> <p>Optional.</p> <p>Note: Do not set the value to an empty string (" "). If you do, TIBCO BusinessEvents sets the queue or topic name to an empty string which creates an exception, and the message is not sent.</p> <p>See How TIBCO BusinessEvents Sets the JMSReplyTo Header in Outgoing Messages on page 91.</p>
JMSType	String	<p>The message type identifier, if used by the provider.</p>
JMSRedelivered	Boolean	<p>If this field is set, it is possible that the message was delivered to the client earlier, but not acknowledged at that time.</p>

Chapter 7 HTTP and SOAP Channels

See [Chapter 5, Channels and Destinations, on page 59](#) for basic information and procedures that apply to all types of channels. This chapter provides additional information about working with HTTP channels, including those configured to serve SOAP messages.

Topics

- [Overview of HTTP and SOAP Channels, page 96](#)
- [Working with HTTP Requests, page 97](#)
- [HTTP Channel Advanced Configuration Settings, page 102](#)
- [Using HTTP Functions to Configure HTTP Request Messages, page 110](#)
- [Configuring TIBCO BusinessEvents as a SOAP Server and Client, page 116](#)
- [Parsing and Building SOAP Messages, page 123](#)
- [Understanding the WSDL to Project Resource Mapping, page 128](#)

Overview of HTTP and SOAP Channels

An HTTP channel is an internal HTTP server. When the TIBCO BusinessEvents engine starts, it starts the internal HTTP server, which listens to the requests on the port specified in the HTTP Connection resource.

Selecting the Server Type

TIBCO BusinessEvents uses a built-in server to serve HTTP requests by default. You can also use a TOMCAT server integrated with TIBCO BusinessEvents.

SOAP Support

SOAP version 1.1 is supported. TIBCO BusinessEvents supports only document/literal type of encoding of SOAP requests.

A SOAP event is an extension of a SimpleEvent. To create a SOAP event, you create a SimpleEvent that inherits from a **SOAPEvent**. This creates a default schema in the event payload. Then you edit the schema and introduce header and body elements as necessary.

Using an HTTP channel and a destination configured to use the SOAP serializer and deserializer, TIBCO BusinessEvents can act as a web services platform that sends and receives SOAP requests, and performs whatever operations are provided by the web service.

TIBCO BusinessEvents can import a WSDL file and create the required project artifacts based on it, such as events, rules, rule functions, channels, and destinations. For more details, see [Creating Resources Using the WSDL Import Utility on page 119](#).

TIBCO BusinessEvents can also export a SOAP based rule function to a WSDL. The export utility scans the project for rule functions that take a SOAP event as the input, and generates a WSDL operation for each one. For more details, see [Exporting Resources as a WSDL File on page 120](#).

See [Configuring TIBCO BusinessEvents as a SOAP Server and Client on page 116](#) for more details.

Working with HTTP Requests

This section explains how an HTTP request is mapped to an event, and how to add an HTTP channel and destination. It assumes basic familiarity with HTTP and with the procedure of setting up a channel. See [Adding Channels and Destinations on page 67](#) if you need basic instructions.

Mapping of HTTP Requests to Events

`RESTMessageSerializer`, which is set while configuring the channel, maps HTTP requests to TIBCO BusinessEvents. HTTP headers and HTTP parameters in the GET method are mapped to similarly named event properties. When both parameters and headers are specified, parameters take precedence.

Each HTTP Header consists of a name and field value, which are mapped into an event property name and value.

The POST data in the request must match with the payload as defined in the corresponding event. If there is no payload defined for the event, the POST data is translated into a `ByteArray` payload, and is not accessible via the mapper. In other words, only XML payloads are visible via the mapper.

The transfer encoding (`charset`) in the `Content-Type` header indicates what type of transformation is applied to the POST data (message body) to safely transfer content between sender and recipient. If the `Content-Type` header is missing, UTF-8 is used as the default transfer encoding.

When you want the REST Serializer to deserialize a GET request into an event with a payload, include the `_payload_` request parameter. The string value of the `_payload_` parameter will always be used as payload in the event.



Avoid sending long requests using GET. For large payload data requests, it is recommended to use the POST method.

For HTTP responses, all event properties are translated to similarly named HTTP headers and the payload is sent as HTTP content.

Sending Non-ASCII Content to Event Properties

The HTTP/1.1 specification states that only ASCII characters can be sent in HTTP headers.

To send non-ASCII event properties in GET methods, use HTTP parameters. HTTP parameters are passed as the `QueryString` of the request URI, that is, the

part of the URI that contains data to be passed to web applications.

To decode the `QueryString`, use either the URI Encoding or the body encoding. The body encoding is specified in the `contentType` HTTP header.

Select either the URI Encoding or Use Body Encoding for URI setting in the HTTP channel Advanced tab.

Mapping of HTTP Request URI to Destination

An HTTP request URI must map to a valid TIBCO BusinessEvents destination. If not, an error is returned, and the message is discarded. Once the destination is established, the HTTP message is converted either into an event based upon `_ns_/_nm_`, or into the default event associated with this destination.

In this case TIBCO BusinessEvents server looks for the destination having the same URI as the `requestURI`.

For example, the `requestURI` for the request

`https://localhost:7000/Transport/Channel/StudentDestination` is `/Transport/Channel/StudentDestination`. TIBCO BusinessEvents engine maps the request with a destination having URI `/Transport/Channel/StudentDestination` if it exists.

Configuring TIBCO BusinessEvents to Receive and Send HTTP Requests

To configure TIBCO BusinessEvents to receive and send HTTP requests, you need to perform the following tasks:

Task A Add an HTTP Connection

Add an HTTP Connection resource to your project. In the Host and Port fields, specify the host and port to which HTTP clients send requests.

In the Host field, enter the name or IP address of the machine running TIBCO BusinessEvents. This is the HTTP server.

In the Port field, enter any available port on the host machine. This is the port on which the server listens for HTTP requests.

To configure an HTTPS (Secure) Connection Check the **SSL** checkbox, click the **Configure SSL** button, and complete the pop-up dialog settings. The server must authenticate to the client. In the Identity field, provide the location of the Server Identity File. (See [Task B](#)).



In one-way SSL, the server authenticates itself to the client using a Server Identity File, but the client does not have to authenticate to the server.

In two-way SSL, the server authenticates itself to the client using a Server Identity File and the client also authenticates to the server using the Client Identity file.

Check the Requires Client Authentication checkbox. This enables the Trusted Certificates Folder field, where you provide the Client certificates.

Task B For SSL Only—Add an Identity Resource

TIBCO BusinessEvents supports use of an identity file for SSL. Before you configure the secure HTTP connection, add an Identity resource to your project and configure it.

In the URL field, specify the project path of the keystore file, which must be within the project folders.

In the File Type field, specify the keystore file type, and in the Password field, provide the password for the keystore file.

Task C Add an HTTP Channel

Add a channel to your project and configure it as follows:

1. At the New Channel Wizard, provide a name and description, and in the Driver field select HTTP. Click **Finish**.
2. In the Channel editor Channel tab, update the description as desired. The Driver field is set to HTTP (as set in the wizard). The Method of Configuration is preset to Resource and can't be changed.
3. In the serverType field select **BUILT-IN** or **TOMCAT**, as appropriate.
4. In the Resource field, browse to the HTTP connection resource you configured in [Task A](#).
5. Click **Advanced**, and configure run-time configuration properties. These properties provide information such as the location of the document root folder. Some properties are for SSL configuration of HTTP Component servers. See [HTTP Channel Advanced Configuration Settings on page 102](#).

Task D Add a Destination

Add a destination to the channel in the usual way. (See [Adding Channels and Destinations on page 67](#) for details.) In the **Serializer** field select the appropriate serializer:

```
com.tibco.cep.driver.http.serializer.RESTMessageSerializer
```

Specify the default event in the usual way as needed by your project requirements.



HTTP clients of the TIBCO BusinessEvents server would use the complete destination URI, after the host and port for example,

```
http://www.acme.com:5560/Transport/Channel/MyDestination.
```

See [Mapping of HTTP Request URI to Destination on page 98](#).

Task E Create Events as Needed and Set Default Destinations

For receiving HTTP requests and sending responses, configure events in the usual way, and select an HTTP-based destination as the default destination.

Task F Configure Rules and Rule Functions

Configure rules and rule functions according to your needs.

For example, in response to a POST request you might do the following:

- Create a concept instance, using XPath functions to extract data from the POST data in the request event payload
- Create a response event and use `Event.replyEvent()` to send back an empty response using the request event's default destination.

As another example, in response to a GET request you might do the following:

- Identify a concept instance using a property in the request event (created on arrival of the GET request message).
- As needed, identify or generate data to return and create a response event to hold that data.
- Return the data using `Event.replyEvent`.

Task G In the CDD, Configure the Processing Unit

In the Cluster Deployment Descriptor (CDD) Configure the processing unit for deployment as needed (see [Chapter 3, CDD Configuration Procedures, on page 7](#) for details). Note the following requirement.



Workers Setting Only **Callers** thread option is supported for TOMCAT based channels. This is because when a client application sends a request, TOMCAT spawns a thread to handle it. If the response is not returned on the *same* caller thread, then this thread returns with an empty response.

HTTP Channel Advanced Configuration Settings

The following settings configure the internal HTTP server used by the channel. They are set in the Advanced tab of the HTTP channel resource editor.

Table 20 HTTP Channel Advanced Configuration Settings (Sheet 1 of 7)

Property	Notes
Settings Common to TOMCAT and BUILT-IN Server Types	
serverType	<p>Displays the server type selected in the Channel tab: BUILT-IN or TOMCAT. Configuration options differ in each case, as shown in this table.</p>
Connection Timeout (msec)	<p>The number of milliseconds the HTTP server waits after accepting a connection, for the request URI line (that is, the first part of the request message) to be presented.</p> <p>The value 0 means no timeout.</p> <p>The default is 60000.</p>
Accept Count	<p>The maximum queue length for incoming connection requests when all request processing threads are in use. Any requests received when the queue is full are refused.</p> <p>The value -1 means that a default of 10 is set.</p> <p>The default is -1.</p>
Socket Output Buffer Size	<p>The size in bytes of the buffer to be provided for socket output buffering.</p> <p>The value -1 means that the use of a buffer is disabled.</p> <p>The default is 9000.</p>

Table 20 HTTP Channel Advanced Configuration Settings (Sheet 2 of 7)

Property	Notes
Max Processors	<p>The maximum number of simultaneous requests that can be handled by the HTTP server.</p> <p>For the BUILT-IN server, set to a positive integer value. If no value is specified, then the BUILT-IN server handles <i>n</i> requests simultaneously, where <i>n</i> is less than the maximum number of open file descriptors (sockets) for the server socket. This number depends on the operating system.</p> <p>For TOMCAT, set to a value of 10 or greater. Values of less than 10 are treated as 10.</p> <p>The value -1 means that the following default values are used:</p> <p>For TOMCAT: 200</p> <p>For BUILT-IN: 5</p> <p>The default is -1.</p>
TCP No Delay	<p>If the checkbox is checked, the <code>TCP_NO_DELAY</code> option is not set on the server socket.</p> <p>If the checkbox is checked, the <code>TCP_NO_DELAY</code> option is set on the server socket. Using <code>TCP_NO_DELAY</code> improves performance under most circumstances.</p> <p>The default is checked.</p>
Use Body Encoding for URI	<p>If checked, the encoding specified in the <code>contentType</code> HTTP header is used to decode the request URI.</p> <p>If unchecked, the value in URI Encoding field (or its default value) is used.</p> <p>If the URI Encoding setting is specified, then the Use Body Encoding for URI checkbox is ignored.</p> <p>See Sending Non-ASCII Content to Event Properties on page 97 for more details.</p> <p>For an example showing <code>contentType</code>, see Working with Outgoing SOAP Messages (Event Payloads) on page 126.</p> <p>The default is unchecked.</p>

Table 20 HTTP Channel Advanced Configuration Settings (Sheet 3 of 7)

Property	Notes
URI Encoding	<p>Specifies the character encoding used to decode the request URI. If not specified, UTF-8 is used.</p> <p>If specified, the Use Body Encoding for URI setting is ignored.</p> <p>The default is UTF-8.</p> <p>See Sending Non-ASCII Content to Event Properties on page 97 for more details.</p>
Document Root	<p>The absolute path where static HTML files are stored. The HTTP server retrieves pages from this location. This setting is used by internal TIBCO BusinessEvents applications and may have limited application for other purposes.</p> <p>No default value.</p>
Document Page	<p>The name of the default static HTML file stored in the document root. This setting is used by internal TIBCO BusinessEvents applications and may have limited application for other purposes.</p>
TOMCAT Server Type Settings	
Connection Linger	<p>Note Displays only for TOMCAT Server Type but not used in this release and not specific to this server type.</p> <p>The number of milliseconds during which the sockets used by the HTTP server linger (that is, not complete immediately) when they are closed. Use of socket linger allows time for a graceful shutdown sequence to complete.</p> <p>To disable use of socket linger, set to -1.</p> <p>The default is -1.</p>

Table 20 HTTP Channel Advanced Configuration Settings (Sheet 4 of 7)

Property	Notes
Enable DNS Lookups	<p>Set to <code>true</code> if you want the calls to <code>request.getRemoteHost()</code> to perform DNS lookups and return the actual host name of the remote client. Set to <code>false</code> to skip the DNS lookup and return the IP address as a string instead (thereby improving performance).</p> <p>By default, DNS lookups are disabled.</p>
Compression	<p>Compression can be used to save server bandwidth. Uses <code>HTTP/1.1 GZIP</code> compression.</p> <p>The compression option set here is used together with the MIME types shown in the Compressible Mime Types setting. Allowable values are:</p> <p>off Disables compression. Values in the Compressible Mime Types setting are ignored.</p> <p>on Enables compression</p> <p>force Forces compression.</p> <p>An integer Specifies a threshold amount of data above which output is compressed. The unit is bytes. For example, if set to <code>2048</code> then any file above 2MB will be compressed. If content length is unknown, the output is always compressed. Compression applies to MIME types shown in the Compressible Mime Types setting that are over the threshold (or unknown).</p> <p>The default is off.</p>
Max KeepAlive Requests	<p>The maximum number of HTTP requests that can be pipelined until the connection is closed by the server.</p> <p>A value of 1 disables <code>HTTP/1.0 keep-alive</code>, as well as <code>HTTP/1.1 keep-alive</code> and pipelining.</p> <p>A value of -1 allows an unlimited amount of pipelined or keep-alive HTTP requests.</p> <p>The default value is -1.</p>

Table 20 HTTP Channel Advanced Configuration Settings (Sheet 5 of 7)

Property	Notes
Max HTTP Header Size	<p>The maximum size of the request and response HTTP header, specified in bytes.</p> <p>The default is 4096, that is 4 KB.</p>
Max HTTP Post Size	<p>The maximum size of the POST data, specified in bytes, which is handled by the container FORM URL parameter parsing. You can disable the limit by setting this to less than or equal to 0.</p> <p>The default is 2097152, that is, 2 MB.</p>
Max HTTP Save Post Size	<p>The maximum size of the POST data, specified in bytes, which is saved or buffered by the container during FORM or CLIENT-CERT authentication. For both types of authentication, the POST data is saved or buffered before the user is authenticated.</p> <p>For CLIENT-CERT authentication, the POST data is buffered for the duration of the SSL handshake and the buffer emptied when the request is processed.</p> <p>For FORM authentication the POST is saved while the user is redirected to the login form and is retained until the user successfully authenticates or the session associated with the authentication request expires.</p> <p>A value of -1 means no limit.</p> <p>A value of 0 disables the saving of POST data during authentication.</p> <p>The default is 4096, that is 4 KB.</p>
Protocol	<p>Specifies the HTTP protocol to use. Options are as follows:</p> <p><code>http</code> HTTP protocol.</p> <p><code>https</code> For secure communications (SSL).</p> <p><code>memory</code> Memory protocol.</p> <p><code>ajp</code> Apache JServ Protocol, a binary protocol.</p> <p>Default is <code>http</code>.</p>

Table 20 HTTP Channel Advanced Configuration Settings (Sheet 6 of 7)

Property	Notes
Max Spare Threads	<p>The maximum number of unused request processing threads that are allowed to exist until the thread pool starts stopping the unnecessary threads.</p> <p>The default is 50.</p>
Min Spare Threads	<p>The number of request processing threads that are created when this <code>Connector</code> is first started. The connector also makes sure that it has the specified number of idle processing threads available. Set this to a value smaller than that set for Max Spare Threads.</p> <p>The default is 4.</p>
Compressible Mime Types	<p>This is a comma-separated list of MIME types for which HTTP compression may be used.</p> <p>This setting works with the Compression setting See notes for that setting for more details.</p> <p>The default value is <code>text/html,text/xml,text/plain</code>.</p>
Restricted User Agents	<p>Used to limit support to specific browsers and versions of specific browsers. Comma-separated list containing one or more browser names, such as Mozilla, Internet Explorer. Prepend version with a slash, for example, <code>Mozilla/4.0</code>.</p>
BUILT-IN Server Type Settings	
Stale Connection Check	<p>Determines whether to perform a stale connection check or not. Checked means enabled and unchecked means disabled.</p> <p>Disabling the stale connection check may result in slight performance improvement at a risk of getting an I/O error when executing a request over a connection that has been closed at the server side.</p> <p>The default is unchecked.</p>

Table 20 HTTP Channel Advanced Configuration Settings (Sheet 7 of 7)

Property	Notes
SSL Server: Key Manager Algorithm	<p>The key manager algorithm for the SSL Service provider.</p> <p>The default is SunX509.</p>
SSL Server: Trust Manager Algorithm	<p>The trust manager algorithm for the SSL Service provider.</p> <p>Ensure that a provider is available for any algorithm other than PKIX.</p> <p>The default is PKIX.</p>
SSL Server: Protocols	<p>The SSL protocols that can be enabled on the server. Add each protocol using a comma-separated list.</p> <p>The default protocols are the ones supported by the SSL Provider. (SSLv3 and TLSv1 are the most widely supported.)</p> <p>Leave blank to use the default protocols for your SSL provider.</p>
SSL Server: Ciphers	<p>The cipher suites (SSL protocols) used by the server. Add each suite name on a separate line.</p> <p>The following cipher suites are supported:</p> <pre>SSL_RSA_WITH_RC4_128_MD5 SSL_RSA_WITH_RC4_128_SHA TLS_RSA_WITH_AES_128_CBC_SHA TLS_DHE_RSA_WITH_AES_128_CBC_SHA TLS_DHE_DSS_WITH_AES_128_CBC_SHA SSL_RSA_WITH_3DES_EDE_CBC_SHA SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA SSL_RSA_WITH_DES_CBC_SHA SSL_DHE_RSA_WITH_DES_CBC_SHA SSL_DHE_DSS_WITH_DES_CBC_SHA SSL_RSA_EXPORT_WITH_RC4_40_MD5 SSL_RSA_EXPORT_WITH_DES40_CBC_SHA SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA SSL_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA</pre> <p>Leave blank to use the default cipher suites for your SSL provider.</p>

Defining Event Properties for Standard HTTP Header Properties

You need to define corresponding event properties for HTTP header names so that the header names are mapped to the event properties at run time.

Standard HTTP header properties can have a dash (-) in their names. While defining the corresponding event properties for such header properties, use an underscore character (_) instead of a dash(-). A dash is *not* allowed in the event property names.

Following is the list of the HTTP header properties. In this a dash is replaced by an underscore while converting from HTTP headers to Event Properties, and vice-versa.

<code>accept-charset</code>	<code>max-forwards</code>	<code>proxy-authenticate</code>
<code>accept-encoding</code>	<code>proxy-authorization</code>	<code>retry-after</code>
<code>accept-language</code>	<code>user-agent</code>	<code>set-cookie</code>
<code>accept-ranges</code>	<code>content-encoding</code>	<code>transfer-encoding</code>
<code>cache-control</code>	<code>content-disposition</code>	<code>proxy-authenticate</code>
<code>content-type</code>	<code>content-language</code>	<code>retry-after</code>
<code>if-match</code>	<code>content-location</code>	<code>set-cookie</code>
<code>if-modified-since</code>	<code>content-md5</code>	<code>transfer-encoding</code>
<code>if-none-match</code>	<code>content-range</code>	<code>www-authenticate</code>
<code>if-unmodified-since</code>	<code>last-modified</code>	

Using HTTP Functions to Configure HTTP Request Messages

This section explains how HTTP functions are used to send secure and non-secure HTTP requests to other servers, and work with responses received.

See [HTTP Channel Advanced Configuration Settings on page 102](#) for details about setting up TIBCO BusinessEvents as a secure server.

Generating a Self-Signed SSL Certificate (Keystore)

HTTPS requires use of an SSL certificate. It creates a keystore file and password for the specified type of SSL certificate (keystore).



It is not recommended for production to use self-signed certificates.

If you use trusted certificates, see [Loading Trusted Certificates on page 111](#).

Signature

Object createKeystore(String *ksFilePath*, String *ksType*, String *ksPassword*)

Description

Creates and returns a keystore object, using the given parameters.

Parameters

Name	Type	Description
ksFilePath	String	The absolute path of the keystore file.
ksType	String	The type of keystore. JKS and PKCS12 are supported.
ksPassword	String	Obfuscated password for the keystore.

Returns

Type	Description
Object	The keystore object

Getting POST Data

Signature	Object getPostData(SimpleEvent <i>event</i>)
Description	Returns the POST data sent in an HTTP POST request (event).

Parameters	Name	Type	Description
	<code>event</code>	<code>SimpleEvent</code>	Request event sent on the HTTP destination. Must not be null.
Returns	Type	Description	
	<code>Object</code>	The POST data	

Loading Trusted Certificates

As explained in [Task A, Add an HTTP Connection, on page 98](#), ensure that certificates from trusted certificate authorities are stored in the project, using an Identity resource in the TIBCO Shared Resources folder. Alternatively use the `BE_GLOBAL_TRUSTED_CA_STORE` global variable to store a location of the certificates outside of the project.

Signature	<code>Object loadTrustedCertificates(String trustedCertsFolder, String passwordToSet)</code>		
Description	HTTPS requires use of an SSL certificate. This function loads a trusted certificate (that is, creates and returns a keystore object) from the trusted certificates folder.		
Parameters	Name	Type	Description
	<code>trustedCertsFolder</code>	<code>String</code>	The project path to the folder containing the certificates. If trusted certificates are stored outside the project, use the following construct in the function: <code>System.getGlobalVariableAsString("BE_GLOBAL_TRUSTED_CA_STORE")</code>
	<code>passwordToSet</code>	<code>String</code>	Obfuscated password for the keystore.
Returns	Type	Description	
	<code>Object</code>	The certificate's keystore object	

Sending an Event

To send an event as a request encapsulating request headers as properties.

Signature

Event sendRequest (String url, SimpleEvent requestEvent, String responseEventURI, long timeoutMillis)

Description

This function sends an event as a response to the request.

Parameters

Name	Type	Description
url	String	The URL for the endpoint that will receive this request.
requestEvent	SimpleEvent	The event to serialize and send to the server.
responseEventURI	String	The fully-qualified path of an event. This event is created when the response is received.
timeoutMillis	Long	The timeout interval for the operation. If the value is -1, the server waits forever.

Returns

Type	Description
Event	An event as a response to the request

Sending an Asynchronous Request (Not Secure)

To send requests to an HTTP server asynchronously, use this function.

Signature

String sendAsynchronousRequest(String url, SimpleEvent requestEvent, String correlationId, String callbackRuleFunctionURL, String methodType)

Description

Sends a request to the server specified by the url parameter. When TIBCO BusinessEvents receives a response, the callback function is called.

Returns a correlation ID, which is either passed as input, or is generated from the server if the parameter is null. This ID enables you to correlate a request with its response.

Parameters

Name	Type	Description
url	String	The URL for the server that will receive this request.
requestEvent	SimpleEvent	The event to serialize and send to the server.

Name	Type	Description
<code>correlationID</code>	String	An optional ID to correlate the request and the response. If not specified, the ID is generated by the server.
<code>callbackRuleFunctionURL</code>	String	The fully-qualified path of a rule function. This rule function is called when the response is received. The response event would contain the correlation ID. The rule function must have correlation ID, RequestEvent, and ResponseEvent as parameters.
<code>methodType</code>	String	The HTTP method type. Valid values are: GET or POST.

Returns

Type	Description
String	A correlation ID

Sending a Secure Asynchronous Request

To send asynchronous requests using SSL, use this function.

Signature

```
String sendSecureAsynchronousRequest(String url, SimpleEvent requestEvent, String correlationID, String callbackRuleFunctionURL, String methodType, Object clientIdKeystore, String clientIdPassword, Object trustedCertsKeystore, String trustedCertsPassword boolean verifyHostName)
```

Description

This function is the same as the `sendAsynchronousRequest()` function, with the addition of the SSL-related parameters, shown below.

Parameters

All of the parameters for `sendAsynchronousRequest()` plus the following:

Name	Type	Description
<code>clientIdKeystore</code>	Object	The keystore object for client identity.
<code>clientIdPassword</code>	String	Password for the client ID keystore.

Name	Type	Description
<code>trustedCertsKeystore</code>	Object	Keystore Object for trusted certificates.
<code>trustedCertsPassword</code>	String	Password for the trusted certificates keystore.
<code>verifyHostName</code>	Boolean	Flag for checking if a host name matches the names stored in the server's certificates.

Returns

Type	Description
String	A correlation ID

Sending a Secure Synchronous Request

To send synchronous requests using SSL, use this function.

Signature `Event sendSecureRequest(String url, SimpleEvent requestEvent, String responseEventURI, Object clientIdKeystore, String clientIdPassword, Object trustedCertsKeystore, String trustedCertsPassword, boolean verifyHostName, long timeoutMillis)`

Description This function is the same as the `sendSecureAsynchronousRequest()` function, except that the requests are synchronous. The parameters are shown below.

Parameters

Name	Type	Description
<code>url</code>	String	The URL for the server that will receive this request.
<code>requestEvent</code>	SimpleEvent	The event to serialize and send to the server.
<code>responseEventURI</code>	String	The fully-qualified path of an event. This event is created when the response is received.
<code>clientIdKeystore</code>	Object	The keystore object for client identity.
<code>clientIdPassword</code>	String	Password for the client ID keystore.

Name	Type	Description
<code>trustedCertsKeystore</code>	Object	Keystore Object for trusted certificates.
<code>trustedCertsPassword</code>	String	Password for the trusted certificates keystore.
<code>verifyHostName</code>	Boolean	Flag for checking if a host name matches the names stored in the server's certificates.
<code>timeoutMillis</code>	Long	The timeout interval for the operation. If the value is -1, the server waits forever.

Returns

Type	Description
Event	An event as a response to the request

Configuring TIBCO BusinessEvents as a SOAP Server and Client

Using an HTTP channel and a destination configured to use a SOAP Serializer, TIBCO BusinessEvents can act as a web services platform, sending and receiving SOAP requests. You can configure the project manually or using a WSDL import utility, which greatly simplifies configuration.

This release supports SOAP version 1.1. Some understanding of SOAP protocol is required in order to work with this feature. See <http://www.w3.org/TR/soap/> for details.

Overview of SOAP Related Resources

The following project resources must be configured so that TIBCO BusinessEvents can send and receive SOAP messages.

SOAP Destinations

When a correctly configured HTTP destination receives a SOAP request, the SOAP serializer deserializes the SOAP message to its corresponding event. That event has to be inherited from a `SOAPEvent`. The event payload contains the SOAP envelope.

SOAP Events

To create a SOAP event, create a `SimpleEvent` that inherits a `SOAPEvent`. It makes event configuration easier. Its payload has a `message` root element having an `Envelope` child element. The root element contains `Header` and `Body` elements, and the `Body` element has a `Fault` element. You can further configure these elements using the payload editor.

SOAP messages (and events) can have attachments.

SOAP Encoding

Only the SOAP document/literal form of encoding is supported.

Rules and Rule Functions

A rule in the project accesses the SOAP event payload as needed using the mapper or SOAP catalog functions. The rule puts the SOAP response into the payload of a `SOAPEvent` that is sent to the client. The SOAP serializer sends the SOAP response to the client.

SOAP Catalog Functions

SOAP catalog functions enable you to access and process the contents of the following elements in the event payload of an incoming SOAPEvent, and add elements to the outbound SOAPEvent:

- Header
- Body
- Fault
- Attachment

See [Parsing and Building SOAP Messages on page 123](#) for more details.

Mapping of SOAP Request URI to Destination

For Soap requests, the header property `SOAPAction` represents the destination name, and the `requestURI` represents the channel URI. TIBCO BusinessEvents combines the `SOAPAction` and `requestURI` to create a destination URI. It maps the request with a destination having same URI as the newly created URI.

For example, if the `requestURI` of a request is `/QueryBooks` and the value of `SOAPAction` is `QueryBooksByAuthor`, then TIBCO BusinessEvents engine maps the request with a destination having `/QueryBooks/QueryBooksByAuthor` URI if it exists.

Manually Creating Resources to Work with SOAP Services

This section explains how you can manually create and configure project resources to work with SOAP services

The section [Creating Resources Using the WSDL Import Utility on page 119](#) explains how to use the WSDL import feature to create the resources.

Also see [Understanding the WSDL to Project Resource Mapping on page 128](#) for details about how the WSDL elements correspond to project resources.

Task A Configure the HTTP Channel and SOAP Destination

Configure an HTTP channel, an HTTP connection resource for the channel. See [Working with HTTP Requests on page 97](#) for details. Connection configuration is the same for SOAP and HTTP channels, except for the serializer. For SOAP destinations, use the following serializer:

```
com.tibco.cep.driver.http.serializer.SOAPMessageSerializer
```

After you complete [Task B](#), set that `SOAPEvent` as the default event for the destination.

Task B Add a SOAPEvent (and Other Ontology as Needed)

Configure the SOAPEvents that will receive the SOAP requests and send out SOAP responses. Also configure any other ontology as needed. SOAPEvent is an event type provided with TIBCO BusinessEvents. However it is created in a two-step manner:

1. Add a simple event in the usual way. See [Adding a Simple Event on page 136](#).
2. In the Inherits From field, select **SOAPEvent**.
3. Set the default destination to the destination you created in [Task A](#).

The payload of a SOAPEvent is automatically configured with the structure of a SOAP message. It has Header and Body elements, and within the Body element, a Fault element. You can further configure the Header and Body elements using the payload editor.

Task C Configure Rules and Rule Functions using SOAP Functions

In general the procedure of serving requests and sending requests to other servers is the same for SOAP services as for other HTTP interactions. See [Working with HTTP Requests on page 97](#). One rule function is used as the event preprocessor.

In addition, you must configure rules and rule functions to access information from the SOAP messages in the inbound events, and to populate outbound events with SOAP message details. Set two SOAPEvents you configured in [Task B](#) as input and return arguments of the preprocessor. See [Parsing and Building SOAP Messages on page 123](#) for details.

Task D Configure an Event Preprocessor

Open the project CDD for editing, and configure an event preprocessor for the destination you configured in [Task A](#). Use the rule function you configured in [Task C](#).

Creating Resources Using the WSDL Import Utility

WSDL (Web Services Description Language) is an XML format for describing web services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information.



- Only document style WSDLs with literal encoding are currently supported.
- Only In-Only and In-Out Message Exchange Patterns (MEPs) are currently supported.

TIBCO BusinessEvents can parse an imported WSDL file and create the required project artifacts based on its contents: the channels and destinations (concrete WSDLs only), events, rules, and rule functions.

Both abstract and concrete WSDL files can be used. When you import an abstract WSDL, channel and transport information is not available and you must create those resources manually. Concrete WSDLs contain information that is used to create these resources automatically.

You then implement the newly created rules and rule functions to provide the expected behavior for the web service described in the WSDL file.

See [Understanding the WSDL to Project Resource Mapping on page 128](#) to understand which sections of the WSDL are used to create each resource.

To Import a WSDL File

First you must import the WSDL file into the project. Then you can use the WSDL import utility.

1. Import the WSDL file into the project, as follows.
 - a. In Studio Explorer, right-click the project folder (or the project root) where you want to store the WSDL file and select **Import**. You see the Import wizard.
 - b. At the Select dialog, select **General > From File System**. Click **Next**.
 - c. At the File system dialog, click **Browse**, select the directory that contains the WSDL file you want to use, and click **OK**.
 - d. The File system dialog displays the directory in the left panel and its files in the right panel. Check the checkbox to select the file you want to use. Click **Finish**.

The WSDL file is imported into the project.

2. Create project artifacts from the WSDL file as follows.
 - a. In Studio Explorer, right-click anywhere in the project and select **Import**. You see the Import wizard.
 - b. At the Select dialog, select **TIBCO BusinessEvents > WSDL**. Click **Next**.
 - c. At the WSDL Import dialog, click **Browse** and select the WSDL file you imported in [step 1](#), then click **OK**.
 - d. The WSDL Import dialog displays again, showing the selected WSDL. Click **Finish**.

If the WSDL file you select is an abstract WSDL, you see a warning letting you know that a channel will not be created. See the section introduction for more details about abstract WSDL files.

The project artifacts are created within a folder named using part of the WSDL file name.

3. Save the project.

Exporting Resources as a WSDL File

The WSDL export utility allows you to export rules and rule functions that have a `SOAPEvent` as a parameter, as a WSDL file.



WSDL Filenames must conform to the `NCName` datatype. See the following page for more details:

<http://www.w3.org/TR/REC-xml-names/#NT-NCName>

Some Japanese characters, such as half-width Katakana, have issues when they are used in XML names. See the following document for more details:

<http://www.w3.org/Submission/japanese-xml/>

[Table 21](#) shows how related project resources are exported as a WSDL file.

Table 21 Exporting Project Artifacts as WSDL

Project Folders and Resources	WSDL Artifact after Exporting
<p>RuleFunction</p> <p>Use the rule function specified as the input event's default destination as the SOAPEvent's preprocessor.</p> <p>That rule function must have the "SOAPEvent for input" as the input and the "SOAPEvent for output" as the return type.</p>	Forms the <code><wsdl:operation></code> operation.
<p>SOAPEvent</p> <ul style="list-style-type: none"> • <code>SOAPEvent</code> for the WSDL input argument OR • <code>SOAPEvent</code> for the WSDL output argument <p>Use the event you configured in Task B.</p>	Creates a message. The contents of Envelope > Header > Body become message parts.
<p>HTTP Channel</p> <p>Channel with HTTP as the Driver, Resource as the Method of Configuration, with a pointer to the HTTP Connection.</p>	The URI of the channel forms the <code><soap:address></code> URI.
<p>HTTP Connection for HTTP Channel</p> <p>Host and port are used.</p>	The name of the connection forms the <code><wsdl:port></code> . Host and port of the connection form the <code><soap:address></code> host and port.
<p>Destination for HTTP Channel</p> <p>With <code>SOAPMessageSerializer</code> as the default serializer.</p> <p>The destination used is the default destination of the SOAPEvent. It b</p>	<p>Forms the <code>soapAction</code> attribute of the operation.</p> <p>The</p> <p>The destination name becomes <code>soapAction</code> specified for that particular operation.</p>
<p>CDD</p> <p>In the Agent Classes tab, Input Destination Collections list (or Collections tab, Input Destinations), the Preprocessor specified for the input SOAPEvent's default destination.</p>	Associates <code>soapAction</code> to the operation in the HTTP binding.

To Export a Rule or Rule Function as a WSDL

1. Right-click the project, and click **Export...**
2. In the Export dialog, select **WSDL** under TIBCO BusinessEvents, and click **Next**.
3. Select the WSDL location by specifying the path, and type a valid `NCName` for the WSDL file.
4. Select the CDD file of the project.
5. Click **Finish**.
6. Save the project.

Parsing and Building SOAP Messages

This section explains how to use the SOAP functions to parse information from incoming requests, and to construct outgoing messages (responses and requests) in your rules and rule functions.

This manual does not explain how to work with the SOAP protocol. For example, you should understand how the `Actor` and `MustUnderstand` attributes in SOAP headers are used to process the message as it passes from its originator, through intermediary applications, to its ultimate destination.

Working with Incoming SOAP Messages (Event Payloads)

An incoming SOAP message can be a request, or a response, depending on whether TIBCO BusinessEvents is acting as the server or the client. TIBCO BusinessEvents can also act as an intermediate node along the path of a SOAP message to its ultimate destination. This section explains how to parse (get) information out of the incoming event payload, which contains the SOAP message.

To Parse the SOAP Envelope

```
String getEnvelope(SimpleEvent inSoapEvent)
```

Given a request `SOAPEvent`, this function returns the SOAP envelope in the request event payload, for example:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.tibco.com/schemas/SoapOverHttp/Schema/Schema.xsd" >
  <soapenv:Header/>
  <soapenv:Body>
    <sch:root>
      <sch:First>1</sch:First>
      <sch:Second>2</sch:Second>
    </sch:root>
  </soapenv:Body>
</soapenv:Envelope>
</message>
```

To Parse (and Optionally Remove) Headers and Header Attributes

```
String[] getHeaders(SimpleEvent inSoapEvent, String actor, Boolean
removeHeaders)
```

If the `actor` parameter has a null value then all the immediate children of the Header element are retrieved:

```
getHeaders(inSoapEvent, null, false)
```

Otherwise, the header specified by the `actor` attribute is retrieved. For example, given this Headers element in a SOAP event payload:

```
<soapenv:Header>
  <t:user xmlns:t="http://schemas/xml.com" soapenv:mustUnderstand="true"
    soapenv:actor="http://localhost:9090/Service">jon</t:user>
  <t:user_surname xmlns:t="http://schemas/xml.com"
    soapenv:mustUnderstand="true"
    soapenv:actor="http://localhost:9090">smith</t:user_surname>
</soapenv:Header>
```

If you specify the following:

```
getHeaders(inSoapEvent, "http://localhost:9090/Service", false)
```

To remove the specified header part or parts, set the final parameter to `true`. (The SOAP specification states that if a header is processed it should be removed. You would remove a header if TIBCO BusinessEvents is acting as an intermediary node and the request created using the SOAP functions will be sent on to another server.)

Then the first Headers element is returned:

```
<t:user xmlns:t="http://schemas/xml.com" soapenv:mustUnderstand="true"
  soapenv:actor="http://localhost:9090/Service">jon</t:user>
```

You can also retrieve the attributes of a SOAP Header element:

```
String[] getSOAPHeaderAttribute(SimpleEvent inSoapEvent, int index,
String attribute)
```

You can also remove all or selected headers using one of these functions:

```
removeHeaderPart()
removeHeaderParts()
```

To Parse the SOAP Body (SOAPBodyParts)

Two functions are available for getting SOAP body parts.

```
String[] getAllSOAPBodyParts(SimpleEvent inSoapEvent)

String[] getSOAPBodyParts(SimpleEvent inSoapEvent, String name,
String namespace)
```

The `getAllSOAPBodyParts()` function simply returns all SOAP body parts.

The `getSOAPBodyParts()` function allows you to specify which parts are of interest. Given a body part name and a namespace for a specified `SOAPEvent`, it returns a String array of matching SOAP body parts in serialized form. Name and namespace parameters can't be null.

Example

Given this function:

```
String[] body_part= getSOAPBodyParts
(soapeventin,"root",http://www.tibco.com/schemas/SoapOverHttp/Schema/Schema.xsd)
```

And this `soapeventin` event payload:

```
<?xml version="1.0" encoding="UTF-8"?>
<message>
  <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://www.tibco.com/schemas/SoapOverHttp/Schema/Schema.xsd"
xmlns:temp="http://temp/">
    <soapenv:Header/>
    <soapenv:Body>
      <sch:root>
        <sch:First>1</sch:First>
        <sch:Second>2</sch:Second>
      </sch:root>
      <sch:parent2>
        <sch:child1>3</sch:child1>
        <sch:child2>4</sch:child2>
      </sch:parent2>
    </soapenv:Body> </soapenv:Envelope>
```

You would get this as the SOAP body part:

```
body_part[0]=
<?xml version="1.0" encoding="UTF-8"?>
<ns0:root xmlns:ns0="http://www.tibco.com/schemas/SoapOverHttp/Schema/Schema.xsd">
  <ns0:First>1</ns0:First>
  <ns0:Second>2</ns0:Second>
</ns0:root>
```

To Parse Attachments

The following functions enable you to work with SOAP attachments in the request message:

```
getNumberOfAttachments(SimpleEvent inSOAPEvent)
getAttachmentContentID(SimpleEvent inSOAPEvent, int Index)
```

```

getAttachmentContentType(SimpleEvent inSOAPEvent, int Index)
getAttachmentContent(SimpleEvent inSOAPEvent, int Index)
getAttachmentContentByContentID(SimpleEvent inSOAPEvent, string
contentID)

```

The content ID is the attachment identifier. You can select which attachment to work with using its index position. First get the count of the attachments using `getNumberOfAttachments()`. Then using the index, you can get the content ID and content type, as well as the attachment content itself.

The content is returned in byte form, so after you get the content, you must then use other functions to make the content human-readable.

For an example, see

<http://www.w3.org/TR/SOAP-attachments#SOAPReferenceToAttachments>.

To Parse SOAP Fault XML Nodes

The following functions enable you to work with the standard SOAP Fault XML nodes from the payload of a SOAP event:

```

getFault (SimpleEvent, soapEvent)
getFaultActor (SimpleEvent, soapEvent)
getFaultCode (SimpleEvent, soapEvent)
getFaultString (SimpleEvent, soapEvent)

```

Working with Outgoing SOAP Messages (Event Payloads)

You can add header parts, body parts, fault parts and attachments to the outgoing SOAP message (whether it is a response or a request).

The signatures of the relevant functions are as follows:

```

addHeaderPart(SimpleEvent outSOAPEvent, String headerPartXml)
addSOAPBodyPart(SimpleEvent outSOAPEvent, String bodyXML)
addSOAPHeaderAttribute(SimpleEvent outSOAPEvent, int index, String
attribute, String value)
addFaultPart(SimpleEvent outSOAPEvent, String faultCode, String
faultMessage, String faultActor, String faultDetailString)
addAttachment(SimpleEvent outSOAPEvent, String contentID, String
content, String contentType, String contentEncoding)

```

TIBCO BusinessEvents adds each type of fragment to the appropriate part of the event payload—header, body, or fault. The fragments must be well-formed XML. You can also add attachments.

For example, to add a body part containing information for a response you would include all the required details including any namespace information:

```
SOAP.addSOAPBodyPart(outSOAPEvent, "<ns0:BookStore  
xmlns:ns0=\"http://www.abc.com/xsd/books\"><ns0:Book><ns0:Author>J.K.Rowling</ns0:A  
uthor></ns0:Book></ns0:BookStore>");
```

The specified body part is added to the correct place in the outline structure of the SOAP message, which is provided by the SOAPEvent. The resulting payload would look similar to the following:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">  
  <SOAP-ENV:Body>  
    <ns0:BookStore xmlns:ns0="HTTP://www.abc.com/xsd/books">  
      <ns0:Book>  
        <ns0:Author>J.K.Rowling</ns0:Author></ns0:BookStore>  
      </ns0:Book>  
    </ns0:BookStore>  
  </SOAP-ENV:Body>  
</SOAP-ENV:Envelope>
```

Understanding the WSDL to Project Resource Mapping

A WSDL file describes a web service. The WSDL Import utility imports a WSDL file and generates TIBCO BusinessEvents project artifacts using elements in the WSDL. TIBCO BusinessEvents can import abstract and concrete WSDL files. The source of the WSDL could be, for example, a ActiveMatrix BusinessWorks SOAPRequestReply activity.

Example WSDL

The table following this example shows which WSDL elements and attributes are used to create TIBCO BusinessEvents project artifacts. Elements and attributes used in the import are highlighted in bold text. Differences between import from abstract and concrete WSDL files are also highlighted. See [Table 22, Imported WSDL Project Artifacts, on page 129](#) for more details.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Created by TIBCO WSDL-->
<wSDL:definitions omitted to keep the example short>
  <wSDL:types>
    <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.books.org" elementFormDefault="qualified"
attributeFormDefault="unqualified" targetNamespace="http://www.books.org">
. . . . . Elements omitted
    </xsd:schema>
  </wSDL:types>
. . . . . Elements omitted
```

Note: *In an abstract WSDL, the following elements are used in the import. However in a concrete WSDL, the <wSDL:binding> elements are used instead.*

```
<wSDL:portType name="GetBookPortType">
  <wSDL:operation name="GetBook">
    <wSDL:input message="tns:GetBookRequestMessage"/>
    <wSDL:output message="tns:GetBookResponseMessage"/>
  </wSDL:operation>
</wSDL:portType>
```

Note: *This is a concrete WSDL example, so the <wSDL:binding> elements are used in the import. (In an abstract WSDL, the <wSDL:portType> element contents are used instead.)*

```
<wSDL:binding name="getBookBinding" type="tns:GetBookPortType">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <wSDL:operation name="GetBook">
    <wSDL:documentation>The operation has no documentation
  </wSDL:documentation>
    <soap:operation style="document" soapAction="/Service/getBook"/>
    <wSDL:input>
      <soap:body use="literal" parts="part1"/>
    </wSDL:input>
  </wSDL:operation>
</wSDL:binding>
```

```

        <soap:header use="literal"
            message="tns:TransactionRecordMessage" part="user"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal" parts="part1"/>
        <soap:header use="literal"
            message="tns:TransactionRecordMessage" part="transactionID"/>
    </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:service name="getBook">
    <wsdl:port name="getBookHttpPort" binding="tns:getBookBinding">
        <soap:address location="http://ACME:9090/Service/getBook"/>
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

Example Project Folder Structure

Suppose you import the example WSDL above into a Studio project called Library. The imported and generated project artifact names would appear as shown in the Project Folders and Resources column in the table below. Folders that are added by TIBCO BusinessEvents are shown in bold. The example is a concrete WSDL. In the WSDL Source column the source of folder and resource names is given for abstract as well as concrete WSDL sources.

Table 22 *Imported WSDL Project Artifacts*

Project Folders and Resources	Project Resource Type	WSDL Source
Library	Project root folder	N/A
getBook/	Folder	<p>For concrete WSDLs: <wsdl:service></p> <p>For abstract WSDLs: <wsdl:service> is not present, so the folder structure starts from the folder created from <wsdl:portType>.</p>
GetBookPortType/	Folder	<p>For abstract and concrete WSDLs: <wsdl:portType></p>
getBook/Events/	TIBCO BusinessEvents folder	

Table 22 Imported WSDL Project Artifacts (Cont'd)

Project Folders and Resources	Project Resource Type	WSDL Source
GetBookRequestMessage	Event	<div><wsdl:input></div> <div>For abstract WSDLs, in <wsdl:portType> section.</div> <div>For concrete WSDLs, in <wsdl:binding> section.</div>
GetBookResponseMessage	Event	<div><wsdl:output></div> <div>For abstract WSDLs, in <wsdl:portType> section.</div> <div>For concrete WSDLs, in <wsdl:binding> section.</div>
getBook/RuleFunctions/	TIBCO BusinessEvents folder	
GetBook	Rule function	<div><wsdl:operation></div> <div>For abstract WSDLs, in <wsdl:portType> section.</div> <div>For concrete WSDLs, in <wsdl:binding> section.</div>
getBook/Rules/	TIBCO BusinessEvents folder	
GetBookPortType	Folder	<div>Abstract WSDL: <wsdl:portType name></div> <div>Concrete WSDL: <wsdl:binding type></div>
GetBook	Rule	<div><wsdl:operation></div> <div>For abstract WSDLs, in <wsdl:portType> section.</div> <div>For concrete WSDLs, in <wsdl:binding> section.</div>

Table 22 Imported WSDL Project Artifacts (Cont'd)

Project Folders and Resources	Project Resource Type	WSDL Source
Import from Concrete WSDLs Only If the import is from a concrete WSDL, the HTTP Connection resource, Channel resource and Destination resource are added using details in the <code><wsdl:service></code> section of the WSDL. If the import is from an abstract WSDL, you must create these resources manually.		
<code>getbook/Transports/</code>	TIBCO BusinessEvents folder	
<code>getBookHTTPPort</code>	HTTP Connection	<code><wsdl:port></code> The host and port come from the <code><soap:address location></code>
<code>Service/</code>	TIBCO BusinessEvents folder	
<code>getBook</code>	Channel See Channel Folders on page 131 .	<code><soap:address location></code> (from the last part of the location URL)
<code>Service_getBook</code>	Destination See Destination Names on page 132	<code><soap:operation soapAction></code>

How Project Artifacts are Named

Channel Folders

Given a concrete WSDL, folders are created for all the elements after the `port`, up to the last forward slash of the location URL. The text after the last forward slash is the channel name. For example, given the following location URL:

```
http://ACME:9090/Service/Trial/getBook
```

The folder structure would be `/Service/Trial` and the channel name would be `getBook`.

Destination Names

In a concrete WSDL, the `SOAPAction` attribute of a `<soap:operation>` element specifies the URL of a destination. It also becomes the destination name. Forward slashes (/), colons (:), and periods (.) are converted to underscore characters (_) to form the name. For example:

```
http://www.acme.com/TNT/webservices/getByteField
```

Becomes:

```
http___www_acme_com_TNT_webservices_getByteField
```

Rules and Rule Functions

For each operation, the import utility creates a rule and a rule function. The rule has no body. The rule functions have `SoapEventOut` as the return type. Null value is returned by default.

For example, the `GetBook` operation becomes a `GetBook` rule in the `GetBookPortType` folder which is in the `Rules` folder, and also a `GetBook` rule function in the `Rulefunctions` folder.

You implement the rules and rule functions in your project according to the web service you want to implement.

Events

The `<wsdl:input>` element becomes a request event and the `<wsdl:output>` element becomes a response event. Each event type inherits from the `SoapEvent` event type.

Event names come from the `message` attributes. In the example, the request event is `GetBookRequestMessage` and the response event is `GetBookResponseMessage`.

Faults

Faults specified in a WSDL are used in the outbound SOAP event, as the `Fault` element.

TIBCO BusinessEvents supports three sorts of events: Simple events (usually referred to just as events); time events, which are timers; and advisory events. This chapter explains how to use simple events.

Topics

- [Overview of Simple Events, page 134](#)
- [Working with Events in Rules, page 135](#)
- [Adding a Simple Event, page 136](#)
- [Simple Event Reference, page 137](#)
- [Simple Event Attributes Reference, page 143](#)

Overview of Simple Events


A simple event defines an object that represents an occurrence of something. When the term "event" is used without the qualifier advisory or time, it refers to a simple event.

See also

- [Effect of Cache Only Cache Mode on page 221](#)
- *TIBCO BusinessEvents Getting Started* provides a practical introduction to events, including use of default events and default destinations.
- Chapter 2, Channels and Events in *TIBCO BusinessEvents Architect's Guide* for a more detailed overview and information about the following topics:
 - Message Acknowledgment
 - Default Destinations and Default Events
 - Simple Events — Time to Live and Expiry Actions
- [Chapter 24, Diagrams, on page 421](#) for information on using event dependency and sequence diagrams, and event model views.

Using Inheritance

Use of inheritance can simplify event configuration. A child event inherits the following:

- All the parent event's properties.
 - The parent event's expiry action (if set). However, an expiry action set in the child event overrides the parent event expiry action.
- 
 - A parent event cannot have a payload.
 - All child events of an event with TTL=0 must also use TTL setting TTL=0

Events that are related to each other directly or indirectly by inheritance cannot have distinct properties that share a common name. Therefore the following restrictions apply:

- If two events are related by inheritance, you cannot create a new property in one with a name that is already used in the other.
- If two unrelated events have properties that share a name, you cannot create an inheritance relationship between the two events.

Working with Events in Rules

This section explains various aspects of using events in rules.

See also [Using Scheduler Functions \(Requires Cache OM\) on page 155](#).

Explicitly Assert Events Created in Rules

At runtime, event instances that are created using rules are not automatically asserted. You must explicitly assert such events, for example using the `Event.assertEvent()` function.

Events that are created from incoming messages, on the other hand, are automatically asserted.

Specifying Default and Non-Default Destinations

TIBCO BusinessEvents includes two functions that allow you to send simple events out to another application: `Event.sendEvent()` and `Event.routeTo()`.

- `Event.sendEvent()` automatically sends the event to its default destination.
- `Event.routeTo` takes a destination as an argument, ignoring the event's default destination.

With `routeTo` you can direct an event to a destination on a different channel from the event's default destination. You can also override the properties of the destination, for example, the subject.

You cannot, however, override the properties of the channel itself, for example, the network field in a Rendezvous channel.

Scheduling Simple Events

You can use two methods to schedule simple events:

- Rule-based time events schedule the assertion of simple events.
- Scheduler functions schedule the sending of simple events to their default channels.

Each method is appropriate in different situations. See [Chapter 9, Time Events and Scheduler Functions, on page 145](#) for details.

Adding a Simple Event

This section provides summary steps for adding a simple event. See [Simple Event Reference on page 137](#) for details on how to complete the values. Also see [Overview of Simple Events on page 134](#).

To Add a Simple Event

1. In Studio Explorer, right click the folder where you want to store the event and select **New > Simple Event**. You see the New Simple Event Wizard.
2. In the Simple Event Name field, type a name for the event. In the Description field, type a description as desired.



You can't change the name in the editor. To change the name of any project element, right-click the element in Studio Explorer and select **Refactor > Rename**. See [Chapter 4, Element Refactoring Operations, on page 45](#) for more details.

3. Click **Finish**. You see the Simple Event Editor.
4. Complete the Standard tab and Advanced tab sections as explained in [Simple Event Reference on page 137](#).
5. Save the resource.



The ability to add metadata groups and properties is provided for customization purposes and is not documented.

Simple Event Reference



Simple Event resources are used to define an object that represents an occurrence, such as sending an invoice, debiting an account, and so on.

You can modify and enrich events before they are asserted into the Rete network. Rule evaluation depends on event values at time of assertion, so they can be changed only before assertion.



If you are working with a project imported from a release earlier than 5.0.0, you may see metadata properties. However, do not use them. Instead use the settings and properties in the Domain Objects section of the CDD file as needed. See *TIBCO BusinessEvents Administration* for details.

Wizard and Configuration (Standard Tab)

The Wizard and the Configuration section have the following fields.

Field	Global Var?	Description
Name	No	<p>Not shown as a field because it can't be changed. The name appears in the Wizard, and in the title of the event.</p> <p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p>
Description	No	Short description of the resource.
Inherits From	No	This event inherits from the event you select here. Leave blank if you don't want to use inheritance. As a convenience, you can open the selected event resource by clicking the underlined label.

Field	Global Var?	Description
Time to Live	No	<p>Specify a numerical value and a time unit. Time units available are milliseconds, seconds, minutes, hours, and days. (Default unit is seconds.)</p> <p>The numerical value is interpreted as follows:</p> <ul style="list-style-type: none">One or higher (>0): the event expires after the specified number of units elapse. <p>Specify the time units in the drop-down list field to the right of the Time to Live field. Choose one of: Seconds, Minutes, Hours, Days, WeekDays, Weeks, Months, Years</p> <ul style="list-style-type: none">Zero (0): the event expires after the completion of the first RTC cycle.A negative integer (<0): the event does not expire, and must be explicitly consumed. The value -1 is generally used to indicate an event that does not expire. <p>See Declaration and Expiry Action (Advanced Tab) on page 139.</p>
Default Destination	No	<p>When the destination is not otherwise specified (for example in rules or rule functions), events of this type are sent to the destination you select here. You can send an event to the default destination of its event type using the <code>Event.sendEvent()</code> function. As a convenience, you can open the selected destination resource by clicking the underlined label.</p>
Retry On Exception	No	<p>When a destination's event preprocessor fails due to an exception, the behavior an event instance of this type is determined by this check box setting:</p> <ul style="list-style-type: none">When this checkbox is checked TIBCO BusinessEvents attempts to reprocess the event instance that failed.When the checkbox is unchecked TIBCO BusinessEvents does not attempt to reprocess the event instance that failed. <p>See Event Preprocessors on page 243 for more information.</p>

Properties (Standard Tab)

The Properties section has the following fields. Event properties generally map to incoming or outgoing message properties.

Field	Global Var?	Description
Name	No	<p>The name to appear as the label for the property. Names follow Java variable naming restrictions. Do not use any reserved words. See Identifiers (Names) on page 275.</p> <p>Note: In addition to standard naming restrictions, do not begin an event property name with <code>_ns_</code> or <code>_nm_</code>. These have a special use. See Mapping Incoming Messages to Non-default Events on page 62.</p> <p>For events used in JMS channels Names beginning with <code>_jms</code> or <code>jms</code> (case insensitive) are used only for JMS header properties. You can, however, use properties beginning <code>jms_</code> (case insensitive) for event properties.</p> <p>See Using JMS Header Properties in Incoming and Outgoing Messages on page 90 for more details. Table 19, JMS Header Field Names, on page 92 shows the list of JMS header properties. Consult the JMS specification for more details.</p>
Type	No	<p>One of: <code>String</code>, <code>Integer</code>, <code>Long</code>, <code>Double</code>, <code>Boolean</code>, <code>DateTime</code></p> <p>Note: For properties of type <code>Double</code>, all NaN (Not a Number) values are converted to <code>0.00</code>.</p>
Domain	No	<p>Click the search button and select the domain model you want to use for this property. See Chapter 13, Domain Models, on page 177 for details on adding domain models.</p>

Declaration and Expiry Action (Advanced Tab)

If the [Time to Live](#) field is zero or higher, define the action or actions to take when an event expires in the Expiry Action section.

If an event is explicitly consumed in a rule, TIBCO BusinessEvents does not execute the expiry action.

The editor in the Expiry Action section is the same as the Rule function editor. See [Rule Function Resource Reference on page 231](#) for details.

See Simple Events — Time to Live and Expiry Actions in *TIBCO BusinessEvents Architect's Guide* for background details.

Payload (Advanced Tab)

An event can have a payload. The payload often corresponds to a message body. Payloads can be defined using an XML schema. In the left panel you add groups (elements) and parameters (attributes). You can add groups as children of a selected group, or at the same level, to define a hierarchy as desired. In the right panel, you define the type of each element or parameter. The table below describes the payload parameters available for each content type. The content types appear in the drop-down list for the Content field in the Payload section:

Table 23 Simple Event Payload Element Parameters (Sheet 1 of 3)

Content/Parameter	Description
Complex Element	An element that contains other elements. This is like a structure in a programming language. The complex element can contain zero or more elements of other types, including other complex elements.
Name	The name of the element.
Cardinality	Values for Cardinality: <ul style="list-style-type: none">• Required: The payload must include an instance of this element.• Optional (?); The element is not required.• Repeating (*); The element is a list that has zero or more instances.• At least one (+): The element is a list that has one or more instances.
Element of Type	An element with a specified data type. You can specify a scalar data type (string, integer, and so on), you can reference an XML type, or you can specify the <code>TIBCO ActiveEnterprise Any</code> data type.
Name	The name of the element.
Cardinality	See Cardinality under Complex Element.
Type	The generic data type. For example, decimal or date/time.
Type	The specific data type. For example, float or month. Refer to the <i>TIBCO ActiveMatrix BusinessWorks Palette Reference</i> for a complete list.
XML Element Reference	A reference to an element in a stored XML schema. See TIBCO Designer documentation for more information about XML schemas.
Cardinality	See Cardinality under Complex Element.

Table 23 Simple Event Payload Element Parameters (Sheet 2 of 3)

Content/Parameter	Description
Schema	Stored XML schema that contains the element or type you want to reference.
Element	The element within the stored XML schema that you want to reference.
Attribute of Type	An attribute with a specified data type. You can specify a scalar data type (string, integer, and so on), you can reference an XML type, or specify the <code>TIBCO ActiveEnterprise Any</code> data type.
Name	The name of the element.
Cardinality	See Cardinality under Complex Element.
Type	The generic data type. For example, decimal or date/time.
Type	The specific data type. For example, float or month. Refer to the <i>TIBCO ActiveMatrix BusinessWorks Palette Reference</i> for a complete list.
Sequence	A sequence of elements. Each item in the sequence is a structure of the sub-elements of this element.
Cardinality	See Cardinality under Complex Element.
Choice	A choice of elements. The data type of this element can be one of the sub-elements defined.
Cardinality	See Cardinality under Complex Element.
All	The data type of this element can be all of the data types of the sub-elements defined.
Cardinality	See Cardinality under Complex Element.
XML Group Reference	A reference to an XML group in a stored XML schema. See <i>TIBCO Designer</i> documentation for more information about XML schema.
Cardinality	See Cardinality under Complex Element.
Schema	Stored XML schema that contains the element or type you want to reference.
Model Group	Select the appropriate model group from the pull-down list.

Table 23 Simple Event Payload Element Parameters (Sheet 3 of 3)

Content/Parameter	Description
Any Element	A reference to any XML Element. You can use the Coercions button to supply a reference to the XML Element for this item when it appears in the input or process data.
Cardinality	See Cardinality under Complex Element.
Validation	Select the level of validation to be performed on the XML Element: <ul style="list-style-type: none">Strict: Must validate by locating a declaration for the element.Skip: Do not validate.Lax: Validate if a declaration for the element is available.

Simple Event Attributes Reference



When using an event in a rule's form editor, type `eventname@` to see the list of its attributes.

You can use the following attributes in rules to return information about a simple event instance.

Attribute	Type	Returns
@id	long	The event's unique internal ID.
@extId	string	<p>The event's unique external ID. Optional.</p> <p>The value of the <code>extId</code> is set at creation time, for example, using the event's ontology function, or the <code>Event.createEvent()</code> function. The value cannot be changed after that.</p> <p>Note: The <code>extId</code> value (if set) must be unique across all objects in the cluster.</p> <p>Tip: You can use the property <code>Agent.AgentClassName.checkDuplicates</code> to check for duplicate <code>extIds</code> across the cluster. See Chapter 3, CDD Configuration Procedures in <i>TIBCO BusinessEvents Administration</i> for details.</p>
@ttl	long	The event's time to live, where the assertion of the event defines the start of the time to live period. You can specify the value in the SimpleEvent resource TTL field. See Simple Event Reference on page 137 .
@payload	string	The payload as a string value. See Payload (Advanced Tab) on page 140 for more on specifying the payload in a SimpleEvent resource.

Time Events and Scheduler Functions

TIBCO BusinessEvents supports three sorts of events: Simple events (usually referred to just as events); time events, which are timers; and advisory events. This chapter explains how to use time events.

In addition you can use functions to schedule events. This method can be preferable to using time events in some cases.

Topics

- [Overview of Time Events, page 146](#)
- [Working With Time Events, page 148](#)
- [TimeEvent Resource Reference, page 150](#)
- [TimeEvent Attributes Reference, page 152](#)
- [Rule Based TimeEvent Function Reference, page 154](#)
- [Using Scheduler Functions \(Requires Cache OM\) on page 155](#)

Overview of Time Events

TIBCO BusinessEvents offers two kinds of time events, repeating and rule-based. In addition you can schedule events using functions.



Time events do not go through an event preprocessor. If you are using cache-only cache mode, ensure that any objects are properly loaded. Events scheduled using scheduler functions, however, are sent through channels and would therefore go through event preprocessors in the usual way.

See Also

- [Chapter 8, Simple Events, on page 133](#) for overview information about events in general.
- [Chapter 24, Diagrams, on page 421](#) for information on using diagrams.

Scheduled Time Events



Repeat (Interval) based time events are not supported in multi-engine mode Time events configured to repeat at intervals are not supported in multiple-agent (multi-engine) configurations. Rule-based time events, however, are supported.

You can configure a time event to repeat at a configurable time interval. For example, if you configure a time event to repeat every thirty seconds, then every thirty seconds TIBCO BusinessEvents creates a new time event of that type.

You can configure a repeating time event to create a specified number of events at each interval.

At engine startup

The time interval begins during engine startup. See [Appendix B, Engine Startup and Shutdown Sequence, on page 279](#) in *TIBCO BusinessEvents Administration* for specific details.

Rule Based Time Events

A rule based TimeEvent resource has only a name and description. You can then use it in a rule to schedule a simple event to be asserted, using its ontology function, `ScheduleTimeEventName()` in a rule (see [Rule Based TimeEvent Function Reference on page 154](#)). You can schedule the event to be asserted after a period of time, and you can pass information to the event and specify its time to live. You can call the `ScheduleTimeEventName()` function in different places with different time delays.

You can use rule based time events in various ways. For example, you might write rules that check for delays in order fulfillment:

1. A new Order event is asserted, and Rule A (which has Order in its scope) creates a time event T and configures it to be asserted in sixty minutes, and passes the order ID as the closure parameter value. (Rule A also sends the order details to another system.)
2. Sixty minutes after Rule A executes, timer event T is asserted.
3. The assertion of time event T triggers Rule B, which has T in its scope. Rule B checks the order status. If the order is delayed, it sends out an alert.

Working With Time Events

This section explains how to create and configure a time event.

Adding a Time Event

See [TimeEvent Resource Reference on page 150](#) for information on how to complete the values.

1. In Studio Explorer, right click the folder where you want to store the time event and select **New > Time Event**. You see the New Time Event Wizard.
2. In the Time Event Name field, type a name for the time event. In the Description field, type a description.



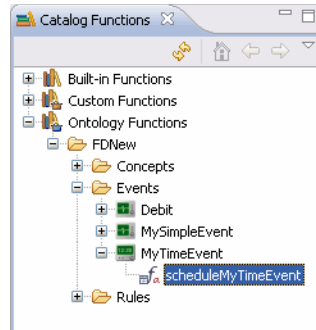
You can't change the name in the editor. To change the name of any project element, right-click the element in Studio Explorer and select **Refactor > Rename**. See [Chapter 4, Element Refactoring Operations, on page 45](#) for more details.

3. Click **Finish**. You see the Time Event editor.
4. In the Type field, select the type of time event you want to create and configure accordingly:
 - ruleBased: Select Rule Based. Use the event in a rule as explained in [Configuring a Rule Based Time Event in a Rule or Rule Function on page 148](#).
 - repeat: The event occurs at specified intervals. Select Repeat and configure the time event following guidelines in [TimeEvent Resource Reference on page 150](#).
5. Click **Apply** and save the resource.

Configuring a Rule Based Time Event in a Rule or Rule Function

1. Open the rule editor for the rule or rule function where you want to use the rule-based time event.
2. Display the Catalog Functions view if it is not available: Select **Window > Show View > Other > TIBCO BusinessEvents > Catalog Functions**.

3. In the Catalog Functions view, expand Ontology Functions and drill down to the rule-based time event. Expand the rule-based time event and select its ontology function (`scheduleTimeEvent`).



4. Drag the time event's ontology function into the Actions area of the rule (or Body area of the rule function) and configure the parameters, following guidelines in [Rule Based TimeEvent Function Reference on page 154](#).
5. Save the resource.

TimeEvent Resource Reference



TimeEvent resources are used as timers. You can configure rule-based timers, which are scheduled in a rule, and repeating timers, which are scheduled in the TimeEvent resource. Use time events to trigger rules.



If you are working with a project imported from a release earlier than 5.0.0, you may see metadata properties. However, do not use them. Instead use the settings and properties in the Domain Objects section of the CDD file as needed. See *TIBCO BusinessEvents Administration* for details.

Wizard and Configuration Tab

The Wizard and Configuration tab have the following fields.

Field	Global Var?	Description
Name	No	<p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p> <p>For events used in JMS channels Names beginning with <code>_jms</code> or <code>jms</code> (case insensitive) are used only for JMS header properties. You can, however, use properties beginning <code>jms_</code> (case insensitive) for event properties. See Using JMS Header Properties in Incoming and Outgoing Messages on page 90 for more details.</p>
Description	No	Short description of the resource.
Type	No	<ul style="list-style-type: none">ruleBased: The time event is scheduled by a rule.repeat: Time events are created periodically. Complete the rest of the settings to define the period and number of events per interval.

Field	Global Var?	Description
Interval	No	<p>Used with repeating time events only. Specify the interval between time events using a numerical value and a time unit.</p> <p>TIBCO BusinessEvents creates the first time event immediately after the engine starts and then creates the next time event based on the interval.</p> <p>A repeating time event expires after the completion of the first RTC cycle (that is, the time to live code is set internally to zero).</p> <p>Time units available are milliseconds, seconds, minutes, hours, and days.</p>
Events per Interval	No	<p>Used with repeating time events only. Enter the number of events to create in each Repeat Every interval.</p> <p>Default is 1.</p>

TimeEvent Attributes Reference



When using an event in a rule’s form editor, type *eventname@* to see the list of its attributes.

You can use the following attributes in rules to return information about an event instance.

Attribute	Type	Returns
@id	long	The event’s unique internal ID.
@closure	String	<ul style="list-style-type: none">For repeating time events: Not used (Null value).For rule based time events: A string that was specified when the event was scheduled.
@interval	long	<ul style="list-style-type: none">For repeating time events: The number of milliseconds between creation of successive time events.For rule based time events: Not used (0 value).
@scheduledTime	DateTime	<p>The time scheduled for asserting an instance of this event into the Rete network.</p> <ul style="list-style-type: none">For repeating time events: calculated based on the time of the last assertion of an instance of this event, and the interval.For rule based time events: specified using the <code>ScheduleTimeEventName()</code> function’s <code>delay</code> parameter. See Rule Based TimeEvent Function Reference on page 154.

Attribute	Type	Returns
@ttl	long	<ul style="list-style-type: none">• For repeating time events: always 0 (zero).• For rule based time events: specified using the ontology function's <code>ttl</code> parameter. See Rule Based TimeEvent Function Reference on page 154.

Rule Based TimeEvent Function Reference

Signature `Event ScheduleTimeEventName(long delay, String closure, long ttl)`

Domain `action`

Description For each rule-based time event you create, an ontology function is also created to enable you to schedule the assertion of an instance of the event in a rule action. The function name follows this format: `ScheduleTimeEventName`.

Parameters	Name	Type	Description
	<code>delay</code>	<code>long</code>	The event is created (and asserted) the specified number of milliseconds after the rule action executes.
	<code>closure</code>	<code>String</code>	The TimeEvent created will simply store the information passed in the closure parameter. you can put the closure string value in rule conditions to specify the time events that will trigger the rule.
	<code>ttl</code>	<code>long</code>	The event’s time to live. Follows the same rules as the time to live setting in a simple event. See Simple Event Reference on page 137 .

Returns	Type	Description
	<code>Event</code>	An instance of the event type specified in the function name (<code>ScheduleTimeEventName</code>).

Using Scheduler Functions (Requires Cache OM)



The TIBCO BusinessEvents Express edition supports only In Memory object management. Therefore it does not support scheduler functions.

You schedule time events as explained in [Overview of Time Events on page 146](#), and you schedule simple events using scheduler functions. Events scheduled using these functions are sent to their default destination.

You might use the event scheduler functions instead of a time event in the following cases:

- If you need to send any event on a schedule, you must use the scheduler. If you schedule a memory-only simple event the schedule is persisted. If you schedule a memory only time event (using the time event's function, not the scheduler functions) then the schedule is not persisted.
- If you need to create many, perhaps thousands, of schedules at one time, performance is better using the event scheduler functions.
- Because schedule management and event sending can be handled by a cache agent, fewer resources in the inference agents are required compared to time events. Additionally, with backing store enabled the schedules are only loaded into memory a batch at a time, reducing the total memory requirement compared to time events.

Task A Create a Scheduler

First you create a scheduler using the function `Cluster.createScheduler()`. You can do this, for example, in a startup rule function, or other rule function or rule:

```
void createScheduler(String schedulerName, long pollInterval, long refreshAhead)
```

- *schedulerName* A unique ID for the scheduler you are creating.
- *pollInterval* TIBCO BusinessEvents checks the scheduler cache every *pollInterval* milliseconds, for scheduler work items whose *scheduledTime* falls in the current interval.
- *refreshAhead* Time in milliseconds (into the future) used to pre-load the scheduled events from the backing store.



In this release, the larger value of *pollInterval* and *refreshAhead* is used for both these settings.

For example:

```
Cluster.createScheduler("myScheduler", 1000, 5000);
```

Task B Schedule the Event to be Sent

You use the function `Cluster.scheduleEvent()` to schedule an event to be sent at a certain time, using a scheduler you created earlier. You can do this in a rule or rule function as needed:

```
void scheduleEvent(String schedulerName, String workKey, SimpleEvent evt, long  
scheduledTime)
```

Where:

- *schedulerName* is the unique ID for the scheduler you created at an earlier time.
- *workKey* is a unique key that identifies the work item (that is, the scheduled task). This key can be used to identify the work item later, for example to cancel it.
- *evt* is the simple event to be scheduled.
- *scheduledTime* is used to specify the time the event is sent to its default destination. The value is interpreted as an absolute time. To schedule a time relative to the present, use the `System.currentTimeMillis()` function, as shown below.

For example:

```
Cluster.scheduleEvent("MyScheduler", myworkKey, Event.createEvent("xslt:// details  
omitted "), System.currentTimeMillis() + 5000)
```

Chapter 10 **Advisory Events**

TIBCO BusinessEvents supports three sorts of events: Simple events (usually referred to just as events); time events, which are timers; and advisory events. This chapter explains how to use advisory events.

Topics

- [Working With Advisory Events, page 158](#)
- [Advisory Event Attributes Reference, page 160](#)

Working With Advisory Events

You never have to add or configure an event of type `AdvisoryEvent`. Advisory events are asserted into the Rete network automatically when certain conditions, for example, exceptions, occur. Add the `AdvisoryEvent` event type to rule declarations to be notified of such conditions. Then use the event attributes in the rule as needed ([Advisory Event Attributes Reference on page 160](#)).

An advisory event expires after the completion of the first RTC cycle (that is, the time to live is set internally to zero).

Uses of Advisory Events

See [Advisory Event Attributes Reference on page 160](#) for details about use of attributes in each of the following uses.

Exceptions in User Code

The TIBCO BusinessEvents engine automatically asserts an advisory event when it catches an exception that originates in user code but that is not caught with the `catch` command of the TIBCO BusinessEvents Exception type.

For information on working with other kinds of exceptions, see [Exception Handling on page 286](#).

ActiveMatrix BusinessWorks Process Fails or Times Out

Advisory events are also used in the container mode method of TIBCO BusinessEvents-ActiveMatrix BusinessWorks integration feature `invokeProcess()` function. Such events are asserted when the ActiveMatrix BusinessWorks process fails or times out (or is cancelled). See [Chapter 21, ActiveMatrix BusinessWorks Integration, on page 347](#)

Engine is Activated

An advisory event (`engine.primary.activated`) is asserted when an engine has finished starting up and executing startup functions, if any (see Appendix B, Engine Startup and Shutdown Sequence in *TIBCO BusinessEvents Administration*).

Adding an Advisory Event to a Rule

1. Open the rule editor for the rule where you want to use the advisory event. (See [Adding a Rule on page 222](#).)

2. Do one of the following:

- In the form-based editor, click the plus icon in the Declarations panel and select Advisory Event from the list of resources.
- In the source editor, add a line like the following in the `declare` block:

```
AdvisoryEvent a;
```

In both cases you can change the alias as desired (from `a` to something else).

3. Save the resource.

The `AdvisoryEvent` event type has no properties. You can use its event attributes in rules to return information about an advisory event. See [Advisory Event Attributes Reference on page 160](#).

Advisory Event Attributes Reference



When using an event in a rule’s form editor, type `eventname@` to see the list of its attributes.

The `AdvisoryEvent` event type has no properties. You can use the following attributes in rules to return information about an advisory event. Add-on products may use additional advisory event types.

Name	Type	Description
@id	long	The event’s unique internal ID.
@extId	String	Null.
@category	String	Broad category of advisory: Exception—Used for exceptions not otherwise caught. Engine—Used for engine events. Also used for TIBCO BusinessEvents-ActiveMatrix BusinessWorks integration projects. Deployment —Used for hot deployment
@type	String	Type of advisory within the category. See Table 24, Attributes Used for Each Type of Advisory Event, on page 160 for details.
@message	String	Message content depends on the type of advisory event. See Table 24, Attributes Used for Each Type of Advisory Event, on page 160 for details.

Table 24 Attributes Used for Each Type of Advisory Event

Description	@category	@type	@message
Exceptions arising from user code	Exception	The Java class name of the exception	The stack trace and a message (if the exception includes a message)
TIBCO ActiveMatrix BusinessWorks Activity Failure	Engine	INVOKE BW PROCESS Indicates a failure or cancellation or timeout of the ActiveMatrix BusinessWorks process.	The error message from the failed ActiveMatrix BusinessWorks process, or the timeout message.

Table 24 Attributes Used for Each Type of Advisory Event

Description	@category	@type	@message
Engine is activated	Engine	engine.primary.activated	Engine <i>EngineName</i> activated.
		Indicates the engine has been activated.	
Hot deployment	Deployment	deployment.hotdeploy.success	Hot deployed project <i>ProjectPath</i>
		deployment.hotdeploy.fail	Failed to hot deploy project <i>ProjectPath</i>

Chapter 11 **Concepts**

This chapter explains how to work with concepts.

Topics

- [Overview of Concepts, page 164](#)
- [Adding Concepts and Concept Relationships, page 165](#)
- [Concept Resource Reference, page 167](#)
- [Concept Attributes Reference, page 170](#)

Overview of Concepts

You can add concept definitions so that information that arrives in events or from other sources can be organized and persisted as needed, and used in rules. You can add definitions manually. You can also import database tables as concept definitions. Concept instances are created in rules.

See Also

- Chapter 3, Concepts in *TIBCO BusinessEvents Architect's Guide* to gain an understanding of concepts including a detailed discussion of concept history, and concept relationships.
- [Chapter 24, Diagrams, on page 421](#) for information on using concept dependency and concept model diagrams.

Concept Serialization and Handling of Null Value Properties at Runtime

By default, when concept instance objects are serialized to XML, properties with null values are excluded. You can change this behavior so that null values are included. You can also change the XSD for a concept object to allow null values, using the nillable attribute. See [Appendix A, Handling Null Properties, on page 459](#) for details.

Concepts and State Machines

If you are using the TIBCO BusinessEvents Data Modeling add-on product, you can associate a concept with a state machine. See *TIBCO BusinessEvents Data Modeling Developer's Guide* for details.

Adding Concepts and Concept Relationships

This section explains how to work with concepts, and how to set up relationships between concepts.

Adding a Concept

See [Concept Resource Reference on page 167](#) for details on completing values.

To Add a Concept

1. In Studio Explorer, right click the folder where you want to store the concept and select **New > Concept**. You see the New Concept Wizard.
2. In the Concept Name field, type a name for the concept. In the Description field, type a description.



You can't change the name in the editor. To change the name of any project element, right-click the element in Studio Explorer and select **Refactor > Rename**. See [Chapter 4, Element Refactoring Operations, on page 45](#) for more details.

3. Click **Finish**. You see the Concept Editor.
4. Do the following:
 - Configure the remaining fields as explained in [Wizard and Configuration Tab on page 167](#).
 - As needed, add and configure properties as explained in [Properties on page 168](#).
5. Save the resource.



The ability to add metadata groups and properties is provided for customization purposes and is not documented. Metadata properties are also used for database concepts, a feature available in TIBCO BusinessEvents Data Modeling. See *TIBCO BusinessEvents Data Modeling Developer's Guide* for details.

Adding Concept Relationships

See Concept Relationships in *TIBCO BusinessEvents Architect's Guide* for information about concept relationships.

To Create an Inheritance Relationship

In the child concept resource's Configuration section, identify the parent concept in the **Inherits From** field.

Note that inheritance is set at the concept level. Other relationships are set at the concept property level.

To Create a Containment Relationship

A concept can be contained by only one concept at a time.

1. In the container concept resource's Property section, add and configure a property of type **ContainedConcept**.
2. Configure the property as follows:

Name: Provide a helpful name. For example, "wheels" could indicate that this concept is contained by a "car" concept (in an appropriate project).

Multiple: Check the checkbox if this property is an array.

Policy: Changes Only or All Values.

History: The number of historical values to keep.

To Create a Reference Relationship

A concept can refer to itself, and can be referred to by more than one concept.

1. In the Property tab of one of the concept resources, create a property of type `ConceptReference`.
2. Configure the property as follows:

Name: Provide a helpful name such as `is a lineitem of` to express the relationship.

Multiple: Check the checkbox if this property is an array.

Policy: Changes Only or All Values.

History: The number of historical values to keep.

See Concept Property History in *TIBCO BusinessEvents Architect's Guide* for more on this topic.

Editing or Deleting Concept Relationships

You can't edit or delete the relationship using the concept view graphical user interface. Instead, work with the concept properties directly in the Concept Editor.

Concept Resource Reference



Concept resources are descriptive entities similar to the object-oriented concept of a class. They describe a set of properties. For example, one concept might be Department, and it could include department name, manager, and employee properties.



If you are working with a project imported from a release earlier than 5.0.0, you may see metadata properties. However, do not use them. Instead use the settings and properties in the Domain Objects section of the CDD file as needed. See *TIBCO BusinessEvents Administration* for details.

Wizard and Configuration Tab

The Wizard and Configuration tab have the following fields.

Field	Global Var?	Description
Name	No	<p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p> <p>This field appears on the New Concept Wizard. The name then appears in the title of the concept editor.</p>
Description	No	<p>Short description of the resource.</p> <p>This field appears on both the New Concept Wizard and in the concept editor.</p>
Inherits From	No	<p>If you want this concept to inherit all the properties of another concept, browse to and select that concept.</p> <p>Concepts that are related to each other directly or indirectly by inheritance cannot have distinct properties that share a common name. Therefore the following restrictions apply:</p> <ul style="list-style-type: none"> • If two concepts are related by inheritance, you cannot create a new property in one with a name that is already used in the other. • If two unrelated concepts have properties that share a name, an inheritance relationship cannot exist between them.

Field	Global Var?	Description
State Models	No	State models that are owned by this concept. Models can be added and removed as needed. Requires TIBCO BusinessEvents Data Modeling.
Auto Start State Model	No	If checked then when a concept is asserted, at runtime, its main state machine (if any) is started. If unchecked, then state machines are started using this function in rules: <code>Instance.startStateMachine()</code> The function has a parameter specifying whether child concepts' state machines are also started. For contained concepts, the value of the Auto Start State Machine is not relevant and is not checked. Requires TIBCO BusinessEvents Data Modeling.

Properties

The Properties tab has the following fields:

Field	Global Var?	Description
Name	No	The property name.
Type	No	Any of the following types: <code>String, Integer, Long, Double, Boolean, DateTime, ContainedConcept, ConceptReference</code> When you create a property of type <code>ContainedConcept</code> , you are creating a containment relationship. The concept that you are currently configuring is the container; the concept you specify as a property is the contained concept. When you create a property of type <code>ConceptReference</code> you are creating a property that references another concept. See Concept Relationships in <i>TIBCO BusinessEvents Architect's Guide</i> for more details. Note: For properties of type <code>Double</code> , when a backing store is used, all NaN (Not a Number) values are converted to <code>0.00</code> .

Field	Global Var?	Description
Multiple	No	<p>Check the Multiple checkbox if this property is an array.</p> <p>Consider, for example, an Order concept: In most cases, an Order concept would allow only one value for the customer property but multiple values for the <code>line_item</code> property. Selecting the Multiple checkbox creates a property array.</p>
Policy	No	<p>TIBCO BusinessEvents can record historical values using either of these policies:</p> <p>Changes Only TIBCO BusinessEvents records the value of the property every time it changes to a new value.</p> <p>All Values TIBCO BusinessEvents records the value of the property every time an action sets the value even if the new value is the same as the old value.</p>
History	No	<p>Determines if historical values are stored, and if so how many.</p> <p>The default maximum value is 1024. You can set a different maximum using the property The maximum value allowed is 1024.</p> <p>Zero (0): TIBCO BusinessEvents does not store historical values for the concept. It stores the value without a time and date stamp</p> <p>One or more (>0): TIBCO BusinessEvents stores the property value when the property changes, along with a date and timestamp, up to the number specified. When the maximum history size is reached, the oldest values are discarded as new values are recorded. See Concept Property History in <i>TIBCO BusinessEvents Architect's Guide</i> for more details.</p> <p>Note: Use of a temporal function with a concept that has a history size of 0 may cause a runtime exception.</p>

Metadata

The Metadata section is used in a special way with database concepts. Requires TIBCO BusinessEvents Data Modeling. See *TIBCO BusinessEvents Data Modeling Developer's Guide* for details. It is not used otherwise (except for customization, which is not documented).

Concept Attributes Reference



When using a concept in a rule’s form editor, type *conceptname*.@ to see the list of its attributes.

You can use the following attributes in rules to return information about a concept instance.

Entity	Attributes	Type	Returns
Concept	@id	long	The concept instance’s unique internal ID.
	@extId	string	<p>The concept instance’s unique external ID. Optional.</p> <p>The value of the <code>extId</code> is set at creation time (for example, using the <code>Instance.createInstance()</code> function) and cannot be changed after that.</p> <p>Note: The <code>extId</code> value (if set) must be unique across all objects in the cluster.</p> <p>Tip: You can use the property <code>Agent.AgentClassName.checkDuplicates</code> to check for duplicate <code>extIds</code> across the cluster. See Chapter 3, CDD Configuration Procedures in <i>TIBCO BusinessEvents Administration</i> for details.</p>
ContainedConcept	@id	long	The contained concept instance’s unique internal ID.
	@extId	string	The contained concept instance’s unique external ID.
	@parent	concept	The parent concept instance. (This is treated as a concept reference in the language.)

Entity	Attributes	Type	Returns
ConceptReference	@id	long	The contained concept instance's unique internal ID.
	@extId	string	The contained concept instance's unique external ID.
	@isSet	boolean	Available for any concept property (not for referenced concepts). For example, you can use: <code>acc.Balance@isSet</code> . It means null or is assigned.

Chapter 12 **Scorecards**

This brief chapter explains how to work with scorecards.

Topics

- [Understanding and Working With Scorecards, page 174](#)

Understanding and Working With Scorecards

A scorecard is a special type of concept. A scorecard serves as a set of static variables that is available throughout the project. You can use a scorecard resource to track key performance indicators or any other information.

Unlike concepts and events, each scorecard resource is itself a single instance — it is not a description for creation of instances. You create the scorecard at design time. Its values can be viewed and updated using rules.

It is more accurate to say there is one instance of a scorecard per inference agent. Each inference agent in an application has its own instance of the score card. Scorecards are not shared between agents.

Any agent that uses scorecards, and also uses Cache OM, must be assigned a unique key so that the correct scorecard can be retrieved from the cache. The key is set in the Processing Unit tab of the CDD. See Chapter 3, CDD Configuration Procedures in *TIBCO BusinessEvents Administration*.

It is not necessary to add scorecards to the declaration of a rule. Because there is only one instance of each scorecard in a deployed TIBCO BusinessEvents agent, any change causes all rules that use the scorecard in their conditions to be evaluated.

See Also

- [Chapter 11, Concepts, on page 163](#).
- [Chapter 24, Diagrams, on page 421](#) for information on using scorecard dependency diagrams.

Adding a Scorecard

Configuring a scorecard is similar to configuring a concept, except that scorecards don't have relationships.

1. In Studio Explorer, right click the folder where you want to store the scorecard and select **New > Scorecard**. You see the New Scorecard Wizard.
2. In the Scorecard Name field, type a name for the scorecard. In the Description field, type a description.



You can't change the name in the editor. To change the name of any project element, right-click the element in Studio Explorer and select **Refactor > Rename**. See [Chapter 4, Element Refactoring Operations, on page 45](#) for more details.

3. Click **Finish**. You see the Scorecard Editor.
4. Add and configure properties as explained in [Concept Resource Reference on page 167](#).
5. Save the resource.



The ability to add metadata groups and properties is provided for customization purposes and is not documented.

Using a Scorecard in Rules

After configuring a scorecard resource, use rules to gather the information you need in the scorecard. To access the scorecard in a rule, use this syntax:

folder . folder . scorecard . property

For Example:

```
int i = SalesFolder.StatsScorecard.numOrdersProperty;
```

Scorecard Resource Reference



Scorecard configuration is the same as concept configuration, except that scorecards have no relationships with each other. Scorecards, therefore, have none of the properties used for setting up relationships. Scorecard properties can be of any primitive type.

See [Concept Resource Reference on page 167](#) for details about scorecard properties

Chapter 13 **Domain Models**

Domain model resources enable you to control user input for decision tables and test data. This chapter describes how to add and import domain models and associate them with entity properties.

Topics

- [Overview of Domain Models, page 178](#)
- [Adding Domain Models, page 179](#)
- [Importing and Exporting Domain Models, page 183](#)
- [Associating Domain Models with a Property, page 186](#)
- [Validating Data in Domain Models, page 187](#)

Overview of Domain Models

You can add domain models for concept, event, and scorecard properties. A domain model specifies the values that you may find useful for defining ontology item properties. For example, instead of typing text for a certain concept property, you can pick a value from a list, or enter a value within a predefined range.

A domain model can extend another domain model. When defining a domain model you can specify which domain model it inherits from.

Domain models can be used in decision tables and in test data for Tester. Decision tables are available in the TIBCO BusinessEvents Decision Manager add-on. See TIBCO BusinessEvents Decision Manager documentation for details about working with decision tables.



You cannot add domain models for concept properties of the `ContainedConcept` or `ConceptReference` type.

Setting up domain models has two steps:

1. Add domain models. For example, create a folder and add all the models you need for a project in this folder. See [Adding Domain Models on page 179](#) for details.

You can also import (and export) domain models. See [Importing and Exporting Domain Models on page 183](#).

2. Associate domain models with properties. See [Associating Domain Models with a Property on page 186](#) for details.

When creating a decision table, domain models control the values you can use for a given property.

Domain Model Value Descriptions for User Friendly Presentation

All domain model values can have optional descriptions that appear in the domain model editor. A preference determines whether domain model values or their descriptions appear in decision table cells. For some applications displaying descriptions can make the table easier for users to understand. For example suppose the value is a code such as 23, and the description is North West. Users will find it is easier to work with the description than the code. As another example, for a Boolean data type, the description can provide words such as Accepted and Rejected for the values True or False.

See TIBCO BusinessEvents Decision Manager documentation for details.

Adding Domain Models

You can store domain models as desired, for example, in a folder called `DomainModels`. For each domain model, you create a set of domain entries, where each entry represents a valid value for the entity property that uses the domain model.

After you add a domain model, associate it with a property. See [Associating Domain Models with a Property on page 186](#)

To Add a Domain Model

1. Right-click the folder where you want to store the domain model, and select **New > Domain Model**. You see the New Domain Model Wizard.

Alternatively, right-click a resource, and select **New > Other**. In the New dialog, select **Domain Model** under TIBCO BusinessEvents.

2. In the Domain Model Name field, type a name for the domain model. In the Description field, type a description.



You cannot change a resource name after you click Finish. (However, you can change the description later.)

3. Click **Finish**. You see the Domain Model editor.
4. In the Domain Type field, select the data type for this domain model. See [Supported Data Types on page 180](#) for supported types.
5. As needed complete the Inherits From field. If this domain model extends another domain model, browse to and select that domain model.
6. In the Domain Entries section, click **Add (+)**. An empty row appears in the table of entries.

You can also select rows and click **Duplicate (🔗)** to duplicate (and then modify) selected rows.

You can select a row and click **Remove (✖)** to remove individual rows that are not needed.

7. For each row you add, enter a description (optional) and in the Details section, define the domain model entry.

The Details section presents appropriate fields for defining the type of domain model entry you selected in [step 4](#). See [To Add Domain Entries on page 180](#) for examples.

8. When you have created the entries for the domain model, save the domain model resource.

To Add Domain Entries

When you add a domain model, you first select its data type. The Domain Model editor then displays an appropriate user interface for defining domain model entries of that data type.

The Domain Entries section is the same for all types. The Details section changes depending on what data type you selected for the domain model.

Supported Data
Types

Domain Models support the following data types:

- String
- Integer
- Long
- Double
- Boolean
- DateTime

Sections below show the user interface for each type.

String

String entries are simple text strings.

Domain Entries

+

Add

×

Remove

⌘

Duplicate

Description:

ABC

Value

London

Capital of England

London

Capital of France

Paris

Details

Simple string:

Paris



Numeric values in a String domain type When you use a numeric value (Integer, Double, or Long) in a domain model of type String, TIBCO BusinessEvents adds double quotes around the value. (These are visible after you save and reopen the domain model.)

Integer, Double,
Long

In a single domain model, you can enter single values, range values, or a mixture of both. Acceptable values for integer, long, and double domain entries are the same as for the equivalent Java datatypes.



Text values in a numeric data type When you enter text in a domain model of type Integer, Double or Long, TIBCO BusinessEvents marks it in red color, and indicates you to correct it.

The user interface is similar for all numeric datatypes. Here are some single values and a range value for an integer domain model:

The screenshot shows the 'Domain Entries' window for an integer domain model. The table lists three entries: 'Size 2' with value 2, 'Size 4' with value 4, and 'Sizes 6 to 12' with value [6,12]. The 'Details' panel on the right shows the 'Range' radio button selected, with 'Lower: 6' and 'Upper: 12', both marked as 'Included'.

Description	Value
Size 2	2
Size 4	4
Sizes 6 to 12	[6,12]

Details:
☐ Single ☒ Range
 Lower: 6 ☒ Included
 Upper: 12 ☒ Included

The next figure shows a set of ranges. Note that you define whether each end of the range is inclusive. For complete coverage, ensure that there is no gap and no overlap between ranges through consistent use of the Included check box. For example, the figure shows that the lower bound of the Senior range, 55, is not included.

The screenshot shows the 'Domain Entries' window for an integer domain model with three range entries: 'Juvenile' [0,17], 'Adult' [18,55), and 'Senior' (55,100]. The 'Details' panel shows the 'Range' radio button selected. For the 'Senior' range, the 'Lower: 55' is marked as 'Not Included' (checkbox unchecked) and the 'Upper: 100' is marked as 'Included' (checkbox checked).

Description	Value
Juvenile	[0,17]
Adult	[18,55)
Senior	(55,100]

Details:
☐ Single ☒ Range
 Lower: 55 ☐ Included
 Upper: 100 ☒ Included

Boolean

Boolean entry values are always `true` or `false`. The description can give the meaning of the pair of choices, such as male or female, supported or unsupported, eligible or ineligible and so on.

The screenshot shows the 'Domain Entries' window for a Boolean domain model. The table lists two entries: 'Eligible' with value true and 'Ineligible' with value false. The 'Details' panel shows the 'Boolean' radio button selected, with 'true' and 'false' options.

Description	Value
Eligible	true
Ineligible	false

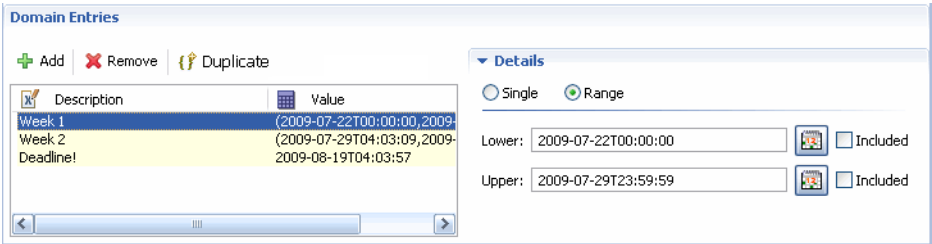
Details:
 Boolean: ☐ true ☒ false

DateTime In a single domain model, you can enter single values, range values, or a mixture of both. You specify a date and a time. A date and time picker enable you to select the date and time:



If you don't want to specify a time of day, set the time to midnight (12:00 AM) for the start date, and to a minute before midnight (11:59:59 PM) for the end date of an inclusive range, or for a single date.

The calendar shows a 12-hour clock. PM numbers are converted to a 24 hour clock format in the value table.



Importing and Exporting Domain Models

You can import domain model information from a database, from a Microsoft Excel spreadsheet, and from the source of a database concept property. (Database concepts are available in the TIBCO BusinessEvents Data Modeling add-on).

After you import a domain model, associate it with a property. See [Associating Domain Models with a Property on page 186](#)

Import From the Source of a Database Concept

You can import all values from the database column that corresponds to a database concept property for use as the domain model entry for that property. This is explained in *TIBCO BusinessEvents Data Modeling Developer's Guide*.

Export To and Import From Excel

You can export a domain model to an Excel spreadsheet. The description for each domain model entry goes in column A, and the value goes in column B. When you import from an Excel spreadsheet, similarly, column A is used as the description, and column B is used for the value.

In TIBCO BusinessEvents Decision Manager, you can export decision tables to Excel and import them from Excel. When you do so, you also export and import domain models associated with properties used in a decision table column. See *TIBCO BusinessEvents Decision Manager User's Guide* for details.

To Import Domain Model Entries from Excel

1. In TIBCO BusinessEvents Studio Explorer, do one of the following:
 - Right-click the folder where you want to create the domain model and select **Import > TIBCO BusinessEvents > Domain Model**.
 - In TIBCO BusinessEvents Studio Explorer, select any item in the project entity and select **File > Import > TIBCO BusinessEvents > Domain Model**.
2. Click **Next**. You see the Import Domain Model Wizard.

If you invoked the import wizard by right-clicking a folder, that folder is selected as the parent folder. You can choose a different one as desired.
3. In the Domain Import Source field, select `EXCEL`.
4. In the File Name field, enter a name for the domain model resource. Optionally enter a description.

5. In the Data Type field, select the appropriate data type for the domain model and click **Next**. You see the next wizard page.
6. In the Select Excel File to Import field, browse and select the Excel file that has the domain model information.
7. Click **Finish**. You see a message, "Domain Import Successful." Click **OK**.
8. You see the Domain Model editor. The column values appear as domain entries. You can add, edit, duplicate, and remove entries as appropriate.

To Export Domain Values to Excel

1. In TIBCO BusinessEvents Studio Explorer do one of the following:
 - Right-click the domain you want to export and select **Export > TIBCO BusinessEvents > Export Domain to Excel**.
 - In TIBCO BusinessEvents Studio Explorer, select any item in the project entity and select **File > Export > TIBCO BusinessEvents > Export Domain to Excel**.
2. Click **Next**. You see the Export to Microsoft Excel File Wizard.
3. In the Select Excel File to Export to field, browse and select the Excel file to which you want to export the domain model information.

To create a new Excel file, specify a filename that does not yet exist. If you specify an existing Excel file, the file contents are replaced with the exported domain model information.
4. In the Select Domain to Export area, expand the display and select the domain model you want to export.

If you invoked the export wizard by right-clicking a domain model, that domain model is selected. You can choose a different one as desired.
5. Click **Finish**. You see a message, "Export Domain Model to File Successful." Click **OK**.

Import From a Database Table

When you import domain model information from a database, the result set from the SQL query is transformed for use as domain model entries. This feature is supported with Oracle Database.

To Import Domain Model Entries from a Database Table

1. Add a JDBC Connection resource and configure it to connect to the database from which you want to import the domain model. See *TIBCO BusinessEvents Developer's Guide* for details.
2. In TIBCO BusinessEvents Studio Explorer, do one of the following:
 - Right-click the folder where you want to create the domain model and select **Import > TIBCO BusinessEvents > Domain Model**.
 - In TIBCO BusinessEvents Studio Explorer, select any item in the project entity and select **File > Import > TIBCO BusinessEvents > Domain Model**.
3. Click **Next**. You see the Import Domain Model Wizard.

If you invoked the import wizard by right-clicking a folder, that folder is selected as the parent folder. You can choose a different one as desired.
4. In the Domain Import Source field, select `DATABASE_TABLE`.
5. In the File Name field, enter a name for the domain model resource. Optionally enter a description.
6. In the Data Type field, select the appropriate data type for the domain model and click **Next**. You see the next wizard page.
7. In the JDBC Resource URI field, browse and select the JDBC Connection resource that connects to the database you want to use.

The connection information from the JDBC Connection resource displays. You can override it here as desired.

Click **Next**. You see a list of tables in the database. Expand the list of tables to see the columns that match the datatype specified in [step 6](#).
8. Do one of the following:
 - Select one or more columns, then click the **Create Domain for Selected Columns** button. Values of all columns are used for the domain model entries.
 - Click **Advanced** and enter an SQL query whose resultset is used to create the domain entries. This option enables you to make use of joins, where clauses, and so on. Then click **Execute Query**.
9. Click **Finish**. You see a message, "Domain Import Successful." Click **OK**.
10. You see the Domain Model editor. The column values appear as domain entries. You can add, edit, duplicate, and remove entries as appropriate.

Associating Domain Models with a Property

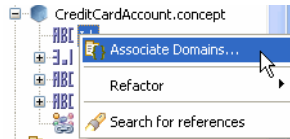
You can associate domain models with concept, event, and scorecard properties.

If domain models have the same data type as that of the properties, you can associate multiple domain models with multiple properties.

For example, an event and a concept have an `OrderID` property defined the same way, and both the `OrderID` properties use an `OrderIDModel` domain model. In this case, the `OrderIDModel` domain model is associated with the `OrderID` property of the event, as well as of the concept. However, the concept or the event property is associated only with the `OrderIDModel` domain model.

To Associate Domain Models with a Property

1. Do one of the following:
 - In TIBCO BusinessEvents Studio Explorer, expand the concept, event, or scorecard to display its properties. Right-click the desired property and select **Associate Domains**.



- In TIBCO BusinessEvents Studio Explorer, open the desired concept, event, or scorecard editor. In the Domain Model field of the desired property, click the browse icon.

The Associate Domains dialog appears.

2. Expand the project tree to display domain models, and select one or more as appropriate.

The displayed domain models have the same data type as that of the property.

3. Save the domain model resource.

Validating Data in Domain Models

You can validate duplicate entries and a mismatch of upper and lower range values while defining domain models.

Duplicate Domain Values Not Allowed

Each domain value must be unique. If you accidentally enter duplicate values, the Problems view displays helpful information. For example:



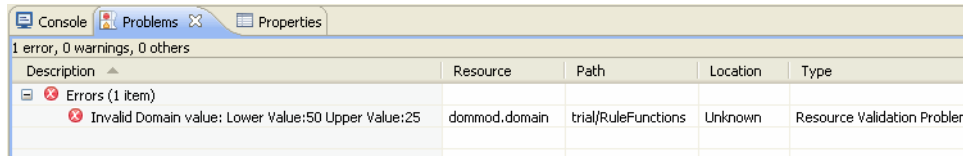
The screenshot shows the 'Problems' view with a table containing one error entry. The table has four columns: Description, Resource, Location, and Problem Code. The entry indicates a duplicate domain value for resource '6' at an unspecified location, with the problem code 'INVALID_ENTRY'.

Description	Resource	Location	Problem Code
✖ Duplicate Domain values.	6		INVALID_ENTRY

The Resource column displays the duplicate value.

Mismatching Range Values Not Allowed

If you accidentally enter mismatching values in the Lower and Upper fields for range values, the Problems view displays helpful information. For example:



The screenshot shows the 'Problems' view with a table containing one error entry. The table has five columns: Description, Resource, Path, Location, and Type. The entry indicates an invalid domain value because the lower value (50) is greater than the upper value (25) for the resource 'dommod.domain' at the path 'trial/RuleFunctions'. The location is 'Unknown' and the type is 'Resource Validation Problem'.

Description	Resource	Path	Location	Type
✖ Invalid Domain value: Lower Value:50 Upper Value:25	dommod.domain	trial/RuleFunctions	Unknown	Resource Validation Problem

Chapter 14 **Shared Resources**

This chapter describes a set of resources that can be used for various purposes in your projects.

Topics

- [Adding a Shared Resource, page 190](#)
- [HTTP Connection, page 192](#)
- [Identity Resource, page 195](#)
- [JDBC Connection, page 197](#)
- [JMS Connection, page 204](#)
- [JMS Application Properties, page 203](#)
- [JNDI Configuration, page 211](#)
- [Rendezvous Transport, page 214](#)

Adding a Shared Resource

See the section [Working with External Library and Custom Function Paths on page 10](#) for actions you must do to run, test, or debug projects that use third-party libraries in TIBCO BusinessEvents Studio.

If you are adding a JDBC connection for a backing store, see the procedure, Add a JDBC Connection Resource (Now or Later), in *TIBCO BusinessEvents Administration*.

To Add a Shared Resource

1. In Studio Explorer, right click the folder where you want to store the shared resource and select **New > Other**.
2. In the Select a Wizard dialog, expand TIBCO Shared Resources and select the resource type you want to add (see list in [step 3](#)) and click **Next**.
3. In the new resource wizard enter a name in the File Name field and click **Finish**.
4. You see the editor for the shared resource you selected. See the following sections for guidelines on completing the fields:
 - [HTTP Connection, page 192](#)
 - [Identity Resource, page 195](#)
 - [JDBC Connection, page 197](#), and also see [Enabling the Test Connection Feature, page 190](#)
 - [JMS Connection, page 204](#), and also see [Enabling the Test Connection Feature, page 190](#)
 - [JMS Application Properties, page 203](#)
 - [JNDI Configuration, page 211](#)
 - [Rendezvous Transport, page 214](#)
5. (Optional) View the Source tab. The source tab shows the XML source format for the shared resource. You can edit the XML source directly, but this is not recommended because of the risk of error. The resource editor provides validation checks.

Enabling the Test Connection Feature

To make the Test Connection feature work for JMS Connection and JDBC Shared Connection shared resources, copy their relevant JAR files to the following

directory:

BE_HOME/lib/ext/tpcl

(This directory is referenced in the extended classpath in the file

BE_HOME/studio/eclipse/configuration/studio.tra.)

For TIBCO Enterprise Message Service copy the `jms.jar` and `tibjms.jar` to the above location.

For DBMS products, copy the supported driver file you are using for the DBMS product.

HTTP Connection

The HTTP Connection resource describes the characteristics of the connection used to receive incoming HTTP requests. This section provides a reference to the fields. For procedures see [Adding a Shared Resource on page 190](#).

The HTTP Connection resource can specify that the HTTPS (secure sockets layer or SSL) protocol must be used by clients. If this is enabled, you can configure the SSL parameters for the HTTP server using the Configure SSL Button. See [Configure SSL on page 193](#) for more information.



If you have multiple HTTP Connection resources specified by multiple HTTP Receiver process starters, the HTTP servers require that all of the connections must be valid to initialize all HTTP Receivers. Therefore, make certain that all HTTP Connection resources have valid configurations before testing or deploying the project.

Wizard and Configuration Tab

The Wizard and Configuration tab have the following fields.

Field	Global Var?	Description
Wizard		
File Name	No	<p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p> <p>Unlike other resource identifiers, however, shared resource identifiers can have spaces in the name.</p> <p>This field appears on the New HTTP Connection Wizard. The name then appears in the title of the resource.</p>
Configuration		
Description	No	Short description of the resource.

Field	Global Var?	Description (Cont'd)
Host	Yes	<p>Specifies the name of the host that accepts the incoming requests. For machines that have only one network card, the default value <code>localhost</code> specifies the current machine. For machines that have more than one network card, this field specifies the host name of the card that will be used to accept incoming HTTP requests.</p> <p>Note Only one HTTP server can be started on each port. Therefore, if you have a machine with multiple network cards, make certain that all HTTP Connection resources that use the same host name specify different port numbers.</p> <p>Note If there is more than one network card on the machine, and the <code>serverType</code> field is set to TOMCAT, and you specify <code>localhost</code> in this field, then all network cards on the machine will listen for incoming HTTP requests on the specified port. If the <code>serverType</code> field is set to BUILT-IN, then only the loopback address will be bound to the HTTP port.</p>
Port	Yes	Port number on which to listen for incoming HTTP requests.
Use SSL	No	<p>Specifies whether incoming requests must use the HTTPS (secure socket layer, or SSL) protocol. This protocol authenticates the server to the client, and optionally authenticates the client to the server.</p> <p>Enabling this field allows you to specify SSL parameters with the Configure SSL button (see Configure SSL on page 193).</p>

Configure SSL

The SSL Configuration for HTTPS Connections dialog (accessed by configuring the Configure SSL button) allows you to specify the SSL parameters for the HTTP connection.

The following are the fields in the SSL Configuration for HTTPS Connections dialog:



The HTTPComponent server type does not support Entrust based SSL.

Field	Description
Requires Client Authentication	<p>Checking this field requires clients to present their digital certificate before connecting to the HTTP server.</p> <p>When this field is checked, the Trusted Certificates Folder becomes enabled so that you can specify a location containing the list of trusted certificate authorities.</p>
Trusted Certificates Folder	<p>This field is only applicable when the Requires Client Authentication field is checked.</p> <p>This field specifies a folder in the project containing one or more certificates from trusted certificate authorities. This folder is checked when a client connects to ensure that the client is trusted. This prevents connections from rogue clients.</p>
Identity	<p>This is an Identity resource that contains the HTTP server's digital certificate and private key. See Identity Resource on page 195 for more information.</p>
Strong Cipher Suites Only	<p>When checked, this field specifies that the minimum strength of the cipher suites used can be specified with the <code>bw.plugin.security.strongcipher.minstrength</code> custom engine property. See <i>TIBCO ActiveMatrix BusinessWorks Administration</i> for more information about this property. The default value of the property disables cipher suites with an effective key length below 128 bits.</p> <p>When this field is unchecked, only cipher suites with an effective key length of up to 128 bits can be used.</p>

Identity Resource

The Identity resource encapsulates information that may be used to authorize a user, connection, and so forth. The information you supply changes depending on the type of Identity resource you want to use. This section provides a reference to the fields. For procedures see [Adding a Shared Resource on page 190](#).

The identity certificate location, its type, and password can be specified as global variables.

Wizard and Configuration Tab

The New Identity Resource Wizard and the Configuration tab of the Identity Resource have the following fields.

Field	Global Var?	Description
Wizard		
File Name	No	<p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p> <p>Unlike other resource identifiers, however, shared resource identifiers can have spaces in the name.</p> <p>This field appears in the New Identity Resource Wizard. The name then appears in the title of the resource.</p>
Configuration		
Description	No	Short description of the resource.
Type	No	The type of identity resource: Identify File, Certificate/Private Key, or Username/Password (the default). See sections below for details.
Identity File		
Use this option if the certificate includes the private key information in the same file.		
URL	No	Location of the certificate (which includes the private key).

Field	Global Var?	Description (Cont'd)
File Type	No	Choose the certificate file type from the drop-down list: Entrust JCEKS JKS PEM PKCS12
Password	Yes	Password for the certificate.
Certificate/Private Key Identity Use this option if the private key and the certificate are in two separate files.		
Certificate URL	No	Location of the certificate. Click the browse icon or type in a URL.
Key URL	No	Location of the private key file associated with the certificate.
Key Password	Yes	Password used for private key.
Username/Password Use this option if you want to use a username and password for authentication and don't want to use a certificate.		
Username	No	Name of the user for this identity.
Password	Yes	Password for the user for this identity.

JDBC Connection

The JDBC Connection resource describes a JDBC connection. JDBC connections are used with backing store (see *TIBCO BusinessEvents Administration*) and with database concepts (available in the TIBCO BusinessEvents Data Modeling add-on). This section provides a reference to the fields. For procedures see [Adding a Shared Resource on page 190](#).

Wizard and Configuration Tab

The Wizard and Configuration tab have the following fields.

Field	Global Var?	Description
Wizard		
File Name	No	<p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p> <p>Unlike other resource identifiers, however, shared resource identifiers can have spaces in the name.</p>
Configuration		
Description	No	Short description of the resource.
Connection Type	No	<p>Specifies the kind of JDBC connection you wish to create. The connection type can be one of the following:</p> <ul style="list-style-type: none"> • JDBC • JNDI • XA <p>The type of connection determines the other configuration fields that appear.</p>

Field	Global Var?	Description (Cont'd)
JDBC Connection Type Configuration Fields		
JDBC Driver	Yes	<p>The name of the JDBC driver class. You can select from a list of drivers or enter a driver manually. Listed drivers are as follows:</p> <pre>oracle.jdbc.OracleDriver (thin)</pre> <pre>com.ibm.db2.jcc.DB2Driver (supported for database concepts only)</pre> <pre>com.microsoft.sqlserver.jdbc.SQLServerDriver</pre> <pre>com.mysql.jdbc.Driver</pre> <pre>com.sybase.jdbc3.jdbc.SybDriver</pre> <p>When you select a driver, the Database URL field is populated with a template for the URL for the driver.</p>
Database URL	Yes	<p>The URL to use to connect to the database. A template of the URL is supplied for the selected JDBC driver. You must supply the portions of the URL that are in angle brackets, for example, the host, port number, and database instance name.</p>
Maximum Connections	Yes	<p>The maximum number of database connections to allocate. The default maximum is 10. The minimum value that can be specified is 1.</p> <p>See Connection Pooling on page 201 for more details, including related settings that override this setting.</p>
User Name	Yes	<p>User name to use when connecting to the database.</p>
Password	Yes	<p>Password to use when connecting to the database.</p>
Login Timeout	Yes	<p>Time (in seconds) to wait for a successful database connection. Only JDBC drivers that support connection timeouts can use this configuration field. If the JDBC driver does not support connection timeouts, the value of this field is ignored. Most JDBC drivers support connection timeouts.</p>
JNDI Connection Type Configuration Fields		
JNDI DataSource Name	Yes	<p>The JNDI name specified for the DataSource.</p>

Field	Global Var?	Description (Cont'd)
Use Shared JNDI Configuration	No	<p>When this field is checked, the JNDI Configuration field appears, allowing you to choose a shared JNDI Configuration resource.</p> <p>When this checkbox is unchecked, configuration fields appear.</p>
JNDI Configuration	No	This field only appears when the Use Shared JNDI Configuration field is checked. This field allows you to choose a JNDI Configuration shared resource that specifies the JNDI connection information.
JNDI Context Factory	No	The initial context factory class for accessing JNDI. (<code>javax.naming.Context.INITIAL_CONTEXT_FACTORY</code>). You can choose from the drop down list of supported classes, or you can type in a different <code>InitialContextFactory</code> class name.
JNDI Context URL	Yes	<p>The URL to the JNDI service provider (<code>javax.naming.Context.PROVIDER_URL</code>). An example URL is provided when one of the supported JNDI context factory classes is selected.</p> <p>See your JNDI provider documentation for the syntax of the URL.</p>
JNDI User Name	Yes	The user name to use when logging into the JNDI server (<code>javax.naming.Context.SECURITY_PRINCIPAL</code>). If the JNDI provider does not require access control, this field can be empty.
JNDI Password	Yes	The password for logging into the JNDI server (<code>javax.naming.Context.SECURITY_CREDENTIALS</code>). If the JNDI provider does not require access control, this field can be empty.

XA Connection Type Configuration Fields

XA DataSource Class	No	<p>The XA DataSource class.</p> <p>Note: TIBCO BusinessEvents attempts to find the class. However, you may need to add the location of the class to the CLASSPATH environment variable in the <code>be-engine.tra</code> file.</p>
---------------------	----	---

Field	Global Var?	Description (Cont'd)
Database URL	Yes	<p>The URL to use to connect to the database. A template of the URL is supplied for the selected XA DataSource class. You must supply the portions of the URL that are in angle brackets (<>). For example, if you select the <code>tibcosoftwareinc.jdbcx.oracle.OracleDataSource</code>, the following appears in the Database URL field:</p> <pre>jdbc:tibcosoftwareinc:oracle://<host>: <port#>;SID=<db_instancename></pre> <p>You must supply the host, port number, and database instance name in the URL.</p>
Maximum Connections	Yes	<p>The maximum number of database connections to allocate. The default maximum is 10. The minimum value that can be specified is 1.</p> <p>See Connection Pooling on page 201 for more information about this field.</p> <p>Default is 5.</p>
User Name	Yes	User name to use when connecting to the database.
Password	Yes	Password to use when connecting to the database.

Connection Pooling

TIBCO BusinessEvents creates a pool of JDBC connections for every JDBC Connection shared resource that uses the JDBC connection type. The maximum size of this pool is specified by the Maximum Connections configuration field.

Resources such as backing stores and database concepts that use this JDBC Connection resource are given a connection from the pool. Once the maximum number of connections is reached, resources requesting a connection cannot proceed. Once a connection is freed by an activity, the connection is returned to the pool. Connections that are left open will eventually time out and be closed. These connections can be reopened at a later time, until the maximum number of connections specified in this field is reached.

For backing store connections, you can use additional connection pool properties, which override equivalent settings in the JDBC Connection resource. See Chapter 15, JDBC Backing Store Setup in *TIBCO BusinessEvents Administration* for details.

Test Connection Button

The Test Connection button allows you to test the connection specified in the configuration of this resource. See [Enabling the Test Connection Feature on page 190](#) for a step you must take to enable the connection to work.

JMS Application Properties

The JMS Application Properties resource describes any JMS message properties that a JMS application expects. These properties can then be added.

For procedures see [Adding a Shared Resource on page 190](#).

Wizard and Configuration Tab

The Wizard and Configuration tab have the following fields.

Field	Global Var?	Description
Wizard		
File Name	No	The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275 . Unlike other resource identifiers, however, shared resource identifiers can have spaces in the name.
Description	No	Short description of the resource.
Configuration		
Properties Table A table listing any application-specific properties. Use the + and x buttons to the right of the table to add and delete properties. Use the up and down arrow buttons to move selected properties to the desired location in the table.		
Property Name	Yes	Name of the column.
Type	Yes	The datatype of the property. Double-click the cell to cause a drop down list of valid JMS datatypes to appear, and choose a value.
Cardinality	No	Specifies whether the property is optional or required. Double-click the cell to cause a drop down list of two values to appear and select optional or required.

JMS Connection

JMS Connection resource describes a JMS connection. This section provides a reference to the fields. For procedures see [Adding a Shared Resource on page 190](#).

Wizard and Configuration Tab

The Wizard and Configuration tab have the following fields.

Field	Global Var?	Description
Wizard		
File Name	No	<p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p> <p>Unlike other resource identifiers, however, shared resource identifiers can have spaces in the name.</p> <p>This field appears on the New JMS Connection Wizard. The name then appears in the title of the resource.</p>
Configuration Tab		
Description	No	Short description of the resource.
User Name	Yes	<p>User name to use when logging into the JMS server.</p> <p>If the JMS provider does not require access control, this field can be empty.</p> <p>Not all JMS servers require user names and passwords. Refer to your JMS provider documentation and consult your system administrator to determine if your JMS server requires a user name and password.</p>
Password	Yes	<p>Password to use when logging into the JMS server.</p> <p>If the JMS provider does not require access control, this field can be empty.</p>

Field	Global Var?	Description (Cont'd)
Auto-Generate Client ID	Yes	<p>Checking this field specifies you wish to automatically generate the client ID if no client ID is specified or if the specified ID is already in use. When this field is checked, if a value is specified in the Client ID field, an attempt is made to use the specified value. However, if the specified value is already in use, a new client ID is generated.</p> <p>If this field is not checked, then the value specified in the Client ID field is used. If no value is specified in the Client ID field, then no client ID is set. If the value specified in the Client ID field is already in use, an exception is thrown.</p>
Client ID	Yes	<p>Client ID for the connection. Typically JMS providers have a provider-specific format for client IDs. See your JMS provider's documentation for more information about client IDs. Each connection must use a unique Client ID.</p> <p>You cannot use the same JMS Connection resource for accessing both topics and queues. You should create separate JMS Connection resources if you wish to access both topic and queue destinations.</p>
Use SSL	No	<p>Specifies whether you wish to use SSL for the connection to the JMS server. SSL is used when the Use SSL checkbox is checked. Click the Configure SSL button to configure the SSL connection parameters.</p> <p>Note: SSL is supported only when using TIBCO Enterprise Message Service.</p> <p>See Configure SSL on page 208 for more information.</p>
Use JNDI for Connection Factory	Yes	<p>Specifies whether JNDI should be used to look up the ConnectionFactory object. If this field is unchecked, the Provider URL and Use XA Connection Factory fields appear. If this field is checked, JNDI configuration fields appear.</p>
Provider URL	Yes	<p>This field is only available when the Use JNDI for Connection Factory field is unchecked.</p> <p>This is the URL to use to connect to the JMS server.</p>
Use XA Connection Factory	No	<p>This field is only available when the Use JNDI for Connection Factory field is unchecked.</p> <p>When checked, this field specifies that an XA connection factory is to be used.</p>

Field	Global Var?	Description (Cont'd)
Connection Factory SSL Password	Yes	<p>This field is only available when the Use SSL checkbox is checked, and the User Shared JNDI Configuration checkbox is unchecked.</p> <p>The SSL configuration is specified in the ConnectionFactory object, except for the client SSL password.</p> <p>You can specify your client SSL password in this field, or you can leave this field empty. If your password is not specified, the private key password is used.</p>
Use Shared JNDI Configuration	No	<p>When this field is checked, the JNDI Configuration field appears. It allows you to choose a JNDI Configuration.</p> <p>When this field is unchecked, JNDI configuration fields appear.</p>
JNDI Configuration	No	<p>This field only appears when the Use Shared JNDI Configuration field is checked.</p> <p>This field allows you to choose a JNDI Configuration that specifies the JNDI connection information. See JNDI Configuration on page 211 for more information.</p>
JNDI Context Factory	Yes	<p>This field only appears when the Use Shared JNDI Configuration field is unchecked. Required. The initial context factory class for accessing JNDI. (<code>javax.naming.Context.INITIAL_CONTEXT_FACTORY</code>). You can choose from the drop down list of supported classes or you can type in a different InitialContextFactory class name.</p> <p>Note: TIBCO BusinessEvents attempts to find the class. However, you may need to add the Java file supplied by your JNDI service provider to the CLASSPATH environment variable to use JNDI.</p>
JNDI Context URL	Yes	<p>This field only appears when the Use Shared JNDI Configuration field is unchecked. Required. This is the URL to the JNDI service provider (<code>javax.naming.Context.PROVIDER_URL</code>). An example URL is provided when one of the supported JNDI context factory classes is selected.</p> <p>See your JNDI provider documentation for the syntax of the URL.</p>

Field	Global Var?	Description (Cont'd)
JNDI User Name	Yes	This field only appears when the Use Shared JNDI Configuration field is unchecked. User name to use when logging into the JNDI server (<code>javax.naming.Context.SECURITY_PRINCIPAL</code>). If the JNDI provider does not require access control, this field can be empty.
JNDI Password	Yes	This field only appears when the Use Shared JNDI Configuration field is unchecked. Password to use when logging into the JNDI server (<code>javax.naming.Context.SECURITY_CREDENTIALS</code>). If the JNDI provider does not require access control, this field can be empty.

Test Connection Button

The Test Connection button allows you to test the connection specified in the configuration of this resource. See [Enabling the Test Connection Feature on page 190](#) for a step you must take to enable the connection to work.

Note that this button is only enabled if JNDI is not used (that is, the Use JNDI for Connection Factory field is unchecked) or if the required JNDI fields on the Configuration and Advanced tab are supplied when JNDI is used.

Advanced Tab

The Advanced tab has the following fields.

Field	Global Var?	Description
Topic Connection Factory	Yes	<p>This field is only available when the Use JNDI for Connection Factory field on the Configuration tab is checked.</p> <p>The <code>TopicConnectionFactory</code> object stored in JNDI. This object is used to create a topic connection with a JMS application.</p> <p>See your JNDI provider documentation for more information about creating and storing <code>TopicConnectionFactory</code> objects.</p>

Field	Global Var?	Description (Cont'd)
Queue Connection Factory	Yes	<p>This field is only available when the Use JNDI for Connection Factory field on the Configuration tab is checked.</p> <p>The <code>QueueConnectionFactory</code> object stored in JNDI. This object is used to create a queue connection with a JMS application.</p> <p>See your JNDI provider documentation for more information about creating and storing <code>QueueConnectionFactory</code> objects.</p>
Optional JNDI Properties	No	<p>Any additional properties to supply for the connection. You specify a name, datatype, and value for each property.</p> <p>These properties are typically vendor-specific. See your JNDI provider documentation for more information about the available properties.</p>

Configure SSL

The SSL Configuration button allows you to configure the SSL connection parameters.



When using JNDI to lookup the JMS Connection factory, the parameters `ssl_identity` and `ssl_verify_host` must be specified in the `factories.conf` file of the Enterprise Message Service server.

The following table describes the SSL Configuration dialog.

Field	Description
Trusted Certificates Folder	<p>Location of the trusted certificates on this machine. The trusted certificates are a collection of certificates from servers to whom you will establish connections. If the server you wish to establish a connection presents a certificate that does not match one of your trusted certificates, the connection is refused.</p> <p>This prevents connections to unauthorized servers.</p> <p>Trusted certificates must be imported into a folder, and then you can select the folder in this field.</p>

Field	Description (Cont'd)
Identity	<p>The location of the client certificate. This is a resource contained in the General palette.</p> <p>You only need to specify the client certificate when the JMS server requires client authentication.</p> <p>See Identity Resource on page 195 for more information.</p>
Trace	<p>Specifies whether SSL tracing should be enabled during the connection. If checked, the SSL connection messages are logged and sent to the console.</p>
Debug Trace	<p>Specifies whether SSL debug tracing should be enabled during the connection. Debug tracing provides more detailed messages than standard tracing.</p>
Verify Host Name	<p>Specifies whether you wish to verify that the host you are connecting to is the expected host. The host name in the host's digital certificate is compared against the value you supply in the Expected Host Name field. If the host name does not match the expected host name, the connection is refused.</p> <p>Note: The default context factories for TIBCO Enterprise Message Service automatically determine if host name verification is necessary. If you are using a custom implementation of the context factories, your custom implementation must explicitly set the verify host property to the correct value. For example:</p> <pre>com.tibco.tibjms.TibjmsSSL.setVerifyHost (false)</pre>

Field	Description (Cont'd)
Expected Host Name	<p>Specifies the name of the host you are expecting to connect to. This field is only relevant if the Verify Host Name field is also checked.</p> <p>If the name of the host in the host's digital certificate does not match the value specified in this field, the connection is refused.</p> <p>This prevents hosts from attempting to impersonate the host you are expecting to connect to.</p>
Strong Cipher Suites Only	<p>When checked, this field specifies that the minimum strength of the cipher suites used can be specified with the <code>bw.plugin.security.strongcipher.minstrength</code> custom engine property. See <i>TIBCO ActiveMatrix BusinessWorks Administration</i> for more information about this property. The default value of the property disables cipher suites with an effective key length below 128 bits.</p> <p>When this field is unchecked, only cipher suites with an effective key length of up to 128 bits can be used.</p>

JNDI Configuration

The JNDI Configuration shared resource provides a way to specify JNDI connection information that can be shared by other resources. This resource can be specified in any resource that permits JNDI connections. For example, [JDBC Connection on page 197](#) and [JMS Connection on page 204](#) can use JNDI connections. This section provides a reference to the fields. For procedures see [Adding a Shared Resource on page 190](#).

Wizard and Configuration Tab

The Wizard and Configuration tab have the following fields.

Field	Global Var?	Description
Wizard		
Name	No	<p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p> <p>Unlike other resource identifiers, however, shared resource identifiers can have spaces in the name.</p> <p>This field appears on the New JNDI Connection Wizard. The name then appears in the title of the resource.</p>
Configuration		
Description	No	Short description of the resource.
JNDI Context Factory	No	<p>The initial context factory class for accessing JNDI. (<code>javax.naming.Context.INITIAL_CONTEXT_FACTORY</code>). You can choose from the drop down list of supported classes.</p> <p>Note: TIBCO BusinessEvents attempts to find the class. However, you may need to add the JAR file supplied by your JNDI service provider to the CLASSPATH environment variable to use JNDI.</p>

Field	Global Var?	Description (Cont'd)
JNDI Context URL	Yes	<p>The URL to the JNDI service provider (<code>javax.naming.Context.PROVIDER_URL</code>). An example URL is provided when one of the supported JNDI context factory classes is selected.</p> <p>See your JNDI provider documentation for the syntax of the URL.</p>
JNDI User Name	Yes	<p>User name for logging into the JNDI server (<code>javax.naming.Context.SECURITY_PRINCIPAL</code>). If the JNDI provider does not require access control, this field can be empty.</p>
JNDI Password	Yes	<p>Password for logging into the JNDI server (<code>javax.naming.Context.SECURITY_CREDENTIALS</code>). If the JNDI provider does not require access control, this field can be empty.</p>

Advanced Section

The Advanced section has the following fields.

Field	Global Var?	Description
Validate JNDI Security Context	No	<p>Some application servers store the security context on the thread used to establish the JNDI connection (at the time of this release, only the WebLogic application server does this). In that case, the first activity to use this resource establishes the security context, then subsequent activities use the same security context, unless this field is checked. Checking this field ensures that each activity that uses this resource examines the security context to determine if the activity uses the same security context as the security context established on the thread. If they are different, the activity's configured security context is used.</p> <p>Checking this field causes additional overhead for activities that use this resource. Therefore, only check this field when necessary.</p>

Field	Global Var?	Description (Cont'd)
Optional JNDI Properties	No	The table in this field contains optional properties to pass to the JNDI server. Use the +, X, and arrow keys to add, delete, and move properties in the list. Each property requires the property name, the datatype for the property, and the value for the property. See the documentation for your JNDI provider for more information about properties that can be passed to the JNDI server.

Rendezvous Transport

The Rendezvous Transport resource describes a TIBCO Rendezvous transport. See the TIBCO Rendezvous documentation for more information about specifying these fields.

This section provides a reference to the fields. For procedures see [Adding a Shared Resource on page 190](#).



Wizard and Configuration Tab

The Wizard and Configuration tab have the following fields.

Field	Global Var?	Description
Wizard		
Name	No	<p>The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275.</p> <p>Unlike other resource identifiers, however, shared resource identifiers can have spaces in the name.</p> <p>This field appears on the New Rendezvous Transport Wizard. The name then appears in the title of the resource.</p>
Configuration		
Description	No	Short description of the resource.
Daemon	Yes	<p>In the case of TIBCO Rendezvous daemon running on the same machine as TIBCO BusinessEvents engine, this is not specified. If Rendezvous is running on a different machine, then the Daemon field is specified as the remote host name followed by the socket number.</p> <p>For example:</p> <pre>ssl:acct:5785</pre>

Field	Global Var?	Description (Cont'd)
Network	Yes	This field contains the host name, IP address, network name, or interface name. For example: ; 224 . 34 . 103 . 4
Service	Yes	The TIBCO Rendezvous service name, specified in one of the following formats: <i>service name</i> or <i>port number</i> .
Use SSL	No	The Use SSL checkbox specifies that Secure Sockets Layer (SSL) should be used when communicating with the TIBCO Rendezvous daemon. When this field is checked, the Configure SSL button is enabled. See Configure SSL Button on page 215 for more information about configuring SSL parameters.

Configure SSL Button

The Configure SSL button allows you to configure the SSL parameters for communicating with the TIBCO Rendezvous daemon. See the TIBCO Rendezvous documentation for more information about how SSL is configured for TIBCO Rendezvous daemons and clients.

The SSL Configuration for TIBCO Rendezvous dialog has the following fields:

Field	Description
Daemon Certificate	<p>File containing one or more certificates from trusted certificate authorities. This file is checked when connecting to a daemon to ensure that the connection is to a daemon that is trusted. This prevents connections to rogue TIBCO Rendezvous daemons that attempt to impersonate trusted daemons.</p> <p>You can retrieve a daemon's certificate using the administration interface in TIBCO Rendezvous. See the TIBCO Rendezvous documentation for more information about obtaining certificates through the administration interface. Once retrieved, you can select a folder in your project and import this certificate into the folder using the Tools > Trusted Certificates > Import Into PEM Format menu item.</p>

Field	Description (Cont'd)
Identity	<p>This is an Identity resource used to authenticate to the TIBCO Rendezvous daemon. The Browse button allows you to select from a list of appropriately configured Identity resources. Only Identity resources whose Type field is set to Identity File or Username/Password are listed.</p> <p>See Identity Resource on page 195 for more information.</p>

Advanced Section

The Advanced section of the UI has the following fields.

Field	Global Var?	Description
RV Type	No	<p>The type of TIBCO Rendezvous connection to use. This can be reliable (standard RV transport), certified (RVCM), or Distributed Queue (RVCMQ).</p> <p>The fields of the Advanced tab correspond to the value selected for this field.</p>
Certified Transport		
CM Name	Yes	The name of the delivery-tracking session. This name is in the same format as TIBCO Rendezvous subject names.
Ledger File	Yes	The name and location of the persistent ledger file that tracks certified messages. If not specified, the certified message ledger is kept in process memory only.
Sync Ledger File	Yes	Specifies whether to keep the ledger file synchronous with the current messages.
Relay Agent	Yes	Name of the relay agent to use. Relay agents are useful when clients are disconnected from the network from time to time. The relay agents store inbound certified messages and labeled messages (and other messages related to certified delivery features) on behalf of their disconnected client programs. When a client is connected, it receives inbound messages immediately.

Field	Global Var?	Description (Cont'd)
Require Old Message	Yes	Check this box if you wish to require the retention of messages for which delivery has not been confirmed. These messages will be resent.
Message Timeout (sec)	Yes	The time limit (in seconds) for certified message delivery.
Distributed Queue		
CMQ Name	Yes	The name of the distributed queue. This name is in the same format as TIBCO Rendezvous subject names.
Worker Weight	Yes	The weight of the worker (this pertains to the worker processing queue requests). Relative worker weights assist the scheduler in assigning tasks. When the scheduler receives a task, it assigns the task to the available worker with the greatest worker weight.
Worker Tasks	Yes	Sets the task capacity for the worker (this pertains to the worker processing queue requests). Task capacity is the maximum number of tasks that a worker can accept. When the number of accepted tasks reaches this maximum, the worker cannot accept additional tasks until it completes one or more of them.
Worker Complete Time	Yes	The amount of time the scheduler waits for a worker process to complete. If the worker process does not complete in the specified period, the scheduler reassigns the message to another worker.
Scheduler Weight	Yes	Weight represents the ability of this member to fulfill the role of scheduler, relative to other members with the same name. Cooperating distributed queue transports use relative scheduler weight values to elect one transport as the scheduler; members with higher scheduler weight take precedence. Acceptable values range from 1 to 65535.
Scheduler Heartbeat	Yes	The scheduler sends heartbeat messages at this interval (in seconds). All members with the same name must specify the same value for this parameter. The value must be strictly positive.

Field	Global Var?	Description (Cont'd)
Scheduler Activation	Yes	When the heartbeat signal from the scheduler has been silent for this interval (in seconds), the member with the greatest scheduler weight takes its place as the new scheduler. All members with the same name must specify the same value for this parameter. The value must be positive

Chapter 15 Rules and Rule Functions

This chapter explains how to work with rules and rule functions.

For documentation on catalog functions see [Chapter 16, Functions, page 247](#)

Topics

- [Overview of Rules and Rule Functions, page 220](#)
- [Adding a Rule, page 222](#)
- [Rule Editor Reference, page 226](#)
- [Adding a Rule Function, page 228](#)
- [Rule Function Resource Reference, page 231](#)
- [Using Variables and Functions in the Rule Editor, page 233](#)
- [Using Priority and Rank to Control Order of Rule Execution, page 237](#)
- [Using the Quick Fix Feature in the Rule Editor, page 239](#)
- [Tips for Working in the Rule Editor, page 241](#)
- [Event Preprocessors, page 243](#)
- [Transaction Error Handler Rule Function, page 244](#)

Overview of Rules and Rule Functions

This chapter explains how to work with the rule editor to create rules and rule functions. It also explains some specific uses for rule functions.

To learn how rules are executed at runtime and for other background information see *TIBCO BusinessEvents Architect's Guide*, especially the section Understanding Conflict Resolution and Run to Completion Cycles.

See Also

- Chapter 4, Rules and Functions in *TIBCO BusinessEvents Architect's Guide*
- [Chapter 16, Functions, on page 247](#)
- [Chapter 17, Rule Language Grammar, on page 273](#)
- [Chapter 18, Rule Language Datatypes, on page 291](#)
- [Chapter 24, Diagrams, on page 421](#) for information on using rule dependency and sequence diagrams.

Form-based and Source Rule Editors

When you work with rules and rule functions, you can choose how to work:

- Using a form-based rule editor, similar to the rule editor in earlier versions of TIBCO BusinessEvents
- Using a source editor, which is closer to a Java programming environment.

You can switch between editors and changes made in one editor are reflected in the other one. You can't switch from the source editor to the form editor if there are validation errors in the code.

Rule Components

A TIBCO BusinessEvents rule has three components:

- **Declaration** — Use the declaration to declare which concepts and events the rule will depend on, and the names by which instances of these entities can be referred to in the conditions and actions. Aliases must be valid identifiers. Declaring multiple terms of the same type allows the rule to consider multiple instances of the corresponding Entity.

- **Conditions** — Each statement in the condition must evaluate to a boolean value. All of these statements must be true for the rule's action to be executed. Assignments and calls to certain functions are disallowed in the condition.
- **Actions** — List of statements that will be executed, when the rule is fired, for each combination of terms that matches all the conditions.

Effect of Cache Only Cache Mode

When using Cache Only cache mode for one or more entities, you must consider how to handle the cache-only entities when you write rules and preprocessor rule functions. See *Working With Cache Modes and Loading Cache Only Objects into the Rete Network* in *TIBCO BusinessEvents Architect's Guide*.

Adding a Rule

These instructions focus on the form-based editor and mention the equivalent settings in the source editor. Adapt the instructions if you are using the source editor.

Figure 2 Rule Form Editor

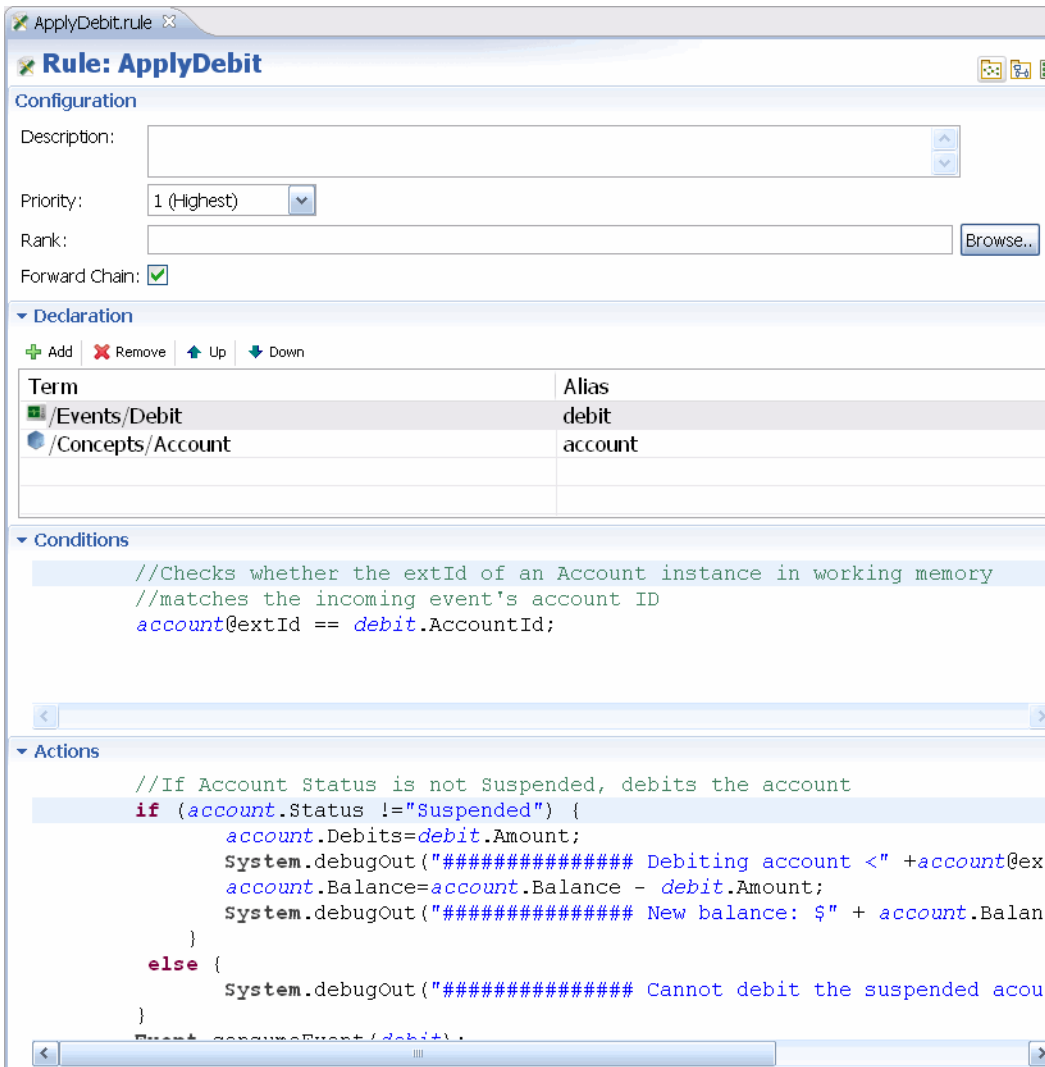


Figure 3 Rule Source Editor

```

* @description
* @author
*/
rule Rules.ProcessDebits.ApplyDebit {
    attribute {
        priority = 1;
        forwardChain = true;

```

Optionally add entry for Rank as needed (Rank=RuleFunction), or enter in Form view.

```

    }
    declare {
        Events.Debit debit;
        Concepts.Account account;
    }
    when {
        //Checks whether the extId of an Account instance in working memory
        //matches the incoming event's account ID
        account@extId == debit.AccountId;
    }
    then {
        //If Account Status is not Suspended, debits the account
        if (account.Status != "Suspended") {
            account.Debits=debit.Amount;
            System.debugOut("##### Debiting account <" +account@extId+ "> by $"
+debit.Amount);
            account.Balance=account.Balance - debit.Amount;
            System.debugOut("##### New balance: $" + account.Balance);
        }
        else {
            System.debugOut("##### Cannot debit the suspended account <"
+account@extId + ">");
        }
        Event.consumeEvent(debit);
    }
}

```

To Add a Rule

See [Rule Editor Reference on page 226](#) for details on the settings.

1. In Studio Explorer, right click the folder where you want to store the rule and select **New > Rule**. You see the New Rule Wizard.

2. In the Rule Name field, type a name for the rule. In the Description field, type a description as desired. (In the source editor the description appears in the `@description` line of the comments at the top of the editor.)



You can't change the name in the editor. To change the name of any project element, right-click the element in Studio Explorer and select **Refactor > Rename**. See [Chapter 4, Element Refactoring Operations, on page 45](#) for more details.

3. Click **Finish**. By default, you see the source rule editor on opening the editor. It shows the outline for a rule's source code. Click the **Form** tab at the bottom of the editor to use the form editor.

At any time you can click the Form and Source tabs at the bottom of the editor to switch between the form-based editor and the source editor as you work, depending on your preference.



Rule Editor Preference To set which mode the editor uses upon first opening, go to Window > Preferences > TIBCO BusinessEvents > Rules and check or uncheck the following checkbox as desired: **Initially show 'Form' tab in Rule Editor**

4. In the Form editor Configuration panel, add or edit a description as desired. (In the source editor the description appears in the * `@description` line of the comments at the top of the editor.)
5. To control the order of rule execution:
 - a. If you want to control the order in which rules execute, set the Priority field accordingly. Highest priority is 1. (In the source editor set `priority = n` where `n` is the priority number.)

If you want to also control the order in which rules with the same *priority* execute, set the Rank field accordingly. Browse to and select the rule function you created for this purpose. (If you want to use the source editor, add `Rank=RuleFunction` in the list of rule attributes.) See [Using Priority and Rank to Control Order of Rule Execution on page 237](#) for details.
6. To disable forward rule chaining (the default behavior) uncheck the Forward Chain checkbox. (In the source editor set `forwardChain = false`.)

7. In the Declarations panel (equivalent to the `declare` statements in the source editor), drag an ontology entity into the Declarations area, OR do the following:
 - a. Click **Add** to add resources that you will be using in your rule. You see the Select Rule Declaration Arguments dialog.
 - b. In the upper half of the Select Rule Declaration Arguments dialog, select the kind of entity you want to use.
 - c. In the lower half of the dialog, select a resource from the filtered ontology tree, and click **OK**. Your selection appears in the Declarations list. TIBCO BusinessEvents assigns an alias to the resource. You can edit the alias.
 - d. To re-order the declarations, highlight a declaration and click the up or down arrow to move it. This is relevant only in rule functions, to order the arguments.
8. In the Conditions panel (equivalent to the `when` statements in the source editor), write condition statements (in the TIBCO BusinessEvents rule language).

Each line is a complete statement. Each condition must evaluate to a Boolean value. Each line is joined to the others with an implicit AND operator. All of a rule's conditions must evaluate to true for the conditions to be satisfied. See Order of Evaluation of Rule Conditions in *TIBCO BusinessEvents Architect's Guide* for more information.

See [Using Variables and Functions in the Rule Editor on page 233](#) and [Tips for Working in the Rule Editor on page 241](#) for more information on working in the rule editor.
9. In the Actions panel (equivalent to the `then` statements in the source editor), write action statements (in the TIBCO BusinessEvents rules language).
10. Save the resource.

Rule Editor Reference



The rule editor and rule function editor are similar. This section focuses on the form-based rule editor. You can adapt the information in the Area and Property section to apply to the different blocks of code in the source editor.

Property	Description
Configuration Section	
Name (Wizard only)	The name to appear as the label for the resource. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275 .
Description (Editor and Wizard)	Short description of the resource.
Priority (Editor and Wizard)	<p>Specify a value between 1 and 10, where 1 is the highest priority and 10 is the lowest priority. Rules with a higher priority execute before rules with a lower priority.</p> <p>See also Rank. Only set priority or rank where there is a reason to control the order of rules. As a general practice, it is more efficient to let the engine determine the order of rule firing.</p>
Rank	<p>Specifies a rule function that controls the order of execution of rules with the same priority. Validity must include Condition (that is, do not select Action Only as the validity.)</p> <p>See Using Priority and Rank to Control Order of Rule Execution on page 237 for details.</p> <p>Default is 0.0.</p>
Forward Chain	<p>Determines if the rule is used in forward chaining. If the checkbox is unchecked, changes made by this rule won't trigger other rules.</p> <p>Default is checked.</p>

Property	Description
Declaration Section	
Term	<p>A concept or event type in the project that you will use in your rule. Types you add to the declaration define the scope of the rule.</p> <p>For example, a concept definition such as <code>/Concepts/Accounts/CheckAccount</code>.</p> <p>It is not necessary to add scorecards in the declaration in order to use them in the rule.</p>
Alias	<p>A name used to refer to the scope element in the body of the rule. You can change the alias. Like resource names, aliases must be valid identifiers See Identifiers (Names) on page 275.</p>
Conditions Section	
	<p>Each line in the Conditions area is a single expression that evaluates to <code>true</code> or <code>false</code>. Each line is joined to the others with an implicit <code>and</code> operator.</p> <p>For the OR operator, use a double pipe (<code> </code>) on the same line.</p> <p>TIBCO BusinessEvents evaluates single conditions from left to right. TIBCO BusinessEvents optimizes the evaluation of multiple conditions (see Order of Evaluation of Rule Conditions in <i>TIBCO BusinessEvents Architect's Guide</i>.)</p>
Actions Section	
	<p>List of statements that will be executed when the rule is fired.</p>

Adding a Rule Function

Regular rule functions have arguments and a body containing the code for the function. Virtual rule functions have arguments but no body. Virtual rule functions are used only with the TIBCO BusinessEvents Decision Manager add-on software. The implementation for virtual rule functions is provided by one or more decision tables. See TIBCO BusinessEvents Decision Manager product documentation for more details.

To Add a Rule Function

See [Rule Function Resource Reference on page 231](#) for details on completing values.

1. In Studio Explorer, right click the folder where you want to store the rule function and select **New > Rule Function**. You see the New Rule Function Wizard.
 - a. In the Rule Function Name field, type a name for the rule function.
 - b. In the Description field, type a description. (In the source editor the description appears in the * @description line of the comments at the top of the editor).
 - c. Set the return type for the rule function. Default is `void`. Browse to select a return type as needed. For return types that require additional configuration, such as `ContainedConcept`, complete the configuration in the Rule Function editor.
 - d. If you want this rule function to be a virtual rule function (to be implemented by a decision table), check the **Virtual** checkbox.



You can't change the name in the editor. To change the name of any project element, right-click the element in Studio Explorer and select **Refactor > Rename**. See [Chapter 4, Element Refactoring Operations, on page 45](#) for more details.

2. Click **Finish**. If you see the source editor, click the **Form** tab at the bottom of the editor to use the form editor as desired.

At any time you can click the Form and Source tabs at the bottom of the editor to switch between the form-based editor and the source editor as you work, depending on your preference.



Rule Function Editor Preference To set which mode the editor uses upon first opening, go to Window > Preferences > TIBCO BusinessEvents > Rules and check or uncheck the following checkbox as desired: **Initially show 'Form' tab in Rule Function Editor**.

3. In the Form editor Configuration panel, add or edit an alias and a description as desired. (In the source editor the description appears in the *`@description` line of the comments at the top of the editor and the Alias appears in the `attribute` list.).
4. If you did not do so in the Wizard, use the Return Type to select the return type of the rule function.
5. If you did not do so in the Wizard, set the Virtual check box according to your need. Check the check box if you are creating a virtual rule function (to be implemented by a TIBCO BusinessEvents Decision Manager decision table).



In the source editor, the signature of a virtual rule function is:

```
virtual void rulefunction folder.RFName
```

Do not add code to the Body block in the source editor of a virtual rule function. If you do, you see error messages if you try to save or to switch to the form-based editor.

6. From the Validity drop-down list, select the value that specifies where the rule function can be used (source editor `attribute` equivalents shown in parentheses):

- Action (`validity=ACTION`)
- Action and Condition (`validity=CONDITION`)
- Action, Condition and Query (`validity=QUERY`)

Virtual rule functions have a non-editable validity setting of Action.

7. If the rule function returns a value, specify the Return Type, otherwise leave this field set to `void`. (Appears in the signature of the rule function in the Source editor.)

Virtual rule functions have a non-editable return type of Void.

8. In the Scope section (`scope` statements in the source editor) you define the arguments of the rule function. Drag entities into the Scope area from Studio Explorer, OR do the following:
 - a. Click **Add** to add resources that you will be using in your rule function. You see the Select Rule Function Scope Arguments dialog.
 - b. In the upper half of the Select Rule Function Scope Arguments dialog, select the type you want to use.
 - c. If you want to specify an array, check the `isArray` checkbox. (You can specify a variable array in the source editor in the usual way, for example, `int[] myArr.`)
 - d. If the type you select is an ontology type, in the lower half of the dialog, select a resource from the filtered ontology tree.
 - e. Click **OK**.

Your selection appears in the list. TIBCO BusinessEvents assigns an alias to it. You can edit the alias.

Add more entities as needed.

9. Add more arguments as needed, and use the up and down arrows to order the arguments as needed.
10. In the Body panel (`body` statements in the source editor), use the TIBCO BusinessEvents rule language to implement the function. (Virtual rule functions have only a signature, and no implementation at design time.)

See [Using Variables and Functions in the Rule Editor on page 233](#) and [Tips for Working in the Rule Editor on page 241](#) for more information on working in the rule editor.

11. Save the resource.

Rule Function Resource Reference



Rule Function resources enable you to write rule functions that you can use in rules, as startup and shutdown actions, and as preprocessors.

Virtual rule functions are decorated with a V.

Property	Description
Configuration Section	
Name (Wizard only)	The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275 .
Description (Editor and Wizard)	Short description of the resource.
Return Type (Editor and Wizard)	If the rule function returns a value, specify the Return Type, otherwise leave set to void. See the list of valid types for the Scope Section area, next.
Virtual (Editor and Wizard)	If set to yes, the rule function is a virtual rule function. Virtual rule functions have arguments but no body. The Body panel is disabled and so is the Return Type field. The body is provided by a decision table. See <i>TIBCO BusinessEvents Decision Manager User's Guide</i> .
Alias	Optionally, enter an alias for the rule function. Used as a short way to refer to the rule function. You can use the alias, for example, to make query strings shorter (if you have the TIBCO BusinessEvents Event Stream Processing add-on).

Property	Description
Validity	<p>Specifies where the rule function can be used. Possible values are as follows:</p> <p>Action Indicates that this rule function can be used only in the Action block of a rule.</p> <p>Action and Condition Indicates that this rule function can be used in the Action and Condition blocks of a rule.</p> <p>Action, Condition and Query Indicates that this rule function can be used in the Action and Condition blocks of a rule, and can also be used in the text of a query (The query language features are available only in TIBCO BusinessEvents Enterprise Suite).</p> <p>Note: Only Action rule functions can be used as start up actions and shutdown actions.</p>
Scope Section	
Term	<p>The type of the argument. Arguments and return type can be any of the following, including arrays of these datatypes:</p> <ul style="list-style-type: none">• Primitive, that is any of: <code>String</code>, <code>int</code>, <code>long</code>, <code>double</code>, <code>boolean</code>, <code>DateTime</code>, <code>Object</code>• Concept• Event• Specific type of Concept• Specific type of Event <p>The Object data type is used to pass parameters between standard and user-defined functions and external Java sources.</p> <p>For more details, see Chapter 18, Rule Language Datatypes, on page 291.</p>
Alias	<p>Each argument requires a type and an alias. Names must be valid identifiers. See Identifiers (Names) on page 275.</p>
Body Section	
List of statements that will be executed when the rule function executes.	

Using Variables and Functions in the Rule Editor

This section provides some tips on working in the rule editor. The rule editor is used for TIBCO BusinessEvents rules, rule functions, and state machine rules.

Using Catalog Functions in the Rule Editor

To use catalog functions in an editor choose one of the following methods.

1. When adding code in the rule editor, do one of the following.
 - Begin to type the function name, beginning with the folder path to the function.

Do not enter the catalog name, which appears in the list of catalog functions in the Catalog functions View, and looks like a top-level folder provided catalog names are CEP Query, RDBMS, and Standard. Type a period at the end of the folder name. A popup window shows all folders and functions within the folder whose name you typed. For example type `Database.` to see a list of all functions in the Database folder.
 - Open the Catalog Functions view. To open the view click as follows:
Window > Show View > Other > TIBCO TIBCO BusinessEvents > Catalog Functions.

Drill down on categories within the catalog to expand to lists of functions and drag the desired function to the rule editor.
2. Provide parameter values as indicated by the tooltip. You can hover the cursor over a function to display a tooltip showing the function's arguments. You can also see the tooltip contents in the online reference, *TIBCO BusinessEvents Functions Reference*.

See [Overview of Catalog Functions on page 248](#) for descriptions of the various function catalogs, and an explanation of the decorations that appear on many function names.

For information about configuring mapper functions see [Using the Function Argument Mapper on page 234](#).

Using Global Variables in the Rule Editor

To use a global variable in the rule editor, use one of the `System.getGlobalVariableAs*` functions. For example:

```
System.getGlobalVariableAsString("Hostname", "Localhost")
```

Where `Hostname` is the name of the variable and `Localhost` is an optional literal value to use if the variable is not found.

Do not use this syntax: `%% Global.Variable.Name%%`.

See [Working with Global Variables on page 17](#) for more details about global variables.

Using the Function Argument Mapper

For functions known as mapper functions you can use the Function Argument Mapper to map inputs from a source to the function arguments.

See [Chapter 19, Mapping and Transforming Data, on page 297](#) for a reference to using the Function Argument Mapper.

To Open the Function Argument Mapping Wizard

You can open the Function Argument Mapping Wizard in various ways. For example here is one way:

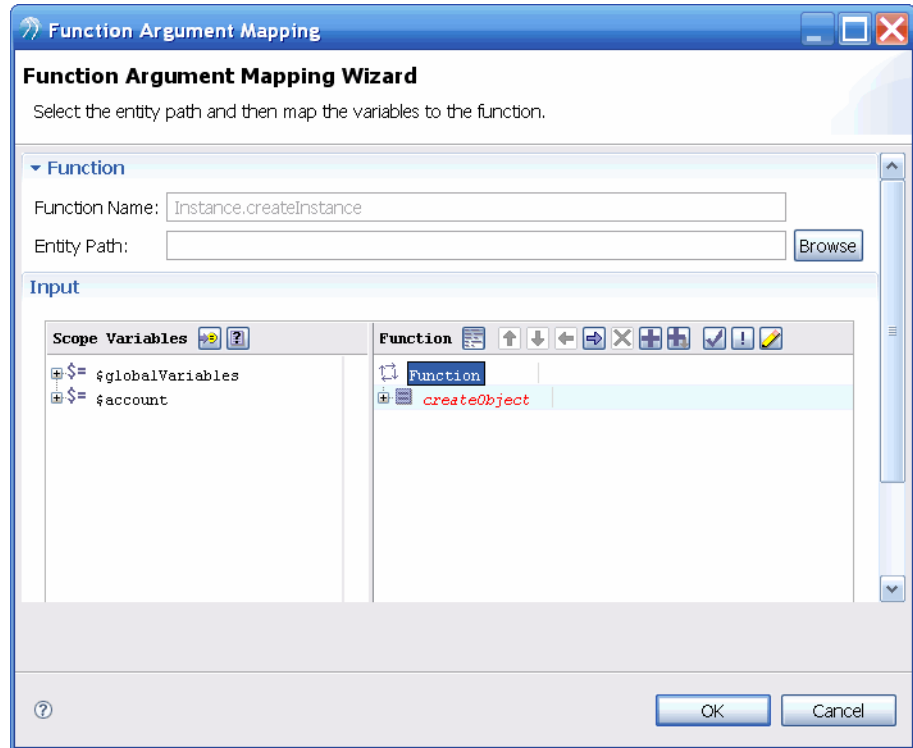
1. In the rule editor, type the category of function you want to use. A list of functions in that category appears.
2. Select a mapper function from the list of functions in that category. The function argument area contains the text `"xslt://"`.
3. Control-click the text `"xslt://"` to display a hypertext link. Click the link to open the Function Argument Mapping Wizard.


Other ways to perform the same action are as follows:

- Type the name of a mapper function category into the **Actions** or **Conditions** areas, then type an open parenthesis ("(") TIBCO BusinessEvents displays `"xslt://"`. Control-click to open the Function Argument Mapping Wizard.
- Type the entire string to specify the function category path and name, followed by `("xslt://")`. Then control-click the text `"xslt://"` to open the Function Argument Mapping Wizard.

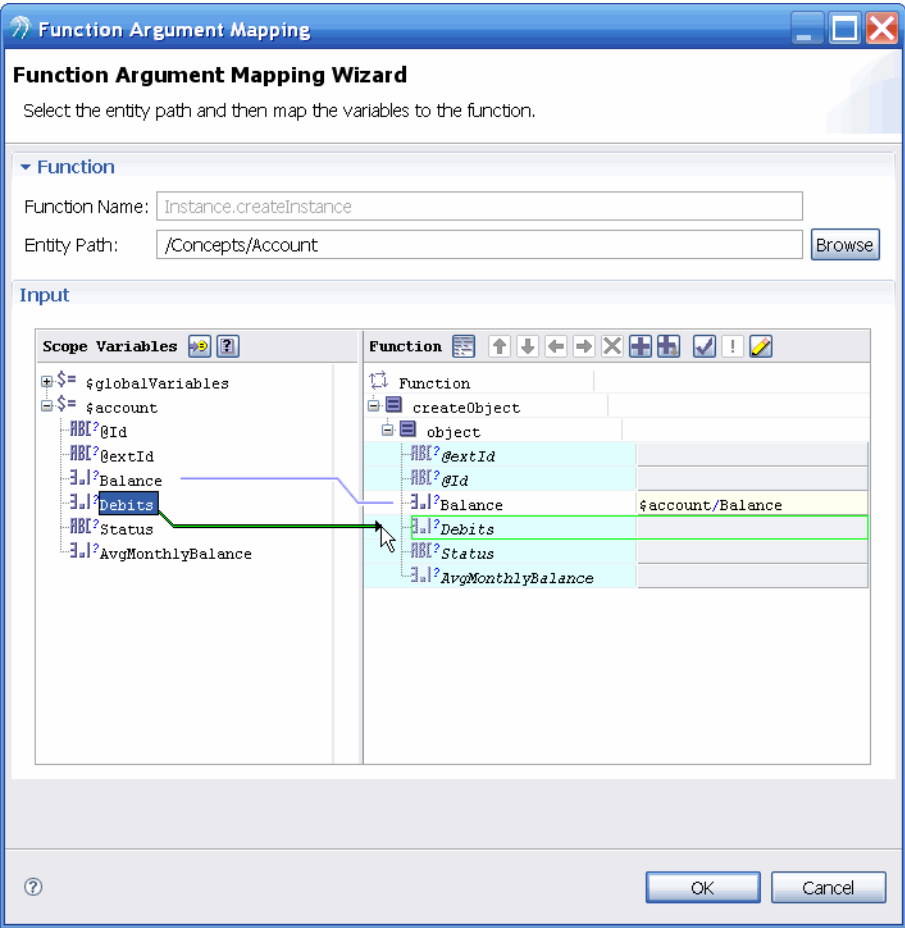
To Work with the Function Argument Mapper Wizard

See [Chapter 19, Mapping and Transforming Data, on page 297](#) for full details on using the wizard. This section provides summary information only.



1. In the Function section, take appropriate action. For example, if you are using `Instance.createInstance()`, you would click the browse icon () to the right of the **Entity Path** field and select a concept type so you can create an instance of it.
2. Select a resource and click **Apply**.
3. Select the **Input** tab. TIBCO BusinessEvents displays a list of variables associated with the project on the left and a list of properties for the selected resource type on the right.
4. Drill down and expand the lists on both sides to expose the variables and properties.

5. Drag variables from the left to the appropriate property on the right, as shown:



6. If you want to define the arguments using more complex logic, type the code or click the XPath Formula Builder (🔗) button to use the XPath Formula Builder. See [Chapter 20, XPath Formula Builder](#), on page 335.
7. Click OK.



Array index difference In the TIBCO BusinessEvents language, array indexes start from zero (0). However, in XSLT and XPath languages, they start from one (1). It's important to remember this difference when using the rule language in the rule editor, and when working in the XSLT mapper and the XPath builder.

Using Priority and Rank to Control Order of Rule Execution

For each RTC, the rule agenda is sorted by priority and then within priority by rank, for those rules that use the same ranking. Use of priority and rank is optional. You can also use priority without using rank.

In the rule's Rank field (or rule attribute, in the source view), you specify a rule function that returns a double. The larger the return value, the higher the ranking. You can specify the same rule function in different rules to perform ranking across tuples of those rules. Here are the requirements:

- The rule function must have a Validity setting that includes Condition (that is, do not set it to Action Only).
- You can assign the same rule function to different rules as long as the following is true:
 - The scope of the rule function includes only parameters found in all the rules that use the same function. This can be a subset of what's in the rule declaration. (As with rules, primitives are not allowed.)
 - The parameters must be used in the same order as they appear in the rule declaration.
- The rule function must return a double value. (The default value for the Rank field is 0.0.)

Examples

For example, suppose two rules declare CustName and SupportLevel (plus other things that can be different). You assign the same rule function to both rules. The function returns 3.0 for Gold support level, 2.0 for Silver, and 1.0 for Bronze. As a result, among rule tuples with priority 1, those for customers with Gold support execute before those for customers with Silver support, which execute before those for customers with Bronze support.

Below is an example showing how rule priority and ranking determine the sort order. Suppose you assign the same rule function for ranking rules 1, 2, and 3. At run-time these are some rule tuples to be sorted in the agenda for an RTC:

Rule 1 (Tuple X)	Priority: 5	Rank: 10.0
Rule 2 (Tuple A, B)	Priority: 1	Rank: 1.0
Rule 2 (Tuple A, C)	Priority: 1	Rank: -1.0
Rule 2 (Tuple A, D)	Priority: 1	Rank: -2.0
Rule 3 (Tuple A)	Priority: 4	Rank: 0.0

They are sorted and executed as follows. (This could change during a conflict resolution cycle, of course, depending on the effect of rule actions.)

Rule 2 (Tuple A, B)	Priority: 1	Rank: 1.0
Rule 2 (Tuple A, C)	Priority: 1	Rank: -1.0
Rule 2 (Tuple A, D)	Priority: 1	Rank: -2.0
Rule 3 (Tuple A)	Priority: 4	Rank: 0.0
Rule 1 (Tuple X)	Priority: 5	Rank: 10.0

Reverse Order Example

If you want the rule with the smallest return value to be ranked highest, multiply by negative one (-1.0) to reverse the size of the values. For example suppose you want rules that were asserted earlier to execute before those that were asserted later. The time values returned by three rules are 100, 150, and 300. Using $-1.0 * time$, they will be fired in the desired order: -100.0 (the largest value), then -150.0, then -300.0.

Using the Quick Fix Feature in the Rule Editor

The Quick Fix feature enables you to create concepts, events, and rule functions without leaving the rule editor. It also enables you to add properties to existing concepts and events. The feature is available when an unknown reference appears in the rule or rule function code.



This feature is also used in the state modeler feature, available in TIBCO BusinessEvents Data Modeling, wherever the rule editor is used.

If a qualified name has multiple unresolved references, for example, `Concepts.Something.Somethingelse`, the Quick Fix feature only applies to the first unresolved reference in the name.

To Use the Quick Fix Feature

1. Type the code in the rule editor as if the entity, property, or rule function you want to use already exists. See [Table 25, Quick Fix Feature Options, on page 239](#) for some examples.

You see a lightbulb icon (💡) in the left margin on any line where there are unknown references that you can configure using Quick Fix.

2. To use the available Quick Fix options, click the lightbulb icon or press Ctrl+1. The options are explained in [Table 25](#):

Table 25 Quick Fix Feature Options

Unknown Reference	Quick Fix Editor
<p>Unqualified name. Example:</p> <pre>SomeName s ;</pre>	<p>A list of appropriate wizards appears so you can create a new concept, simple event, or time event. The wizard you select opens so you can define the entity.</p>
<p>Unqualified name followed by parentheses. Example:</p> <pre>SomeName ()</pre>	<p>A link to the rule function creation wizard appears. Click it to open the wizard and define the rule function.</p>
<p>Qualified name. Examples:</p> <pre>Rules.SomeName () ;</pre> <pre>Concepts.SomeName s ;</pre>	<p>A Quick Fix option appears so you can automatically create the entity or rule function in the given folder. The folder must already exist in the project.</p>

Table 25 Quick Fix Feature Options

Unknown Reference	Quick Fix Editor
Qualifier is a concept or event and reference is unknown. Example: <code>Concepts.Person person; person.someProperty = "123"</code>	A Quick Fix option appears with the name: Create a property definition in the entity' <i>entityName</i> '. Click the option to open the New Property Definition dialog, and configure the details of the property. In the example, the <code>someProperty</code> property does not already exist as a property of <code>Concepts.Person</code> .

Tips for Working in the Rule Editor

This section has some tips to help you work in the rule editor.

Switching between Form and Source Editors

You can freely switch between form and source editors for rules and rule functions. In each case the editors stay synchronized with the latest changes.

You can't switch from the source editor to the form editor if there are any syntax or resolution errors in your code.

The Priority setting is used by the runtime engine when determining the order in which rules are fired. Rules with a number closer to one fire first. When there is no reason to force rules to execute in a particular order, leave the Priority set to the default and let the runtime engine determine rule order.

Declaring multiple terms of the same type Allows the rule to consider multiple instances of the corresponding entity. Specify different aliases to keep the terms distinct

Scorecards Scorecards are like concepts except that there is only one instance of a scorecard (or more accurately, one instance per agent when multi-engine features are used). It is therefore not necessary to put scorecards in the declaration of a rule because a scorecard never requires an alias. You can use scorecard properties in conditions (just as you would concept properties). However, because a scorecard doesn't have an alias, refer to it like a function, for example,

```
Folder.Folder.Scorecard.prop1
```

Standard Eclipse Features

In addition to some TIBCO BusinessEvents-specific features, the source and form rule editors support standard Eclipse functionality such as the following: Undo and redo; copy and paste; breakpoint features; standard text annotations (which can be changed using Preferences); text folding (source editor only); Java outline view (source editor only).



When you're working in the source editor, press **Ctrl+Shift+L** to see a list of keyboard shortcuts available in that context. (This is a general Eclipse feature.)

Information Highlighted

Keywords, variables, and functions are highlighted in the text. Also, when you hover the mouse over a resource such as a concept, event or function, information about it displays in a tooltip.

Syntax and resolution errors are automatically flagged by visual cues. They are underlined and also display in the vertical and overview rulers.

Miscellaneous Tips

Some other tips are highlighted below.

Table 26 Tips for Working in the Rule Editor

To do this...	Do this...
Switch Between Source and Form Editors	Click the bottom tabs to switch between the source and form editors. The code remains synchronized. However, you can't switch if there are errors in the code. First resolve the errors, then switch.
To Use Content Assist to Complete Values	<p>The content assist feature helps you complete values using information that is available in resources. For example, if you type the name of a concept type (or its alias) and then a period, a list of the concept type properties appears for you to select from.</p> <p>The selection list also appears when the cursor is in an appropriate location and you press Ctrl+Space, or if you right click and select Edit > Content Assist from the context (right-click) menu.</p>
To Comment (and Uncomment) a Line	To comment out a line, or uncomment a line, press Ctrl+/** or select Edit > Toggle Comment from the context menu
To Search for References	When the cursor is placed in an appropriate item in the code such as an entity or function name, you can find all references to that item references in the rule code. Press Ctrl+Shift+G , or select Search > Search for References from the context (right-click) menu. The item references are highlighted in the text, and an arrow appears in the vertical ruler.
To Jump to the Definition of an Item	<p>To jump to the location where an item is defined, you can use two methods.</p> <ul style="list-style-type: none">Click in the item name and press F3, or right-click and select Open Declaration.Press and hold the Control key while you move (hover) the mouse pointer the text. When you hover over an item that displays an underline, Ctrl+click the item to jump to the place where it is defined. <p>For example, you would jump from an alias to the declaration, and from an entity or entity property to the entity's editor.</p>

Event Preprocessors

Event preprocessors are rule functions with one argument of type simple event. They perform tasks after an incoming message is transformed into a simple event but before it is asserted into the Rete network.

An event preprocessor is assigned to a destination and acts on all events arriving at that destination.



See Event Preprocessors in *TIBCO BusinessEvents Architect's Guide* important information on the use of preprocessors.

If an event preprocessor fails due to an exception, the Retry on Exception field setting on the event type determines subsequent behavior (see [Simple Event Reference on page 137](#)).

Loading Cache Only Entities If you are using the Cache Only cache mode for any entities, it is recommended that you load the relevant entities in the event preprocessor. See [Guidelines for Use of Cache Related Functions on page 260](#).

Configuring an Event Preprocessor

To configure a preprocessor you associate the desired rule function with a destination. This configuration is done using the Cluster Definition Descriptor editor. See Chapter 3, CDD Configuration Procedures, Collections Tab — Destinations Settings and Properties, in *TIBCO BusinessEvents Administration*.

Transaction Error Handler Rule Function

You can create a transaction error handler callback rule function that enables you to identify which post RTC transactions failed or which events were not sent out during the post RTC phase. The transaction error handler rule function is invoked each time a database transaction exception occurs, and each time a send event exception occurs.

This feature requires cache-aside database write strategy. It is not supported when the following property value is set:

```
Agent.agentclassname.enableParallelOps=false
```

The error handler rule function provides full details of the failed transactions. The transaction details provided in this callback function can be used for audit trail purposes. The engine continues retrying the transaction or action as usual.

To Register the Rule Function

To register the rule function, add the following property to the be-engine.tra / project CDD file at an appropriate level (for example, at the cluster level to affect all deployed engines). Set the value to the project path of the error handler rule function:

```
be.engine.txn.error.function=projectPathToErrorHandlerRuleFunction
```

Rule Function Signature

The required signature for the error handler rule function is:

```
int Name (Object txns, int errorType, int errCode, String errMsg, long retryCount)
```

The return value must be 0 (zero).

The following table explains the arguments.

Argument	Notes
Object <i>txns</i>	<p>You can turn this argument into an <code>Object []</code> within the rule function, for example:</p> <pre>Object [] array = txns;</pre> <p>The resulting array can be used to obtain useful data:</p> <ul style="list-style-type: none"> • <code>array[0]</code> is a <code>Concept []</code> containing all the Concept objects that: <ul style="list-style-type: none"> — have been created in the current RTC, — and have not been deleted. • <code>array[1]</code> is a <code>Concept []</code> containing all the Concept objects that: <ul style="list-style-type: none"> — existed before the current RTC, — have been modified in the current RTC, — and have not been deleted. • <code>array[2]</code> is a <code>long []</code> containing the id of all the Concept objects that: <ul style="list-style-type: none"> — existed before the current RTC, — and have been deleted in the current RTC. • <code>array[3]</code> is a <code>SimpleEvent []</code> containing all the SimpleEvent objects that: <ul style="list-style-type: none"> — have been created in the current RTC, — and have not been deleted. • <code>array[4]</code> is a <code>long []</code> containing the id of all the Event objects that: <ul style="list-style-type: none"> — existed before the current RTC, or arrived and started the RTC, — and have been deleted in the current RTC.
int <i>errorType</i>	<p>Value can be one of:</p> <ul style="list-style-type: none"> -1 for errors happening during database operations. -2 for errors happening when sending an event.

Argument	Notes
<code>int <i>errCode</i></code>	<p>The error code, which is dependent on <code>errorType</code>:</p> <ul style="list-style-type: none"> • For database related errors: <ul style="list-style-type: none"> — The error code as returned by <code>SQLException</code>, if available — Else <code>-100</code> if TIBCO BusinessEvents determines that the database is not available — Else <code>-200</code>, which means that this is an unknown error • For sent event errors, the value is <code>-1</code>, meaning an internal error condition.
<code>String <i>errMsg</i></code>	The associated exception message.
<code>long <i>retryCount</i></code>	The number of times this transaction has been retried before the present call to the callback.

Chapter 16 **Functions**

This chapter explains how to work with functions in TIBCO BusinessEvents.

Topics

- [Overview of Catalog Functions, page 248](#)
- [Function Tooltips and Decorations, page 252](#)
- [Temporal Functions and Their Parameters, page 254](#)
- [VRF Functions, page 256](#)
- [Guidelines for Use of Cache Related Functions, page 260](#)
- [Guidelines for Use of Oracle Coherence Functions, page 262](#)
- [Adding Custom Functions, page 265](#)
- [Restrictions on Use of Custom Functions, page 267](#)
- [Structure of a Function Catalog, page 268](#)

Overview of Catalog Functions

The functions registry includes various catalogs of functions provided with the product, and each catalog organizes functions into various related categories. You can use functions in rule conditions and actions and in rule function bodies.

To View the Catalog functions

All catalogs appear in the Catalog Functions view. To open the view navigate to Window > Show View > Other > TIBCO BusinessEvents and select Catalog Functions. The catalog view appears on the right, by default.

Built-in Functions

For all the built-in functions, this section lists the main categories in each function catalog (but not sub-categories). See [TIBCO BusinessEvents Functions Reference](#) for full details.

Standard Functions

The most generally useful catalog, the standard function catalog include the following categories:

- **BusinessWorks** functions are used in ActiveMatrix BusinessWorks integration projects. See [Chapter 21, ActiveMatrix BusinessWorks Integration, on page 347](#).
- **Channel** functions return information about destinations, and can resume and suspend a destination.
- **Cluster functions** help with multi-engine functionality
 - **DataGrid** functions are for use with Cache object management. See [Guidelines for Use of Cache Related Functions on page 260](#).
- **Date** functions allow you to compare two DateTime values using only the date portion of the value.
- **DateTime** functions allow you to perform these date/time related tasks and more: add units of time to a DateTime, compare, retrieve, and format dates and times.
- **Engine** functions allow you to retrieve information about the engine, for example, available memory or the number of rules fired.

- **Event** functions allow you to assert, create, and send simple events and perform other event-related tasks, for example, return the default destination URI of a simple event.
- The **Exception** function enables you to create an exception.
- **File** functions provide various useful functions used when working with files.
- **HTTP** functions are used with the HTTP channel
- **Instance** functions allow you to create and delete concept instances and perform other instance-related tasks, for example, return an instance given an internal ID.
- **Math** functions allow you to perform advanced mathematical operations.
- **Number** functions allow you to perform type conversions from and to numbers and return the maximum and minimum values for a numeric type.
- **SOAP** functions enable you to work with SOAP messages sent through an HTTP channel.
- **String** functions allow you to perform comparisons, searches, conversions, and other operations with strings.
- **System** functions allow you to send messages to a debug log, retrieve global variables, retrieve system properties, and write data to a file.
 - **IO** functions allow the writing and closing of specific files.
- **Temporal** functions allow you to examine and perform calculations on values stored in a property's history. For information about using temporal functions, see [Temporal Functions and Their Parameters on page 254](#).
- **Util** functions category has one sub-category for working with HashMaps.
- **VRF** functions (that is, Virtual Rule Function functions) allow you to work with decision tables. See [VRF Functions on page 256](#) and see *TIBCO BusinessEvents Decision Manager User's Guide* for details.
- **XPath** functions allow you to evaluate XPath expressions.

CEP Pattern Functions

Pattern functions are used with the pattern matcher language for identifying patterns in events. See TIBCO BusinessEvents Event Stream Processing add-on documentation for details.

CEP Query Functions

Query functions are used with the query language for querying data in the cache. See TIBCO BusinessEvents Event Stream Processing add-on documentation for details.

Communication Functions

Communication functions provide a set of catalog functions that enables TCP communication. You can create a local TCP server and a TCP client so that TIBCO BusinessEvents can communicate with data sources not otherwise available through channels, using TCP. See [Communicating with Other Sources using TCP on page 70](#) for details.

RDBMS Functions

Database functions are provided for working with database concepts. See TIBCO BusinessEvents Data Modeling for more on database concepts.

Security Functions

These functions are used internally by the TIBCO BusinessEvents Decision Manager add on, for authentication.

Ontology functions are generated by TIBCO BusinessEvents based on the concepts, events, and rules in your project. There are three types of ontology functions:

- Constructors — Allow you to create a simple event or concept instance.
- Time events — Allow you to create and schedule a time event. See [Chapter 9, Time Events and Scheduler Functions, on page 145](#).
- Rule functions — Allow you to invoke a rule function. See [Rules and Rule Functions on page 219](#).

The Ontology Functions area uses the same folder structure as the project (or rather, a subset of that structure).

Custom Functions

You can also create custom functions. Custom functions appear in the Custom Function catalog. For information about custom functions, see [Adding Custom Functions on page 265](#).

Enabling Extended Functions

Extended functions (sometimes called hidden functions) may be made available by TIBCO Support to address customer-specific use cases. They are also sometimes used to make legacy features available to customer who wish to continue using them. To make them visible in the Catalog Functions view, do the following:

1. Open the following file for editing:

```
BE_HOME/studio/eclipse/configuration/config.ini
```

2. Add a property of the following format:

```
TIBCO.BE.function.catalog.function-catalog-name=true
```

For example to enable the Oracle Coherence function category you would add

```
TIBCO.BE.function.catalog.coherence=true
```

Save the file and restart TIBCO BusinessEvents Studio.

Function Tooltips and Decorations

Tool Tips

When you float the cursor over a function in the registry, TIBCO BusinessEvents displays the description and syntax in a tool tip next to the cursor. Tooltips also form the online reference to the function catalogs. See *TIBCO BusinessEvents Functions Reference* in the HTML documentation.

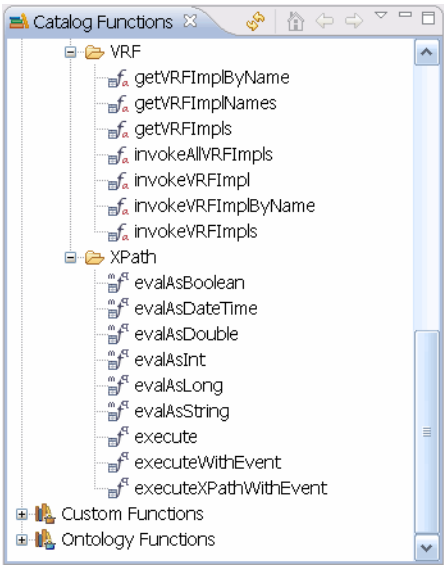


You can create your own tool tips for custom functions. See [Adding Custom Functions on page 265](#).


You can turn off the tool tip display too using Window >Preferences >TIBCO BusinessEvents preferences

Decorations Indicating Where Functions can be Called

Some functions are decorated with small letters that indicate useful information about use of the function. A function can have zero, one, or more decorations. the following figure shows all the available decorations. They are described in the sections after the figure:



Mapper Functions

Functions that bring up the XSLT mapper and XPath builder are identified by a small *m* at the upper left of the *f*, for example:  `evalAsString` . For more on using mapper functions, see [Using the Function Argument Mapper on page 234](#).

Functions That Can Be Used in TIBCO BusinessEvents Decision Manager


Functions that can be used in TIBCO BusinessEvents Decision Manager are marked with a small table icon,  `getVRFImpls` for example (which is also an action-only function).

Validity Decorations


Some functions can be used only in rule actions, some in rule actions and rule conditions, and some in actions, conditions, and in queries.

Functions that can be used in actions and conditions have no decoration. They are considered to have the default validity.

Action-Only Functions

These functions are for use in rule actions only. Some of these functions have side effects, for example they can change values. Other functions are limited to actions for other reasons. These action-only functions are identified by a small *a* at the bottom right of the *f*. For example:  `setDateTime` .

Functions That Can Be Used in Queries or with Pattern Matcher

Functions that can be used in queries or with Pattern Matcher are marked with a blue *q* for example,  `getMonth` , for example. You can call such functions in a query string.

They are also valid in rule actions and conditions.

See documentation for TIBCO BusinessEvents Event Stream Processing add-on software for details on pattern matcher and query features.

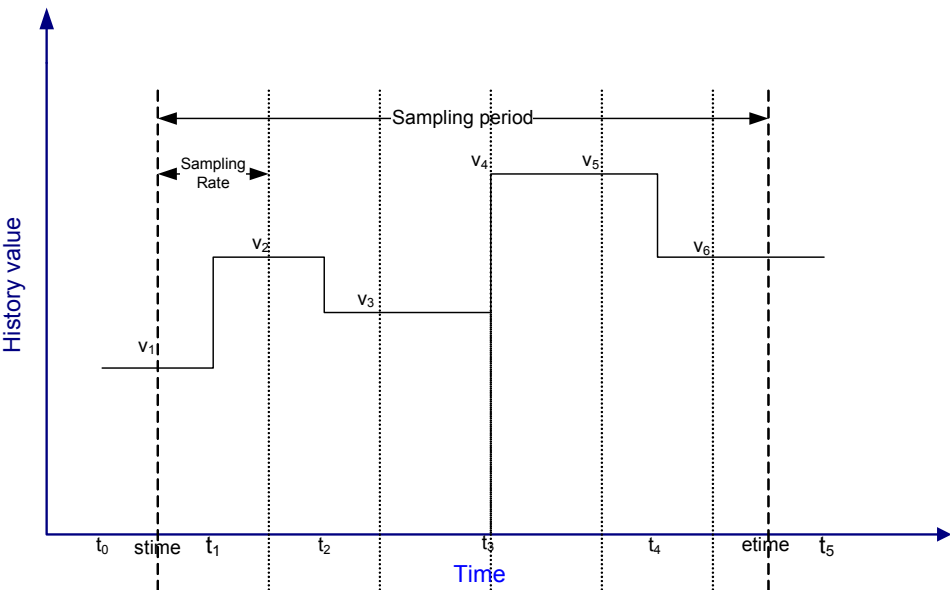
Temporal Functions and Their Parameters

The set of Built-In > Standard functions that come with TIBCO BusinessEvents includes functions that allow you to perform calculations on numeric values sampled over time. These functions are called temporal functions and they work exclusively with concept properties that store numeric values. Temporal functions make use of the history ring buffer to sample a property’s values over time.



Use of a temporal function with a concept that has a history size of 0 may cause a runtime exception.

Figure 4 Temporal Functions Parameters



All temporal functions include these parameters, illustrated in [Figure 4](#):

- **property** — The property for which you want to sample values.
- **stime** — The time from which you want to begin sampling values (the start time) measured in milliseconds since 00:00:00 UTC on January 1, 1970.
- **etime** — The time at which you want to stop sampling values (the end time) measured in milliseconds since 00:00:00 UTC on January 1, 1970.
- **sample_rate** — The number of milliseconds between samples.

- **bound_by_stime** — A flag indicating whether the start-time is flexible:
 - **True** indicates that if the start time you provide is earlier than the timestamp for the oldest available value, you want to perform the calculation starting with the oldest available value.
 - **False** indicates that if the start time you provide is earlier than the timestamp for the oldest available value, you want to abort the calculation.

VRF Functions

These functions are used only with TIBCO BusinessEvents Decision Manager.

The VRF category of functions (within the Built-in > Standard Functions) provide flexibility when you are working with virtual rule function implementations.

Virtual rule functions are implemented by decision tables.



Default Implementation When you deploy multiple implementations (tables) for one virtual rule function, but use a function that doesn't specify the implementation by name, for example if you use `Functions.MyVirtualRuleFunction()`, the default implementation is used. The default implementation is whichever was the last implementation to be deployed. However, if you use hot deployment, it may not be possible to determine which implementation was deployed last.

The VRF category of functions contains the following:

```
getVRFImpByNames()
getVRFImpNames()
getVRFImps()
invokeAllVRFImps()
invokeVRFImp()
invokeVRFImpByName()
invokeVRFImps()
```

Common arguments used in the above functions are described in [Table 27, Common Arguments for VRF Functions, on page 258](#).

To Add a Virtual Rule Function

1. Right-click the folder where you want to store the virtual rule function and select **New > Rule Function**. You see the New Rule Function Wizard.
 - a. In the Rule Function Name field, type a name for the rule function.
 - b. In the Description field, type a description. (In the source editor the description appears in the `* @description` line of the comments at the top of the editor).
 - c. Check the **Virtual** checkbox.



You can't change a new resource name after you click Finish. (You can change the description, however.)

2. Click **Finish**.

Click the tabs at the bottom of the editor to switch between the form-based editor and the source editor as you work, depending on your preference. These instructions use the form editor and mention the equivalent settings in the source editor.

3. In the Form editor Configuration panel, add or edit a description as desired. (In the source editor the description appears in the `* @description` line of the comments at the top of the editor).
4. If you did not do so in the Wizard, check the Virtual check box.



In the source editor, the signature of a virtual rule function is:

```
virtual void rulefunction folder.RFName
```

Do not add code to the Body block in the source editor of a virtual rule function. If you do, you see error messages if you try to save or to switch to the form-based editor.

In virtual rule functions, the Validity field is set to Action and the Return Type is set to Void. column, select where the rule function can be used (source editor equivalents shown in parentheses):

5. In the Scope section (`scope` statements in the source editor), drag an ontology entity into the Scope area, OR do the following:
 - a. Click **Add** to add resources that you will be using in your rule function. You see the Select Rule Function Scope Arguments dialog.
 - b. In the upper half of the Select Rule Function Scope Arguments dialog, select the type you want to use.
 - c. If the type you select is an ontology type, in the lower half of the dialog, select a resource from the filtered ontology tree.
 - d. Click **OK**.

Your selection appears in the Declarations list. TIBCO BusinessEvents assigns an alias to it. You can edit the alias.

Add more entities as needed.

6. Save the project.

For example, here is the source view for a simple virtual rule function:

```
/**
 * @description Action to take when account is suspended
 */
virtual void rulefunction Rules.FollowUp {
    attribute {
        validity = ACTION;
```

```
    }  
    scope {  
        Concepts.Account account;  
    }  
    body {  
    }  
}
```

VRF Function Arguments

The VRF functions use various subsets of the following common arguments:

Table 27 Common Arguments for VRF Functions

Name	Type	Notes
vrfURI	String	The universal resource identifier (URI) for the virtual rule function. This is typically the full path to the virtual rule function within the project directory. For example, in the <code>CreditCardApplication</code> example, the virtual rule function <code>Person_VirtualRuleFunction()</code> has the following URI: <code>/Virtual_RF/Person_VirtualRuleFunction</code>
vrfImpl	Object	An object representing a virtual rule function implementation. This argument is required when invoking specific virtual rule function implementations.
implName	String	The name of a decision table (also known as a virtual rule function implementation). For example, in the <code>CreditCardApplication</code> example, the virtual rule function <code>BankUser_VirtualRuleFunction</code> has an implementation (decision table) called <code>bankUser</code> . The <code>implName</code> argument is used to retrieve a corresponding implementation object, or to execute an implementation.

Table 27 Common Arguments for VRF Functions (Cont'd)

Name	Type	Notes
<code>args</code>	Object array	The arguments to be passed to one or more virtual rule function implementations on invocation. These objects consist of the concepts, events, scorecards, and so on. that are needed by the implementation or implementations. For example, the <code>processApplication</code> implementation in the <code>CreditCardApplication</code> example project requires concepts of type <code>Application</code> , <code>BankUser</code> , and <code>CreditCardApplication</code> to be passed as arguments. In order to invoke the <code>processApplication</code> implementation, an instance of each concept type must be passed in the <code>args</code> array.
<code>returnValues</code>	Object array	<i>Not used in this release.</i> This argument is used only for the <code>invokeVRFImpls</code> function. When invoking multiple implementations, the return value of each implementation is stored in this array. The array will contain a null entry for each implementation that does not return a value.

Guidelines for Use of Cache Related Functions



TIBCO BusinessEvents Express This section relates to Cache OM functionality and is not relevant if you are using TIBCO BusinessEvents Express edition.

Various standard functions in the Standard catalog `cluster.DataGrid` category enable you to work with objects in the cache. Cache load functions load items into the Rete network so they are available.

Tool tips in the user interface (and reproduced in the *TIBCO BusinessEvents Functions Reference*) explain the details of how to use the functions. This section explains certain patterns of use.

These core functions are used with both the Oracle and TIBCO cache providers.



The previous versions of these core functions (which began with `c_`) are deprecated. Migrate projects created in earlier releases to use the current names (see [Migrating Core Coherence Functions on page 52](#)).

Query functions are not available for the TIBCO BusinessEvents DataGrid provider. Similar functionality can be achieved using the TIBCO BusinessEvents Event Stream Processing add-on.

Functions for Loading Entities to Rete from Cache and Backing Store

```
CacheLoadConceptByExtId
CacheLoadConceptsByExtId
CacheLoadConceptById
CacheLoadConceptsById
CacheLoadEventByExtId
CacheLoadEventById()
CacheLoadParent()
```

`CacheLoad*()` functions are required for working with cache only cache mode. They load entities from the cache (or backing store if not found in the cache) into the Rete network. See Loading Cache Only Objects into the Rete Network in *TIBCO BusinessEvents Architect's Guide* for details.

Use the `CacheLoad*()` functions in an event preprocessor. Only use them in rules for cases where the `id` or `ExtId` is not known in advance (in the preprocessor).

After cache-only concepts are loaded in this way, you can then use `Instance.getByExtId()` in rules.

Never use `Instance.getByExtId()` unless you have first loaded the concept. `Instance.getByExtId()` does not assert the concept but just returns it for use in rules, for example, as read-only reference data.

Locking Functions

```
Lock()
Unlock()
```

In the event preprocessor, use the `Lock()` function to prevent other threads or engines from operating on the same entity. The lock is automatically released at the end of the RTC.

Use `Unlock()` only in a preprocessor and only to handle cases where you need to release the lock, for example because some information is missing that would be required to go forward.

See *Using Locks to Ensure Data Integrity Within and Across Agents* in *TIBCO BusinessEvents Architect's Guide* for details on use of locks.

Indexing Function

`Index()` creates an index on the specified property, which is useful when you run queries. (See [Guidelines for Use of Oracle Coherence Functions on page 262](#).)

Specialty Functions Not for General Use

The following functions are used only in certain applications:

```
CacheLoadConceptIndexedByExtId()
CacheLoadEntity()
CacheReevaluate()
EnableCacheUpdate()
```

Guidelines for Use of Oracle Coherence Functions



TIBCO BusinessEvents Express This section relates to Cache OM functionality and is not relevant if you are using TIBCO BusinessEvents Express edition.

This section is provided for those using Oracle Coherence as the cache provider. For information on using Oracle Coherence see [Enabling Use of Oracle Coherence as the Cache Provider on page 22](#).

Constants and Extractors Categories

The Coherence category has Constants and Extractors functions which are used in conjunction with functions in the Filters category.

Extractor functions return values for properties of different types.

Constants functions wrap constants so they can be used in filter functions. For example if a filter checks for `x = 10`, you would first wrap `10` using `C_IntConstant()`.

Filters Category

Functions in this category enable you to use various criteria to identify a set of objects in the cache for a query. You can pass the filter to a query function.

The `C_RuleFunction()` function allows you to specify a rule function containing a custom filter condition.

Query Category



Query category functions operate only on the cache Unlike the `C_CacheLoad*()` functions, the query functions do not look in the backing store if objects are not found in the cache.

If a backing store is used Do not use query functions that delete or modify values if a backing store is used. Instead use a query to return the IDs of the entities you want to delete and use `Instance.deleteInstance()` or `Event.consumeEvent()` as needed.

```
C_CacheInvoke()
C_CacheOnlyMode_DeleteConcepts
C_CacheOnlyMode_DeleteEntities
C_CacheOnlyMode_QueryConcepts
C_CurrentContext
```

```

C_EntryHasNext
C_EntryIterator
C_EntryNextValue
C_KeyHasNext
C_KeyIterator
C_KeyNextValue
C_QueryAction
C_QueryAndLoadConcepts
C_QueryConcepts
C_QueryEvents
C_QueryEvents_Order
C_QueryIDs

```

Query functions take various actions for a specified entity or set of entities. For example, `C_CacheInvoke()` allows you to invoke a rule function for all matching entities in the cache.

For some functions you can specify the entities by passing a filter (from the Filters category).

Tool tips in the user interface (and reproduced in the *TIBCO BusinessEvents Functions Reference*) explain how to use the functions singly or in combination to achieve the desired results.

The `C_CacheOnlyMode*` functions are for use with entities that use cache-only cache mode.

Query Optimization

When you use Oracle Coherence as the cache provider, you can index concept and event properties to make searches faster.

The query optimization you set up is used by `C_Query*` functions and by snapshot queries (available in TIBCO BusinessEvents Event Stream Processing add-on).

The snapshot query manager rewrites parts of the query's `where` clause to use Coherence `IndexAwareFilters`. Coherence, however, may or may not use the index, depending on how complex the filter is. Complex `where` clauses containing function calls and joins will not be optimized. Only simple filters, such as `age > 60`, or `name in ("a", "b", "c")`, are re-written to use indexes.

Only query agents enabled for query optimization use this feature. You can define the indexes in two ways. Setup is explained next.

To Enable Query Optimization

In the project CDD file, add the following property to the query agent properties:

```
be.agent.query.enable.filter.optimizer=true
```

Only agents with this property set to true will attempt to use indexing that you define.

To Use Query Optimization

You can set an index on an entity property in either of two ways: using a Coherence catalog function in your code, or setting a metadata property in the project's Cluster Deployment Descriptor (CDD) file.

Using a Coherence Catalog Function You can define an index using the following function in your code:

```
void C_Index(String cacheName, Object property, boolean isOrdered)
```

where:

cacheName is a String returned by `C_CacheName()`.

property is the object Returned by the appropriate `C_DatatypeAtomGetter` functions, for example, `C_StringAtomGetter()`.

isOrdered is a Boolean: set to true to order the contents of the indexed information, and set to false if you want to use an unordered index.

Using a Property Metadata Setting Do the following to create an unordered index on a property:

1. Open the project CDD in TIBCO BusinessEvents Studio and go to Domain Objects > Overrides.
2. Edit or create the override as needed for the desired entity or entities
3. In the Properties Metadata section, check the **Present in Index** checkbox for the property you want to index.
4. Save.

Adding Custom Functions

TIBCO BusinessEvents allows you to write your own custom functions in Java and add them to the function registry.

When you have added the custom functions to TIBCO BusinessEvents Studio, their catalog appears automatically in the Catalog Functions view, along with the built-in function catalogs.

TIBCO BusinessEvents documentation does not contain detailed instructions for creating custom functions or programming in Java. It is assumed that you are already familiar with programming in Java, and have at a minimum already implemented a class and a static function.

See also:

- [Restrictions on Use of Custom Functions on page 267](#)
- [Structure of a Function Catalog on page 268](#)

Task Summary

The steps below summarize the tasks required to integrate your own custom functions with TIBCO BusinessEvents Studio. An additional action is required make the functions available at runtime.

1. Write a custom static function in Java and compile it.
2. Create a file called `functions.catalog`, an XML file that makes it possible to access your custom functions from TIBCO BusinessEvents Studio. You can also include information for a tool tip for each function. See [Structure of a Function Catalog on page 268](#) for details.
3. Create a `.jar` file that includes the following:
 - The class files with your implementation.
 - The XML file described in [Structure of a Function Catalog on page 268](#).
4. Add the location of the JAR file or files (and any dependent JAR files) to the project properties. See [Working with External Library and Custom Function Paths on page 10](#) for details.

The functions are then available for use in any rule editor. (It is not necessary to restart TIBCO BusinessEvents Studio before you can use your functions.)



The locations of the custom function jars are stored in the `.beconfig` configuration file, which is located at the root level of the TIBCO BusinessEvents Studio project. You can check this configuration file into a version control system so that it can be shared with other project developers.

Note that the `.beconfig` file contains other information as well, such as the location of project libraries and so on.

Restrictions on Use of Custom Functions

Note the following restrictions that pertain to using custom functions.

Static and Non-Static Functions

Custom functions must be written in Java and have public static modifiers.

As a workaround, encapsulate a non-static function in a static function and compile the encapsulating class to get the `.class` file.

Return Types

TIBCO BusinessEvents custom functions support the following return types:

- Java types supported are: `Object`, `String`, `Calendar` (which displays in TIBCO BusinessEvents as `DateTime`), `Integer`, `Long`, `Double`, `Boolean`, `int`, `long`, `double`, and `boolean` (but not `byte`, `short`, `float`, or `char`).
- Entity and its subtypes, such as `Concept` or `Event`.
- Arrays are supported, with the exception of multidimensional arrays.

Name Overloading

The `functions.catalog` file makes functions available for use in TIBCO BusinessEvents Studio. The structure of the file requires each function within a class to have a unique name. Because of this structure, you cannot refer to an overloaded function in `functions.catalog`.

For example, the standard Java library has several `String.valueOf()` functions overloaded for each primitive type (`String.valueOf(int i)`, `String.valueOf(double d)` and so on). However, the TIBCO BusinessEvents standard function catalog has a separate function name for each data type: `valueOfBoolean()`, `valueOfDouble()`, `valueOfInt()`, and `valueOfLong()`.

See [Structure of a Function Catalog on page 268](#), for more about the `functions.catalog` file.

Editing Custom Functions

If you need to edit custom functions, you must do so in a Java editor or project, and then re-export them to a custom function JAR. Add the JAR to the TIBCO BusinessEvents project, as explained in [Adding Custom Functions on page 265](#).

Structure of a Function Catalog

A function catalog is an XML file that conforms to `function_catalog.xsd` — a schema. This allows TIBCO BusinessEvents to integrate your custom functions with the function registry in TIBCO BusinessEvents Studio. The function catalog must be in the XML format described in [Table 28](#) to map properly to the schema.



- Name the function catalog `functions.catalog`.
- Place `functions.catalog` in the root folder of the required Java archive resource (`.jar`) file.

Elements in the Function Catalog

[Table 28](#) lists and describes the elements used in the function catalog. Each element’s horizontal position within the [Element Name](#) column indicates the correct nesting position within the XML file.

Table 28 Function Catalog Elements

Element Name	Sub-Elements	Description
<code><catalog name="name"></code>		The root element. Attribute: <code>name="name"</code> where <i>name</i> is a name you provide for this functions catalog. Example: <code><catalog name="custom"></code>
<code><category></code>		This is a sub-element of <code><catalog></code> . <code><category></code> is a nesting container for a set of related functions within this functions catalog.
	<code><name></code>	A name you provide for this category.
<code><function></code>		A container for the information about a single function.
	<code><name></code>	The name of the function.
	<code><class></code>	The java class that implements the function.
	<code><desc></code>	Optional. A description of the function.
	<code><args></code>	A comma separated list of descriptive names for the function’s arguments. TIBCO BusinessEvents takes the argument type from the function itself.

Table 28 Function Catalog Elements (Cont'd)

Element Name	Sub-Elements	Description
	<code><async></code>	<p>Not used in this release.</p> <p>Required. Set to <code>false</code>.</p>
	<code><isActionOnly></code>	<p>If this function has side effects, for example, if it can modify values, you can only use it in action rules. Set this parameter to <code>true</code> to alert TIBCO BusinessEvents that this function has side effects.</p> <p>Optional. Valid values: <code>true, false</code></p>
	<code><reevaluate></code>	<p>Relevant only when a function is used in a condition.</p> <p>If set to <code>true</code> then:</p> <ul style="list-style-type: none"> TIBCO BusinessEvents does not memorize the result of the evaluation of the condition that contains this function. If any of the conditions is re-evaluated, then this function is also re-evaluated. <p>For example, <code><reevaluate></code> is set to <code>true</code> for <code>currentTimeMillis()</code>. Given this condition:</p> <pre>stock.price > 10.0; currentTimeMillis() - stock.time > 600000;</pre> <p>If the condition <code>stock.price > 10.0;</code> is re-evaluated, then <code>currentTimeMillis()</code> is also re-evaluated.</p> <p>If set to <code>false</code> then:</p> <ul style="list-style-type: none"> TIBCO BusinessEvents calls the function during the first evaluation and stores the result is stored and used for subsequent condition evaluations. TIBCO BusinessEvents Studio re-evaluates the condition only if another part of the same condition changes. <p>In the above stock price example, if <code><reevaluate></code> were set to <code>false</code>, then the condition would be re-evaluated only if <code>stock.time</code> changes.</p> <p>Optional. Valid values: <code>true, false</code></p>

Table 28 Function Catalog Elements (Cont'd)

Element Name	Sub-Elements	Description
	<code><isValidInBui></code>	<p>If this function can be used in decision tables, set this element to <code>true</code>.</p> <p>Optional. Valid values: <code>true</code>, <code>false</code></p>
	<code><isValidInQuery></code>	<p>If this function can be used in queries, set this element to <code>true</code>.</p> <p>Optional. Valid values: <code>true</code>, <code>false</code></p>
<code><tooltip></code>		<p>This is a sub-element of <code><function></code>. This element contains the elements of a tool tip. A tool tip provides information about a function when the user floats the cursor over the name of the function in the functions catalog in the rule editor.</p>
	<code><synopsis></code>	<p>A brief description of the function for display within the tool tip.</p> <p>Example:</p> <pre><synopsis>Takes value of "amount" and returns item name.</pre>
	<code><args></code>	<p>A container for descriptive information about the function's arguments. The information you provide will be displayed in the function's tool tip.</p>
	<code><paramdesc></code>	<p>A sub-element of <code><args></code>. Provides descriptive information for one argument. Use this tag once for each argument.</p> <p>Attributes:</p> <ul style="list-style-type: none"><code>name='arg_name'</code><code>type='arg_type'</code> <p>Example:</p> <pre><paramdesc name='amount' type='int'>arg1</paramdesc></pre>
	<code><returns></code>	<p>Describes the return value.</p> <p>Attribute: <code>type='return_type'</code></p> <p>Example:</p> <pre><returns type='string'>item name</returns></pre>

Example Function Catalog

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<catalog name="Custom">
  <category>
    <name>Categories</name>
    <category>
      <name>SimpleOnes</name>
      <function>
        <name>firstSample</name>
        <method>firstSample</method>
        <class>com.tibco.be.functions.custom.CustomJavaHelper</class>
        <args></args>
        <tooltip>
          <synopsis>You could write java here</synopsis>
        </tooltip>
      </function>
    </category>
    <category>
      <name>TimeBasedPropertyValues</name>
      <function>
        <name>snapshotOverTime</name>
        <method>snapshotOverTime</method>
        <class>com.tibco.be.functions.custom.CustomJavaHelper</class>
        <args>entity, startTime, endTime, namespace</args>
        <isActionOnly>true</isActionOnly>
        <tooltip>
          <synopsis>This is the synopsis</synopsis>
          <args>
            <paramdesc name="propertyDouble"
              type="PropertyAtomDouble">Property to serialize.</paramdesc>
            <paramdesc name="startTime" type="long">Start time for
              serialising the history values.</paramdesc>
            <paramdesc name="endTime" type="long">End time for
              serialising the history values.</paramdesc>
          </args>
          <returns type="String">An XML String Serialization
            of the PropertyAtomDouble passed.</returns>
        </tooltip>
      </function>
    </category>
    <category>
      <name>Serialize</name>
      <function>
        <name>serializeConcept</name>
        <method>serializeConcept</method>
        <class>com.tibco.be.functions.custom.CustomJavaHelper</class>
        <args>concept, changedOnly, namespace, root</args>
        <isActionOnly>true</isActionOnly>
        <tooltip>
          <synopsis>Serializes the Concept passed to an XML String
            which is returned.</synopsis>
          <args>
            <paramdesc name='concept' type='Concept'>The Concept to
              serialize.</paramdesc>
            <paramdesc name='changedOnly' type='boolean'>true - data
              modified since last conflict resolution cycle
              serialized</paramdesc>
            <paramdesc name='namespace' type='String'>A path describing
              the namespace to save in.</paramdesc>
            <paramdesc name='root' type='String'>The name of
              the serialized data.</paramdesc>
          </args>
        </tooltip>
      </function>
    </category>
  </category>
</catalog>
```

```
        <returns type='String'>An XML String Serialization of  
            the Concept passed.</returns>  
    </tooltip>  
</function>  
</category>  
</category>  
</catalog>
```

Chapter 17 **Rule Language Grammar**

This chapter describes the grammar for TIBCO BusinessEvents rules.

Topics

- [Rule Language Basics, page 274](#)
- [Keywords and Other Reserved Words, page 280](#)
- [Attributes, page 281](#)
- [Accessing Concept and Event Properties, page 283](#)
- [Exception Handling, page 286](#)
- [Flow Control, page 289](#)

Rule Language Basics

Whitespace

Whitespace is used to separate tokens (identifiers, keywords, literals, separators, and operators) just as it is used in any written language to separate words. Whitespace is also used to format code.

These are whitespace characters, excluding line terminators:

- the ASCII SP character, also known as "space"
- the ASCII HT character, also known as "horizontal tab"
- the ASCII FF character, also known as "form feed"

Line terminators include these characters:

- the ASCII LF character, also known as "newline"
- the ASCII CR character, also known as "return"
- the ASCII CR character followed by the ASCII LF character

Comments

Comment rules as shown:

`/* text */` TIBCO BusinessEvents ignores the text from “`/*`” to “`*/`”.

`// text` TIBCO BusinessEvents ignores the text from “`//`” to the end of the line.

Separators

The following tokens are used for separators:

- ; Statement separator for conditions and actions.
- (Expression Grouping begin, or function argument list begin.
-) Expression Grouping end or function argument list end.
- , Argument list separator.

Identifiers (Names)

An identifier (or *name*, to use the user interface label) is an unlimited-length sequence of letters and digits, the first of which must be a letter. Letters include uppercase and lowercase ASCII Latin letters A-Z, a-z, and the underscore (_).



Do not use the dollar sign (\$).

Identifiers are case sensitive.

Identifiers cannot have spaces (except shared resource identifiers).

Identifiers may not be the same as any literal, keyword, or other reserved word. See [Keywords and Other Reserved Words on page 280](#) and [Literals on page 277](#).

Letters and digits may be drawn from the entire Unicode character set, which supports most writing scripts in use in the world today, including the large sets for Chinese, Japanese, and Korean. This allows programmers to use identifiers in their programs that are written in their native languages.

Digits include the ASCII digits 0-9.

Two identifiers are the same only if they have the same Unicode character for each letter or digit. Note that some letters look the same even though they are different Unicode characters. For example, a representation of the letter A using \u0041 is not the same as a representation of the letter A using \u0391.

Two example identifiers: `new_order E72526 creditCheck`



Here is a more succinct way for programmers to understand the requirements:

```
<Identifier>  :=  [ <ID_START> { <ID_PART> }* ]
<ID_START>   :=  except '$', any character for which
java.lang.Character.isJavaIdentifierStart() returns true
<ID_PART>    :=  except '$', any character for which
java.lang.Character.isJavaIdentifierPart() returns true
```

Local Variables

You can use local variables of the following types in action rules and rule functions:

- Primitives
- Concepts
- Events

You can also use primitive arrays, which are fixed length. Here are examples of array declaration, initialization, and array creation expressions:

- Array declaration and initialization:

```
int i;                // int
int[] ii = {1,2,i};   // array of int
```

- Array creation with initialization expression:

```
ii = int[] {1,2,3};
```

- Array creation without initialization expression:

```
int[] arr = int[5] {};  
arr = int[5] {};
```

- Getting the length of the array:

```
int arrLength = arr@length;
```


Literals

The TIBCO BusinessEvents rule Language supports these literals:

- **int** — One or more digits without a decimal. May be positive or negative.
Examples: `4` `45` `-321` `787878`
- **long** — An integer literal suffixed with the letter L. The suffixed L can be either upper or lower case, but keep in mind that the lower case L (l) can be difficult to distinguish from the number one (1).
Examples: `01` `0777L` `0x100000000L` `2147483648L` `0xC0B0L`
- **double** — A number that can represent fractional values. D suffix is optional unless there is no decimal point or exponent.
Examples: `1D` `1e1` `2.` `.3` `0.0D` `3.14` `1e-9d` `1e137`
- **String** — Zero or more characters enclosed in double quotes (""). The string must be on one line. Use `\n` for newlines. Use the plus sign (+) to concatenate string segments.
Examples: empty string: `" "` (quotes with no space). Space character: `" "` (quotes with one or more spaces). Strings with values: `"P0QSTN3"` `"The quick brown fox had quite a feast!"` Strings spanning multiple lines:
`"The quick brown fox " +`
`"had quite a feast!"`
- **boolean** — One of these two values: `true` `false`
- **Null** — This value: `null`

Escape Sequences

You can represent characters in literals using these escape sequences:

Table 29 *Escape Sequences*

Character	Escape Sequence
backspace	\b
horizontal tab	\t
linefeed	\n
form feed	\f
carriage return	\r
double quote	\"
single quote	\'
backslash	\\

Operators

The language defines the following operators. Operators in this list that are also used in the Java language work the same as the Java operators:

Table 30 *Operators in the TIBCO BusinessEvents Rule Language*

Operator	Notes
++ --	increment, decrement
+ -	unary plus, unary minus
* \ %	multiplication, division, remainder
+ -	addition, subtraction
> < >= <=	greater than, less than, greater than or equal to, less than or equal to
=	assignment
== !=	equality, inequality (Does deep string comparison, unlike Java.)

Table 30 Operators in the TIBCO BusinessEvents Rule Language (Cont'd)

Operator	Notes
&& !	boolean AND, OR, NOT
- = += *= /= %=	combined operation and assignment. As an example, <i>expr</i> += 2; is the same as <i>expr</i> = <i>expr</i> + 2; += works on strings as well as numbers. For example, you have a variable <code>String s = "abc";</code> and then you use <code>s+= "def";</code> and so the value of <i>s</i> becomes "abcdef"
instanceof	Tests whether an object is an instance of specified type. Restricted to use with concepts and events. Example: <code>boolean b = customer instanceof USCustomer;</code>
.	property access
@	attribute access

Keywords and Other Reserved Words

Do not use the words listed in this section as identifiers, resource names, or folder names. Case sensitivity depends on context (as noted in documentation). The list also includes keywords and other reserved words for all add-on products.

\$lastmod	enum	null	union
abs	Event	object	unique
abstract	except	offset	using
accept	exists	or	validity
ACTION	extends	order	virtual
AdvisoryEvent	false	order	void
alias	final	pending_count	volatile
all	finally	package	when
and	first	policy	where
as	float	priority	while
asc	for	private	
assert	forwardChain	protected	
attribute	from	public	
avg	goto	purge	
backwardChain	group	QUERY	
between	having	rank	
body	hours	requeue	
boolean	if	return	
break	implements	rule	
by	import	rulefunction	
byte	in	scope	
case	instanceof	seconds	
catch	int	select	
char	interface	short	
class	intersect	SimpleEvent	
Concept	last	sliding	
CONDITION	is_defined	static	
const	is_undefined	strictfp	
ContainedConcept	key	String	
continue	last	sum	
count	latest	super	
date	like	switch	
DateTime	limit	synchronized	
days	lock	then	
dead	long	this	
declare	maintain	throw	
default	max	throws	
delete	milliseconds	time	
desc	min	TimeEvent	
distinct	minutes	timestamp	
do	mod	transient	
double	moveto	true	
emit	native	try	
else	new	tumbling	
entity	not	undefined	

Attributes

TIBCO BusinessEvents provides attributes that you can use in rules to access information of various kinds. The attributes are listed in [Table 31](#). Use the @ operator to access attributes.

Table 31 Attributes

Entity	Attributes	Type	Returns
SimpleEvent	@id	long	The event's unique internal ID.
	@extId	string	The event's unique external ID.
	@ttl	long	The time to live of the event as specified in the configuration. (This is not the time-to-live remaining.)
	@payload	string	The payload as a string value.
Repeating TimeEvent	@id	long	The time event's unique internal ID.
	@closure	string	null.
	@interval	long	The number of units between creation of successive time events.
	@scheduledTime	dateTime	The time scheduled for asserting into the Rete network.
	@ttl	long	0.
Rule-Based TimeEvent	@id	long	The time event's unique internal ID.
	@closure	string	A string that was specified when the event was scheduled.
	@interval	long	0.
	@scheduledTime	dateTime	The time scheduled for asserting into the Rete network.
	@ttl	long	The time to live of the event as specified when scheduling the event. (This is not the time-to-live remaining.)

Table 31 Attributes (Cont'd)

Entity	Attributes	Type	Returns
Advisory Event	id	long	The advisory event's unique internal ID
	extId	String	Null
	category	String	Broad category of advisory, for example, an exception.
	type	String	Type of advisory within the category.
	message	String	Message for the user.
Concept	@id	long	The concept instance's unique internal ID.
	@extId	string	The concept instance's unique external ID.
ContainedConcept	@id	long	The contained concept instance's unique internal ID.
	@extId	string	The contained concept instance's unique external ID.
	@parent	concept	The parent concept instance. (This is treated as a concept reference in the language.)
PropertyAtom	@isSet	boolean	True if the property value has been set. Otherwise, false.
PropertyArray	@length	int	The number of PropertyAtom entries in the array.



The internal ID is automatically generated by TIBCO BusinessEvents. You cannot set it.

Accessing Concept and Event Properties

This section describes how to access concept properties and event properties using the TIBCO BusinessEvents language.

Concept Property Atom

This is the syntax for accessing a concept property atom:

instanceName.propertyName

where *instanceName* is the identifier of the concept instance, and *propertyName* is the name of the concept property that you want to access.

For example to get the current value of the cost propertyAtom:

```
int x = instanceA.cost;
```

For example, to set a value with the current system timestamp:

```
instanceA.cost = value;
```



If the history size is 0, TIBCO BusinessEvents does not record a timestamp.

Get and Set PropertyAtom Value With User-Specified Time

You can get and set PropertyAtom values as follows:

- You can specify a time and get the PropertyAtom value stored in the history at that time using one of the standard functions:

```
type Instance.PropertyAtom.get type(PropertyAtom propertyName, \
                                     long time)
```

where *type* is the type of the PropertyAtom and *propertyName* is the name of the PropertyAtom, and *time* is the time from which you want to retrieve the value.

- You can set a value in the PropertyAtom History using one of the standard functions:

```
Instance.PropertyAtom.set type(PropertyAtom propertyName, \
                                type value, long time)
```

where *type* is the type of the PropertyAtom and the type of the new value, *propertyName* is the name of the PropertyAtom, *value* is the value to store in the ring buffer, and *time* is the timestamp for the new entry.

TIBCO BusinessEvents manages these requests as follows:

- **If the ring buffer has vacancies**, TIBCO BusinessEvents inserts the new entry into the correct place based on its timestamp, shifts the older values out one place, and returns True.
- **If the ring buffer is full, and the new value has a more recent timestamp than the oldest value**, TIBCO BusinessEvents inserts the new value into the correct place, shifts older values if necessary, drops the oldest value, and returns True.
- **If the ring buffer is full, and the new value has a timestamp that is older than the oldest value in the ring buffer**, TIBCO BusinessEvents does not insert the new value into the ring buffer, and it returns False.

Concept Property Array

This is the syntax for accessing a concept property array:

instanceName.propertyName

where *instanceName* is the identifier of the concept instance, and *propertyName* is the name of the concept property that you want to access.

Accessing a Value in the Property Array

To access a value in a property array, identify the position in the array of the value as shown:

instanceName.propertyName [*indexPosition*]

For example:

```
String x = instanceA.lineItem[0];
```

This gets the current value of the first property atom in the array, `lineItem`, and assigns it to the local variable, `x`.



Array index difference In the TIBCO BusinessEvents language, array indexes start from zero (0). However, in XSLT and XPath languages, they start from one (1). It's important to remember this difference when using the rule language in the rule editor, and when working in the XSLT mapper and the XPath builder.

Setting the Value for an Existing Array Position

You can set the value of an existing position in an array. For example:

```
int[] ii = {1,2,3};
ii[2] = 1;
```


Adding a Value to a Property Array

You can append a value to the end of a property array. You cannot, however, add a value to any other position in an array. This is the syntax:

```
instanceName.propertyName[indexPosition] = value
```

To use the syntax shown above you must know the index position of the end of the array. You can append a value to the end of an array without knowing the index position of the end of the array using the `@length` attribute as shown:

```
instanceName.propertyName[instanceName.propertyName@length] = value
```

Event Property

This is the syntax for accessing an event property:

```
eventName.propertyName
```

For example:

```
String x = eventA.customer;
```

where *eventName* is the identifier of the concept instance and *propertyName* is the name of the event property that you want to access.

Exception Handling

The TIBCO BusinessEvents rule Language includes try/catch/finally blocks and has an Exception type. The `try/catch/finally` blocks behave like their same-name Java counterparts.



Advisory Events You can also use the special AdvisoryEvent event type to be notified of exceptions that originate in user code but that are not caught with a `catch` block. To use the AdvisoryEvent, click the plus sign used to add a resource to the declaration. AdvisoryEvent is always available in the list of resources. See [Chapter 10, Advisory Events, on page 157](#).

This section describes the `try/catch/finally` commands.

Syntax

These combinations are allowed:

- `try/catch`
- `try/finally`
- `try/catch/finally`

```
try      try {
           try_statements
        }

catch    catch (Exception identifier) {
           catch_statements
        }

finally  finally {
           finally_statements
        }
```

Examples

This section provides some examples to demonstrate use of exception handling.

try/finally Example

```
String localStatus = "default status";
try {
    //readStatus might throw an exception
    localStatus = readStatus();
} finally {
    //If readStatus() throws an exception,
    //MyScorecard.status will be set to "default status"
    //but the exception won't be caught here.
    //Otherwise MyScorecard.status will be set to the
    //return value of readStatus()
    MyScorecard.status = localStatus;
}
```

try/catch/finally Example

```
String localStatus = "default status";
try {
    //readStatus might throw an exception
    localStatus = readStatus();
} catch(Exception exp) {
    System.debugOut("readStatus() threw an exception with message"
        + exp.getMessage());
} finally {
    //If readStatus throws an exception,
    //MyScorecard.status will be set to "default status"
    //Otherwise MyScorecard.status will be set to the
    //return value of readStatus()
    MyScorecard.status = localStatus;
}
```

try/catch Example

```
String localStatus = "default status";
try {
    //readStatus might throw an exception
    localStatus = readStatus();
} catch(Exception exp) {
    System.debugOut("readStatus() threw an exception with message "
        + exp.getMessage());
}

//If readStatus throws an exception,
//MyScorecard.status will be set to "default status"
//Otherwise MyScorecard.status will be set to the
//return value of readStatus()
MyScorecard.status = localStatus;
```

Flow Control

The TIBCO BusinessEvents rule Language includes commands to perform conditional branching and iteration loops. This section describes these commands.

if/else

The `if/else` command allows you to perform different tasks based on conditions.

Syntax:

```
if ( condition ) {  
  code_block;  
}  
else {  
  code_block;  
}
```

for

The `for` command allows you to create a loop, executing a code block until the condition you specify is false.

Syntax:

```
for ( initialization; continue condition; incrementor ) {  
  code_block;  
  [break;]  
  [continue;]  
}
```

`break` allows you to break out of the loop.

`continue` allows you to stop executing the code block but continue the loop.

For example:

```
for(int i=1; i<10; i=i+1){  
  System.debugOut("Hello World!");  
}
```

This example prints "Hello World!" to `debugOut` ten times.

while

The `while` command allows you to perform one or more tasks repeatedly until a given condition becomes false.

Syntax:

```
while ( condition ) {  
    code_block;  
    [break;]  
    [continue;]  
}
```

`break` allows you to break out of the loop.

`continue` allows you to stop executing the code block but continue the loop.

Chapter 18 **Rule Language Datatypes**

This chapter provides datatype conversion tables, information about operators and types, and information about how TIBCO BusinessEvents handles inconstancy problems with datatypes.

Topics

- [Concept Properties to XML Datatype Conversions, page 292](#)
- [Compatibility of Operators with Types, page 293](#)
- [Correcting Inconsistencies of Type, page 295](#)

Concept Properties to XML Datatype Conversions

Table 32 Concept Properties to XML Datatype Conversions

Property Type	Int	Long	Float	Double	Boolean	String	DateTime	ComplexType	@ref
int	L	L	L	L		L			
long	N	L	N	N		L			
double	N	N	N	L		L			
String	L	L	L	L	L	L	L		
boolean					L	L			
Datetime						L	L		
ContainedConcept								D	
ConceptReference									ID

N - Numeric conversion, loss of information possible (see note below).

L - Shallow copy — Copies only the current value.

D - Deep copy — Copies the entire structure of the contained concept (current values of all properties only).

ID - Basically a shallow or reference-only copy. The copy refers to the same instance of the concept.



- History is never copied.
- Data loss is possible for conversions from String to a number datatype if the string represents a very large number that would have to be clipped.
- Datatype conversion tables for events are located in the TIBCO Rendezvous and TIBCO Enterprise Message Service documentation.

Compatibility of Operators with Types

Table 33, Operator Matrix, defines the compatibility of operators with types.

Table 33 Operator Matrix

		Right Side of Operator							
		str	int	lon	dou	boo	ent	obj	dat
Left Side of Operator	str	=, +, eq, cmp, inst	+	+	+	+	+	=, +, eq, cmp, inst	+
	int	+	=, math, eq, cmp	=, math, eq, cmp	=, math, eq, cmp			=, math, eq, cmp	
	lon	+	=, math, eq, cmp	=, math, eq, cmp	=, math, eq, cmp			=, math, eq, cmp	
	dou	+	=, math, eq, cmp	=, math, eq, cmp	=, math, eq, cmp			=, math, eq, cmp	
	boo	+				=, eq		=, eq	
	ent	+					=, eq, inst	=, eq, inst	
	obj	=, +, eq, cmp, inst	=, math, eq, cmp	=, math, eq, cmp	=, math, eq, cmp	=, eq	=, eq, inst	=, eq, inst	=, eq, inst
	dat	+,						=, eq, inst	=, eq, cmp, inst

Abbreviation	Meaning and Notes
boo	Boolean.
cmp	Comparison operators: <, >, <=, >=

Abbreviation	Meaning and Notes
dat	Date/Time
dou	Double
ent	Entity. Type includes Concept, Event and Scorecard. Both operands must either be of the same type or have a subtype-supertype relationship
eq	Equality operators: ==, !=
inst	instanceof
int	Integer
lon	Long
math	Numerical operators: unary +, unary -, =, -, *, /, %
obj	Object
str	String

Correcting Inconsistencies of Type

TIBCO BusinessEvents attempts to correct inconsistencies of type whenever possible by converting expressions to the appropriate type. TIBCO BusinessEvents converts expression types in the following cases:

- An expression uses the plus sign (+) with a string operand.
- An arithmetic expression includes numbers of differing types.
- The value of an expression is assigned to a variable of a different type.
- The value of an expression is passed to a function that declares a different type.

There are some inconsistencies of type that TIBCO BusinessEvents cannot correct. For example, all expressions within conditions must be of type boolean. If an expression within a condition evaluates to anything other than boolean, it would be illogical for TIBCO BusinessEvents to convert the expression to boolean. In cases like this, TIBCO BusinessEvents returns an error at compile time.

String Operands

When an expression uses the plus sign (+) with a string operand, TIBCO BusinessEvents treats the expression as a request for concatenation rather than addition. It converts the second operand to a string and concatenates the two strings.

For example:

```
"area code: " + 650    becomes
"area code: 650"
```

Arithmetic Expressions

The following information applies to these operators:

```
*  /  %  +  -  <  <=  >  =  ==  !=
```

When an expression uses one of the above arithmetic operators with two numbers of different numeric types, TIBCO BusinessEvents promotes one of the two operands to the numeric type of the other. It makes these promotions as follows:

- If either operand is a double, TIBCO BusinessEvents promotes the other to a double.
- Otherwise, if either operand is a long, it promotes the other to a long.

Assignment Conversion

If the value of an expression is assigned to a variable, TIBCO BusinessEvents converts the expression's type to that of the variable. This might include, for example, converting a double to an int, or converting a generic model type to a more specific model type.

Function Argument Conversion

Conversions of function arguments are handled in the same way as assignment conversions.

Chapter 19 Mapping and Transforming Data

This chapter describes how to map variables in the scope of a rule or rule function to arguments of a function used in that rule or rule function.

See also [Chapter 20, XPath Formula Builder, on page 335](#) for related information.

See [Chapter 15, Rules and Rule Functions, on page 219](#) for information about working with rules and rule functions.

Topics

- [Overview of Mapping and Transformation, page 298](#)
- [Buttons, Menus, and Icons, page 301](#)
- [Specifying Constants, page 307](#)
- [Data Validation, page 308](#)
- [Repairing Incorrect Mappings, page 309](#)
- [Shortcuts, page 310](#)
- [Examples of Mappings, page 314](#)
- [XSLT Statements, page 330](#)

Overview of Mapping and Transformation

The Function Argument Mapper allows you to supply the data that a function expects as input.

For instructions on accessing the Function Argument Mapper in the rule editor see [Using the Function Argument Mapper on page 234](#)

Function Section

The function section, in the upper part of the dialog, shows the view-only name of the function you are working with and the editable entity path to the item whose properties and attributes you want to map to the function arguments.

Input Section

Scope Variables Panel

The scope variables panel shows the list of properties and attributes available to the function, as well as global variables defined in the project.

Function Panel

The function panel uses an Extensible Stylesheet Language Transformation (XSLT) template that specifies how scope variables should be transformed to provide the expected input. Normally, you do not need detailed knowledge of XSLT to specify a function's expected output. However, if you are familiar with XSLT and you wish to see the actual code, you can right-click on any item in the Function panel and choose Copy from the popup menu. Then open a blank text document and choose Paste. The XSLT is displayed in your text document.


You can also use your own XSLT templates to perform transformations instead of using the techniques described in this chapter. You can paste XSLT into your function input fields, or into the XPath Formula field in XPath Formula Builder. (You cannot, however, paste XSLT directly into the function argument in the rule editor.)

Mapping and Transforming Data to Function Input



Assigning an empty string (" ") to a field in a mapper function results in a null string.

To map data, select an item in the Scope Variables panel, then drag and drop that item into the desired schema element in the Function panel.

Simple mappings appear in the formula area to the right of the input element after you release the mouse button. For more complex mappings, click the Edit Statement () button.


Most options in the Edit Statement dialog are straightforward. However, there are some complex scenarios that require multiple steps. Many of these situations are described in the section [Examples of Mappings on page 314](#). You may also wish to refer to [XSLT Statements on page 330](#) for a reference of XSLT statements when deciding which XSLT statement can be used to achieve the result you desire.

You can specify XPath formulas to transform an element if you need to perform more complex processing. The XPath Formula Builder allows you to easily create XPath formulas. For more advanced use of XPath, see [Chapter 20, XPath Formula Builder, on page 335](#). There are also a variety of third-party books and resources about XSLT and XPath.

The datatypes of the function's arguments display as hints. Once a mapping or formula is specified, a hint becomes an XSLT statement. See [Statements, Hints, and Errors on page 299](#) for more information about hints and statements.

Statements, Hints, and Errors

When you display the Function tab, the existing statements are examined, and any input elements that do not have a statement are displayed as hints. Hints are reminders that you can specify a statement for the input element, but they are not stored as part of the XSLT template for the function's input. Hints are displayed in italics with a light blue background. Once you specify a mapping or a formula for a hint, the input element becomes a statement. You can also drag the hint to the left past the dividing line between the panels and the hint becomes a blank statement.

Once you specify a statement in the Function panel and click OK, it becomes part of the XSLT template used to create the input data. Statements are only deleted if you manually delete them using the delete button, or if you use the Mapper Check and Repair () button to automatically fix errors. Therefore, if the input schema for the function changes, your statements may no longer be valid. See [Repairing Incorrect Mappings on page 309](#) for more information about using the Mapper Check and Repair button to fix statements in the Function panel.

Any statement or hint that has an error is displayed in red. A hint is only displayed in red if it is a required input element. All required input elements must have statements specified. The Mapper Check and Repair button can help you automatically fix some errors. See [Repairing Incorrect Mappings on page 309](#) for more information about fixing errors.

Buttons, Menus, and Icons

The Input tab contains several toolbar buttons, popup menus, and icons. This section describes the various graphical elements of the Input tab.

Toolbar and Right-Click Menu on the Input Section

The Scope Variables panel and the Function panel have several buttons for performing various functions. There is also a popup menu when you right-click on elements in each panel. [Table 34](#) describes the buttons and right-click menu items available in the panels of the Input tab.

Table 34 Input tab toolbar buttons




Button	Right-Click Menu	Description
Scope Variables Panel		
		Coercions. Allows you to specify a type for Scope Variables elements that are not a specific datatype. For example, a choice element can be coerced into one of the possible datatypes for the element, or an element of datatype <code>any</code> can be coerced into a specific datatype.
		Type Documentation. Allows you to specify or view documentation for schema elements.
	Expand	This menu item has two sub-menus: Content and All. Expand > Content expands the current element so that all elements that are currently used in a mapping are visible. Expand > All expands all sub-elements of the currently selected element.
	Show Connected	Expands the elements in the Function area to display elements that are mapped to the currently selected element or its sub-elements.
	Delete	Deletes the selected element.
	Copy	Copies the selected element. The element can be later pasted.
Function Panel		
		Shows or hides the mapping formulas for the input elements.

Table 34 Input tab toolbar buttons








Button	Right-Click Menu	Description
		Move Up. Moves the selected element up in the Function tree.
		Move Down. Moves the selected element down in the Function tree.
	Move Out	Move Out. Promotes the selected element to the next highest level in the Function tree.
	Move In	Moves the currently selected element into a new statement. This displays the Move Into New Statement dialog that allows you to choose the statement you wish to move the element into. See XSLT Statements on page 330 for more information about XPath statements.
	Delete	<p>Deletes the mapping for the selected element. If no mapping is defined, the element itself is deleted (along with all child elements).</p> <p>Note Elements are predefined. Do not delete elements. Deletion of an element causes mapper validation errors due to the mismatch of the right panel's content with its schema.</p>
		<p>Insert. Click Insert to pop-up a New XSLT Statement dialog where you can define an XSLT statement. The statement is inserted in the function input schema on the same level of the hierarchy as the currently selected element.</p> <p>You can add one XSLT statement at a time with this button. The right-click menu item Statement provides a shortcut for multi-line statements, such as Choice or If. See the description of the Statement menu item below for more information.</p> <p>Note Elements are predefined. Do not add new elements. Doing so causes a mismatch of the right panel's contents with its schema.</p> <p>See XSLT Statements on page 330 for more information about XSLT statements.</p>
		Add Child. Adds a statement for a child element to the currently selected element.

Table 34 Input tab toolbar buttons




Button	Right-Click Menu	Description
		<p>Mapper Check and Repair. Verifies the XSLT template you have created in the Function panel against the expected input. A list of errors and warnings appear and you can choose which items you wish to fix. TIBCO BusinessEvents attempts to fix simple problems such as adding missing items that are expected.</p> <p>See Repairing Incorrect Mappings on page 309 for more information.</p>
		<p>Edit Statement. Allows you to modify an XSLT statement for the element.</p> <p>See XSLT Statements on page 330 for more information about XSLT statements.</p>
	Edit	<p>XPath Formula Builder. Invokes the XPath formula builder. You can use this editor to create an XPath statement for this input element. See Chapter 20, XPath Formula Builder, on page 335 for more information about XPath and the XPath formula builder.</p>
	Expand	<p>This menu item has three sub-menus: Content, Errors, and All. Expand > Content expands the current element so that all sub-elements that have a mapping or expression are visible. Expand > Errors expands the current element so that all sub-elements that have an error in their expression are visible. Expand > All expands all sub-elements of the currently selected element.</p>
	Show Connected	<p>Expands the elements in the Scope Variables panel to display elements that are mapped to the currently selected element or its sub-elements.</p>
	Statement	<p>This menu item contains shortcuts that allow you to easily add the desired XSLT statement(s) with one menu item instead of adding the statement(s) with the Insert button. See Statement Menu Options on page 310 for a description of the sub-items of this menu.</p>
	Undo operation	<p>Rolls back the last operation performed. The name of the last operation is shown.</p>
	Redo operation	<p>Performs the last operation that was undone with the Undo menu item. The name of the last operation is shown.</p>
	Cut	<p>Deletes the selected element. The element can be later pasted to a new location.</p>

Table 34 Input tab toolbar buttons

Button	Right-Click Menu	Description
	Copy	Copies the selected element.
	Paste	Pastes the last element that was copied or cut.

Icons for Schema Element Datatypes

Schema elements also have a set of associated icons to indicate their type. [Table 35](#) describes the icons used for schema items.







You can use the Type Documentation button to obtain any available documentation on any node in the Scope Variables or function input schema trees.

Table 35 Icons for schema items

Icon	Description
	Complex element that is a container for other datatypes. This is also called a <i>branch</i> in the schema tree.
	Simple string or character value.
	Simple integer value.
	Simple decimal (floating point) number.
	Simple boolean value.

Table 35 Icons for schema items

Icon	Description
	Simple Date or Time. This can be any of the following datatypes: <ul style="list-style-type: none"> • Time • Date • Date & Time • Duration • Day • Month • Year • Month & Year • Day & Month
	Simple binary (base 64) or hex binary value.
	Represents a schema item that can be any datatype. Data in this schema element can be any datatype.
	Choice. Specifies that the actual schema element can be one of a specified set of datatypes.

Qualifier Icons





Schema elements can have additional icons that specify qualifications. The qualifier icons have different meanings depending upon where they appear. For example, a question mark icon signifies an element is optional in the Scope Variables schema or in a hint in the Function panel. However, in an XSLT statement, the question mark signifies the statement is "collapsed" and an implicit "if" statement is set, but not displayed in the Function panel.

[Table 36](#) describes the additional qualifiers that appear next to the name of schema items.

Table 36 Additional icons for hints

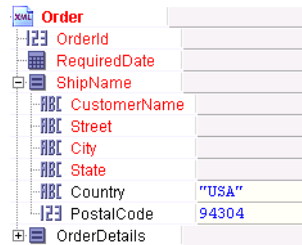
Qualifier	Scope Variables or Hint	Statement
	No qualifier indicates the element is required.	N/A

Table 36 Additional icons for hints

Qualifier	Scope Variables or Hint	Statement
	A question mark indicates an optional Item.	An implicit "if" statement is set for this statement. This occurs when you map an optional element from the Scope Variables to an optional element in the function input schema or if you specify Surround element with if test on the Content tab of the Edit Statement dialog.
	An asterisk indicates the item repeats zero or more times.	N/A
	A plus sign indicates the item repeats one or more times.	N/A
	A null sign indicates the item may be set to null.	<p>A null sign indicates the item is explicitly set to null.</p> <p>You can set an element explicitly to null by clicking the Edit Statement button for the element, then checking the Set Explicit Nil field on the Content tab of the Edit Statement dialog.</p>

Specifying Constants

For each element in the Function input schema tree, you can specify a constant. Constants can be strings or numeric values. To specify a string, enclose the string in quotes. To specify a number, type the number into the schema element's mapping field. The following illustrates specifying the string "USA" for the `Country` item and 94304 for the `PostalCode` item of a function input schema.



Constants can also be used in functions and search predicates. To learn more about complex XPath expressions that use functions and search predicates, see [Chapter 20, XPath Formula Builder, on page 335](#).

Date and Datetime Strings in Constants

In constant expressions used in bindings, datetime values are read in according to the ISO 8601 standard, as described in the XML Schema specification. For example, the value:

```
"2002-02-10T14:55:31.112-08:00"
```

is 55 minutes, 31 seconds and 112 milliseconds after 2pm on February 10th, 2002 in a timezone that is 8 hours, 0 minutes behind UTC.

If no timezone field is present, the value is interpreted in the timezone of the machine that is performing the parsing. This can lead to complications if you are processing data from different timezones, so you are encouraged to always use timezones.

When TIBCO BusinessEvents generates datetime strings UTC time is always used. The output for the example above is:

```
2002-02-10T14:55:31.112Z
```

which is the equivalent time in the UTC timezone.

Data Validation

Data passed as input to a function is validated to ensure that it conforms to its specified datatype.

Table 37 describes the validation behavior. Datatype validation listed with the prefix `xsd:` is defined in the namespace <http://www.w3.org/2001/XMLSchema>. See *XML Schema Part2: Datatypes* specification at <http://www.w3.org/TR/2004/PER-xmlschema-2-20040318/> for more information on the proper representation of these datatypes. Datatype validation listed with the prefix `xdt:` is defined in the namespace <http://www.w3.org/2003/11/xpath-datatypes>. See *Xquery 1.0 and Xpath 2.0 Functions and Operators* specification at <http://www.w3.org/TR/2003/WD-xpath-functions-20031112/> for more information on the proper representation of these datatypes.

Table 37 Datatype validation

Data Type	Validation
boolean	xsd:boolean
double	xsd:double
string	xsd:string
dateTime	xsd:dateTime
long	xsd:long
int	xsd:int

Repairing Incorrect Mappings

Any incorrect statements are displayed in red in the Function panel. Errors can occur for a number of reasons. For example,


- a required element has no statement, and therefore must be specified
- the function's input schema has changed and existing statements may no longer be valid
- the XPath formula for an element may contain an error

You should correct any errors before attempting to test or deploy your process definition.

To help find potential problems in your mappings, click the Mapper Check and Repair button. This button displays a dialog with all potential problems in the specified mappings. You can select the Fix checkbox for potential errors, and TIBCO BusinessEvents will attempt to fix the problem.

Some potential problems in the Mapper Check or Repair dialog cannot be fixed easily, and therefore there is no checkbox in the Fix column for these items. For example, if an element expects a string and you supply a complex type, the corrective action to fix the problem is not clear, and therefore TIBCO BusinessEvents cannot automatically fix the problem. You must repair these items manually.

If you want to return to the original expected Function input and remove all of the currently specified mappings, perform the following:

1. Delete the root element of the function's input by selecting it and clicking the Delete button.
2. Click the Mapper Check and Repair () button.
3. In the Mapper Check and Repair dialog Fix column, select the checkboxes for all items.
4. Click OK.

Alternatively, you can simply select the root input element and press the delete key on your keyboard as a shortcut for the procedure above.

After deleting all mappings and schema items and then repairing the input schema, the function's input reverts to the state before you open the Argument Mapping Wizard for the first time for this function.

Shortcuts

The Move In, Insert, Add Child, and Edit Statement buttons on the toolbar are ways to manually manipulate XSLT statements in the Function panel. These buttons, however, only add or modify one statement at a time. Also, there are some situations where you wish to convert a hint into a statement without performing any mapping. This section describes shortcuts for manipulating XSLT statements.

Statement Menu Options

When you select an element in the function input schema and right-click, a menu appears. The Statement menu item contains several sub-items that are useful shortcuts for creating XSLT statements.

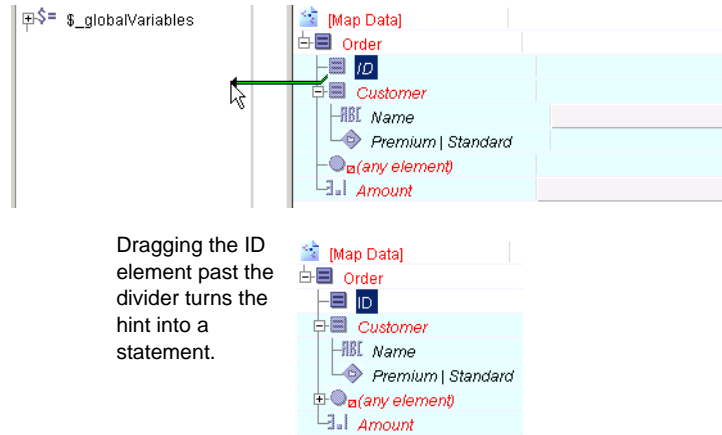
- Surround with Choice — a shortcut for adding a choice statement and its associated conditions or otherwise statements around the currently selected element.
- Surround with If — a shortcut for adding an if statement and placing the currently selected element as the sub-element of the if.
- Surround with For-Each — a shortcut for moving the current element into a For-Each statement.
- Surround with For-Each-Group — a shortcut for moving the current element into a For-Each-Group statement and adding a Group-By grouping statement.
- Duplicate — a shortcut for creating a duplicate of the currently selected element (including any mappings or XPath formulas for the element). The duplicate is added below the currently selected element.
- Insert Model Group Content — a shortcut for inserting the contents of a selected model group into the mapper tree. The selected element in the function input schema is replaced by the contents of the model group you select.

Dragging to the Left

Dragging an element in the function input schema to the left past the divider between the two areas of the Input tab changes a hint into an XSLT statement. [Figure 5](#) illustrates dragging an element to the left.

Figure 5 *Dragging to the left to change a hint to a statement*

Before dragging, the ID element is a hint.



This shortcut is useful in the following situations:

- When you have a complex element with no sub elements and no content.
- When you have a choice element, dragging to the left brings up the Mapping Wizard and allows you to choose a type for the element.
- When you have an element of type Any, dragging to the left brings up a dialog that allows you to specify the type for the element.

Cutting and Pasting

The Function panel is an XSLT template for specifying the function's input schema. You can choose any element in the Function panel and select Copy from the right-click menu or press the Control-C keys to copy the XSLT statement for the element. Once the XSLT is copied, you can paste it into a text editing tool to view or modify the code.

You can also paste arbitrary XSLT code into the Function panel using the right-click menu or the Control-V keys. Pasting XSLT code from the copy buffer places the code above the currently selected element in the Function panel.

Automatic Testing (at Runtime)

When you map Scope Variables elements to Function elements, the behavior of the mapping depends upon the types of elements you are mapping. In the simplest case of mapping a required element in the Scope Variables schema to a required Function element, the value of the Scope Variables element is assigned to the required Function element.

However, when elements are optional or nillable, more complex tests are necessary. When you drag the Scope Variables element to the Function element, the necessary tests are automatically placed into the Function XSLT template.

This section describes the result of mapping different types of elements. The types of mappings are described, then an example is given that illustrates these mappings and shows the XSLT code that is generated automatically when these mappings are performed at runtime.

Required to Required

Specifies that the statement should always include the required Function element and its value should be obtained from the required Scope Variables element that the element is mapped to.

Optional to Optional

Specifies that the statement should test if the Scope Variables element is present, and if so, include the optional element in the function's input. If the Scope Variables element is not present, the optional element is omitted from the function's input.

Nillable to Nillable

Specifies that both the Scope Variables and Function elements can be nil. Therefore, the value of the Function element is set to the value of the Scope Variables element. The value of the Function element is set explicitly to nil if that is the value of the Scope Variables element.

Optional to Nillable

Specifies that the statement should test if the optional Scope Variables element exists. If the element exists, the Function element should be created and set to the value of the Scope Variables element. If the Scope Variables element does not exist, the element is omitted from the function input schema.

Nillable to Optional

Specifies that the statement should test if the Scope Variables element has a value specified, and if so, the optional element in the Function panel should be set to the value of the Scope Variables element. Otherwise, if the Scope Variables element is nil, the optional element is omitted from the Function panel.

Optional & Nillable to Optional & Nillable

Specifies that if the optional Scope Variables element exists, then include the optional Function element in the input schema. If the Scope Variables element is nil, set the value of the Function element explicitly to nil. If the Scope Variables element is not nil, set the value of the Function element to the value of the Scope Variables element. If the Scope Variables element is not present, then omit the optional element from the function input schema.

Examples of Mappings


Some mappings require several steps to achieve the desired results. This section describes some complicated mapping scenarios and how to achieve the desired mappings using the tools available.

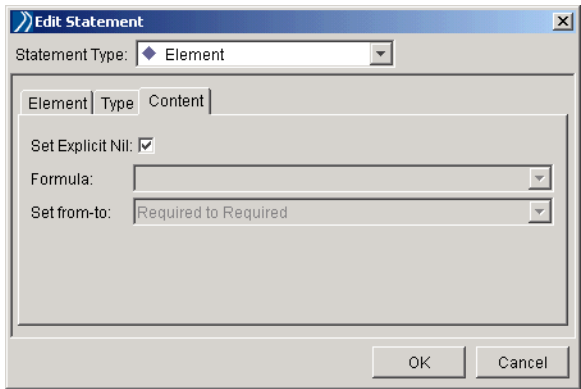


There are many methods to insert or modify XSLT statements in the function input schema. The examples in this section illustrate the simplest procedures to obtain the desired results. However, you do not have to follow the same procedures outlined in this section to achieve the correct set of statements.

Setting an Element Explicitly to Nil

In some situations, you may wish to set an element explicitly to nil. One situation is when you wish to insert a row into a database table and you wish to supply a NULL for one of the columns. To set an input element explicitly to nil, perform the following:

1. Select the input element you wish to set to nil.
2. Click the Edit Statement () button on the Input tab toolbar.
3. Click the Content tab of the Edit Statement dialog.
4. Check the checkbox in the Set Explicit Nil field.

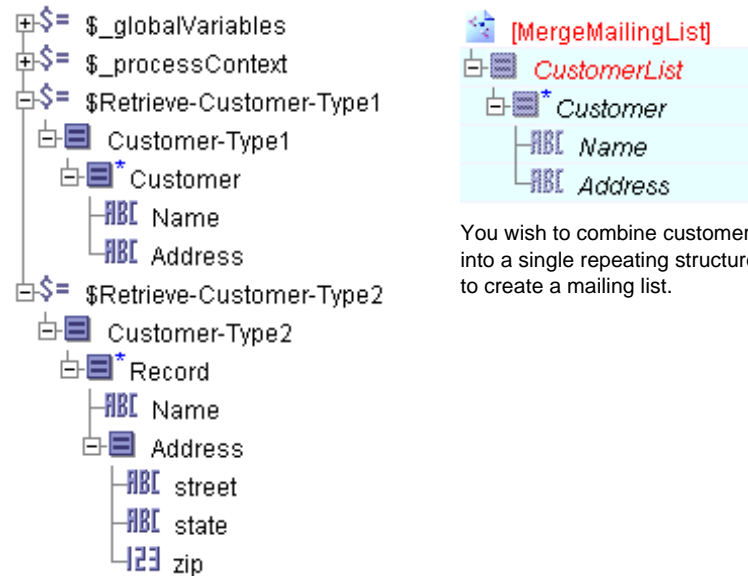


The element's formula becomes blank and is not editable (because nil is the value of the element) and the explicit nil qualifier icon appears next to the statement



Merging Input from Multiple Sources

You may have multiple items in the Scope Variables that you wish to map to one repeating element in the Function panel. For example, you may have multiple formats for customer records and you wish to create a single, merged mailing list containing all customers in one format. In this example, the schemas are the following:



Multiple types of customers are retrieved, each having a different format for the address.

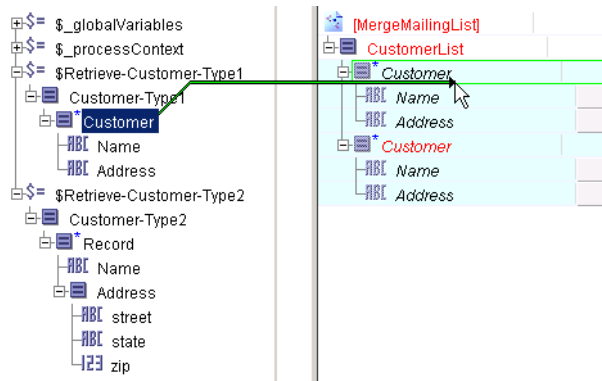
The following procedure describes how to map multiple elements into a single repeating element.

1. Select the repeating element in the Function panel, right-click, and select **Statement > Duplicate** from the menu.

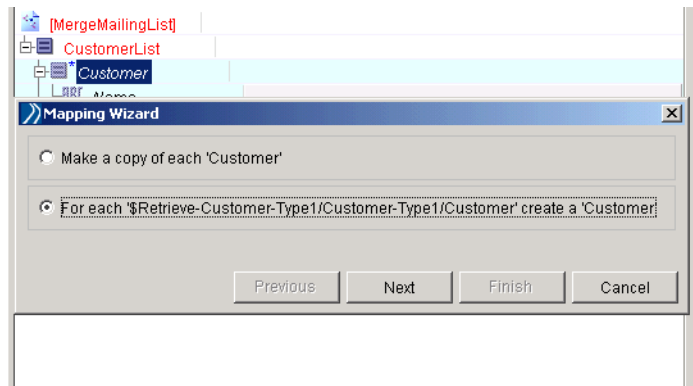
Because you are creating two different formulas for mapping, you need two copies of the repeating element, one for each format. The resulting output contains only one repeating customer element, but the two copies in the Function panel make it simpler to perform two different mappings.

2. Map one of the elements from the Scope Variables to the first copy of the repeating element in the Function. For example, map

```
$Retrieve-Customer-Type1/Customer to  
MergeMailingList/CustomerList/Customer.
```



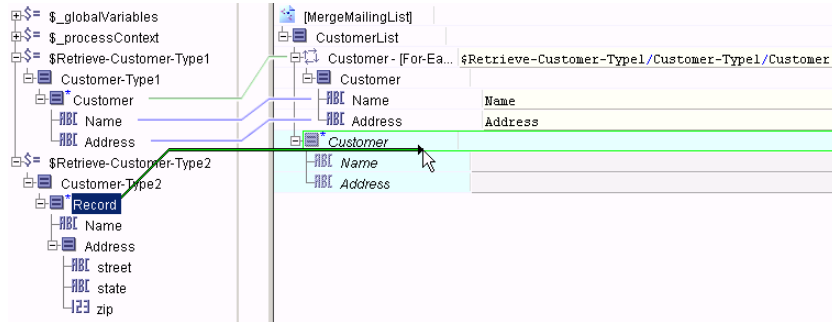
The Mapping Wizard dialog appears and presents choices for what you would like to accomplish. Choose the For Each option and click **Next**.



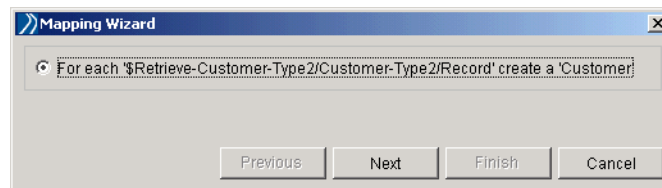
The mapping wizard asks if you wish to automatically map items with the same names. Click **Finish** to accept the default mappings.

- 3. Map the other element from the Scope Variables to the second copy of the repeating element in the Function. For example, map

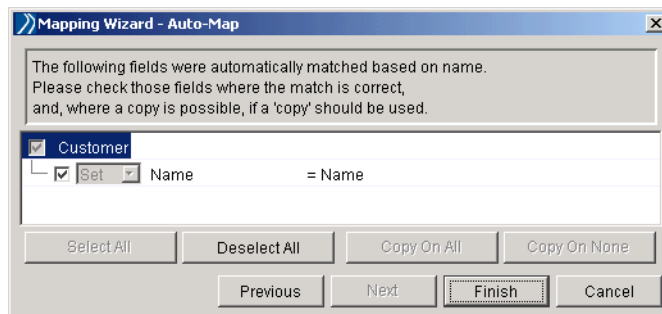
\$Retrieve-Customer-Type2/Record to
MergeMailingList/CustomerList/Customer.



In the Mapping Wizard dialog, choose the For Each option and click **Next**.

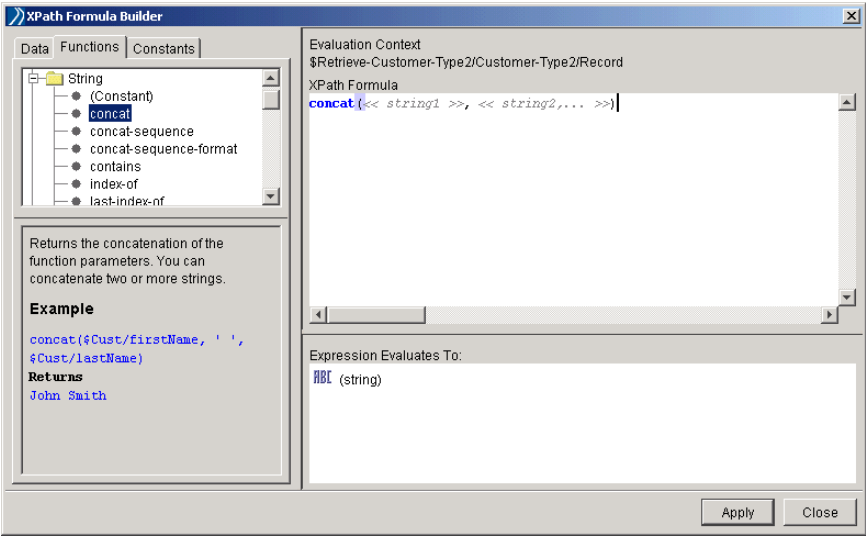


The mapping wizard presents you with an option to automatically map elements with the same name. Click **Finish** to accept the default mappings.

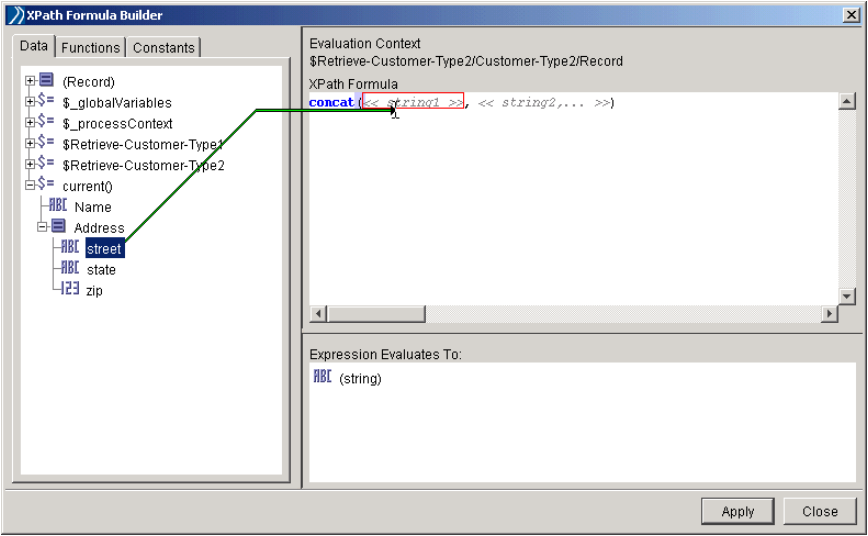


4. Select the `Address` element and click the XPath Formula Builder icon in the Input tab toolbar. In the XPath Formula Builder, drag a `concat()` function into the XPath Formula field. This function is used to concatenate the three

elements in the `Record` element in the Scope Variables area to one `Address` element in the function's input.

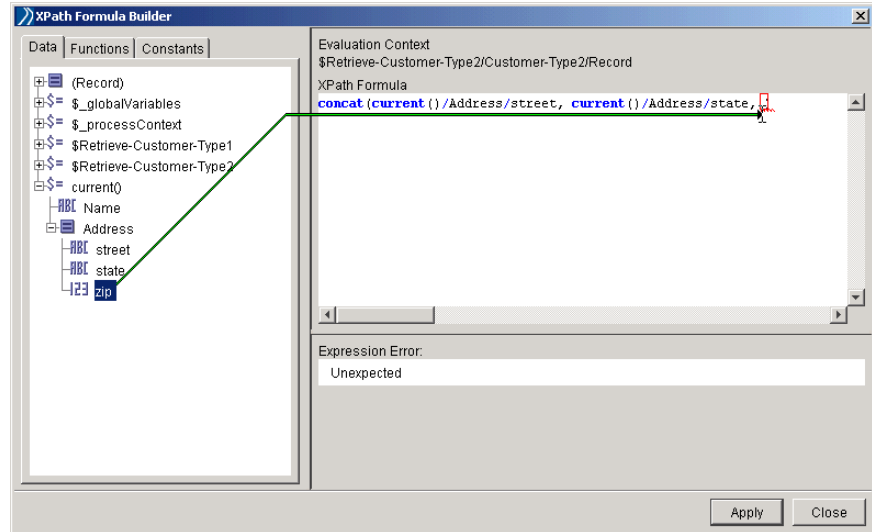


Click the Data tab, then drag the `$current() /Address/street` element into the `<< string1 >>` placeholder in the `concat()` function.

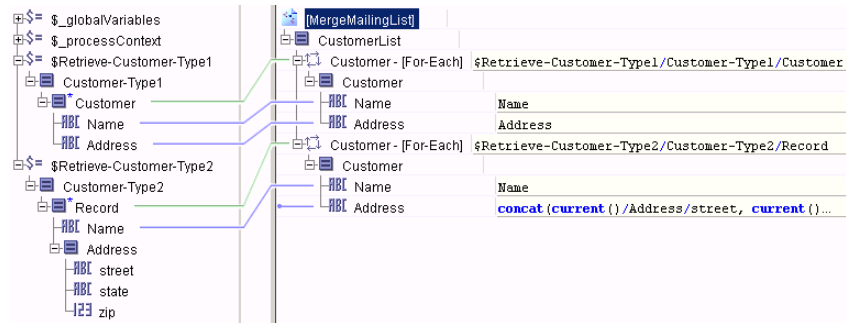


Drag the `$current() /Address/state` element into the `<< string2 >>` placeholder in the `concat()` function. Then, add a comma to the end of the function to include a third string to concatenate. Drag the

`$current() /Address/zip` element into the position of the third string in the `concat()` function.



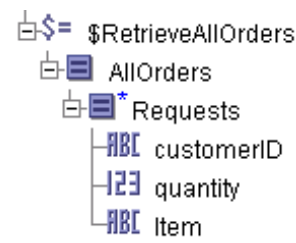
5. Click **Apply**, then click **Close** to dismiss the XPath Formula Builder dialog.
6. This results in the following mapping:



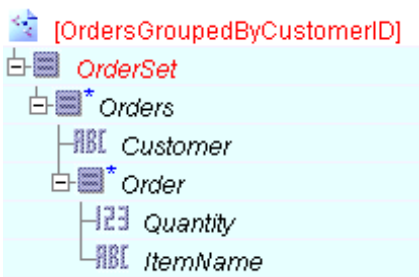
Converting a List Into a Grouped List

You may need to convert a flat list of items into a more structured list. For example, you may have a list of all orders that have been completed. You may want to organize that list so that you can group the orders placed by each customer.

In this example, the schemas are the following:



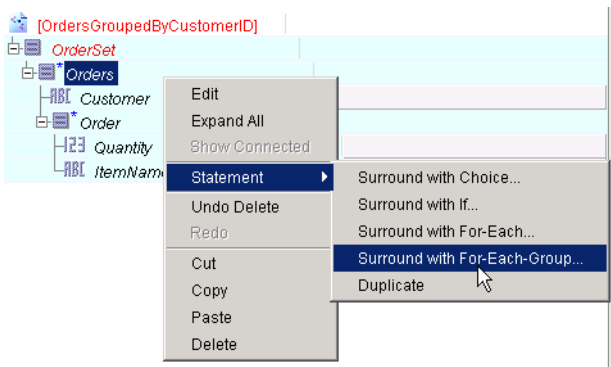
Flat list of orders placed by all customers. Each item in the repeating element Requests lists an order. The same customer can have multiple orders in this list.



The resulting schema is a repeating list of Orders, each item in the list is a customer ID. Each customer ID will, in turn, contain a list of orders.

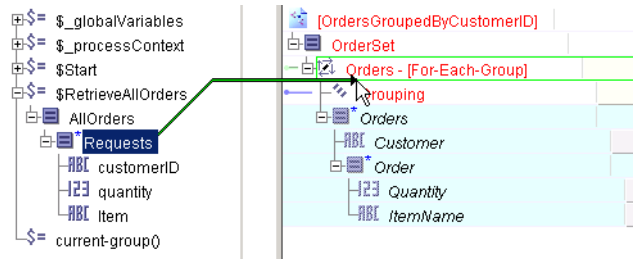
The following procedure describes how to map the flat list of orders into a list grouped by customer ID.

1. Choose the repeating element in the function input schema that holds the grouped data. In this example, that element is `Orders`. Right-click on this element and choose **Statement > Surround with For-Each-Group...** from the pop-up menu. This is a shortcut to create a For-Each-Group statement with the `Orders` element as a child element and a Grouping statement to contain the element you wish to group-by.

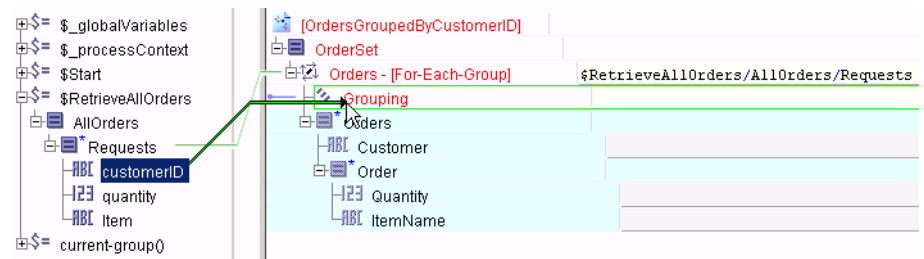


Adding the Grouping statement creates the `$=current-group()` element in the Scope Variables area. The Grouping statement creates the list grouped by the desired element, and the `current-group()` function allows you to access the items in the `Requests` repeating element that correspond to the group that is currently being processed.

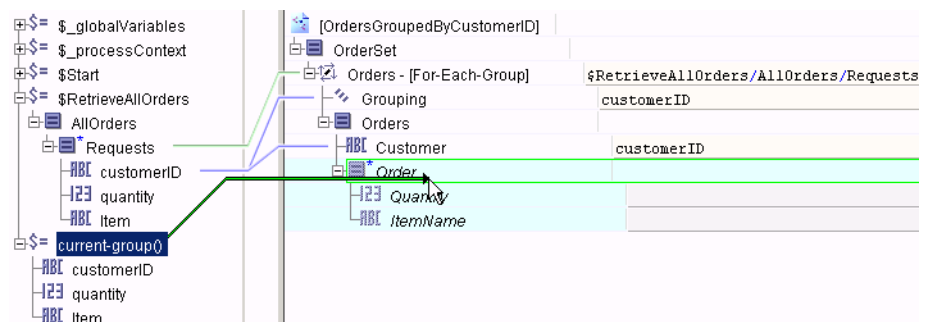
2. Drag the repeating element from the Scope Variables area to the For-Each-Group statement.



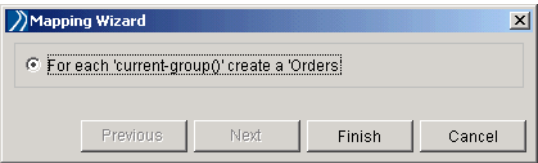
3. Drag the element you wish to group by from the Scope Variables area to the Grouping statement in the Function panel. In this example, `customerID` is the grouping element.



4. Map the `current-group()` element in the Scope Variables area to the repeating element `Order` under the `Customer` element in the Function panel.

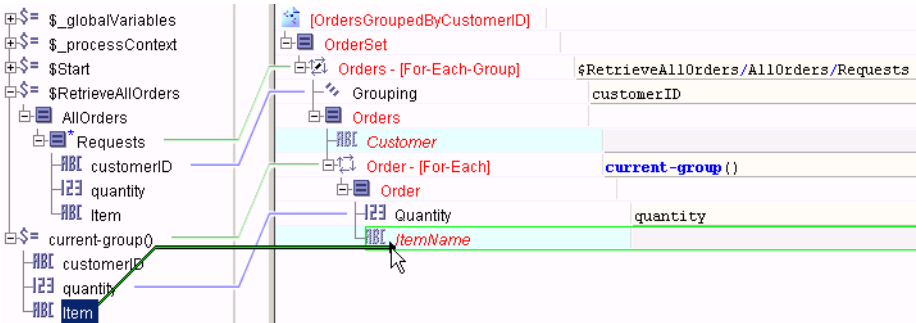


The default choice in the mapping wizard for this mapping is to create a For-Each. Choose this option in the mapping wizard.

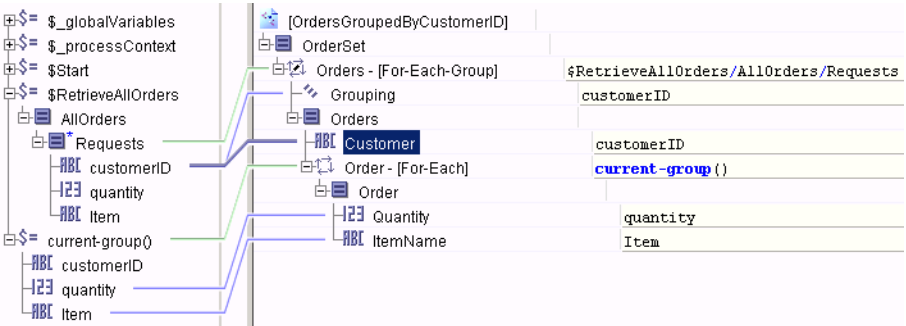


This creates an item in the `Order` list for each item in the current customer ID group that is being processed. The mapping wizard asks if you wish to map items with the same name in the current group and the orders group.

- 5. Map the remaining element from the `current-group()` element into the desired element in the For-Each group. In this case, `quantity` would map to `Quantity` automatically, and `Item` must be mapped to `ItemName`.

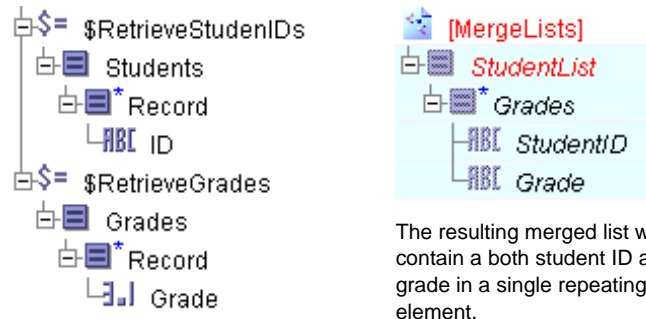


- 6. Map the `customerID` element in the `Requests` element into the `Customer` element in the Function panel.



Merging Two Corresponding Lists

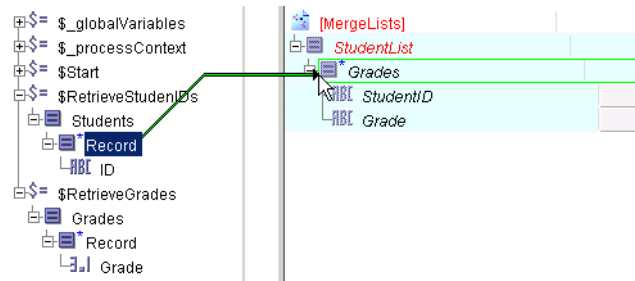
You may need to merge two lists that have corresponding items. For example, you may have a list of student IDs and a list of grades, each grade corresponds to the student ID in the same position in the student ID list. In this example, the schemas are the following:



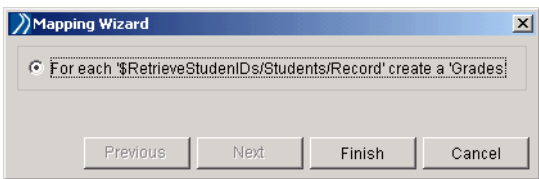
Two repeating lists contain corresponding student IDs and grades.

The following procedure describes how to merge the two repeating elements containing corresponding data into one repeating element.

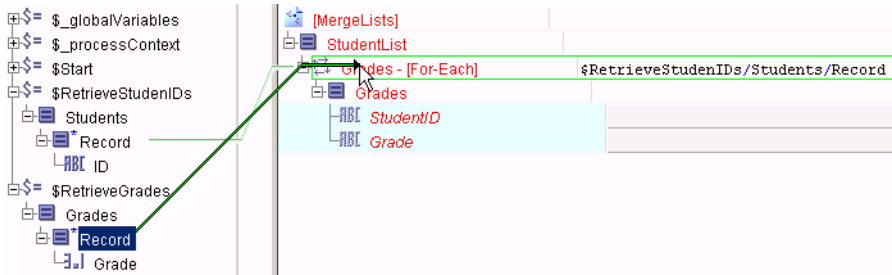
1. Map the first repeating element from the Scope Variables area into the Grades repeating element in the Function panel. In this example, the `$RetrieveStudentIDs/Students/Record` is the first repeating element.



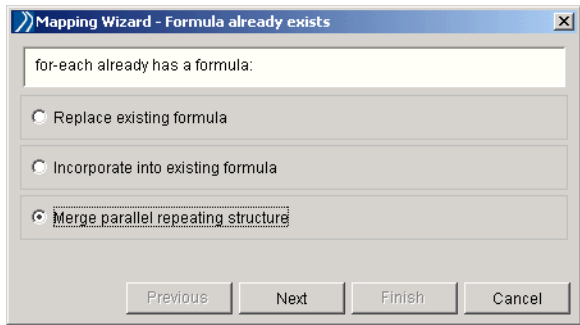
This brings up the mapping wizard with the default choice of creating a For-Each statement. Click **Finish** in the Mapping Wizard dialog to create the For-Each statement.



2. Drag the second repeating element into the For-Each statement.

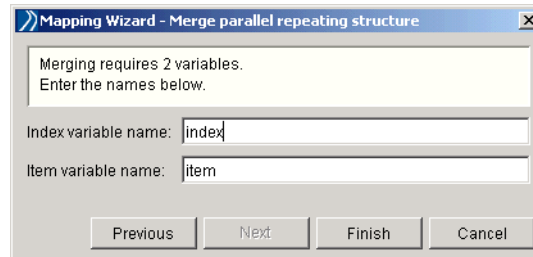


3. The Mapping Wizard dialog appears asking you to chose an option. Choose the Merge parallel repeating structure option and click **Next**.



4. Merging two parallel repeating structures requires two variables. The mapping wizard prompts you to name these two variables. One variable is to hold the position number of the current item being processed, and the other variable is to hold the item in the second list that corresponds to the position of the item in the first list. Create the variables with the default names

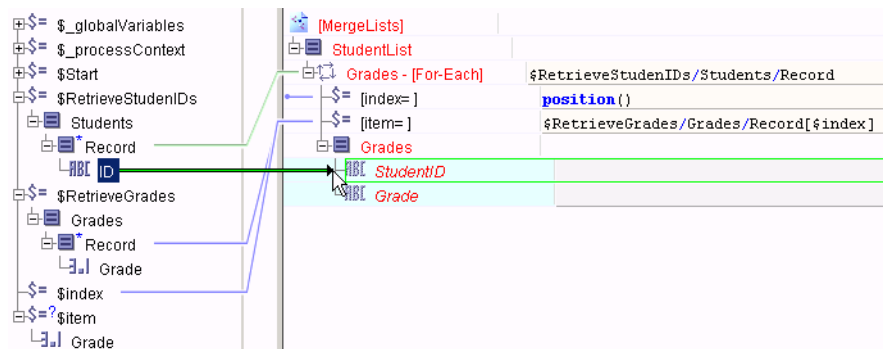
supplied by the mapping wizard, or choose your own names for these variables. Click **Finish** to proceed.



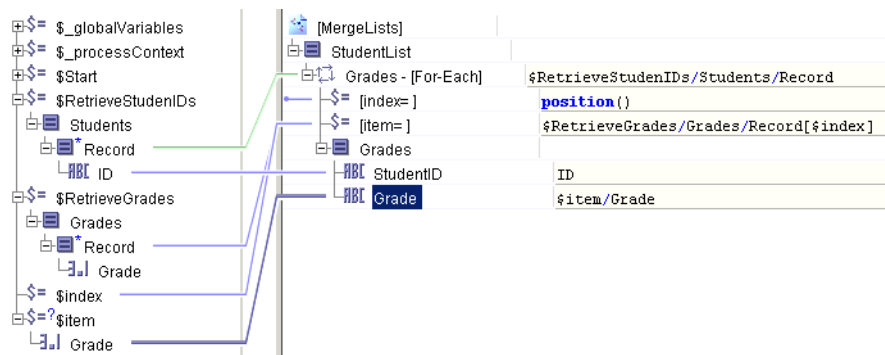
The two variables appear in the Scope Variables area once you have completed this step. The two variables also appear in the Function panel with the correct XPath statement to produce the desired result.

The `$=[index=]` element contains the XPath formula `position()` to set the element with the current position number of the list item being processed. The `$=[item=]` element contains a statement to retrieve the item in the second repeating element that corresponds to the position of the item in the first list that is currently being processed.

5. Map the `ID` element to the `StudentID` element in the function arguments.



6. Map the `$.item/Grade` element to the `Grade` element in the Function panel.

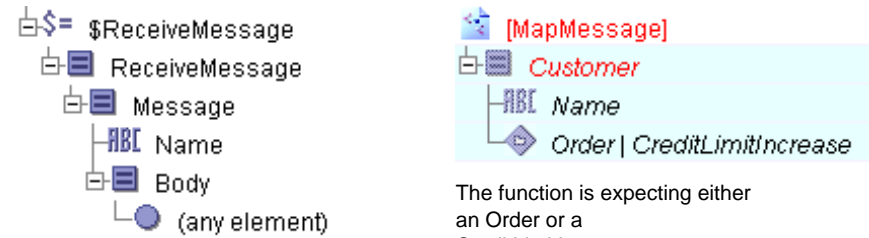


Coercions

In some situations, the datatype of a Scope Variables element may be undefined. In these situations, you may know the datatype of the element, and you can coerce the element into a specific type. The Coercions button in the Input tab toolbar allows you to create and manage your coercions.

The following example illustrates a schema with an element defined as the "any element" datatype. The schema is for a generic incoming message that can have any type of body. In the example, however, the any element is coerced into an Order type so that it can be mapped to a choice element.

In this example, the schemas are the following:



The incoming message can have a body of type any element.

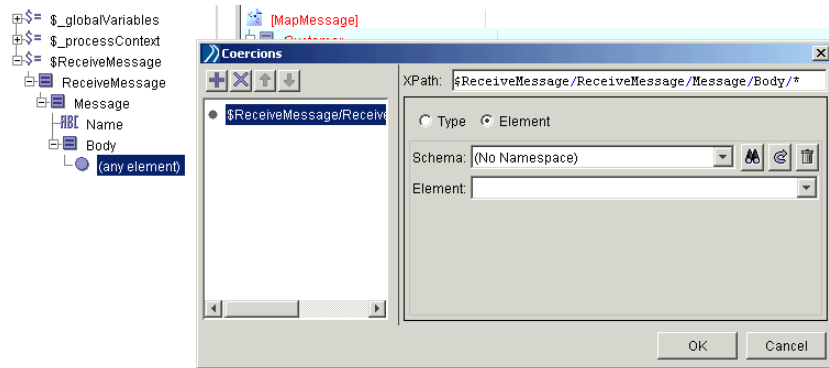
The function is expecting either an Order or a CreditLimitIncrease.

The following procedure describes how to coerce the `Body` element of the incoming message into a specific datatype and map it to a choice element.



There are many ways of accomplishing the same result as this example. This example attempts to illustrate the simplest method to achieve the desired result.

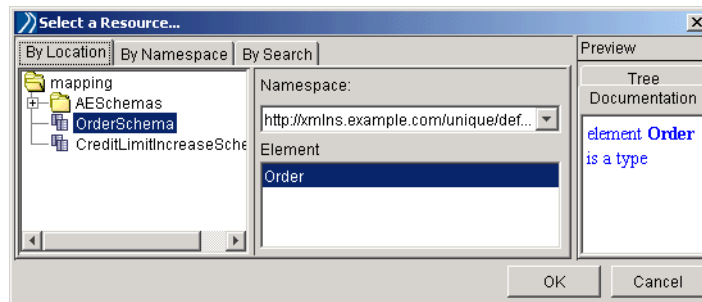
1. Select the element of type any element in the Scope Variables schema. Click the Coercions button in the Input tab toolbar. In the Coercions dialog, click the Insert button (+) to add a coercion for the currently selected element.



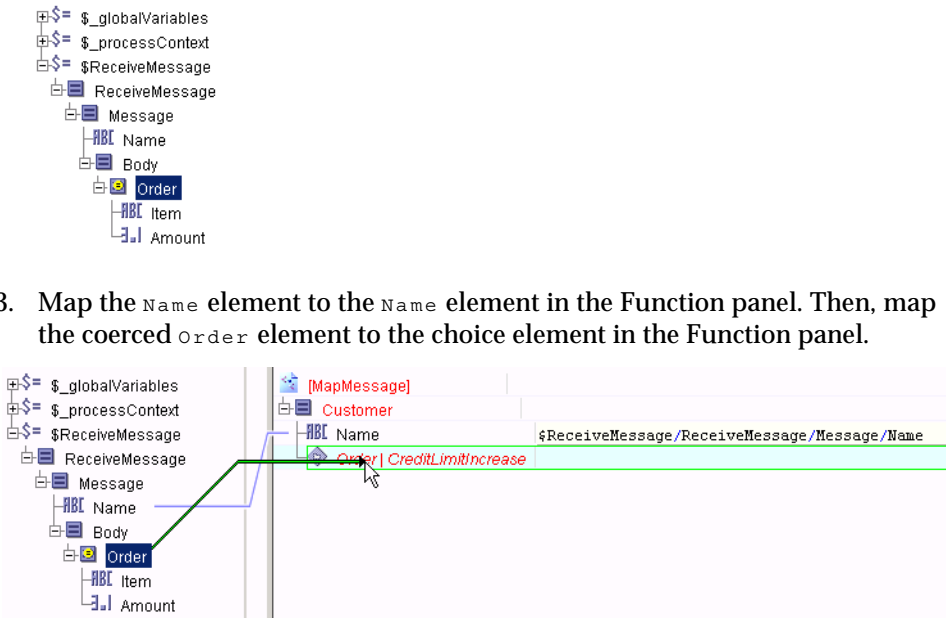
The Coercions dialog allows you to manage all of your coercions for a function in one dialog. You can create, modify, or delete coercions for any element in the Scope Variables schema using this dialog, not just the currently selected element. If you are creating a coercion for an element that is not currently selected, use the XPath field to specify the location of the element.

Click the Element radio button to specify that you are specifying a schema element.

2. Click the Browse Resources button next to the Schema field to browse a list of schemas that can be used. In the Select a Resource... dialog, select the schema that you would like to specify

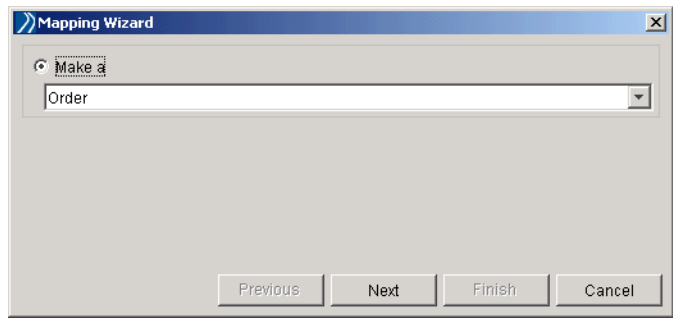


Click OK to coerce the element into the datatype of the selected schema element. The following would be the resulting schema where the element of the datatype any element has been replaced with the `Order` schema.



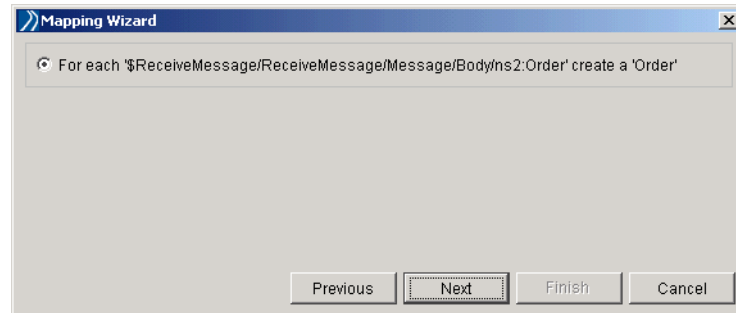
3. Map the `Name` element to the `Name` element in the Function panel. Then, map the coerced `Order` element to the choice element in the Function panel.

The Mapping Wizard dialog appears and asks if you wish to create an `Order` or a `CreditLimitIncrease` element. Select `Order` and click **Next**.

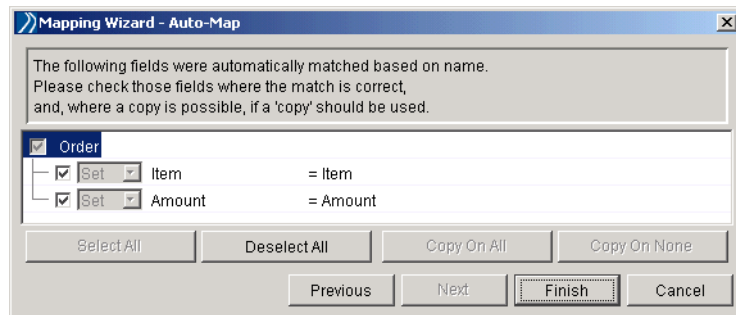


The Mapping Wizard then asks you to create a For Each. Even though there is only one element in the Scope Variables schema (the `Message` element is not

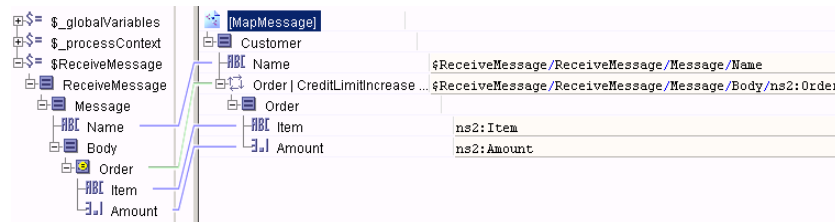
repeating), a For Each is used because this construct allows you to map the individual items of the Order element. Click **Next** to continue.



The Mapping Wizard then asks if you wish to automatically map elements with the same name. Click **Finish** to accept the default mappings.



4. The following is the completed mapping.



XSLT Statements

The following sections describe the XSLT statements you can add to your mapping. You can add or edit these statements by clicking the Edit Statement (🔧) button or these statements can be added automatically by selecting them from the dialogs that appear when you drag and drop elements from the Scope Variables tree to the function argument tree.

The following sections discuss statement types (available in the Statement Type drop-down list in the Edit Statement dialog).

Attribute

Allows you to specify an attribute, and optionally the namespace for the attribute. You can also specify the type of value for the attribute.

XSLT Equivalent

The following is an attribute named "lastname".

```
<ns:attribute namespace="mns" name="lastName"/>
```

When attributes are created, you can optionally specify the kind of value the attribute will have and whether the attribute should be surrounded by an if statement. For example, you can specify the value of the last name attribute to be a constant, like so:

```
<ns:attribute namespace="mns" name="lastName"/>
  "Smith"
</ns:attribute>
```

Choose

Provides a way to select transformation to perform based on an expression. Specify the condition in the `when` element as an XPath expression. You can optionally specify an `otherwise` condition for processing all elements that do not meet any of the specified `when` conditions.

XSLT Equivalent

The following determines if the node set for FilesTransferred contains any files, and if so, performs an action. If the node set is empty (no files were transferred), a different action is performed.

```
<ns0:choose xmlns:ns0="http://www.w3.org/1999/XSL/Transform">
  <ns0:when test="$FTP-Put/FTPputOutputFile/FileTransferred" >
```

```

        < something here ... >
    </ns:when>
    <ns:otherwise>
        < something here ...>
    </ns:otherwise>
</ns:choose>

```

Comment

Places a comment in the XSLT template. Comments are delimited by `<!--` and `-->`.

XSLT Equivalent

```
<!-- comment here -->
```

Copy

Copies the selected node to the current node in the input tree. Only the node is copied, no children of the node are copied.

XSLT Equivalent

```

<ns:copy xmlns:ns0="http://www.w3.org/1999/XSL/Transform"
select="$Query/resultSet"/>

```

Copy-Contents-Of

Copies the selected node's contents. This is useful if you wish to copy an element to a new element with a different name.

XSLT Equivalent

```

<ns:element namespace="foo" name="bar">
    <ns:copy-of select="null/@*" />
    <ns:copy-of select="null/node()" />
</ns:element>

```

Copy-Of

Creates a copy of the selected node, including the node's children. Both the copied node and the destination node must have the same name and structure.

XLST Equivalent

```
<ns0:copy-of xmlns:ns0="http://www.w3.org/1999/XSL/Transform"
select=""/>
```

Element

Creates an element with the specified name.

XSLT Equivalent

```
<elementName>value</elementName>
```

For-Each

Performs the specified statements once for each item in the selected node. This is useful if you wish to process each item of a repeating element once.

XSLT Equivalent

The following iterates over the list of files transferred from a ActiveMatrix BusinessWorks FTP Put activity and outputs an element with the name of each file for each file transferred.

```
<ns:for-each select="$FTP-Put/FTPputOutputFile/FileTransferred">
  <fileName>
    <ns:value-of
select="$FTP-Put/FTPputOutputFile/FileTransferred/Name"/>
  </fileName>
</ns:for-each>
```

For-Each-Group

Groups the items in a list by a specified element. This statement requires a Grouping statement to specify which element to group-by. See [Converting a List Into a Grouped List on page 319](#) for an example of using the For-Each-Group statement.

XSLT Equivalent

```
<ns0:for-each-group
xmlns:ns0="http://www.w3.org/1999/XSL/Transform" select=""/>
```

Generate Comment

Places a comment element into the XSLT template. This comment will be generated into the function's output.

Comment elements have the following syntax:

```
<ns0:comment xmlns:ns0="http://www.w3.org/1999/XSL/Transform"/>
```

Generate PI

Places a processing instruction into the XSLT template.

XSLT Equivalent

```
<ns0:processing-instruction
xmlns:ns0="http://www.w3.org/1999/XSL/Transform" name="/">
```

If

An if statement is used to surround other statements in an XSLT template to perform conditional processing. If the test attribute evaluates to true, the statements in the if are output, otherwise they are not output.

XSLT Equivalent

The following if statement surrounds an attribute for processing order items.

```
<ns:if xmlns:ns="http://www.w3.org/1999/XSL/Transform"
test="not(position()=last())">
  <ns:attribute name="OrderItem">
    <ns:value-of select=
      "$GetOrderInformation/OrderInformation/OrderDetails/OrderItem"/>
    </ns:attribute>
  </ns:if>
```

Value-Of

Specifies a value-of statement. This is normally done implicitly by specifying the formula for an element (field) in the mapping, but you may insert this statement explicitly.

XSLT Equivalent

```
<ns:value-of xmlns:ns="http://www.w3.org/1999/XSL/Transform"
select="/">
```

Variable

Adds a local variable for use in the current mapping. You can specify the name of the variable and whether you wish the variable to have a select attribute.

When you add a local variable, it appears in the Function and Scope Variables panels. You can supply any XPath expression to the new variable in the Function panel (either through mapping or through the XPath Formula Builder).

Once the variable's contents have been supplied, the variable (in the Scope Variables area) can be mapped to any item.

Adding a variable is useful when you wish to join two repeating elements into a single list, then map the combined list to an item. Adding a variable is also useful if you perform the same computation repeatedly. You can map the results of the computation to several items instead of recreating the computation for each item.

Variables can also improve performance of mappings for large data structures. For example, if you have a process variable with 40 sub-elements, and you map each of the sub-elements to a corresponding input item, TIBCO BusinessEvents must retrieve the current process variable for each XPath expression, in this case 40 times. If this mapping appears in a loop, the retrieval of the current process variable occurs 40 times per iteration of the loop. With a variable, the data is retrieved only once and used for all mappings containing the variable. Therefore, to improve performance, create a local variable to hold process variables with a large number of elements and use the local variable in XPath expressions instead of the process variable.

XSLT Equivalent

```
<ns0:variable xmlns:ns0="http://www.w3.org/1999/XSL/Transform"
name="var" select="$RetrieveResults/resultSet"/>
```

Chapter 20 XPath Formula Builder

TIBCO BusinessEvents uses XPath in the XPath Formula Builder, available in the Function Argument Mapper tool (see [Chapter 19, Mapping and Transforming Data, on page 297](#)). You can use XPath, for example, when defining payloads for events TIBCO BusinessEvents also uses XPath as the language for defining conditions and transformations. This section covers the basics of XPath and its use in TIBCO BusinessEvents.

Topics

- [XPath Basics, page 336](#)
- [The XPath Formula Builder, page 339](#)
- [String Representations of Datatypes, page 342](#)
- [Date and Time Functions, page 343](#)

XPath Basics

XPath (XML Path Language) is an expression language developed by the World Wide Web Consortium (W3C) for addressing parts of XML documents. XPath also has basic manipulation functions for strings, numbers, and Boolean values.

To use XPath in TIBCO BusinessEvents, you need only be familiar with the basic XPath concepts, but you may wish to learn more about XPath when building complex expressions.

For a complete description of XPath, refer to the XPath specification (which can be obtained from www.w3.org).

TIBCO BusinessEvents uses XPath (XML Path Language) to identify elements whose content may be used, for, example in an event payload. You can also use XPath to perform basic manipulation and comparison of strings, numbers, and Boolean values.

Addressing Schema Elements

All Scope Variables and Function arguments are represented as an XML schema. Regardless of where the data comes from or its format, TIBCO BusinessEvents represents the data as a schema tree. The data can be simple (strings, numbers, Boolean values, and so on), or it can be a complex element. Complex elements are structures that contain other schema elements, either simple elements or other complex elements. Both simple and complex elements can also repeat. That is, they can be lists that store more than one element of the given type.

XPath is used to specify which schema element you would like to refer to. Each schema has its own associated structure, for example, a set of simple values or simple data and other complex data.

To reference a particular data item in a schema, you start with the root node and then use slashes (/) to indicate a path to the desired data element. For example, if you wish to specify the `Street` attribute in the `ShipName` complex element that is in the `GetOrderInformation` node, you would use the following syntax:

```
$GetOrderInformation/ShipName/Street
```

The path starts with a dollar sign to indicate it begins with a root node, then continues with node names using slashes, like a file or directory structure, until the desired location is named.

Evaluation Context

XPath also has a method for referencing relative paths from a particular node. If you have an *evaluation context*, or a particular starting node in a schema tree, you can specify the relative path to other elements in the tree.

For example, if your evaluation context is `$GetOrderInformation/ShipName`, then you can reference the sub-items of `ShipName` without specifying the entire path. If you wish to reference `$GetOrderInformation/RequiredDate`, the relative path would be `../RequiredDate`. The path is relative to the evaluation context — `RequiredDate` is one level higher in the schema tree than the elements of `ShipName`.

Namespaces

Some schema elements must be prefixed with their namespace. The namespace is automatically added to elements that require this when creating mappings or when dragging and dropping data in the XPath formula builder.

Search Predicates

An XPath expression can have a search predicate. The search predicate is used to locate a specific element of a repeating schema item. For example, a `$GetOrderInformation/OrderDetails/OrderItem` item is a repeating element. If you wish to select only the first item in the repeating element, you would specify the following:

```
$GetOrderInformation/OrderDetails/OrderItem[1]
```

The `[1]` specifies the first element of a repeating item.

Sub-items can also be examined and used in a search predicate. For example, to select the element whose `ProductId` is equal to "3A54", you would specify the following:

```
$GetOrderInformation/OrderDetails/OrderItem[ProductId="3A54"]
```

You can also use functions and expressions in the search predicate. For example, if you wish to find all elements after the first, you would specify the following:

```
$GetOrderInformation/OrderDetails/OrderItem[position()>1]
```

See the Functions tab of the XPath Formula Builder for a list of available functions available and online documentation.

Testing for Nil

Some elements can be explicitly set to nil. You can test an element to determine if it is set to nil or not. For example, the following XPath expression returns true if the `$Order/Item/OnSale` element is set to nil:

```
$Order/Item/OnSale/@xsi:nil="true"
```

Comments


You can add comments to XPath expressions using the XPath 2.0 syntax for comments. The syntax is:

```
{ -- <comment here> -- }
```

For example, the following XPath expression contains a comment:

```
$GetOrderInformation/ShipName/Street {-- returns the street --}
```

The XPath Formula Builder

You access the XPath Formula Builder using a button in the Function Argument Mapping Wizard. First select an item in the Function Argument panel (in the Input section). Then click the XPath Formula Builder button ().

The XPath formula builder allows you to drag and drop schema elements and XPath functions to create XPath expressions. The schema elements, when dragged into the XPath Formula field, automatically become valid XPath location paths for the desired item. If a function is dragged into the XPath formula window, there are placeholders for each parameter of the function. You can drag and drop schema elements over the parameter placeholders to replace each placeholder.

Table 38 describes the different areas of the XPath formula builder.

Table 38 XPath Formula Builder Reference

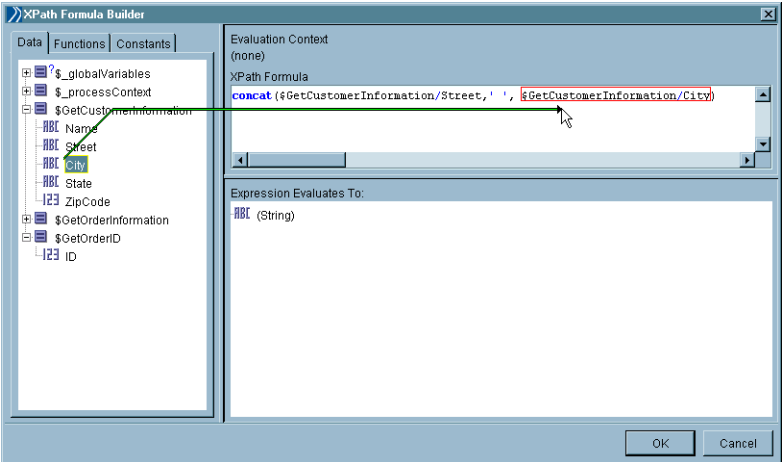
Element	Description
Data tab	Displays the Scope Variables schema tree. All elements in this tree are available to drag and drop into the XPath Formula field.
Functions tab	<p>Displays the available XPath functions. These are categorized into groups and each function can be dragged from the function list into the XPath Formula field.</p> <p>When the function is placed into the XPath formula, placeholders are displayed for the function's parameters. You can drag and drop schema elements from the Data tab into the function's placeholders.</p> <p>The result of evaluating the function is displayed in the "Expression Evaluates To" panel. If there are any errors in the expression, they are listed there as well.</p> <p>For more information about XPath functions, see the description of the function that is displayed when it is selected in the XPath formula builder.</p>

Table 38 XPath Formula Builder Reference

Element	Description
Constants tab	<p>Displays the constants available for use in XPath expressions. These are categorized into groups and each constant can be dragged from the constants list into the XPath Formula field.</p> <p>Constants are useful for inserting special characters, such as tabs, symbols, and so on, into XPath formulas. Constants are also defined for commonly used items, such as date formats.</p>
Documentation panel	<p>Appears below the Functions and Constants tabs. Describes each selected function. As you click on a function in the Function tab, the documentation panel gives a brief description of the function and one or more examples. Similarly documentation for constants in the Constants tab appears.</p>
Evaluation Context field	<p>Displays the evaluation context of the expression field that the editor was invoked from. See Evaluation Context on page 337 for more information about the evaluation context.</p>
XPath Formula field	<p>Displays the XPath formula you wish to create. You can drag and drop items from the Data tab or the Functions tab to create the formula.</p>
Expression Evaluates To Panel	<p>Displays the result of evaluating the formula shown in the XPath Formula field. If there are errors in the formula, they are displayed here.</p>

[Figure 6](#) illustrates using the XPath formula builder to create a valid function. The function concatenates the data elements `$GetCustomerInformation/Street` and `$GetCustomerInformation/City` and places a space between the two elements.

Figure 6 Creating an XPath formula



String Representations of Datatypes

When data must be represented in the input or output of an activity, the data is represented as a string. This section explains the string representations of various datatypes. TIBCO BusinessEvents follows the XPath 1.0 standard for representing all numeric datatypes. TIBCO BusinessEvents follows the XML Schema canonical format for all other datatypes.

Numeric Datatypes

Numeric datatypes include all types derived from `xs:integer`, `xs:decimal`, `xs:float`, and `xs:double`.

All decimal, float, and double numbers are compressed to an integer when represented, if there are only zeros following the decimal point (for example, "1.000" is represented as 1). Scientific notation is never used to represent a floating point number as a string (for example, "`xs:double('1.234E05')`" is represented as 123400). Data is truncated if the number of digits exceeds the maximum precision for the datatype (for example, "`xs:float('1.23456789')`" is represented as 1.2345679).

Both zero and negative zero are represented as 0. Positive and negative infinity are represented as `Infinity` and `-Infinity`. Not a number is represented as `NaN`.

Boolean

The boolean datatype is used to indicate a true or false state.

`xs:boolean('true')` and `xs:boolean('1')` are represented by true. The XPath function `true()` is also represented as `true`.

`xs:boolean('false')` and `xs:boolean('0')` are represented by false. The XPath function `false()` is also represented as `false`.

Date Datatypes

TIBCO BusinessEvents Function Argument Wizard (also known as the function argument mapper) implements dates in one of two ways. Either a date is stored as the number of milliseconds since January 1, 1970, or the date is implemented according to the XPath 2.0 or XQuery 1.0 standards as a set of normalized components (`xs:date`, `xs:time`, `xs:dateTime`, and so on) with an optional time zone offset. Activities that are associated with Java (for example, Java Code, Java Method, and so on) use the first implementation. Activities that are associated with XML (for example, Mapper, Parse XML, and so on) use the second implementation. The second implementation supports arbitrary precision of the seconds component.

Conversion between these representations may result in a loss of information either because of the difference in time zone representation or the precision of the seconds.

Date and Time Functions

There are some functions in the XPath formula builder that allow you to parse or format strings that represent dates and times. These functions are:

- `format-dateTime (format, dateTime)`
- `format-date (format, date)`
- `format-time (format, time)`
- `parse-dateTime (format, string)`
- `parse-date (format, string)`
- `parse-time (format, string)`

The *format* parameter of these functions is based on the format patterns available for the `java.text.SimpleDateFormat` Java class. In the format parameter, unquoted alphabetic characters from `A` to `Z` and `a` to `z` represent the components of the date or time string. You can include non-pattern alphabetic characters in the string by quoting the text with single quotes. To include a single quote, use `' '`. [Table 39](#) describes the alphabetic characters and their associated presentation in a date or time string.

Table 39 Formatting characters in date or time strings

Letter	Description	Example
G	Era Four or more <code>G</code> s return the full name of the era.	AD
Y	year Two <code>Y</code> s return two-digit year.	2003; 03
M	Month in year Three or more <code>M</code> s return text name.	August; Aug; 08
w	Week in year	48
W	Week in month	3
D	Day in year	254
d	Day in month	28

Table 39 *Formatting characters in date or time strings*

Letter	Description	Example
F	Day of week in month	3
E	Day in week Four or more <code>Es</code> return the full name of the weekday.	Friday; Fri
a	AM/PM marker Four or more <code>as</code> return the full name.	AM
H	Hour in day (0-23)	23
k	Hour in day (1-24)	1
K	Hour in AM/PM (0-11)	11
h	Hour in AM/PM (1-12)	1
m	Minute in hour	59
s	Second in minute	48
S	Millisecond	456
z	Time zone represented as a GMT offset.	GMT-08:00
Z	RFC 822 four-digit time zone format	-0800
all other letters	Reserved	—

For any format pattern letter that returns a numeric value (for example, `w`, `h`, and `m`), the number of letters in the format pattern represents the minimum number of digits. For formatting functions, if the date or time has fewer digits than the number of pattern letters, the output is padded with zeros. For parsing functions, when the date or time has fewer digits than the number of characters in the format pattern, the extra characters are ignored, unless they are needed to determine the boundaries of adjacent fields.

Table 40 illustrates some example date and time format patterns and the resulting string.

Table 40 Example date and time format patterns

Date/Time Pattern	Result
"yyy.MM.dd G 'at' HH:mm:ss"	2003.3.11 AD at 09:43:56
"EEE, MMM d, ''yy"	Tue, Mar 11, '03
"hh 'o''clock' a, zzzz"	9 o'clock AM, GMT-8:00
"K:mm a"	0:08 PM
"yyMMddHHmmssZ"	010704120856-700

Chapter 21 **ActiveMatrix BusinessWorks Integration**

This chapter explains how to use the TIBCO ActiveMatrix BusinessWorks plug-in, to integrate ActiveMatrix BusinessWorks and TIBCO BusinessEvents applications.

Topics

- [Overview of Integration with ActiveMatrix BusinessWorks, page 348](#)
- [Integration Components, page 350](#)
- [Design Considerations, page 352](#)
- [Configuring the Environment for ActiveMatrix BusinessWorks Containers, page 355](#)
- [Configuring the Environment For TIBCO BusinessEvents Containers, page 357](#)
- [Configuring a RuleServiceProvider Configuration Resource, page 362](#)
- [TIBCO BusinessEvents RuleServiceProvider Configuration Resource Reference, page 363](#)
- [Working With the TIBCO BusinessEvents Activities, page 365](#)
- [Receive Event Resource Reference, page 366](#)
- [Send Event Resource Reference, page 368](#)
- [Wait for Event Resource Reference, page 369](#)
- [Invoking a TIBCO BusinessEvents Rule Function from a Process, page 372](#)
- [Working With Invoke RuleFunction Activities, page 374](#)
- [Invoke RuleFunction Resource Reference, page 376](#)
- [Working with the BusinessWorks Functions, page 378](#)

Overview of Integration with ActiveMatrix BusinessWorks

If TIBCO ActiveMatrix BusinessWorks used in your environment, you can take advantage of integration components provided by TIBCO BusinessEvents. Integration between TIBCO BusinessEvents and ActiveMatrix BusinessWorks enables each product to take advantage of the strengths of the other. For example, ActiveMatrix BusinessWorks can use TIBCO BusinessEvents as a light-weight rules engine, and TIBCO BusinessEvents can use transports available in ActiveMatrix BusinessWorks.

This guide assumes you know how to work with ActiveMatrix BusinessWorks processes. See *TIBCO BusinessWorks Process Design Guide* if you need detailed guidance.

Integration components include an ActiveMatrix BusinessWorks plug-in, which provides a palette of TIBCO BusinessEvents activities, and various TIBCO BusinessEvents functions. See [Integration Components on page 350](#) for details. You can use these components to do the following:

- More easily send and receive TIBCO BusinessEvents events using ActiveMatrix BusinessWorks activities.
- Invoke TIBCO BusinessEvents rule functions from ActiveMatrix BusinessWorks
- Call ActiveMatrix BusinessWorks processes from TIBCO BusinessEvents.

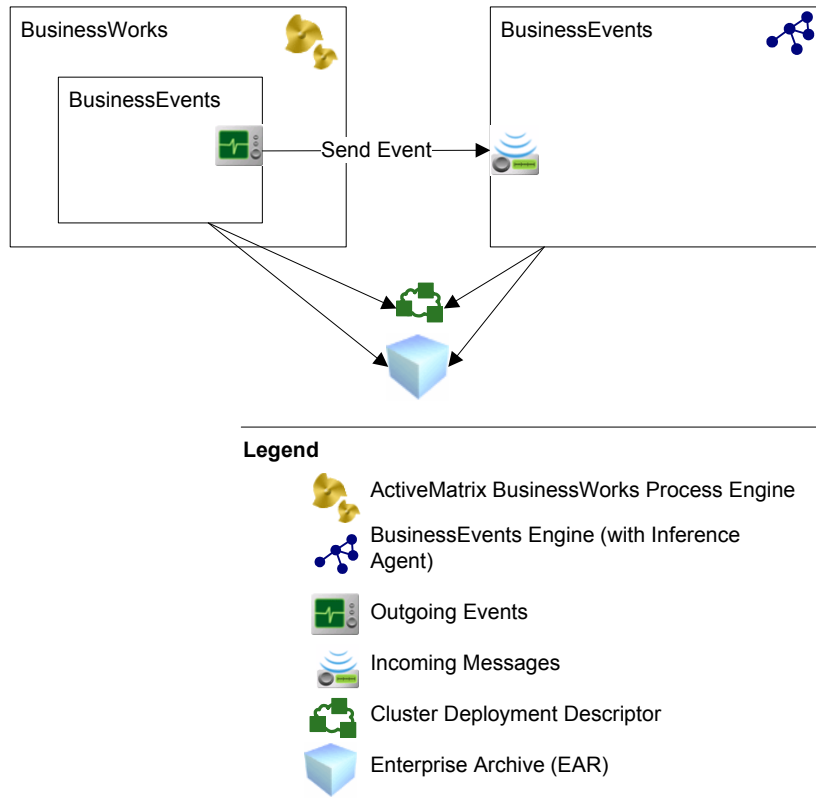
This chapter explains how to configure the environment and use these features.



See the product readme for the supported version of TIBCO ActiveMatrix BusinessWorks.

TIBCO BusinessEvents requires JRE 6. You must also use JRE 6 with ActiveMatrix BusinessWorks in order to integrate with TIBCO BusinessEvents.

The Container and the Contained Engine



You can design integration projects in different ways. You can start an ActiveMatrix BusinessWorks engine inside a TIBCO BusinessEvents container, or you can start a TIBCO BusinessEvents engine inside a ActiveMatrix BusinessWorks container — or you can do both.

In all cases, TIBCO BusinessEvents and ActiveMatrix BusinessWorks must be deployed on the same machine, and both must reference the same CDD file and EAR file (as shown in the figure above).

Which engine runs as the container affects project design and runtime behavior, as explained in [Design Considerations on page 352](#).

Integration Components

Integration components include a palette of activities in ActiveMatrix BusinessWorks, and a category of functions in TIBCO BusinessEvents.

ActiveMatrix BusinessWorks Activities Palette

RuleServiceProvider Configuration Resource

The RuleServiceProvider Configuration resource is used to identify the source of the ontology definitions (in an EAR file) and the location of the TIBCO BusinessEvents application at runtime.

Event Related Activities

The Receive Event, Send Event, And Wait for Event activities enable you to work more easily with TIBCO BusinessEvents events within ActiveMatrix BusinessWorks. The ontology definitions from the TIBCO BusinessEvents application are available to these activities so you can easily configure them. See the following sections:

- [Receive Event Resource Reference on page 366](#)
- [Send Event Resource Reference on page 368](#)
- [Wait for Event Resource Reference on page 369](#)

Invoke RuleFunction Activity

An Invoke RuleFunction activity invokes a TIBCO BusinessEvents rule function in a specified agent instance and passes information to it. The Invoke RuleFunction activity can be combined with any process starter. Execution is synchronous.

TIBCO BusinessEvents Functions

TIBCO BusinessEvents provides a BusinessWorks category of functions, as follows:

`BusinessWorks.invokeProcess()`: Invokes an ActiveMatrix BusinessWorks process in synchronous mode and waits for completion of the process before returning to the rule or rule function. Starts the process engine if it is not already started. Returns an event, or null. Generates an advisory event if it times out.

`BusinessWorks.startProcess()`: Invokes an ActiveMatrix BusinessWorks process in asynchronous mode and returns the process ID (job ID). Upon completion, the ActiveMatrix BusinessWorks process passes an event to the rule function specified in an argument of `startProcess()`. Starts the process engine if it is not already started.

`BusinessWorks.cancelProcess()`: Cancels the specified ActiveMatrix BusinessWorks process. Useful for canceling a long running job. (Cancellation is not guaranteed because the process may complete before receiving the cancellation command.)

`BusinessWorks.init()`: Used as needed to initialize the ActiveMatrix BusinessWorks engine.

`BusinessWorks.shutdown()`: Used as needed to stop the ActiveMatrix BusinessWorks engine.

Design Considerations

This section presents some considerations to keep in mind when you are designing your integration project.

Integration Scope

You can only use TIBCO BusinessEvents resources that are included in the TIBCO BusinessEvents EAR file referenced in the RuleServiceProvider Configuration resource.

TIBCO BusinessEvents and ActiveMatrix BusinessWorks must be installed on the same machine.

Thread Management

TIBCO BusinessEvents generates its own threads to execute the rule function that the ActiveMatrix BusinessWorks process calls through the Invoke RuleFunction activity. Then the ActiveMatrix BusinessWorks thread is released and the process is set to a pending state. When the rule function returns, the ActiveMatrix BusinessWorks process resumes its Ready state.

Avoiding Threading Issues

It is possible to configure a complex execution path in your project, for example, one in which TIBCO BusinessEvents invokes an ActiveMatrix BusinessWorks process, which in turn invokes a TIBCO BusinessEvents rule function, and so on.

The `BusinessWorks.InvokeProcess()` rule function and the Invoke RuleFunction activity both operate synchronously. Both attempt to acquire a lock on the TIBCO BusinessEvents RTC.

Check your execution path carefully to ensure that there are no threading issues leading to deadlock. In general, make sure that no action in the entire execution path attempts to use the same working memory that is already locked.

For example, if you execute `BusinessWorks.InvokeProcess()` function in TIBCO BusinessEvents, then in the ActiveMatrix BusinessWorks process it calls, you cannot use an Invoke RuleFunction activity to invoke a rule function in the same agent. You could invoke a rule function running in a different agent, however.

Design Considerations Related to Container

The engine used as the container is responsible for state and object management, fault tolerance, and logging. The contained engine delegates management of these features to the container. The main points to keep in mind are listed below.

Table 41 *Design Considerations Related to Container (Integration with ActiveMatrix BusinessWorks)*

Item	TIBCO BusinessEvents Containers	ActiveMatrix BusinessWorks Containers
State and OM	TIBCO BusinessEvents manages state. See also Fault Tolerance below.	ActiveMatrix BusinessWorks manages state. Use In Memory object management in TIBCO BusinessEvents. Other OM methods are not supported.
Engine TRA file	The ActiveMatrix BusinessWorks engine TRA file is not used. Provide any properties needed, such as classpath and palette path, in the TIBCO BusinessEvents engine TRA file (see Configuring the Environment for ActiveMatrix BusinessWorks Containers on page 355).	The TIBCO BusinessEvents engine TRA file is not used. Provide any properties needed for TIBCO BusinessEvents in the ActiveMatrix BusinessWorks property file (see Configuring the Environment for ActiveMatrix BusinessWorks Containers on page 355).
CDD file	Provide configuration values for TIBCO BusinessEvents.	The CDD file is also used. Provide configuration values for TIBCO BusinessEvents.
TIBCO Hawk microagent	With configuration of a property in the TIBCO BusinessEvents CDD file, you can use TIBCO Hawk methods for TIBCO BusinessEvents and ActiveMatrix BusinessWorks. See the procedure in the section Configuring the Environment for ActiveMatrix BusinessWorks Containers on page 355 for details.	Only the TIBCO Hawk methods for ActiveMatrix BusinessWorks are used.
Fault Tolerance	TIBCO BusinessEvents fault tolerance is used. See Fault Tolerance With a TIBCO BusinessEvents Container on page 354 .	ActiveMatrix BusinessWorks fault tolerance is used. Do not configure TIBCO BusinessEvents in fault tolerant mode.

Table 41 Design Considerations Related to Container (Integration with ActiveMatrix BusinessWorks) (Cont'd)

Item	TIBCO BusinessEvents Containers	ActiveMatrix BusinessWorks Containers
Logging	TIBCO BusinessEvents manages logging.	ActiveMatrix BusinessWorks manages logging

Fault Tolerance With a TIBCO BusinessEvents Container

When TIBCO BusinessEvents is the container, ActiveMatrix BusinessWorks checkpointing can be used in a limited way. Do not use ActiveMatrix BusinessWorks checkpointing in any process that is called by `startProcess()` or `invokeProcess()`, or in any process called by such processes.

When an active TIBCO BusinessEvents processing unit (PU) fails, standby agents in other processing units become active and take over for the agents that failed. The newly active agents (which are of the same agent class) start the ActiveMatrix BusinessWorks engine as needed.

On failback, agents of lower priority become standbys. During deactivation they stop the ActiveMatrix BusinessWorks engine they were running.

Tips for Working With ActiveMatrix BusinessWorks Containers

- Keep the following points in mind when designing your project.
- The TIBCO BusinessEvents hot deployment feature is not available.
 - The internal TIBCO BusinessEvents engine that ActiveMatrix BusinessWorks starts communicates only with the ActiveMatrix BusinessWorks process that invokes it. TIBCO BusinessEvents channels do not listen for incoming messages.
 - Do not execute any action in a TIBCO BusinessEvents startup rule function that results in use of `BusinessWorks.invokeProcess()` or `BusinessWorks.startProcess()` functions. The ActiveMatrix BusinessWorks engine may not be fully initialized when these functions are executed in a startup rule function (or in any rule in the RTC cycle of a startup rule function).

Configuring the Environment for ActiveMatrix BusinessWorks Containers

To use the integration features you must configure the environment as explained in this section.



- TIBCO BusinessEvents Studio and ActiveMatrix BusinessWorks must be installed on the same machine.
- If you are upgrading an integration project created in an earlier release, see Chapter 4, Migrating Projects from Earlier Versions in *TIBCO BusinessEvents Installation* for actions you must take.
- The values shown in this section work with various test projects. Depending on the ActiveMatrix BusinessWorks services used in your project, you may have to make additional changes.

Task A Configure the TRA Files

You must add various items to the ActiveMatrix BusinessWorks class path and palette path. The `designer.tra` file is used at design time. The `bwengine.tra` file is used at runtime. Update them as explained next.



Remember to add a path separator variable, `%PSP%` to separate the paths, when you paste path content into a TRA file.

If any properties have been modified after installation, examine the paths carefully. Ensure that all paths are specified with appropriate variables.

1. Open these files in a text editor:

- `BE_HOME/bin/be-engine.tra` (as the source of values)
- `TIBCO_DESIGNER_HOME/bin/designer.tra`
- `BW_HOME/bin/bwengine.tra`

2. Copy the `tibco.env.BE_HOME` environment variable from `be-engine.tra` to the `designer.tra` and `bwengine.tra` files.

(If you add this variable manually, set to the value of `BE_HOME`. Use forward slashes, for example: `tibco.env.BE_HOME c:/tibco/be/version.`)

3. Update the custom class path in `designer.tra` and `bwengine.tra`: locate the `tibco.env.CUSTOM_CP_EXT` property and append the following:

- A `%PSP%` separator
- The entire contents of `be-engine.tra` property `tibco.env.STD_EXT_CP`
- The following text:

```
%PSP%%BE_HOME%/hotfix/lib/palettes%PSP%%BE_HOME%/lib/palettes%PSP%%BE_HOME%/hotfix/
lib/ext/tpcl/jide%PSP%%BE_HOME%/lib/ext/tpcl/jide
```

4. Update the custom palette path in the `designer.tra` file: append the following text to the `tibco.env.CUSTOM_PALETTE_PATH` property:

```
%PSP%%BE_HOME%/hotfix/lib/palettes%PSP%%BE_HOME%/lib/palettes
```

5. Also append the following text to `tibco.env.CUSTOM_CP_EXT` in `bwengine.tra`:

```
%PSP%%BE_HOME%/hotfix/lib/palettes%PSP%%BE_HOME%/lib/palettes
```


Configuring the Environment For TIBCO BusinessEvents Containers

Using a TIBCO BusinessEvents container requires additional configuration so that TIBCO BusinessEvents can locate ActiveMatrix BusinessWorks resources at runtime.

First perform all steps in the section [Configure the TRA Files on page 355](#) and then all steps in this section.

Task A Add the Repo URL for BusinessWorks to the CDD

This task assumes all other cluster deployment descriptor (CDD) configuration has been done. For more information about editing the CDD file, see Chapter 3, CDD Configuration Procedures, in *TIBCO BusinessEvents Administration*.

1. Open the TIBCO BusinessEvents project in TIBCO BusinessEvents Studio.
2. Open the CDD Editor resource and select the Agent Classes tab.
3. For all inference agent classes that will run ActiveMatrix BusinessWorks, edit the BusinessWorks Repo URL field: add the path to the runtime TIBCO Designer project.
4. Save.

Use this CDD when deploying the TIBCO BusinessEvents project.

Example Repo URL Values	The repo URL format depends on the deployment transport used. Supported formats for the URL are <code>tibcr</code> , <code>http</code> , <code>https</code> , and <code>file</code> .
----------------------------	---

Rendezvous transport format:

```
tibco.bwrepourl=tibcr@domain name- deployment name:service=repo rvService:daemon=repo
rvDaemon:userName=uid:server=domain_name:password=encrypted_ password
```

Local transport (also known as file transport) format:

```
tibco.bwrepourl=domain_home>/ domain_name/datafiles/ deployment_name_root
```

HTTP transport format:

```
tibco.bwrepourl=http://machine_name:domain_http_port?domain_name-deployment-name&server=domain_name&timeout\=600&userName=uid&password\=encrypted_password
```

Task B Modify the TIBCO BusinessEvents TRA File

- be-engine.tra
1.

Open the *BE_HOME*/bin/be-engine.tra file for editing.

2.

Add the following properties and their values. Take values from the *BW_HOME*/bin/bwengine.tra file, if they are present there:

tibco.env.BW_HOME

tibco.env.BW_MIGRATION_APPEND_VERSION

tibco.env.BW_PLUGINS_HOME_OLD

tibco.env.BW_PLUGINS_HOME

tibco.env.TPCL_HOME

tibco.env.TRA_HOME

tibco.env.TRA_APPEND_VERSION

tibco.env.HAWK_HOME

java.property.palettePath

3.

Copy the following properties and their values from the bwengine.tra file to the be-engine.tra file and rename them: add BW_ to the property name as shown.

In bwengine.tra	In be-engine.tra
tibco.env.CUSTOM_EXT_PREPEND_CP	tibco.env.BW_CUSTOM_EXT_PREPEND_CP
tibco.env.CUSTOM_EXT_APPEND_CP	tibco.env.BW_CUSTOM_EXT_APPEND_CP
tibco.env.STD_EXT_CP	tibco.env.BW_STD_EXT_CP
tibco.env.PATH	tibco.env.BW_PATH
tibco.env.LD_LIBRARY_PATH	tibco.env.BW_LD_LIBRARY_PATH
tibco.env.SHLIB_PATH	tibco.env.BW_SHLIB_PATH
tibco.env.LIBPATH	tibco.env.BW_LIBPATH

4.

In the value of all properties listed in [step 3](#), remove references to any TIBCO BusinessEvents library, if there are any.
- The be-engine.tra file (as shipped) uses the above names in various path properties, so you do not have to make any additional modifications.
5.

Save the file.

Task C Configure for Non-Default TIBCO Hawk Microagent Names

When TIBCO BusinessEvents is the container, the ActiveMatrix BusinessWorks TIBCO Hawk microagent (HMA) name is the same as the TIBCO BusinessEvents HMA name, appended with `-bw`. The name is defined using the `Hawk.AMI.DisplayName` property (see [Table 42, ActiveMatrix BusinessWorks integration Properties for TIBCO BusinessEvents Containers, on page 360](#) for details).

If you want to use default HMA names, skip this task.

Choose a method of specifying non default names:

- If you want to use the same non-default names, except that the ActiveMatrix BusinessWorks HMA is appended with `-bw`: Define the non-default name by adding the `Hawk.AMI.DisplayName` property to the Processing Units tab property sheet in the Cluster Configuration editor.
- If you want to use different names for the ActiveMatrix BusinessWorks and TIBCO BusinessEvents HMAs, do the following: Define the name to be used by the ActiveMatrix BusinessWorks HMA using `Hawk.AMI.DisplayName` property. Whatever name you specify will be automatically appended with `-bw`.
- Add a property called `be.hawk.microagent.name` and define its value as the desired name of the TIBCO BusinessEvents HMA.

Task D Configure for ActiveMatrix BusinessWorks Checkpointing (if Used)

Used only when TIBCO BusinessEvents is the container. See [Design Considerations Related to Container on page 353](#) for advice on using checkpointing in this case.

By default, an ActiveMatrix BusinessWorks instance running inside TIBCO BusinessEvents has the same name as the TIBCO BusinessEvents instance. If you will use ActiveMatrix BusinessWorks checkpointing, and if TIBCO BusinessEvents engine instances are deployed with different names, you must add a property that specifies the same ActiveMatrix BusinessWorks instance name to all processing units. Add the property to all processing units, using the Cluster Configuration editor, Processing Units tab property sheet.

`tibco.bwengine.name` *all-same-engine-name*

You can specify the property and its value in the CDD, or in the TIBCO Administrator Advanced tab at deploy time.

Table 42 ActiveMatrix BusinessWorks integration Properties for TIBCO BusinessEvents Containers

Property	Notes
<code>tibco.bwengine.name</code>	<p>The name of the ActiveMatrix BusinessWorks engine. Used for ActiveMatrix BusinessWorks integration projects where TIBCO BusinessEvents is the container, and used only if the following is true:</p> <ul style="list-style-type: none">• You will use ActiveMatrix BusinessWorks checkpointing• TIBCO BusinessEvents engine instances are deployed with different names. <p>Use this property to ensure that the same ActiveMatrix BusinessWorks engine name is specified in all nodes.</p> <p>Available in TIBCO Administrator by default. No default value.</p>
<code>Hawk.AMI.DisplayName</code>	<p>The name of the TIBCO Hawk microagent (HMA) instance used for TIBCO BusinessEvents. You can change the name as desired. The default name uses this format:</p> <pre>com.tibco.Adapter.{be-engine bw-engine}.DOMAIN. DEPLOYMENT.COMPONENT_INSTANCE</pre> <p>For container mode TIBCO BusinessEvents-ActiveMatrix BusinessWorks integration projects where TIBCO BusinessEvents is the container, TIBCO BusinessEvents internally appends <code>-bw</code> to this name and uses it for the ActiveMatrix BusinessWorks HMA name. However, if <code>be.hawk.microagent.name</code> is used the behavior is different.</p> <p>See <code>be.Hawk.microagent.name</code> for related information.</p>

Table 42 *ActiveMatrix BusinessWorks integration Properties for TIBCO BusinessEvents Containers (Cont'd)*

Property	Notes
<code>be.hawk.microagent.name</code>	<p>This property is used only for container mode TIBCO BusinessEvents-ActiveMatrix BusinessWorks integration projects where TIBCO BusinessEvents is the container. This property is not present by default. You must add it if you want to use it.</p> <p>If you want to use different names for the ActiveMatrix BusinessWorks and TIBCO BusinessEvents TIBCO Hawk microagents (HMAs), but you don't want to use the <code>Hawk.AMI.DisplayName</code> naming scheme (where both HMAs have the same name, except that the ActiveMatrix BusinessWorks name is appended with <code>-bw</code>), then do the following:</p> <ul style="list-style-type: none"> • Define the name of the TIBCO BusinessEvents HMA using <code>be.hawk.microagent.name</code>. • Define the name of the ActiveMatrix BusinessWorks HMA using <code>Hawk.AMI.DisplayName</code> (or just use the default).

Configuring a RuleServiceProvider Configuration Resource

You reference a RuleServiceProvider Configuration resource in various activities so ActiveMatrix BusinessWorks can locate the TIBCO BusinessEvents resources used in the integration.

1. Open the ActiveMatrix BusinessWorks project in TIBCO Designer and open the project folder where you want to store the resource.
2. Right-click in the design panel and select **Add Resource > TIBCO BusinessEvents Activities > TIBCO BusinessEvents RuleServiceProvider Configuration**.
3. Enter values in the Configuration tab, following guidelines in [TIBCO BusinessEvents RuleServiceProvider Configuration Resource Reference on page 363](#).
4. **Click** Apply and save the resource.

TIBCO BusinessEvents RuleServiceProvider Configuration

Resource Reference



This resource specifies which TIBCO BusinessEvents project to use for the integration. See also [Configuring a RuleServiceProvider Configuration Resource on page 362](#).

Configuration

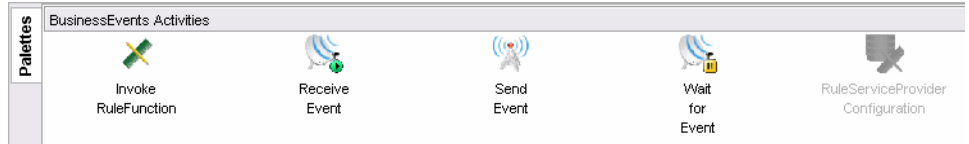
The Configuration tab has the following fields.

Field	Global Var?	Description
Name	No	The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275 .
Description	No	Short description of the resource.
Repo URL/EAR Path	Yes	<p>One of the following:</p> <p>Path to and name of the TIBCO BusinessEvents EAR file. Ontology definitions in this EAR file are used by activities in the TIBCO BusinessEvents activities palette.</p> <p>(At design or testing time you can use an exact copy of the EAR that is used for production. You can use a global variable so that you can define the location of the EAR at deploy time.)</p> <p>or</p> <p>The server-based repository URL for a TIBCO BusinessEvents project that was deployed to a TIBCO Administrator domain.</p>

Field	Global Var?	Description
Deployment Descriptor Path	Yes	<p>Path to the CDD (Cluster Deployment Descriptor) file used by the deployed TIBCO BusinessEvents application.</p> <p>At design or testing time you can use an exact copy of the CDD that is used for production. You can use a global variable so that you can define the location of the CDD at deploy time.</p>
Processing Unit ID		ID of a processing unit in the specified CDD whose values are used for the integration.

Working With the TIBCO BusinessEvents Activities

This guide assumes you know how to work with ActiveMatrix BusinessWorks processes. See *TIBCO BusinessWorks Process Design Guide* if you need detailed guidance.



The TIBCO BusinessEvents Activities palette allows you to easily send and receive TIBCO BusinessEvents events in TIBCO ActiveMatrix BusinessWorks applications.

Before you work with other activities, configure a RuleServiceProvider Configuration resource so that the ActiveMatrix BusinessWorks project can locate TIBCO BusinessEvents resources. See [TIBCO BusinessEvents RuleServiceProvider Configuration Resource Reference on page 363](#).

To Work with the TIBCO BusinessEvents Activities

1. Open the folder in which you want to add a process definition. Right-click in the design panel, and select **Add Resource > Process > Process Definition**.
2. Double-click the Process Definition resource to open it. You see the Start and End activities.
3. Right-click in the design panel, and select **Add Resource > TIBCO BusinessEvents Activities**. Select activities as needed to configure the process. See the following for guidelines:
 - [Receive Event Resource Reference on page 366](#)
 - [Send Event Resource Reference on page 368](#)
 - [Wait for Event Resource Reference on page 369](#)
 - [Invoke RuleFunction Resource Reference on page 376](#)
4. Click **Apply** and save the resource.

Receive Event Resource Reference



This process starter starts a process when the specified simple event arrives from its default destination channel, or from a destination specified in the configuration tab.

Configuration

The Configuration tab has the following fields.

Field	Global Var?	Description
Name	No	The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275 .
Description	No	Short description of the resource.
RuleServiceProvider Configuration		Browse to and select the RuleServiceProvider Configuration resource used in this process. The lists of events and destinations shown in the Event Reference and Custom Destination fields comes from the EAR file referenced in the selected RuleServiceProvider Configuration resource.
Event Reference	No	Browse to and select the simple event you want to receive.
Custom Destination	No	If you do not want to listen for the specified event on its default destination, browse to and select an alternate destination.

Misc

The Misc tab has the following fields.

Field	Description
Sequencing Key	<p>This field can contain an XPath expression that specifies which processes should run in order. Process instances with sequencing keys that evaluate to the same value will be executed sequentially in the order the process instance was created.</p> <p>See <i>TIBCO ActiveMatrix BusinessWorks Process Design Guide</i> for more information about controlling the execution order of process instances and about XPath expressions.</p>
Custom ID	<p>This field can contain an XPath expression that specifies a custom ID for the process instance. This ID is displayed in the View Service dialog of TIBCO Administrator, and it is also available in the <code>\$_processContext</code> process variable.</p>

Output

The Output tab has the following fields.

Output Item	Data Type	Description
<code>BEReceiveEventOutput</code>	complex	Expand <code>BEReceiveEventOutput</code> to show the name, properties, and attributes of the event received.

Send Event Resource Reference



Data from the ActiveMatrix BusinessWorks process context is used to send an event of a specified type to its default destination or to a destination specified in the configuration tab.

Configuration

The Configuration tab has the following fields.

Field	Global Var?	Description
Name	No	The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275 .
Description	No	Short description of the resource.
RuleServiceProvider Configuration		Browse to and select the RuleServiceProvider Configuration resource used in this process. The lists of events and destinations shown in the Event Reference and Custom Destination fields comes from the EAR file referenced in the selected RuleServiceProvider Configuration resource.
Event Reference	No	Browse to and select the simple event you want to send.
Custom Destination	No	If you do not want to send the specified event to its default destination, browse to and select an alternate destination.

Input

The Input tab has the following fields.

Input Item	Data Type	Description
BESendEventInput	complex	Expand BESendEventInput to show the name, properties, and attributes of the event type selected in the Configuration tab.

Wait for Event Resource Reference



This activity listens to the default destination of a specified event type or to a destination specified in the configuration tab, and waits for a matched simple event.

Configuration

The Configuration tab has the following fields.

Field	Global Var?	Description
Name	No	The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275 .
RuleServiceProvider Configuration		<p>Browse to and select the RuleServiceProvider Configuration resource used in this process.</p> <p>The lists of events and destinations shown in the Event Reference and Custom Destination fields comes from the EAR file referenced in the selected RuleServiceProvider Configuration resource.</p>
Description	No	Short description of the resource.
Event Reference	No	Browse to and select the simple event you want to wait for.
Custom Destination	No	If you do not want to listen for the specified event on its default destination, browse to and select an alternate destination.

Event

The Event tab has the following fields.

Input Item	Global Var?	Description
Candidate Event Key	No	<p>Use to filter events. Only events whose candidate event key matches the key provided will trigger the activity.</p> <p>For example you can use the event's extID as the event key. You can also use any of the event's properties as a matching candidate event key. There can be multiple keys, but only the first one that matches is considered.</p> <p>As an example use of this property, suppose you want "Wait for Event" to fire only for the 10th occurrence of an event. You would set the Candidate Event Key value to 10. In the input tab, you would map the input item <code>key</code> to a global variable that is incremented by 1 each time the event occurs.</p> <p>Refer to <i>ActiveMatrix BusinessWorks Palette Reference</i> sections on other "Wait For" types of activities (for example, for JMS queues or topics, or for Rendezvous) for more information.</p>
Event Timeout	No	<p>If the event is received before the process reaches this activity, the event waits for this number of milliseconds. If the timeout period ends before the process reaches this activity, the event is discarded.</p>

Input

The Input tab has the following fields.

Input Item	Data Type	Description
<code>key</code>	String	Candidate Event Key value (specified on the Event tab
<code>processTimeout</code>	Integer	The <code>processTimeout</code> value is the Event Timeout from Event tab

Output

The Output tab has the following fields.

Input Item	Data Type	Description
<code>BEReceiveEventOutput</code>	complex	Expand <code>BEReceiveEventOutput</code> to show the name, properties, and attributes of the event type selected in the Configuration tab.

Invoking a TIBCO BusinessEvents Rule Function from a Process

To enable an ActiveMatrix BusinessWorks process to call a TIBCO BusinessEvents rule function, you need to configure an Invoke RuleFunction activity.

See [Integration Components on page 350](#) for an introduction to the integration features.



- See [Thread Management on page 352](#) for important information about avoiding deadlock.
- Be careful in using the `ThreadLocal` variable because the thread of execution may not be same as that in the prior activity.
- Concepts passed to the rule function are not automatically asserted. Similarly, events passed to the rule function are not automatically asserted. You must assert them explicitly.
- `Object` support is not provided. You can't set `Object` in the scope of the rule function or as its return value. You must specify a specific type of TIBCO BusinessEvents object or a primitive (other than `Object`).

Specifying Input Arguments

All TIBCO BusinessEvents primitives (except `Object`) are supported. Appropriate boxing and unboxing of XML primitive types to TIBCO BusinessEvents primitive types is handled. For more details, see [Chapter 18, Rule Language Datatypes, on page 291](#).

You can pass null values. Select the argument in the Activity Input panel of the Input tab. Click the Edit Statement (exclamation point) button. In the Edit Statement dialog, Content tab, check the Set Explicit Nil checkbox. If `xsi:nil==true`, then a null is passed as the argument.

Base Concept and Base Event are supported as argument types. You must map a specific type to the base concept or base event. Open the Edit Statement dialog, and in the Type tab, check the Type Substitution check box. In the Type field, select XML Type Reference. Select a Schema and a Type.

Using Synchronous Invocation

Execution is synchronous. The process waits for the rule function to return a value. The rule function acquires a lock on TIBCO BusinessEvents working memory until the RTC cycle completes and then returns a value to the process.

You can use more than one Invoke RuleFunction activity in a process only if each uses a different agent. See [Thread Management on page 352](#) for more details.

Using the lockWM Parameter

By default the working memory is locked during the invocation (that is, the value is true). You can change this behavior by setting the `lockWM` parameter to false.

When you set the value to false, the rule function executes outside the context of the working memory.

The results must not modify concept instances or scorecards (as in the case of a preprocessor).

Set the `lockWM` parameter to false for optimization purposes only and use with care. Set it to false only if the call does not modify the working memory in any way. For example, use it to perform stateless calls or calculations, or to create new concept instances or events, or for lookups and so on.

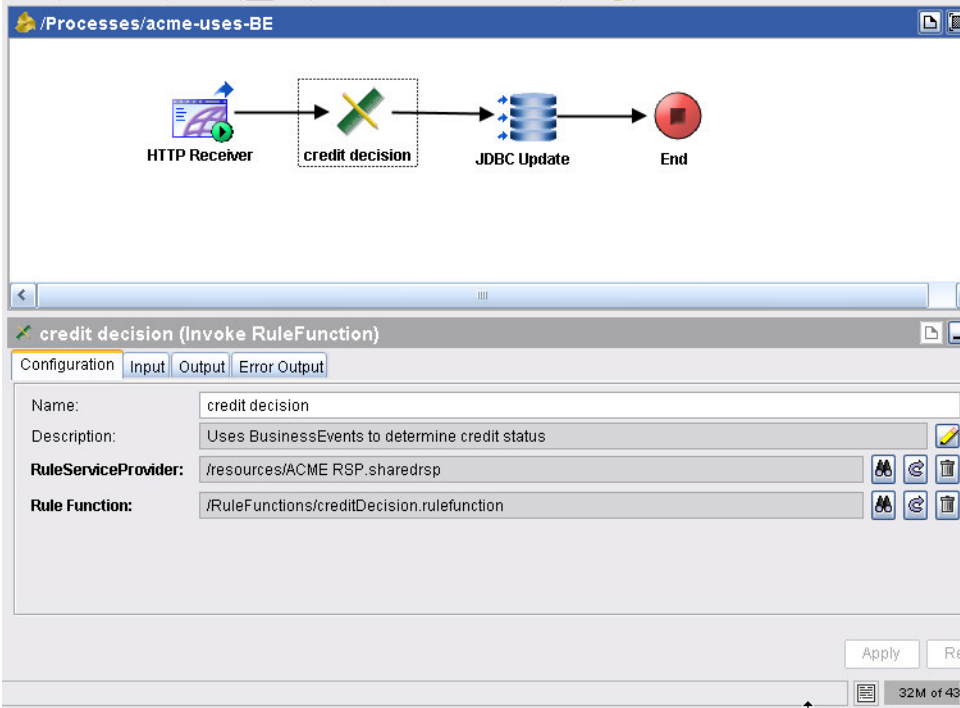
Overriding the Rule Function at Runtime

You can override which function is called at runtime. To do so, enter a global variable as the value for `rulefunction` in the input tab. At runtime, you can specify a rule function with a different name but the same signature as the one specified in the activity's Configuration tab.

You can only use rule functions that are in the TIBCO BusinessEvents EAR file deployed for the integration project.

Working With Invoke RuleFunction Activities

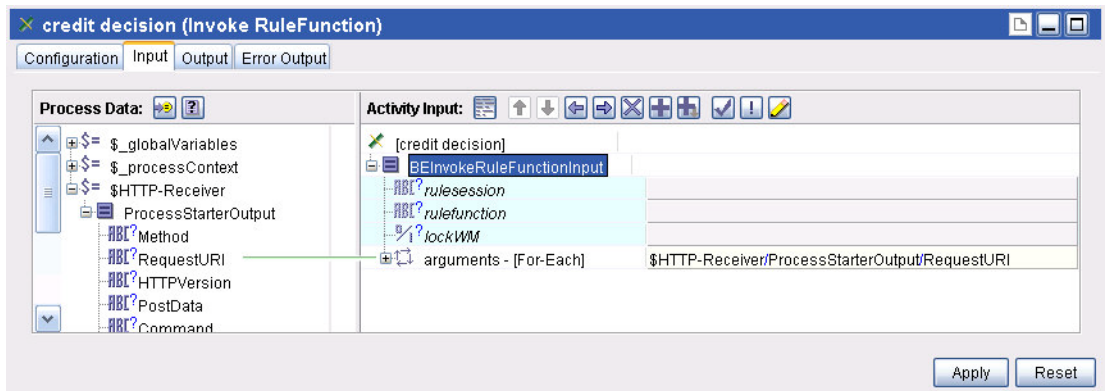
See [Invoking a TIBCO BusinessEvents Rule Function from a Process on page 372](#) for important information about using the Invoke RuleFunction activity in your ActiveMatrix BusinessWorks process.



To Configure an Invoke RuleFunction Activity

1. Start TIBCO Designer and open (or add) the ActiveMatrix BusinessWorks process in which you want to use an Invoke RuleFunction activity.
2. Right-click in the design panel and select **Add Resource > TIBCO BusinessEvents Activities > Invoke RuleFunction**. Link the activity to other activities as appropriate.

3. Complete the Configuration tab, following guidelines provided in [Invoke RuleFunction Resource Reference on page 376](#).
 - Name the activity and give it a description as desired.
 - Browse to and select the RuleServiceProvider Configuration resource that contains the ontology for the project you are integrating with.
 - Browse to and select the TIBCO BusinessEvents rule function you want to invoke.
4. Select the **Input** tab. Expand each item to see all fields. Enter values or map values from process data. Follow guidelines provided in [Invoke RuleFunction Resource Reference on page 376](#).



5. Click **Apply** and save the resource.

Invoke RuleFunction Resource Reference



To enable an ActiveMatrix BusinessWorks process to call a TIBCO BusinessEvents rule function, you configure one or more Invoke RuleFunction activities according to your needs. Execution is synchronous. See [Invoking a TIBCO BusinessEvents Rule Function from a Process on page 372](#) and [Working With Invoke RuleFunction Activities on page 374](#) for more details.

Configuration

The Configuration tab has the following fields.

Field	Global Var?	Description
Name	No	The name to appear as the label for the resource. Names follow Java variable naming restrictions. Do not use any reserved words. Names must be unique within a folder. See Identifiers (Names) on page 275 .
Description	No	Short description of the resource.
RuleServiceProvider Configuration		Browse to and select the RuleServiceProvider Configuration resource used in this process. The list rule functions you can select for the Rule Function field comes from the EAR file referenced in the selected RuleServiceProvider Configuration resource.
Rule Function	No	Select the desired rule function from the TIBCO BusinessEvents project.

Input

The Input tab has the following fields.

Field	Global Var?	Description
<code>rulesession</code>	Yes	Specifies the agent where the rule function is to execute. Optional if the processing unit has only one agent To specify a value, enter the agent name (as specified in the CDD file).
<code>rulefunction</code>	Yes	Optional. Allows you to override the rule function specified in the Configuration tab. Enter a global variable that is used to specify the name of the rule function at runtime. Do not enter the <code>.rulefunction</code> extension. The rule function you specify must have the same signature as the function specified in the Configuration tab (and a different name). See Overriding the Rule Function at Runtime on page 373
<code>lockWM</code>	Yes	Default value is true. The working memory is locked during the invocation. You can set the value to false only if conditions explained in the section Using the lockWM Parameter on page 373 are met.
<code>arguments</code>	No	Displays the arguments for the rule function, if it has any. Map the available process data to the activity input, or enter values in the fields as appropriate for the datatype of each argument.

Output

The Output tab has the following fields.

Output Item	Data Type	Description
Return	(Varies)	The return type for the specified rule function. Return can be void or any of the following types: String, integer, long, double, Boolean, datetime, concept, event.

Working with the BusinessWorks Functions

The integration model provides a BusinessWorks category of functions. This category is located in the General Functions library. Use and configuration of the functions in the BusinessWorks category are described in this section. See [Integration Components on page 350](#) for a brief introduction to these features.

Providing Paths to TIBCO BusinessEvents Project Resources Using Schemas

The `BusinessWorks.invokeProcess()` and `BusinessWorks.startProcess()` functions can send events to and receive events from ActiveMatrix BusinessWorks processes. However, the paths to TIBCO BusinessEvents project resources are not available by default in TIBCO Designer. To use these functions successfully, you must generate the event schemas in TIBCO BusinessEvents and import them into the TIBCO Designer project. See [Exporting \(Generating\) Concept and Event Schema \(XSD\) Files on page 7](#) for details on exporting the event schemas.

In the TIBCO Designer project, import these schemas and use them in the relevant processes, so that the event resource path is known to TIBCO Designer. See ActiveMatrix BusinessWorks documentation for details.

Using invokeProcess()

Purpose	Use this function to take advantage of ActiveMatrix BusinessWorks features. For example, you could use <code>invokeProcess()</code> to send customer information to an ActiveMatrix BusinessWorks process that gets the discount information from a database and returns it to TIBCO BusinessEvents.
Runtime Behavior	<p>If the ActiveMatrix BusinessWorks engine is not already started, TIBCO BusinessEvents will start it before <code>invokeProcess()</code> executes. See Using init() on page 382 for another way to start the engine.</p> <p>When a rule containing or calling <code>invokeProcess()</code> is triggered, <code>invokeProcess()</code> calls the specified ActiveMatrix BusinessWorks process and passes it an event. The <code>invokeProcess()</code> function executes synchronously, so the TIBCO BusinessEvents rule engine waits. The ActiveMatrix BusinessWorks process performs its work and returns an event or null at completion of the process, or it times out if you set a timeout that is exceeded. It generates an advisory event if it times out. Events received are not automatically asserted.</p>

Error Handling and Advisory Events

TIBCO BusinessEvents asserts an advisory event and returns `null` if the ActiveMatrix BusinessWorks process fails, or if the invocation times out. The advisory event category is `Engine`, and the type is `INVOKE BW PROCESS`. The message contains the error message from the failed ActiveMatrix BusinessWorks process, or the timeout message. The advisory event is also created if the process is cancelled using `cancelProcess()`, because the system can't differentiate between different causes for the process stopping before completion.

Because the `invokeProcess()` function returns `null` when an error occurs, you must handle the possibility of a `null` return in rules that use any property or attribute of the returned event.

See [Chapter 10, Advisory Events, on page 157](#) for more information about advisory events.



Do not use `invokeProcess()` more than once in the same thread of execution. See [Thread Management on page 352](#).

Configuring the Function

When configuring the `invokeProcess()` function in a rule or rule function, specify the parameters as follows:

- The ActiveMatrix BusinessWorks process that the function invokes. The specified process must not contain a process starter.
- The event to pass to the process (or specify `null` if you don't want to pass an event to the process).
- A timeout (or specify zero if you don't want to use a timeout).

For example, you send customer information from an event (alias `neworder`) to an ActiveMatrix BusinessWorks process, which returns an event with the discount level to offer:

```
Events/Discountlevel Discount;
Discount = Businessworks.invokeProcess("/Processes/CustInfo",
neworder, 0);
```



Timeouts If you set a timeout period and a timeout occurs, the rule or rule function containing `InvokeProcess()` continues without waiting for the ActiveMatrix BusinessWorks process to complete. If you use a timeout, set it to a period long enough for the ActiveMatrix BusinessWorks process to complete. Include logic to handle the case that the `invokeProcess()` function does time out.

Configuring the Process

Configure the Start activity of the specified ActiveMatrix BusinessWorks process to accept the event passed by the `invokeProcess()` function, if an event is passed. If you specify `null`, of course, this step is not required.

Configure the rest of the activities in the process to carry out whatever processing is desired using the data passed into it by `invokeProcess()`.

Similarly, in the End activity, specify the event type to return to the `invokeProcess()` process.

The returned event is then used as needed by the logic of the rule or rule function.



Events returned are not asserted. You must explicitly assert them as needed.

See [Providing Paths to TIBCO BusinessEvents Project Resources Using Schemas](#) above for additional setup you must do.

Using `startProcess()`

Purpose Use the `startProcess()` function when you want to invoke an ActiveMatrix BusinessWorks process that performs work that can be completed asynchronously. The `startProcess()` function invokes an ActiveMatrix BusinessWorks process in asynchronous mode and immediately returns the job ID of the process. Rule processing continues. When the ActiveMatrix BusinessWorks process completes, it passes an event to a callback rule function that is specified in a `startProcess()` argument. For example, you send order information to an ActiveMatrix BusinessWorks order fulfillment process. When the order ships, notification is returned to TIBCO BusinessEvents, which updates a customer concept instance.

Runtime Behavior If the ActiveMatrix BusinessWorks engine is not already started, TIBCO BusinessEvents will start it before `startProcess()` executes. See [Using `init\(\)` on page 382](#) for another way to start the engine.

When a rule containing or calling `startProcess()` is triggered, `startProcess()` calls the specified ActiveMatrix BusinessWorks process and passes it an event. `startProcess()` returns the `jobID` of the ActiveMatrix BusinessWorks process. (You can use the returned job ID, for example, in the `cancelProcess()` rule function.) The function executes asynchronously, so the TIBCO BusinessEvents rule engine continues while at the same time the ActiveMatrix BusinessWorks process executes. The ActiveMatrix BusinessWorks process performs its work and passes an event to the callback (`ruleFnURI`) rule function specified in the `startProcess()` function arguments. The `ruleFnURI` rule function performs its work, for example, creating and asserting the event.



The `ruleFnURI` rule function must not modify concept instances or scorecards.

Configuring the Function	<p>When configuring the <code>startProcess()</code> function in a rule or rule function, specify the following in the parameters:</p> <ul style="list-style-type: none"> • The ActiveMatrix BusinessWorks process that the function invokes. The specified process must not contain a process starter. • The event to pass to the process (or specify null if you don't want to pass an event to the process). • The TIBCO BusinessEvents rule function that the process calls on completion (the callback rule function). The required signature for the callback rule function is shown below.
Configuring the ruleFnURI Rule Function	<p>The callback rule function, specified in the <code>startProcess()</code> <code>ruleFnURI</code> argument, is called when the ActiveMatrix BusinessWorks process completes. Add the <code>ruleFnURI</code> rule function to the TIBCO BusinessEvents project.</p> <p>For convenience, this rule function is referred to as the <code>ruleFnURI</code> rule function in this section. The <code>ruleFnURI</code> rule function must have the following signature, and the Validity field (in the Configuration tab) must be set to Action:</p> <pre>void ruleFn(long jobID, int status, Event outputEvent, Object closure)</pre> <ul style="list-style-type: none"> • <code>jobID</code> Type: <code>long</code>. The job id can be used to correlate the information passed to <code>ruleFnURI</code> with related information in TIBCO BusinessEvents. • <code>status</code> Type: <code>int</code>. Returns 0 (zero) if the process completed successfully, and -1 if the process did not complete successfully (for example because <code>cancelProcess()</code> was called). • <code>outputEvent</code> Type: <code>Event</code>. An event passed to the rule function by the ActiveMatrix BusinessWorks process. It can be created by the process, or it can be an existing event. If the process fails or is cancelled (for example because <code>cancelProcess()</code> was called), an advisory event is returned. • <code>closure</code> Type: <code>Object</code>. A closure object could be, for example, a value from the original context that has to be passed back to the TIBCO BusinessEvents engine, for example, a loan rate that has been promised. (Note that type is <code>Object</code>, so you can't pass an event or concept.)
Configuring the Process	<p>Configure the Start activity of the specified ActiveMatrix BusinessWorks process to accept the event passed by the <code>startProcess()</code> function, if an event is passed. If you specify null, of course, this step is not required.</p> <p>Configure the rest of the activities in the process to carry out whatever processing is desired using the data passed into it by <code>startProcess()</code>.</p> <p>In the End activity, specify the event type that is to be passed to the specified <code>ruleFnURI</code> rule function.</p>

See [Providing Paths to TIBCO BusinessEvents Project Resources Using Schemas on page 378](#) above for additional setup you must do.

The returned event is then used as needed by the logic of the rule or rule function. Note that if you want to assert the event that is returned, you must explicitly assert it.

Using cancelProcess()

Purpose Use the `cancelProcess()` function to cancel a long running process, specified by `jobID`.

Cancellation may fail if the process has already completed before receiving the cancellation command. In this case, the following exception is thrown:

```
java.lang.Exception: Job JobId not found
```

Configuration The job ID for the process you want to cancel is provided in the return value of `startProcess()`.

Using init()

Purpose The `init()` function initializes the ActiveMatrix BusinessWorks engine if it is not already running. Use of `init()` when the engine is already running is harmless.

Use of `init()` is optional. It is provided as a convenience. For example you can use it in a TIBCO BusinessEvents startup rule function to initialize the ActiveMatrix BusinessWorks engine at startup, so as to save valuable time later.

If `invokeProcess()` or `startProcess()` are executed when the ActiveMatrix BusinessWorks engine is not already started, they will start the engine at that time.

Configuration A general good practice is to call `init()` to start the ActiveMatrix BusinessWorks engine in a startup rule function (specified in the Startup/Shutdown tab of the BAR). The `init()` function takes no parameters. The rule function might simply contain this code:

```
BusinessWorks.init();
```

Using shutdown()

Purpose The `shutdown()` function shuts down the ActiveMatrix BusinessWorks engine if it is running. Use of `shutdown()` when the engine is already shut down is harmless.

Use of `shutdown()` is optional. It is recommended that you use it only when you have finished using the ActiveMatrix BusinessWorks engine, and won't need to start it again. Stopping and restarting the engine is not necessary and can affect performance

Configuration Simply place the shutdown command so that it is executed when needed. The `shutdown()` function takes no parameters. The rule function might simply contain this code:

```
BusinessWorks.shutdown();
```


TIBCO BusinessEvents Performance Profiler

This chapter explains how you can run a profiler utility to gather statistics about activities that occur during each RTC cycle. This information helps to identify bottlenecks in the project, which can often be addressed by redesigning rules or other aspects of a project.

Topics

- [Overview of Profiler, page 386](#)
- [Changing the Delimiter Character, page 387](#)
- [Turning Profiler On and Off, page 388](#)
- [Profiler Reference, page 394](#)

Overview of Profiler

The profiler utility collects statistics relating to the run to completion (RTC) rule evaluation cycle in an inference agent. The utility does not collect data about object management. It also does not collect data for other types of agents. It helps you understand, for example, internal execution times and frequencies of rules.

The profiler records time spent during each RTC on activities such as number of times each condition or action is performed, and total time spent on each condition and action. A complete RTC includes conditions and actions, although any individual RTC might contain only conditions or only actions.

Statistics are collected for each completed RTC. When the profiler is directed to stop during an RTC, it continues to collect data for the current RTC until that RTC is completed.

When the profiler is turned off, it continues to write statistics for the current session until that session is completed. So the RTC in progress is always completed, even if the profiler is directed to stop during an RTC.

After the profiler finishes, statistics data is written to the specified (or default) file and cleared from memory.

You can execute the profiler and turn it off in three ways:

Using properties Profiler turns on when the agent initializes at system startup. Used to profile RTC time, including startup rule functions.

Using TIBCO BusinessEvents catalog functions Profiler turns at the beginning of the next RTC after the function call, if it has not already been enabled. (There is no effect if the profiler is already on). Used to turn the profiler on and off inside a rule or rule function.)

Using a TIBCO Hawk method Profiler turns on by invoking a TIBCO BusinessEvents microagent Hawk method. Profiler is turned on at the beginning of the next RTC after the method call, if it has not already been enabled. (There is no effect if the profiler is already on). Used to dynamically turn the profiler on and off.

Using a TIBCO BusinessEvents Monitoring and Management method Similar to using a Hawk method.

See [Turning Profiler On and Off on page 388](#) for details.

See [Profiler Reference on page 394](#) for a reference to all output data file column headings.

Changing the Delimiter Character

The profiler is tab-delimited by default. The delimiter character can be changed adding the following property in the CDD file:

```
be.engine.profile.delimiter
```

Specify the delimiter using a String value. Enclose the value in double quotes (the quotes are not used as part of the delimiter).

For example to use an open curly brace as the delimiter, you would specify "{" as the value. Do not choose a character used in rule conditions.

Use a single character if the application into which you will import the output uses a one-character delimiter. When importing the file into Excel, do not check the "Treat consecutive delimiters as one" option. Consecutive delimiters indicate a column that is empty.



Also, when importing the file into Excel, set the timestamp field to Text (and not General, which is the default).

Turning Profiler On and Off

This section explains the different ways you can turn the profiler on and off.

To Turn Profiler On and Off Using TIBCO BusinessEvents Monitoring and Management

If you have deployed the processing unit using TIBCO BusinessEvents Monitoring and Management (MM), you can turn the profiler on and off using the MM Console.

Use of MM is documented in the *TIBCO BusinessEvents Administration* guide. In particular, see the section To Execute a Method and the section Profiler Group, both in Chapter 9, Monitoring and Managing a TIBCO BusinessEvents Cluster.

To Turn Profiler On and Off Using Properties

Set the following properties in the Cluster Deployment Descriptor (CDD) Processing Unit tab, for all processing units (engines) whose RTC performance you want to profile.

Table 43 Profiler Configuration Properties (Sheet 1 of 3)

Property	Notes
<code>be.engine.profile.Agent_Class_Name.enable</code>	<p>If set to true, enables profiler for the specified agent class (<i>Agent_Class_Name</i>) when the agent initializes.</p> <p>Default is false.</p>
<code>be.engine.profile.*.enable</code>	<p>If set to true, enables the profiler for all agents when each agent initializes, even when a specified agent class profiler is disabled.</p> <p>Default is false.</p>

Table 43 Profiler Configuration Properties (Sheet 2 of 3)

Property	Notes
<code>be.engine.profile.Agent_Class_Name.file</code>	<p>The name of output file that the profiler writes to, for the specified agent (<i>Agent_Class_Name</i>).</p> <p>Optional, if <code>be.engine.profile.*.file</code> is specified.</p> <p>If <code>be.engine.profile.*.file</code> is specified and <code>be.engine.profile.Agent_Class_Name.file</code> is not specified, then the file name is the value of <code>be.engine.profile.*.file</code>, with the <i>Agent_Class_Name</i> appended.</p> <p>If the properties <code>be.engine.profile.*.file</code> and <code>be.engine.profile.Agent_Class_Name.files</code> are not specified the following occurs:</p> <p>The file name is <code>be-profile.csv</code>, followed by an underbar, followed by the <i>Agent_Class_Name</i>: <code>be-profile_ Agent_Class_Name.csv</code></p>
<code>be.engine.profile.*.file</code>	<p>The default (prefix for the) name of the output file that the profiler writes to. In all cases, the appropriate <i>Agent_Class_Name</i> is appended.</p> <p>Default name is <code>be-profile.csv</code> and it is located under the current working directory, if file name is not specified.</p>
<code>be.engine.profile.Agent_Class_Name.duration</code>	<p>Specifies the duration of profile data collection in seconds, for the specified <i>Agent_Class_Name</i>).</p> <p>When the duration period ends, the profiler continues to collect statistics for the current RTC until the RTC is completed, then outputs data and stops. So the RTC in progress is always completed, even if the profiler is directed to stop during an RTC.</p> <p>If you set duration to a value of zero or less (≤ 0), then profiling continues until agent stops or profiler is explicitly turned of using a function or Hawk method.</p> <p>Default is -1.</p>

Table 43 Profiler Configuration Properties (Sheet 3 of 3)

Property	Notes
<code>be.engine.profile.*.duration</code>	<p>Specifies the duration of profile data collection in seconds, for all agents.</p> <p>When the duration period ends, the profiler continues to collect statistics for the current RTC until the RTC is completed, then outputs data and stops. So the RTC in progress is always completed, even if the profiler is directed to stop during an RTC.</p> <p>If you set duration to a value of zero or less (≤ 0), then profiling continues until the agents stop or profiler is explicitly turned of using a function or Hawk method.</p> <p>When <code>be.engine.profile.Agent_Class_Name.duration</code> and <code>be.engine.profile.*.duration</code> are both present, the duration specified in <code>be.engine.profile.Agent_Class_Name.duration</code> takes precedence.</p> <p>Default is -1.</p>
<code>be.engine.profile.Agent_Class_Name.level</code>	<p>Level of depth that profile data will be collected for the specified agent (<i>Agent_Class_Name</i>) :</p> <ul style="list-style-type: none">• -1: all levels of profile data are collected, including RTC level and conditions and actions within the RTC.• 1: Only RTC level of profile data will be collected (and no condition and action data). <p>Default is -1.</p>
<code>be.engine.profile.*.level</code>	<p>Level of depth that profile data will be collected for all agents:</p> <ul style="list-style-type: none">• -1: all levels of profile data are collected, including RTC level and conditions and actions within the RTC.• 1: Only RTC level of profile data will be collected (and no condition and action data). <p>When <code>be.engine.profile.Agent_Class_Name.level</code> and <code>be.engine.profile.*.level</code> are both present, the level specified in <code>be.engine.profile.Agent_Class_Name.level</code> takes precedence.</p> <p>Default is -1.</p>

To Turn Profiler On and Off Using Functions

This section assumes you understand how to use TIBCO BusinessEvents functions. It tells you which functions to use and the effect of each function.



You can turn the profiler on using the engine properties, and turn it off using a function or Hawk method, as desired. See notes for `be.engine.Agent_Class_Name.profile.duration` and `be.engine.*.profile.duration` in [Table 43, Profiler Configuration Properties, on page 388](#).

To turn the profiler on In your rule or rule function, use the following function to turn on the profiler:

```
Engine.Profiler.startCollectingToFile(String fileName, int level, long duration)
```

The above function turns on the TIBCO BusinessEvents Profiler and starts collecting data for the specified duration for the agent in which the rule or rule function that calls this function is executed. The profiler starts collecting data at the beginning of next RTC.

Profile data is output to the specified file in comma-separated value format at the end of the duration period, unless the profiler is turned off before the end of the duration, in which case it is output at the end of the RTC that completes after the profiler is turned off.

Input arguments are the same as the engine properties show in [Table 43, Profiler Configuration Properties, on page 388](#):

`String fileName`: See `be.engine.profile.Agent_Class_Name.file`

`int level`: See `be.engine.profile.Agent_Class_Name.level`

`long duration`: See `be.engine.profile.Agent_Class_Name.duration` (Here `Agent_Class_Name` is the current agent in which the rule or rule function that calls this function is executed.)

To turn the profiler off In your rule or rule function, use the following function to turn off the profiler:

```
Engine.Profiler.stopCollecting()
```

The above function turns off the TIBCO BusinessEvents profiler and writes the profile data to a file for the agent in which the rule or rule function that calls this function is included (the file is output at the end of the RTC that completes after the profiler is turned off). There is no effect if the profiler is not on.

To Turn Profiler On and Off Using TIBCO Hawk Methods

This section assumes you understand how to use TIBCO Hawk methods. It tells you which methods to use and the effect of each method



You can turn the profiler on using properties, and turn it off using a function or Hawk method, as desired. See notes for `be.engine.profile.duration` in [Table 43, Profiler Configuration Properties, on page 388](#).

Before you Begin Ensure that the property `hawk.enabled` is set to true in the CDD at the cluster level before the TIBCO BusinessEvents engine starts.

To turn the profiler on Use the following method to turn on the profiler:

```
StartFileBasedProfiler(String session, String fileName, int level, long duration)
```

The above method turns on the TIBCO BusinessEvents profiler for the specified agent. The profiler starts collecting data at the beginning of next RTC for the specified duration.

This method works the same way as the

`Engine.Profiler.startCollectingToFile()` function (see [To Turn Profiler On and Off Using Functions on page 391](#)), except that it requires you to specify an agent class.

Input arguments are the same as the engine properties shown in [Table 43, Profiler Configuration Properties, on page 388](#):

String session: If you want to monitor multiple agents, execute the method once for each, specifying the agent class name in each case. If there is only one agent, the session parameter is optional.

String fileName: See `be.engine.profile.Agent_Class_Name.file`

int level: See `be.engine.profile.Agent_Class_Name.level`

long duration: See `be.engine.profile.Agent_Class_Name.duration`

If you attempt to turn on the profiler when it is already running, an error is returned, but the running profiler is not affected.

To turn the profiler off In your rule or rule function, use the following function to turn off the profiler:

```
StopFileBasedProfiler(String session)
```

The above method turns off the TIBCO BusinessEvents profiler and writes the profile data into a file for the specified agent when the current RTC has completed. You must execute the method once for each session, as needed.

If you attempt to turn off the profiler when it is already off, an error is returned, but there is no effect on the profiler.

Profiler Reference

The table in this section explains each of the columns in the profiler report. Data is grouped by `RTC_Stats_Type` and Description. (Description contains information about the specific RTC.) All data collected for conditions and actions performed during each RTC is listed within each RTC grouping.

Three rows of column headers for the RTC, condition and action are listed at the beginning of the file:

- One for statistics relating to the overall RTC
- One for statistics relating to conditions
- One for statistics relating to actions.

The first column of each data line is always the statistic type, which begins with one of `RTC-`, `CONDITION-`, or `ACTION-`.

Data is also grouped, one row for the overall RTC, and zero or more rows for different conditions or actions or both, as appropriate.

Table 44 Profiler Column Heading Reference (Sheet 1 of 5)

Column Heading	Notes
Statistics Relating to the Overall RTC	
<code>RTC_Stats_Type</code>	Type of rule evaluation cycle (RTC) There are 10 different types: RTC: <code>RTC-Object-Asserted</code> <code>RTC-Object-Modified</code> <code>RTC-Object-Deleted</code> <code>RTC-Event-Expired</code> <code>RTC-Execute-Rule</code> <code>RTC-Invoke-Action</code> <code>RTC-Invoke-Function</code> <code>RTC-Post-Process</code> <code>RTC-Repeat-TimeEvent</code> <code>RTC-Reevaluate-Element</code>
<code>Timestamp</code>	The time at which the first RTC begins.
<code>Description</code>	Information relating to the current <code>RTC_Stats_Type</code> . For example, the description of type <code>RTC-Object-Asserted</code> is the name of the object being asserted.
<code>NumExecuted</code>	Total number of times the same RTC has been executed.

Table 44 *Profiler Column Heading Reference (Sheet 2 of 5)*

Column Heading	Notes
TotalRtcTime	Total time in milliseconds spent on the total number of executions of the same RTC.
AvgRtcTime	$\text{TotalRtcTime} / \text{NumExecuted}$
MaxRtcTime	The maximum time in milliseconds spent on a single RTC.
MinRtcTime	The minimum time in milliseconds spent on a single RTC.
MaxResolvedTime	The maximum time in milliseconds spent to resolve a single RTC, including condition evaluation and action execution, but excluding operations related to object management (OM).
MinResolvedTime	The minimum time in milliseconds spent to resolve a single RTC, including condition evaluation and action execution, but excluding operations related to object management (OM).
Statistics Relating to Conditions	
CONDITION_Stats_Type	Type of rule Condition. One of the following: CONDITION-Filter Condition-Join
Timestamp	The timestamp of the first time the RTC begins.
RuleDescription	Name of the rule containing the condition, or name of state machine transition rule containing the condition.
ConditionDescription	Condition statement of a rule or a state machine transition rule for user-defined condition, or predefined condition name for internal conditions. When a user-defined rule condition has a commented-out line, the ConditionDescription of the next condition is //... Only applies to CONDITION_Stats_Type
NumEvaluated	Total number of times this condition is evaluated in the same RTC.
NumEvalTrue	Total number of times the Join condition is evaluated to true. This value is the sum of NumEvalTruePropagatedLeft and NumEvalTruePropagatedRight.

Table 44 Profiler Column Heading Reference (Sheet 3 of 5)

Column Heading	Notes
TotalTime	Total time in milliseconds spent on the total number of condition evaluations.
AvgTime	$\text{TotalTime} / (\text{NumLeftSearch} + \text{NumRightSearch})$
MaxTime	The maximum time in milliseconds spent on a single condition evaluation.
MinTime	The minimum time in milliseconds spent on a single condition evaluation.
NumEvalPropagatedLeft	Number of times the join condition evaluation is triggered by object assertion propagated from the left side of the condition.
NumEvalTruePropagatedLeft	Number of times the join condition evaluates to true and evaluation is triggered by object assertion propagated from the left side of the condition.
AvgRateEvalTruePropagatedLeft	Average rate that the condition evaluates to true and evaluation is triggered by object assertion propagated from the left side of the condition.
MaxNumEvalTruePropagatedLeft	Maximum number of times the join condition evaluates to true and evaluation is triggered by object assertion propagated from the left side of the condition.
MinNumEvalTruePropagatedLeft	Minimum number of times the join condition evaluate to true and evaluation is triggered by object assertion propagated from the left side of the condition.
NumEvalPropagatedRight	Number of times the join condition evaluation is triggered by object assertion propagated from the right side of the condition.
NumEvalTruePropagatedRight	Number of times the join condition evaluates to true and evaluation is triggered by object assertion propagated from the right side of the condition.
AvgRateEvalTruePropagatedRight	Average rate that the condition evaluates to true and evaluation is triggered by object assertion propagated from the right side of the condition.

Table 44 Profiler Column Heading Reference (Sheet 4 of 5)

Column Heading	Notes
MaxNumEvalTruePropagatedRight	Maximum number of times the join condition evaluates to true and evaluation is triggered by object assertion propagated from the right side of the condition.
MinNumEvalTruePropagatedRight	Minimum number of times the join condition evaluate to true and evaluation is triggered by object assertion propagated from the right side of the condition.
Statistics Relating to Actions	
ACTION_Stats_Type	<p>Type of Actions.</p> <p>There are 10 RTC types and four action types. The four action types are:</p> <p>ACTION-Rule-Action ACTION-Event-Expiry ACTION-Invoke-Action ACTION-Invoke-Function</p>
Timestamp	The timestamp of first time the action execution begins.
Description	<p>Information about the action corresponding to current action type.</p> <p>For example, description of type ACTION-Rule-Action is the name of the rule.</p>
NumExecuted	<p>Total number of times the same action has been executed.</p> <p>A complete action has two phases, action execution and operation.</p>
TotalActionTime	<p>Total time in milliseconds spent on the total number of actions.</p> <p>$\text{TotalActionTime} = \text{TotalExecutionTime} + \text{TotalOperationTime}$</p>
AvgActionTime	$\text{TotalActionTime} / \text{NumExecuted}.$
MaxActionTime	The maximum time in milliseconds spent on a single action.
MinActionTime	The minimum time in milliseconds spent on a single action.

Table 44 Profiler Column Heading Reference (Sheet 5 of 5)

Column Heading	Notes
TotalExecutionTime	Total time in milliseconds spent on the total number of action execution phases. Action execution time is the time Rete network spends on executing the action, for example, time spent in creating new objects, deleting existing objects, and so on.
AvgExecutionTime	TotalExecutionTime / NumExecuted.
MaxExecutionTime	The maximum time in milliseconds spent on a single action execution.
MinExecutionTime	The minimum time in milliseconds spent on a single action execution.
TotalOperationTime	Total time in milliseconds spent on the total number of action operation phases. Action operation time is the time TIBCO BusinessEvents spends on applying changes to the Rete network, for example, time spent asserting newly created objects, or retracting deleted objects.
AvgOperationTime	TotalOperationTime / NumExecuted.
MaxOperationTime	The maximum time in milliseconds spent on a single action operation.
MinOperationTime	The minimum time in milliseconds spent on a single action operation.
MaxAgenda	The maximum size of rule agenda as a result of all action operations.
MinAgenda	The minimum size of rule agenda as a result of all action operations.

Chapter 23 **Testing and Debugging Projects**

This chapter explains how to test and debug projects within TIBCO BusinessEvents Studio.

The sections on debugger assume some familiarity with Eclipse Java debugger, as well as TIBCO BusinessEvents.

Topics

- [Overview, page 400](#)
- [Preparing to Run \(Test\) or Debug a Project, page 403](#)
- [Adding and Working with Launch \(Debug or Run\) Configurations, page 405](#)
- [Launch Configurations Reference, page 408](#)
- [Creating and Working With Test Data, page 410](#)
- [Setting Breakpoints and Running Debugger, page 413](#)
- [Running Tester, page 415](#)
- [Viewing the Results, page 417](#)

Overview

You can run and debug projects in TIBCO BusinessEvents Studio, using test data to understand how TIBCO BusinessEvents rules behave in response to inputs. Debugger shows potential problems in execution of rules and rule functions. You can also simply test a project by running it against test data. You can run multiple engines at the same time, for example, one engine's processing unit might run an inference agent while another's runs a cache agent.



The TIBCO BusinessEvents Decision Manager add-on product also uses a tester feature, Table Analyzer, to enable technical and non-technical users to test rules, rule functions, and decision tables. See TIBCO BusinessEvents Decision Manager documentation for details.

Debugging

TIBCO BusinessEvents Debugger allows you to control the execution of a project by setting breakpoints, stepping through your code, suspending launched programs, examining the contents of variables, providing rule input, and so on.

TIBCO BusinessEvents Debugger integrates with the Eclipse Java development toolkit debugger. Much of the functionality is standard Eclipse debugger functionality. See the Eclipse help for details on features such as breakpoint preferences and other functionality. This chapter explains only the TIBCO BusinessEvents-specific features.

You can debug local projects using their CDD and EAR files, and you can also debug remote engines.



Declarative Programming and Stepping Step into, step over, and step return may not behave as you might expect. In a declarative rule language like TIBCO BusinessEvents, unlike with a procedural language, there is no predetermined path through the code. The inference engine logic determines the next action (next rule) based on various conditions and settings, not the rule itself.

Testing Projects

Running projects to test them works in a way similar to debugging, without the ability to add break points and so on. For testing purposes, you choose options to run the engine, rather than to debug the engine.

Launch Configurations

Before you debug or test a project, you define settings needed to launch the engine, using a *launch configuration* file.

You can add multiple launch configurations for one project, each configured for a different purpose. For example, you may want to test the effect of different startup arguments. Or if you are using cache OM, you could create one configuration for the processing unit (PU) that runs an inference agent, and one for the PU that runs the cache agent.

Test Data

When you test or debug an engine, you must assert test data directly to working memory. The test data in the working memory triggers rules in the normal way, so you can observe conflict resolution and run to completion cycles in the engine.

You can provide test data to the engine in two ways:

Test Data You can create and save concept and event instance data for later use. You can then assert this data selectively when running or debugging the engine.

Rule Data You can provide data as it is expected by rules in the Rule Data tab of the Debugger perspective. See [Working with Rule Data on page 412](#).

See [Creating and Working With Test Data on page 410](#).

The Rule Agenda and Variable Views

When you run the debugger and reach a breakpoint in a rule or rule function the execution is suspended. You can now examine the state of things:

- The Variable view displays value of all the variables in scope.
- The Rule Agenda view displays rules to be executed from the rule agenda, excluding the current rule (where execution is suspended due to the breakpoint).

For example, Event1 is asserted, Event1, which causes Rule1 to execute. Rule1 creates an instance of Concept1. Rule2, Rule3, and Rule4 have Event1 and Concept1 in scope, and their conditions are satisfied by the instance of Event1 and Concept1. If a breakpoint suspends execution in Rule2, then you see Rule3 and Rule4 in the Rule Agenda view.

For details about how the rule agenda is built, see Understanding Conflict Resolution and Run to Completion Cycles in *TIBCO BusinessEvents Architect's Guide*.

Viewing and Understanding Results

After every run, TIBCO BusinessEvents creates a consolidated results file in XML format. It has detailed information on the test run. This results XML file is easy to view and compare.

An editor also appears that displays the results in TIBCO BusinessEvents Studio.

See [Viewing the Results on page 417](#).

Preparing to Run (Test) or Debug a Project

You can do these tasks in any order. For information on setting up Tester Preferences, see [Tester Preferences on page 457](#).

Build an EAR File

1. In TIBCO BusinessEvents Studio, select the project in the Studio Explorer panel, then select **Project > Build Enterprise Archive**.



You need to keep history value for the property in concept/event/scorecard to at least 2.

Also the history policy needs to be set to “All Values”.

2. If you want to build an EAR with debug information, check the **Generate Debug Info** checkbox (checked by default).



Step 2 is not required if you are working with test data. If you are using Rule Input data, follow the steps below.

3. In the File Location field provide the path to the EAR file and specify the EAR file name, for example, c:\myprojects\myproject.ear.
4. Generate the EAR file.

Create Test Data (as Desired)

You can create test data for use across multiple sessions, or you can provide rule input data while the engine is running. See [Creating and Working With Test Data, page 410](#).

For Remote Debugging Only, Configure Java Debug Interface (JDI)

To configure for remote debugging you configure the TIBCO BusinessEvents engine property file on the remote engine so the engine uses Java Debug Interface (JDI) for remote debugging.

You then configure a debug configuration for remote debugging, as explained in [Adding and Working with Launch \(Debug or Run\) Configurations on page 405](#). You must specify the same JDI port number in the TRA file and in the debug configuration.

To Configure Java Debug Interface (JDI)

For each TIBCO BusinessEvents engine you want to enable for remote debugging, do the following.

1. Open the *BE_HOME/bin/be-engine.tra* file for editing.
2. Specify the port on which you want the engine to listen, using the environment variable `tibco.env.JDI_PORT`, for example:

```
tibco.env.JDI_PORT 5192
```

Where `5192` is the default value. If multiple engines run on the same machine, ensure that each has a unique port.

3. Uncomment the following line:

```
-Xrunjdwp:transport=dt_socket,address=%JDI_PORT%,suspend=na,server=y
```

4. Start or restart the engine with `-d` or `-debug` command.

Adding and Working with Launch (Debug or Run) Configurations

Launch configurations are more specifically referred to as *run configurations* and *debug configurations*. All launch configurations have the same basic fields, but debug configurations have extra settings for remote debugging.



Before you begin Ensure that you know the EAR file location, the CDD file location, and the name of the processing unit (configured in the CDD file) that you want to use.

To Add and Work with Launch Configurations



For remote debugging you first have to set properties in the remote engine TRA file. See [For Remote Debugging Only, Configure Java Debug Interface \(JDI\) on page 403](#).

A reference to the settings is provided in [Launch Configurations Reference on page 408](#).

1. Do one of the following:
 - To create a debug configuration, click the down-arrow to the right of the debugger button () on the toolbar. From the drop-down list, and select **Debug Configurations**. Or, select **Run > Debug Configurations**.
 - To create a run configuration, click the down-arrow to the right of the Run button () on the toolbar. From the drop-down list, select **Run Configurations**. Or, select **Run > Run Configurations**.

You see the Debug Configurations or Run Configurations dialog.

2. Select an option from the list on the left:
 - For testing or local debugging, select **TIBCO BusinessEvents Application**
 - For remote debugging, select **Remote TIBCO BusinessEvents Application**.

3. Do one of the following depending on your needs:
 - To edit a configuration, expand TIBCO BusinessEvents Application or Remote TIBCO BusinessEvents and select an existing debug configuration.
 - To add a new configuration, click the **New Configuration** (📄+) button.
 - To duplicate a configuration, select the configuration and then click the **Duplicate** (📄) button. Modify, then save as a new configuration.
 - To delete a configuration, select the configuration and then click the **Delete** (✖) button.

When you add, edit or duplicate a configuration, Configuration fields appear in the right panel.

4. If you selected TIBCO BusinessEvents Application in [step 2](#), do the following:
 - Select the **Main** tab and configure values as explained in [Launch Configurations Reference on page 408](#).
 - Select the **Classpath** tab and configure the classpath for external libraries or custom functions as needed, for example if the project uses Rendezvous or JMS channels. See [Working with External Library and Custom Function Paths on page 10](#) for details.
 - Select the **Environment** tab and configure environment variables as needed, to run or debug the project in TIBCO BusinessEvents Studio. You can add new variables. You can select and then edit existing variables. You can append your edited variable to the existing environment variable, or you can replace the existing environment variable with it. For example if a custom function depends on a native library, you can add the path to that library using the `PATH`, `LD_LIBRARY_PATH`, `SHLIB_PATH`, or `LIBPATH` variable, as appropriate for your operating system.



On Linux platforms when TIBCO BusinessEvents DataGrid is used, you must set `LD_LIBRARY_PATH` to `AS_HOME/lib` in the Environment tab.

- For information on standard Eclipse features incorporated into this area, see Eclipse help.
5. If you selected Remote TIBCO BusinessEvents Application in [step 2](#), you must connect to the remote engine. Click the **Remote** tab and specify the host name or IP address and port the engine is running on. Use the same JDI port number in the TRA file and in the debug configuration (see [For Remote Debugging Only, Configure Java Debug Interface \(JDI\) on page 403](#)).
 6. Click **Apply** to save configuration settings.

7. Do one of the following:
 - Click **Close** and save the configuration you worked on.
 - For debug configurations, click **Debug** to save the configuration and launch the debugger. See [Setting Breakpoints and Running Debugger on page 413](#) for details.
 - For run configurations, click **Run** to save the configuration and run the engine.

You can then assert test data as desired in order to observe the effect of the data on the engine.

Launch Configurations Reference

See [Adding and Working with Launch \(Debug or Run\) Configurations on page 405](#) for the related procedure.



The Source and Common tabs are standard Eclipse dialogs. See Eclipse help for details on use of those tabs. If the project uses third party JARs, you must also reference them in the Classpath tab, and update the Environment as needed.

For Testing and Local Debugging

Field	Notes
Name	A descriptive name. It appears in the drop-down list of configurations.
Main Tab	
Project	Browse to select the name of the TIBCO BusinessEvents project to use for this configuration. You can select from projects in the workspace. The project currently selected in Studio Explorer appears by default.
VMArguments	Optional. Provide any options and parameters just as you would if you were starting the TIBCO BusinessEvents engine at the command-line.
CDD File Location	Browse to select the CDD file to be used for this launch configuration.
Processing Unit Name	Select the name of the processing unit (PU) whose values are used for this launch configuration. The drop-down list displays PUs available in the CDD specified in the CDD File Location setting.
Working Directory	<p>The location of the working directory for the TIBCO BusinessEvents engine. Used to store temporary files and logs. Browse to and select an existing directory.</p> <p>Path names that do not start with the root directory are assumed by the operating system to start from the working directory.</p>
EAR File	<p>Browse to select the EAR file to be used for this launch configuration.</p> <p>The EAR file must be generated with the Generate Debug Info option checked.</p>

For Remote Debugging

You can debug a running TIBCO BusinessEvents engine on the current machine or another machine

Field	Notes
Name	A descriptive name. It appears in the drop-down list of configurations.
Main Tab	
Project	Browse to select the name of the TIBCO BusinessEvents project to use for this configuration. You can select from projects in the workspace. The name of the TIBCO BusinessEvents project in the workspace. It must be the same as the project that is running remotely.
Remote Tab	
Remote Connection	Enter VM arguments for running the remote VM. Alternatively, add these to the <code>java.extended.properties</code> property in the remote application's runtime properties (TRA) file. Default value is: <code>-Xdebug</code> <code>-Xrunjdwp:transport=dt_socket,address=25192,suspend=n,server=y</code>
Host	The host name or IP address of the remote computer where you are running the TIBCO BusinessEvents engine. Default is <code>localhost</code> .
Port	Communication port for the debugger, on the remote machine. Default is <code>25192</code>

Creating and Working With Test Data

You can create data for assertion into the running engine's Rete network during testing and debugging.

You can create data while you are working in the debug perspective. You can also create and save test data ahead of time and save it in your project for later use.

You can create two kinds of data for use in testing using the following tabs:

- **Tester Data** This is concept and event instance data. You create it in the TIBCO BusinessEvents Development perspective, right-click an entity and select Create Test Data. You use it in the Debug perspective: select the Rule Input View > Tester Data tab.
- **Rule Data** This is data directly provided as rule input. It is created and edited in the Rule Input View > Rule Data tab (in the Debug perspective).

How you use test data depends on what aspect of a project you want to test or debug.

You can enter data in the Test Data editor for event payloads, and for concept, event and scorecard properties, including properties that are primitive types, array types, contained concepts, and reference concepts. If the concept, event or scorecard properties are associated with a domain model, then the test data gets populated with the values in the domain model. You can use global variables.

Working with Concept and Event Test Data

To Create Concept and Event Instance Test Data

1. In Studio Explorer, right-click an event or concept and click **Create Test Data**.

The Test Data editor appears showing the event or concept properties as column headers.

2. In the Test Data editor, click **Add** to add rows for new instances. You can add your own unique **extId** values to the test data input, as needed. You can also use global variables.

You can also remove existing rows by selecting one or more rows and clicking **Remove**.

3. Click **Save**. The entity's test data is saved to an XML file stored within the `TestData` folder in the project root.

The `/TestData` folder is the default location (see [Tester Preferences on page 457](#)).


To Edit Test Data for Concepts and Events in Studio Explorer

To add more test data or edit test data you created earlier, do the following.

1. In Studio Explorer, expand the `TestData` folder in the root of the project.
The `/TestData` folder is the default location (see [Tester Preferences on page 457](#)).
2. Drill down to the test data you want to edit. The folder structure matches the project's event and concept folder structure.
3. Double-click the name of the test data file you want to edit. The test data editor opens. Add, remove, and edit rows of test data as desired.

Test data filenames use the format `eventName.eventtestdata` and `conceptName.concepttestdata`.


To Edit Concept and Instance Test Data in the Rule Input View

1. Open the Rule Input view, if it is not already shown. To make the Rule Input View visible, do one of the following:
 - Select the Debug perspective: Select **Window > Open Perspective**, or click the Open Perspective () button). Then select **Other > Debug**. The views associated with the Debug perspective open.
 - In the **Window** menu, click **Show View > Other**. Expand **TIBCO BusinessEvents**, and select **Rule Input**. Click **OK**.
2. Click the **Rule Input** tab and then select the **Tester Data** bottom tab.
For each concept or event for which you created test data, you see one row showing the project path to that concept or event.
3. Double-click the row for the concept or event whose test data you want to edit.

The Test Data editor appears, showing the rows of test data already created.

Working with Rule Data

To Create and Save Rule Data

1. Open the Rule Input view, if it is not already shown. To make the Rule Input View visible, do one of the following, as needed:
 - Select the Debug perspective: Select **Window > Open Perspective**, or click the Open Perspective () button). Then select **Other > Debug**. The views associated with the Debug perspective open.
 - In the **Window** menu, click **Show View > Other**. Expand **TIBCO BusinessEvents**, and select **Rule Input**. Click **OK**.
2. Provide input from the mapper as explained in [Using the Function Argument Mapper on page 234](#).
3. Specify the Launch Target. This is generally the locally running engine. The other fields become active.
4. Specify an event or concept in the Entity URI field. If you specify an event, then specify the Destination URI. Specify the Rule Session (agent) to use.
5. Do one of the following:
 - Click **Save** to save the input values.
You can reuse these saved values for repetitive tests.
 - Click **Load** to load the input values from an existing XML file.
 - Click **Assert** to assert the data to the running engine.

Setting Breakpoints and Running Debugger

After you have done the setup (see [Preparing to Run \(Test\) or Debug a Project on page 403](#)), add breakpoints in your code and run the debugger as explained below.

To Add Breakpoints in the Rule and Rule Function Code

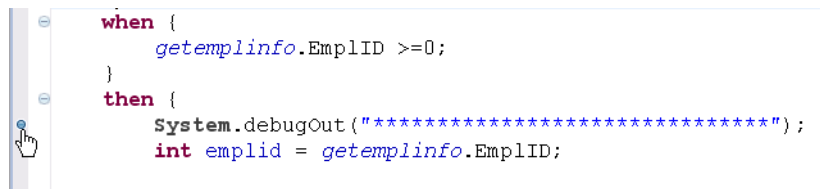
Setting breakpoints is an Eclipse feature. This section provides only basic information. You can also use advanced features such as importing and exporting breakpoints, and using class prepare breakpoints. See Eclipse help for more details on all breakpoint functionality. You can set or change breakpoints during a debug session also.

1. In TIBCO BusinessEvents Studio, open the source editor for a rule or rule function.

You can work with breakpoints in the debugger perspective as well as in the TIBCO BusinessEvents Studio development perspective.

2. To add a breakpoint put your cursor in the left margin (gray area) next to a row where you want to add a breakpoint. Do one of the following:
 - Right click and select Toggle Breakpoint.
 - Double-click in the left margin.

A breakpoint icon appears in the left margin:




3. Set and adjust breakpoints as needed. Select a break point, right click, and do any of the following:
 - To disable a breakpoint, select Disable.
 - To remove a breakpoint, select Toggle Breakpoint. (Or just double click the breakpoint.)
 - To edit a breakpoint's properties, select Breakpoint properties. A dialog displays (with mostly runtime options). For example, you can use a class prepare breakpoint (so the running program is suspended when the specified class or interface is first loaded by the Java VM).

A breakpoint may not be set exactly where you place it. This is because TIBCO BusinessEvents ensures that breakpoints fall on an executable statement, and moves any that don't to the nearest executable statement.





Executing step code line does not match rule editor code line This situation happens when there is a mismatch between the debug line information stored in the EAR, and the information in the open rule editor. To resolve the problem, recompile the EAR file.

To Run Debugger

1. As needed, switch to Debug perspective. Select **Window > Open Perspective**, or click the Open Perspective () button). Then select **Other > Debug**.

Alternatively, wait till TIBCO BusinessEvents prompts you to change to debug perspective. This happens when the debugger reaches the first breakpoint.

2. Launch a configuration: Click the down-arrow to the right of the debugger () button. You see a drop-down list. Do one of the following:
 - Select a debug configuration from the list. (To add configurations to this list, select Organize Favorites from the debugger drop-down list.)
 - Select Debug Configurations. At the Debug Configurations dialog select a debug configuration and click **Debug**.
 - Click the debugger () button or Select **Run > Debug**. (Only if you have already launched debugger with a configuration.)

Debugger starts a TIBCO BusinessEvents engine, using parameters provided in the launch configuration, if any were provided.

3. As appropriate, generate messages to trigger rules using one of the following methods:
 - Send messages to the channels in the usual way (for example from a Enterprise Message Service server)
 - Generate rule input data internally using the Tester Data and Rule Data tab. See [Creating and Working With Test Data on page 410](#).
4. Use the standard Eclipse commands such as step into (F5), step over (F6), step return, step return, and so on, depending on the level of detail you want to examine.



See the options in the Run menu and in the Breakpoints tab for more options, and use Eclipse help for details.


Running Tester

After you have done the setup (see [Preparing to Run \(Test\) or Debug a Project on page 403](#)) and created the test data (see [Creating and Working With Test Data on page 410](#)), run the tester as explained below.

Running the Engine


To Run an Engine

1. As needed, switch to Debug perspective. Select **Window > Open Perspective**, or click the Open Perspective () button). Then select **Other > Debug**.
2. Select **Run > Run Configurations** or click the down-arrow to the right of the Run () button and choose **Run Configurations**.

If you have configured favorites, you can click the down-arrow to the right of the Run () button and choose a favorite.

3. At the Run Configurations dialog select a run configuration and click **Run**.



If you have already started an engine using a run configuration and want to start it again, click the Run () button or Select **Run > Run**.

A TIBCO BusinessEvents engine starts, using parameters provided in the run configuration, if any were provided.

Asserting Rule Input Data



To perform any task in the **Rule Input** tab, you must keep the engine running.

See also [Creating and Working With Test Data on page 410](#) for details about creating and saving rule input data.

You can assert either tester data, or rule data.

To Assert Tester Data

1. Select the Rule Input tab and then the Tester Data bottom tab.
2. In the **Select Test Data** panel, select sets of test data as desired.

In the test data editor you can also select which test data rows you want to assert.

You can also edit the test data associated with each event and concept as needed. See [Working with Concept and Event Test Data on page 410](#) for details. Save the file and close the editor when you have finished editing.

3. In the Input panel, Launch Target field, specify which engine's agents you want to assert the data to.

Multiple engines can run in tester at the same time, for example, a cache agent engine, and an engine running inference agents.

4. In the Input panel, Rule Session field, specify which agent to assert the data to. One engine (processing unit) can have multiple inference agents.
5. Click **Start Test**.

To Run the Tester with Rule Data

1. Open the Rule Input view, if it is not already shown. In the **Window** menu, click **Show View > Other**. Expand **TIBCO BusinessEvents**, and select **Rule Input**. Click **OK**.
2. Do one of the following:
 - Provide input from the mapper as explained in [Using the Function Argument Mapper on page 234](#). Optionally you can click Save and save the values to an XML file.
 - Click **Load** to load the input values from an existing XML file.
3. Specify the **Launch Target**, **Entity URI**, **Destination URI** and **Rule Session**.
4. Click **Assert**.

See [Viewing the Results on page 417](#) for details about understanding the results of a test run.

Viewing the Results

After asserting the tester data or rule data (see [Running Tester on page 415](#)), the following occurs:

- The Console tab displays engine console messages.
- The result data editor appears showing the results of the run. The editor title displays the result data filename with the format `Run-n.resultdata`.
- The results of this test are stored in an XML format in a `.resultdata` file in the `/TestData/Project Name/Processing Unit Name` folder.

This is the default location. See [Tester Preferences on page 457](#) for information on changing the location where test data is stored.

You can also open the result data in its editor by double-clicking the `.resultdata` file in Studio Explorer.

Test Result: FraudDetection/default/Run-1

Created

- /Concepts/Account{id = 5}
- /Rules/ProcessDebits/CreateAccount [Invoked]
 - /Concepts/Account{id = 4} [Causal]

Modified

- /Concepts/Account{id = 8}
- /Rules/ProcessDebits/ApplyDebit [Invoked]
 - /Concepts/Account{id = 8} [Causal]
 - /Events/Debit{id = 14} [Causal]

Deleted

- /Events/Debit{id = 14}

Result Test Data

After

Account @id	Account @Extid	Balance @	Debits @	Status @	AvgMonthlyBal... @
8	ActB	27000.0	3000.0	Normal	15000.0

Before

Account @id	Account @Extid	Balance @	Debits @	Status @	AvgMonthlyBal... @
8	ActB	30000.0	0.0	Normal	15000.0

Understanding Result Data

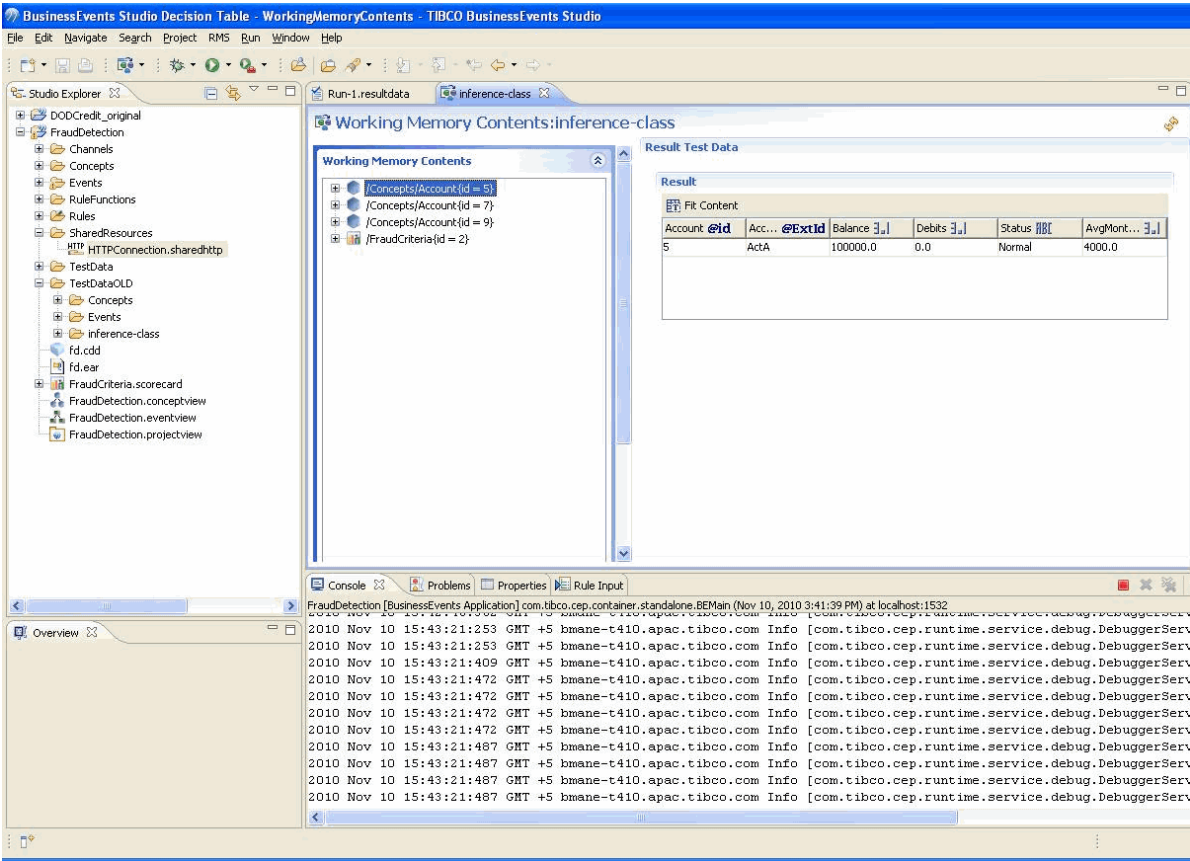
The Result Data editor shows the **Created**, **Modified**, and **Deleted** entities on the left side.

Expanding each entity shows the rule that affected the entity, and also the causal object that triggered the rule.

Click any entity to display its test data in the **Test Data** field, and **Result** displays the results of the test run. The **Test Data** field displays the data entered by the user. The **Result** field displays the result of the test run on that entity.

Viewing and Understanding Working Memory Contents

The working memory contents list all the entities available in the working memory at any instance whether or not they are affected by the test run. You can refresh the working memory by clicking the Refresh (🔄) button on the top-right corner of the editor.



To see the Working Memory Contents

1. Click the Working Memory Contents icon on the tool bar, or from the **Project** menu.
2. Select the appropriate rule session.

The number of objects in the working memory contents is specified in the Preferences. For more details, see [Tester Preferences on page 457](#).

Chapter 24 **Diagrams**

This chapter covers TIBCO BusinessEvents diagrams, including a summary of the diagrams available, how to access and work them, and what options and preferences are available for working with them.

Topics

- [Overview of Diagrams, page 422](#)
- [Working with Diagrams, page 424](#)
- [Project Analyzer and Selected Entity Project Diagrams, page 432](#)
- [Dependency Diagrams, page 436](#)
- [Sequence Diagrams, page 438](#)
- [Concept Model Diagrams, page 439](#)
- [Event Model Diagrams, page 440](#)
- [Diagram Options and Tools Reference, page 441](#)

Overview of Diagrams

TIBCO BusinessEvents provides several kinds of diagrams, which are visualization tools that help you to understand and analyze even very large and complex projects.



The diagrams in TIBCO BusinessEvents are based on Unified Modeling Language (UML), but are NOT completely compliant to UML.

The diagrams allow you to show or hide details. You can open editors for any project component in a diagram. You can also create snapshot images of diagrams to share project information with other personnel.

The main types of diagrams, and the project elements that use them, are as follows:

Selected Entity Project diagrams These diagrams display all the selected project resources, and how they interact with one another. External resources such as XSD and WSDL files are not shown. You can either create a project diagram for an entire project, or only for the selected resources of a project. You can also choose to run the project analyzer while creating a project diagram, which analyzes the project resources and gives a report in the Problems view.

Dependency diagrams These diagrams show the relationships between the selected project resource and its dependent resources. They are available for channels, concepts, state models, scorecards, events (all types except advisory events), rule functions, and rules.

Sequence diagrams These diagrams show how project resources are called into use at runtime. They are available for events (all types except advisory events), rule functions, and rules.

Concept model diagrams These diagrams show the concept model for a project. You can view a model diagram for all the concepts in a project.

Event model diagrams These diagrams show the event model for a project. You can view a model diagram for all the events in a project, except for advisory events.

State model diagrams are available in the TIBCO BusinessEvents Data Modeling add-on. Metrics are available in the TIBCO BusinessEvents Views add-on. See the add-on product documentation for more details.

Table 45 Types of diagrams available for each type of project element

Project Resource	Project	Dependency	Sequence	Concept Model	Event Model
Channel	Yes	Yes			
Concept	Yes	Yes		Yes	
Domain	Yes	Yes			
State Model	Yes	Yes			
Scorecard	Yes	Yes			
Event	Yes	Yes	Yes		Yes
Rule Function	Yes	Yes	Yes		
Rule	Yes	Yes	Yes		
Shared Resource	Yes	Yes			
Metric	Yes	Yes			

Working with Diagrams

You can work with diagrams in a variety of ways for your own information, and you can use them to share aspects of your project with others in a visual way that is easy to understand. This section presents some common tasks and lists the various tools you can use to work with all types of diagrams.

Configuring Diagram Preferences and Properties

Diagram Preferences

See [Diagram Preferences on page 452](#) to understand and set preferences so that diagrams display as you desire.

Diagram Properties

Use the following properties to handle diagrams that are too complex to display in a timely manner. You can adjust the values in the

BE_HOME/studio/eclipse/configuration/studio.tra file to define upper limits:

```
#Diagram properties
be.studio.project.diagram.nrEdges=3000
be.studio.project.diagram.nrMaxEdges=1500
```





If the number of edges in the diagram exceeds the limit defined by `be.studio.project.diagram.nrEdges`, then the usage links in the diagram are hidden. Usage links are links from a rule or rule function to other rules or rule functions it uses.

If the number of edges in the diagram exceeds the limit defined by `be.studio.project.diagram.nrMaxEdges`, then symmetric layout is run as default rather than hierarchical because symmetric layout is faster and less expensive.

Different Ways to Create Diagrams

In TIBCO BusinessEvents Studio you can create diagrams in three ways. The [Table 46, Creating Diagrams](#) lists these three ways and the diagrams you can create using them.


Table 46 Creating Diagrams

How to create a diagram	Diagrams you can create
In Studio Explorer, right-click a resource or a project, and select the appropriate menu option	Selected Entity Project Diagram, Concept Model Diagram, Event Model Diagram
In Studio Explorer, select a resource and right-click	Selected Entity Project Diagram, Concept Model Diagram, Event Model Diagram, Dependency diagram
In a resource editor, click the appropriate diagram button in the top right corner Concept Model Diagram  Event Model Diagram  Dependency Diagram  Sequence diagram 	Concept Model Diagram, Event Model Diagram, Dependency Diagram, Sequence Diagram
In a Selected Entity Project diagram, right-click a resource	Dependency Diagram
In a Selected Entity Project diagram, right-click anywhere on the canvas	Selected Entity Project diagram

Performing Common Tasks

You can perform the following common tasks on all types of diagrams:

- Hover the mouse over a resource to view a tooltip with details
- Double-click any resource represented in the diagram to open its editor
- Drag resources to any location in order to change the layout as desired
- Drag any link to change the layout as desired

- Click any arrowhead to move the arrow to any other element in the project. When you hover the mouse over an arrowhead, you see a small graphic on it. ()
- Double-click the black and white plus signs in a dependency or sequence diagram to unfold the next level of dependency

Using Diagram Tools

You can work with diagrams with the help of the following tools available from the Diagram menu:

- [Interactive Tools](#), such as Select, Pan, Magnify, Marquee Zoom, Interactive Zoom, Link Navigator
- [Layout](#), such as Default Layout, Circular Layout, Orthogonal Layout, Symmetric Layout, Rectilinear Hierarchical Layout, and Oblique Hierarchical Layout
- Print tools: Print Setup, Print Preview, Print
- Export to Image
- Labeling
- Link Routing
- Incremental Layout
- Fit In Window

The above tools are also available on the toolbar when you create or display a diagram. The toolbar also has three additional tools listed below:

- Refresh Diagram
- Zoom Percentage
- Search Diagram Entities

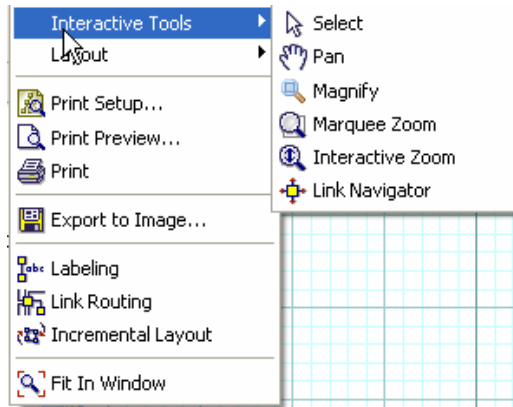


Diagram menu and the toolbar described in this section are available only when you create or open a diagram.

For more details about the diagram tools, see [Diagram Options and Tools Reference on page 441](#).

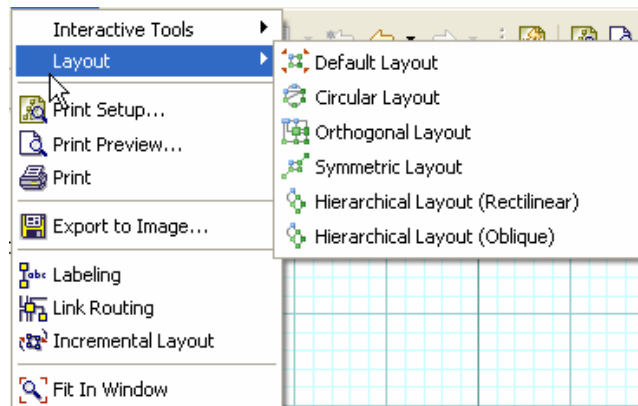
Interactive Tools

To access the Interactive Tools, select Interactive Tools from the Diagram menu. They are also available on the toolbar when you create or display a diagram. Interactive Tools let you select, pan, and magnify diagrams. They also include tools such as marquee zoom, interactive zoom, and link navigator. For more details about these tools, see [Diagram Options and Tools Reference on page 441](#).



Layout

You can access the Layout menu from the Diagram menu. They are also available on the toolbar when you create or display a diagram. Layout options let you change the layout of a diagram to Circular, Orthogonal, Symmetric, or Hierarchical. For more details about these tools, see [Diagram Options and Tools Reference on page 441](#).



Context Menu Diagram Tools

Diagram tools are available as context menus for diagram canvas and objects. [Table 47, Context Menu Options for Canvas and Objects](#) lists these context menu options.

Table 47 Context Menu Options for Canvas and Objects

Context Menu Option	Available for
Selected Entity Project Diagram	Canvas – Selected Entity Project Diagram
Export to Image...	Canvas – All diagrams
Print Setup...	Canvas – All diagrams
Print Preview...	Canvas – All diagrams
Fit In Window	Canvas – All diagrams
Create Dependency Diagram	Object – Selected Entity Project Diagram
Fold One Level	Object – Selected Entity Project Diagram
Fold N Levels...	Object – Selected Entity Project Diagram
Fold All Levels	Object – Selected Entity Project Diagram
Edit	Object – All diagrams

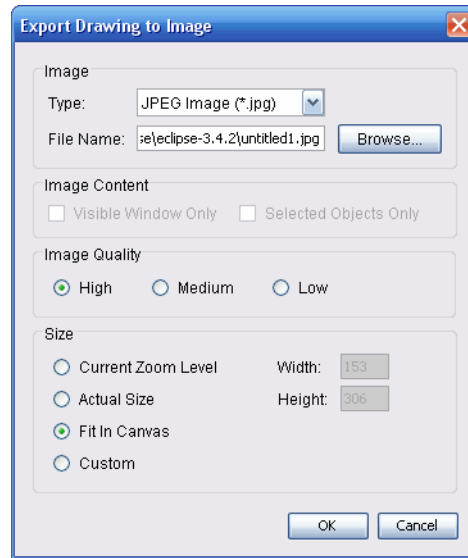
For more details about the diagram tools, see [Diagram Options and Tools Reference on page 441](#).

Exporting a Diagram to an Image

The Export to Image option enables you to save the diagrams locally for later use.

To Export a Drawing (Diagram) to Image

1. From the top menu, select **Diagram > Export to Image**. You see the Export Drawing to Image dialog.



2. Select the type of the image from the Type list.
3. In the File Name field, specify the file name along with the location.
4. Check the check boxes Visible Window Only and Selected Objects Only as appropriate to specify the content of the image.



Selected Objects Only gets enabled only if you select an object in the diagram.

5. Specify the Image Quality as appropriate.
6. Specify the size, and click OK.

The diagram is exported to an image, and saved on the system.

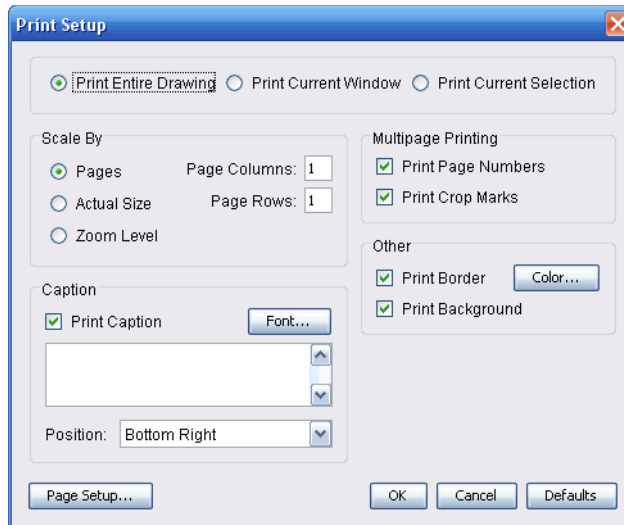
Printing a Diagram

You can set up preferences, preview the print job, and then print the diagram.

To Set up Printing Preferences

The Print Setup dialog enables you to specify the print setup for a diagram.

1. Display the diagram, and from the top menu select **Diagram > Print Setup**.



2. Select an appropriate option to print the entire diagram, or to print only the current window, or to print the selection.
3. In the **Scale By** section, select an option to scale the diagram by pages, by actual size or by zoom level.
4. If you want multiple prints, you can specify if you want the page numbers and crop marks to be printed, by clicking to select the respective checkboxes.
5. If you want a caption for your diagram, you can type a caption in the text box that appears on enabling the **Print Caption** checkbox. You can also select a Font for this caption, and its position on the page.
6. If you want to have borders for your diagrams, click to select the **Print Border** checkbox. Click **Color...** to specify the color of the border.
7. If you want to print the background of the diagram, click to select the **Print Background** checkbox.
8. Use **Page Setup** to specify the size, orientation, and other page properties.
9. If you want to reset all the options to the defaults, click **Defaults**.
10. Click **OK**.

To Preview and Print a Diagram

1. From the top menu, select **Diagram > Print Preview**. An image of the diagram to be printed displays.
2. From the top menu select **Diagram > Print**.

Setting Diagram Preferences

You can set the preferences for all types of diagrams.

To Set Diagram Preferences

For more information on Preferences, see [Chapter 25, TIBCO BusinessEvents Studio Preferences, on page 445](#).

1. From the top menu, select **Window > Preferences**.
2. In the Preferences dialog, expand **TIBCO BusinessEvents**, and then expand **Diagram**.
3. Select any type of diagram.

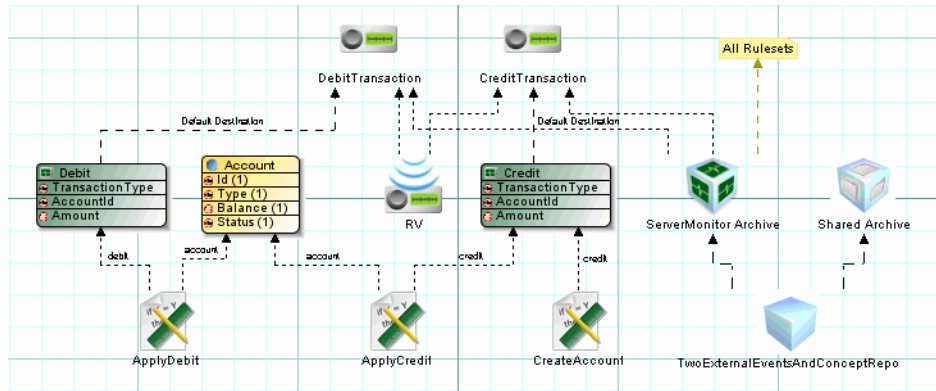
On the right side, you see preferences for the selected diagram type.

4. Edit the preferences, and click **Apply**.

OR

To set all preferences to their default values, click **Restore Defaults**.

Project Analyzer and Selected Entity Project Diagrams



Project Analyzer and the Selected Entity Project diagram work together to provide insights into your project. Project Analyzer is a document generated in the Problems view area. Selected Entity Project diagram provides a visualization of the whole project or of selected project elements, including dependencies. Selected Entity Project diagrams can display all the resources in a project, how they interact with one another, and how they use or update other resources at run time. You can see concepts, events, rules, rule functions, channels, destinations, score cards, shared resources, and all other resources of a particular project in a Selected Entity Project diagram. It also shows all dependencies in a project.

You can set preferences to determine whether project analyzer runs when you create a Selected Entity Project diagram or not. You can use a menu option to create the project analyzer report without creating the Selected Entity Project diagram.

Types of Links

A Selected Entity Project diagram has three types of links:

- Links, represented by continuous lines, indicate inheritance or containment
- Scope Links, represented by dashed lines, indicate resources in the scope
- Usage Links, represented by dashed lines with dots, indicate resources that can be used

Advantages of Project Analyzer and Selected Entity Project Diagrams

Using Project Analyzer and Selected Entity Project Diagrams together helps you understand your project in these ways:

- Helps you understand how all the selected project resources are connected and how they interact
- Analyzes rules and how they modify ontology
- Lets you search entities in a diagram
- Lets you choose the entities to be displayed in a diagram by using Project Filters
- Displays statistics of project resources (number of concepts, events, state machines, and so on) in the properties view
- Enables you to share project design, logic, deployments with others in a format they can read without having to have TIBCO BusinessEvents Studio, as well as print the entire project according to page setup

The Project Analyzer report helps you analyze the project by performing the following tasks:

- Finds rules that can never be fired
- Finds rule functions that are never used
- Finds events that are never used/may never be fired
- Finds destinations with no default events
- Finds domain models that are never used, or that are not associated with entity properties
- Finds state models that are orphaned, or that are not associated with any concepts

Working with Project Analyzer and Selected Entity Project Diagrams

You can show the diagram for an entire project or for selected elements only. You can also filter the diagram to show only certain types of project element.



You can choose whether to run Project Analyzer whenever you create a Selected Entity Project diagram. To do this, from the **Window** menu, click **Preferences**, expand **TIBCO BusinessEvents**, expand **Diagrams**, and then click **Project**. Check the **Run Analysis When Creating View** checkbox on the right side.

You can also run Project Analyzer separately in either case. See [Project Analyzer and Selected Entity Project Diagrams](#), page 432.

To Run Project Analyzer Without Selected Entity Project Diagram

Right-click the top level project folder, and click **Analyze**. Project Analyzer displays the report in the Problems view.

To Create a Selected Entity Project Diagram

1. Do one of the following to create a Selected Entity Project diagram for the entire project:
 - In Studio Explorer, right-click a project resource and select **Create Selected Entity Project Diagram**.
 - Double-click the `*.projectview` file in the Studio Explorer.
- OR, to create a Selected Entity Project diagram for selected resources
- In Studio Explorer, select resources of interest, right-click, and select **Create Selected Entity Project Diagram**.



When you select a folder all elements in that folder are selected.

The Selected Entity Project diagram appears in the editor area.

The diagram tab label reflects the project name and the type of diagram:
`projectName.projectview`, for example, `ExternalEventRepo.projectview`.

By default, the Selected Entity Project diagram displays the selected objects and their immediate dependencies. You can change the depth of dependencies shown using preferences.

To Filter a Selected Entity Project Diagram

To filter what you see, check and uncheck the options in the Project Filter list within the Palette view, and click **Apply**.

Options show or hide elements of the specified type, or group them:

Show Options Show Concepts, Show Events, Show Decision Tables, Show Domain Model, Show State Models, Show Archives, Show Rules, Show Rule Functions, Show Scorecard, Show Channels, Show Scope Links, Show Usage Links, Show Archived Destinations, Show Archived Rules, Show Archived Rules (All), Show Rules in Folders, Show Tooltips. If add-on products are used additional options may appear.

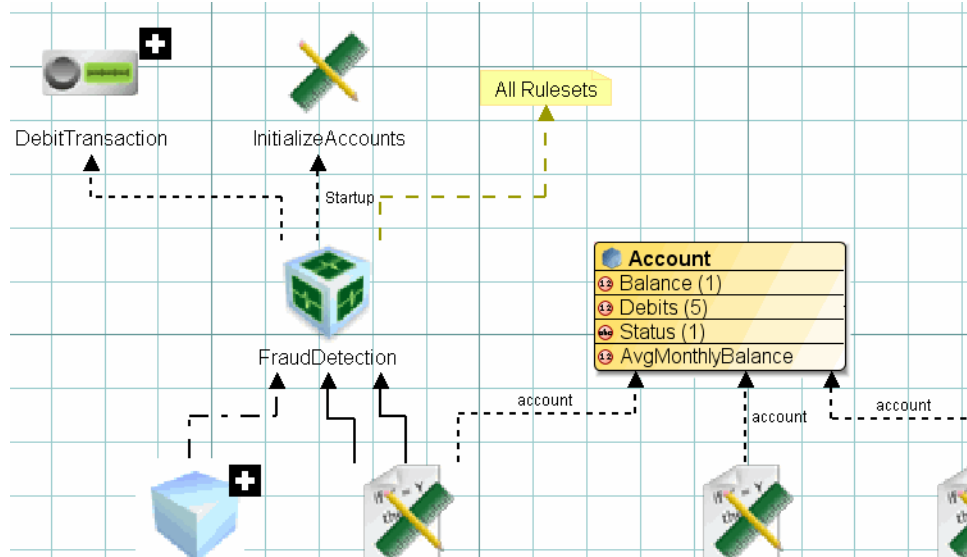
Group Options Group Concepts, Group Events, Group Rules, Group Rule Functions

The filter options in the Palette view are loaded from diagram preferences. See [Diagram Preferences on page 452](#) for more details.

Dependency Diagrams


A dependency diagram shows the dependencies for a selected project resource or for the entire project (Selected Entity Project Diagram).

When two resources are dependent and one changes, it can cause changes to the other resource as well. You can control how many levels of dependencies to view in a dependency diagram - one, two, or all levels.



Rule actions are not represented in rule dependency diagrams. Use Debugger to analyze rule actions. See [Chapter 23, Testing and Debugging Projects, on page 399](#).

To Create a Dependency Diagram

1. Do one of the following:
 - In TIBCO BusinessEvents Studio Explorer, right-click a project resource and select **Create Dependency Diagram**.
 - Open the project element for editing and click the Dependency Diagram () button in the top right of the editor.
 - In a Selected Entity Project diagram, right-click a resource and select **Create Dependency Diagram**.

The dependency diagram appears in the editor area. By default, one level of dependency is shown (showing only direct dependents). White crosses

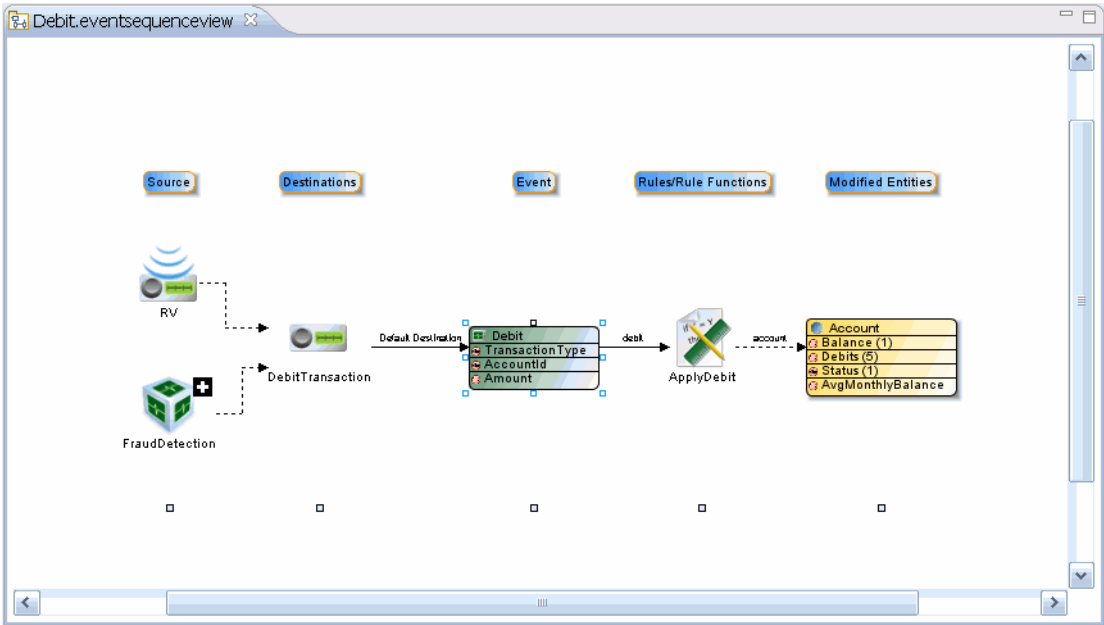
indicate hidden dependencies. The default appearance is also driven by the diagram preferences. See [Diagram Preferences on page 452](#) for more details.

The diagram tab label reflects the item being graphed and the type of diagram: *itemName.item-type*`dependencyview`, for example, `Transaction.eventdependencyview`.


2. As desired, select a dependency level from the palette:
 - One: Shows the direct dependencies of the selected item
 - Two: Shows the direct dependencies of the selected item as well as the direct dependencies of its direct dependencies
 - All: Shows all dependency relationships that involve the selected item

Sequence Diagrams

Sequence diagrams capture the behavior of objects and the messages that are passed between them. You can view sequence diagrams for rule functions, rules, and events.



To Create a Sequence Diagram

Open an event, rule function, or rule for editing and click the Sequence diagram () button in the top right of the editor.

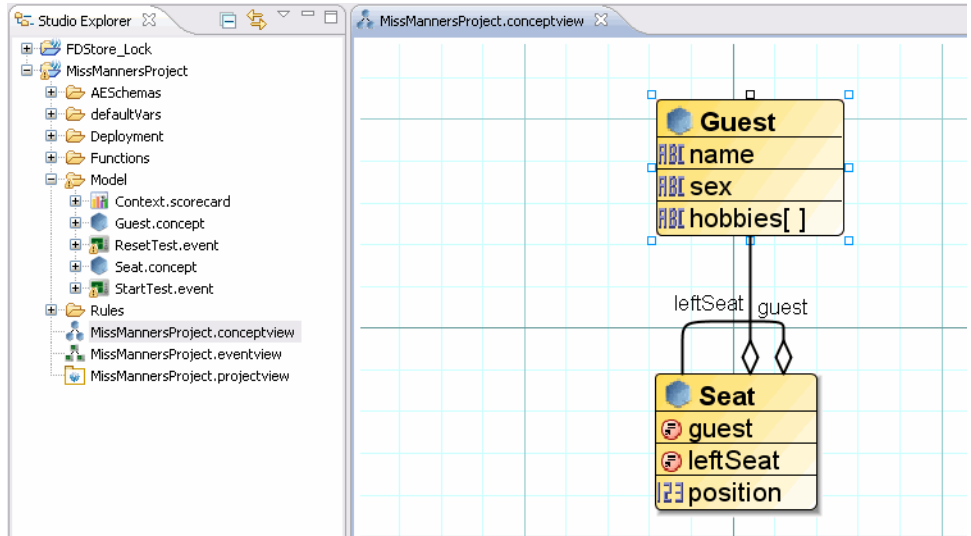
The sequence diagram for the item appears in the editor area.

The diagram tab label reflects the item being graphed and the type of diagram: *itemName.item-type*sequenceview. For example, Transaction.eventsequenceview.

Concept Model Diagrams


The concept model diagrams show inheritance, containment, and reference relationships between all concepts in a project.

Concept model diagrams are created fresh each time you view them (they do not persist on disk).



To Create a Concept Model Diagram

Do one of the following:

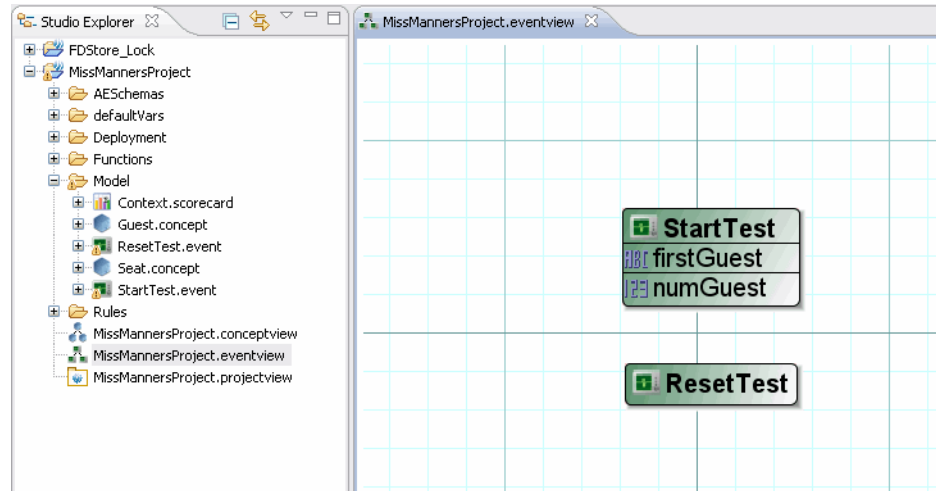
- Right-click a resource or a project in TIBCO BusinessEvents Studio Explorer and select **Create Concept Model Diagram**
- Open a concept editor and click the Concept Model Diagram () button in the top right of the editor
- Double-click the *.conceptview file in the Studio Explorer.

The concept model diagram for that project appears, showing only its direct dependents (Dependency level One).

Event Model Diagrams

The event model diagrams show inheritance relationships between all types of events, except for advisory events.

Event model diagrams are created fresh each time you view them (they do not persist on disk).



To Create an Event Model Diagram

Do one of the following:


- Right-click a resource or a project in TIBCO BusinessEvents Studio Explorer and select **Create Event Model Diagram**.
- Open an event editor and click the Event Model Diagram () button in the top right of the editor. The event model diagram for that project appears, showing only its direct dependents (Dependency level One).
- Double-click the *.eventview file in the Studio Explorer.

Diagram Options and Tools Reference

Most diagram options and tools are self-explanatory, and are just shown in the images. This section provides details for a few that require some explanation.

Table 48 TIBCO BusinessEvents Diagram Tools Reference






















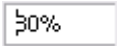

Button	Description
Interactive Tools Menu Options and Toolbar Buttons	
	Select (Pointer) tool.
	Pan tool.
	Magnify Select Magnify and hover over the diagram to display a magnified section of the diagram.
	Zoom Drag the cursor diagonally to define the area to which you want to zoom.
	Interactive Zoom Drag the cursor up to zoom out. Drag the cursor down to zoom in.
	Link Navigator Shows the path of a link using a simple animation.
Layout Menu Options and Toolbar Buttons	
	Default Layout Returns the diagram to the default layout. To display the layout list, click the arrow next to the button and select a new layout from the list of options.
	Circular Layout It is available in the layout list. See Circular Layout on page 443 .
	Orthogonal Layout It is available in the layout list. See Orthogonal Layout on page 443
	Symmetric Layout It is available in the layout list. See Symmetric Layout on page 443 .
	Hierarchical Layout (Rectilinear) It is available in the layout list. See Hierarchical Layout on page 444 .

Table 48 TIBCO BusinessEvents Diagram Tools Reference

Button	Description
	Hierarchical Layout (Oblique) It is available in the layout list. See Hierarchical Layout on page 444 .
Other Diagram Menu Options and Toolbar Buttons	
	Print Setup. Sets options for printing the diagram or elements of the diagram. See Printing a Diagram on page 429 .
	Print Preview. Lets you see how the printed diagram would look.
	Print. Prints the diagram.
	Export to Image. Exports the diagram to one of the five available image file formats, and lets you save the image file locally for future reference. See Exporting a Diagram to an Image on page 428 .
	Labeling Prevents labels from overlapping with other elements and labels in the diagram.
	Link Routing Redraws the diagram, attempting to redraw only the links, leaving resource nodes in the same size and position where possible. The behavior of link routing is affected by the preference options Fix Node Size and Fix Node Position. See Diagram Preferences on page 452 for details.
	Incremental Layout See Incremental Layout on page 444 .
	Fit In Window Zooms the diagram to fit the size of the current diagram editor view.
Additional Toolbar Buttons	
	Refresh Diagram. If you make changes in the project, clicking Refresh Diagram updates the diagram with the changes.
	Zoom Percentage Zooms the diagram to the specified percentage.
	Search Entities. Allows you to search for TIBCO BusinessEvents entities in the diagram.

Layout Options

Layout options refer to the style of the diagram that is rendered in the diagram editor, Rule Debugger, and Dependency panel.



Additional Layout Options

Various options affect the appearance of a diagram, no matter what layout is chosen. (The hierarchical layout offers additional options.) See [Diagram Preferences on page 452](#) for more details.

The main layout options are:

- [Circular Layout, page 443](#)
- [Orthogonal Layout, page 443](#)
- [Symmetric Layout, page 443](#)
- [Hierarchical Layout, page 444](#) (default)
- [Incremental Layout, page 444](#)

Circular Layout

This layout is useful for ontologies where nodes tend to have a clustered (ring or star) structure (where each main node has a starburst of related nodes).

Orthogonal Layout

The orthogonal layout style uses only horizontal and vertical links. Diagrams are drawn quickly.

The algorithm places highly connected nodes closer together, resulting in a more compact diagram. Even when lines overlap, the algorithm ensures that they are still easy to follow. Because this style has no hierarchical or other visual constraints, the resulting diagrams are often very clear.

Symmetric Layout

The symmetric layout style looks for and emphasizes the symmetries in a project topology. It can produce a pleasing visual result, if there are no reasons to arrange the nodes. For example, there is no hierarchy or ring-clustering inherent in the structure of the nodes.

See [Configuring Diagram Preferences and Properties on page 424](#) for a property that affects whether symmetric or hierarchical layout is used as the default, depending on number of edges.

Hierarchical Layout

The hierarchical layout style indicates dependencies by positioning the nodes at different levels. The hierarchical layout style is useful when you need to show precedence relationships in the ontology.

Hierarchical layout is the default. However, see [Configuring Diagram Preferences and Properties on page 424](#) for a property that affects whether symmetric or hierarchical layout is used as the default, depending on number of edges. You can use Layout Preferences to determine in the direction of the hierarchy.

Preferences and options specific to the hierarchical layout are:

- Orientation Options (Edit > Preferences)—Left to Right, Top to Bottom, Right to Left, Bottom to Top
- Routing Options (View > Layout)—Orthogonal or normal (polyline) routing
- Routing Options (Edit > Preferences)—Orthogonal or polyline routing.

See [Diagram Preferences on page 452](#) for more details.


Incremental Layout

Choose Incremental Layout if you want the main arrangement of the diagram to remain stable when you make changes to your project and only re-route the entities and links that have changed since the last rendering.

Keeping most things in the same place makes it easier to see how changes you have made in the project affect the diagram.

If you want to refresh the entire diagram, click the desired layout option. The program has greater freedom to optimize the layout. The result is likely to be a more pleasing arrangement.

To perform an incremental layout update, do one of the following:

- Select View > Layout > Incremental Layout
- Click the Incremental Layout button  on the toolbar

Chapter 25 **TIBCO BusinessEvents Studio Preferences**

This chapter shows the preferences available in TIBCO BusinessEvents Studio.

Topics

- [Setting Preferences, page 446](#)
- [Decision Table Related Preferences, page 448](#)
- [Diagram Preferences, page 452](#)
- [Tester Preferences, page 457](#)

Setting Preferences

To access preferences select **Window > Preferences** and expand **TIBCO BusinessEvents**. Also see [Setting Diagram Preferences on page 431](#) for a more detailed procedure.

To Set Preferences:

1. From the Window menu, select **Preferences**.
2. Expand the options on the left and select each option to view a panel of options on the right.

Details on each set of options are provided in sections following.

3. Change the options from the list as desired, and then click **Apply**.
4. Click **OK** to see Decision Manager with your preferred settings.

TIBCO BusinessEvents

At the top level two preferences allow you do the following:

- Show or hide tooltip (hover) information in Catalog Functions view.
- Show or hide the confirm open perspective pop-up dialog.

Code Generation Shared Archive Ignored Resources Preferences

Located under TIBCO BusinessEvents > Code Generation > Shared Archive > Ignored Resources. These preferences define what file types are excluded from the EAR file shared archive section at build time.

Compare/Merge, Decision Table, and Decision Tree

Compare/Merge preferences are used for comparing two decision tables. See XX for details.

Decision Tree preferences are not supported in this release.

Diagram Preferences

In addition to a global preferences section, there are sections for each type of diagram. See [Diagram Preferences on page 452](#) for details.

Printing Preferences

These preferences apply to printing of diagrams, windows, and selections.

Error/Warning Preferences

You can enable and disable error and warning messages for each type of resource in a project. Resource extensions are shown so you can select which ones to enable or disable. By default all are checked (enabled). If you make changes, a prompt appears asking if you want to rebuild the project.

Rules Preferences and Rule Debug Preferences

In addition to the standard Eclipse Run/Debug preferences, TIBCO BusinessEvents requires some rule debug preferences.

Rules Management Server Preferences

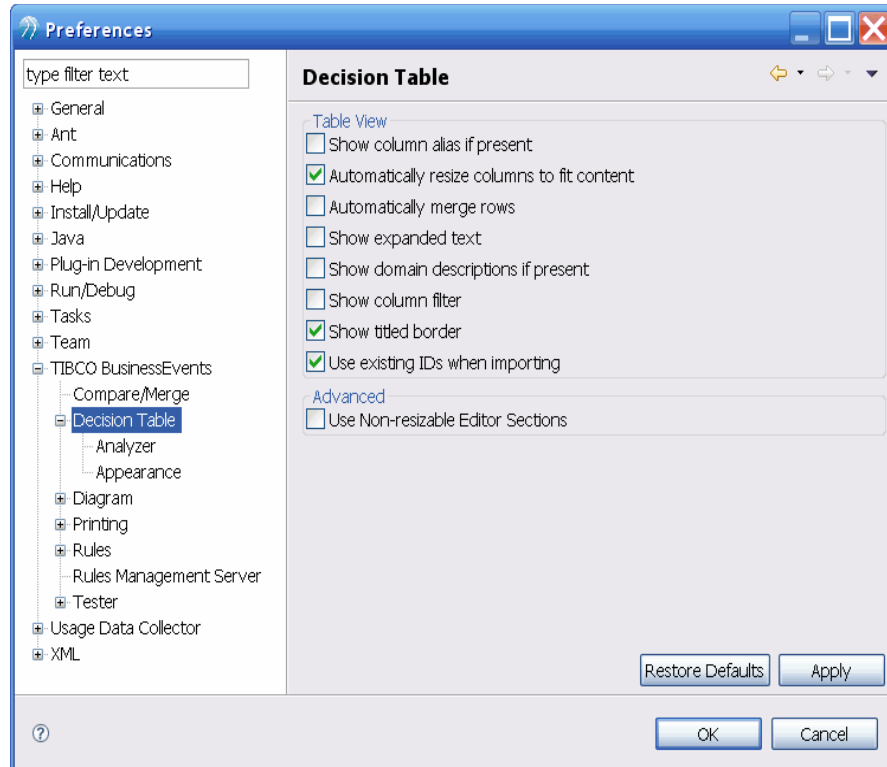
These preferences are used with TIBCO BusinessEvents Decision Manager.

Tester and Tester Appearance Preferences

See [Tester Preferences on page 457](#).

Decision Table Related Preferences

Decision Table Preferences



The following Table View preferences set the initial state of the following features. Text in parentheses shows the names of corresponding buttons in the Decision Table editor. The settings can be toggled in the UI, but their initial state is set by preferences you set:

- Show column alias if present
- Automatically resize columns to fit content (Fit Content button)
- Automatically merge rows (Merge Rows button)
- Show expanded text (Show Text button)
- Display domain descriptions instead of property names.
- Show column filter
- Show titled border

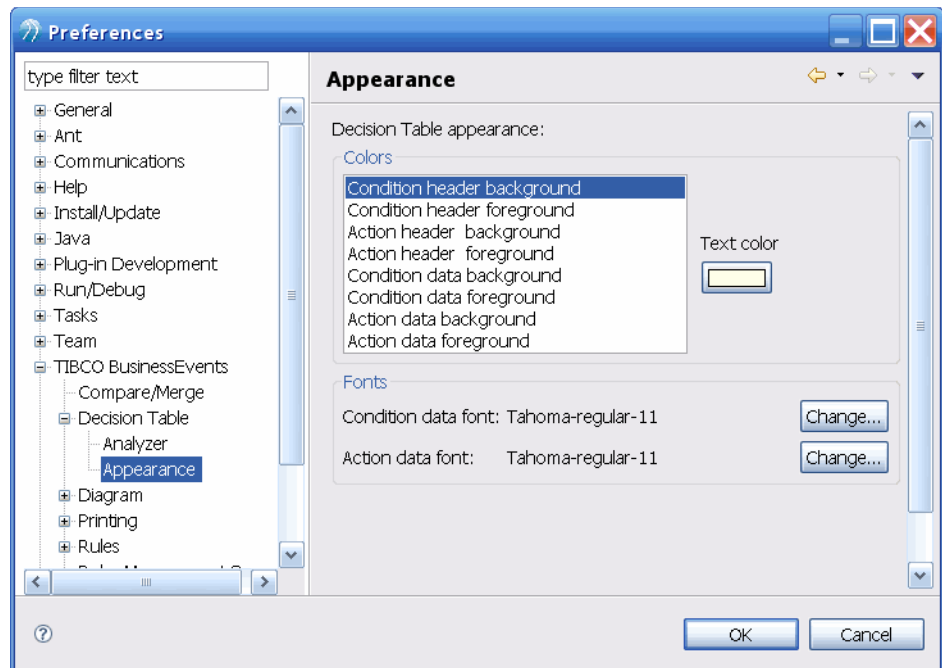
- Use existing IDs when importing
- The Use non-resizable editor sections option prevents or allows users from resizing editor sections.

Decision Table Analyzer Preferences

You can control the following behavior of the decision table analyzer, as well as its appearance:

- Initially show the Table Analyzer view
- Highlight partial ranges
- Use domain model for table completeness
- Analyze while opening table.

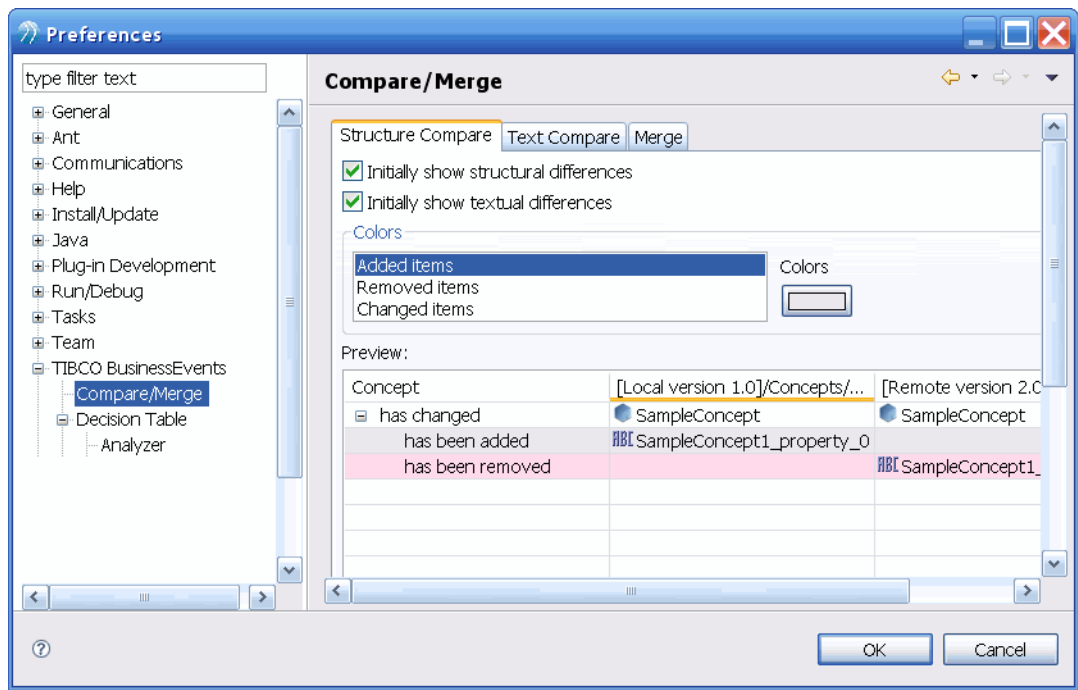
Decision Table Appearance Preferences



These preferences let you set the color scheme for decision tables. For each item in the list, you can set color preferences. For condition data and action data you can specify a font.

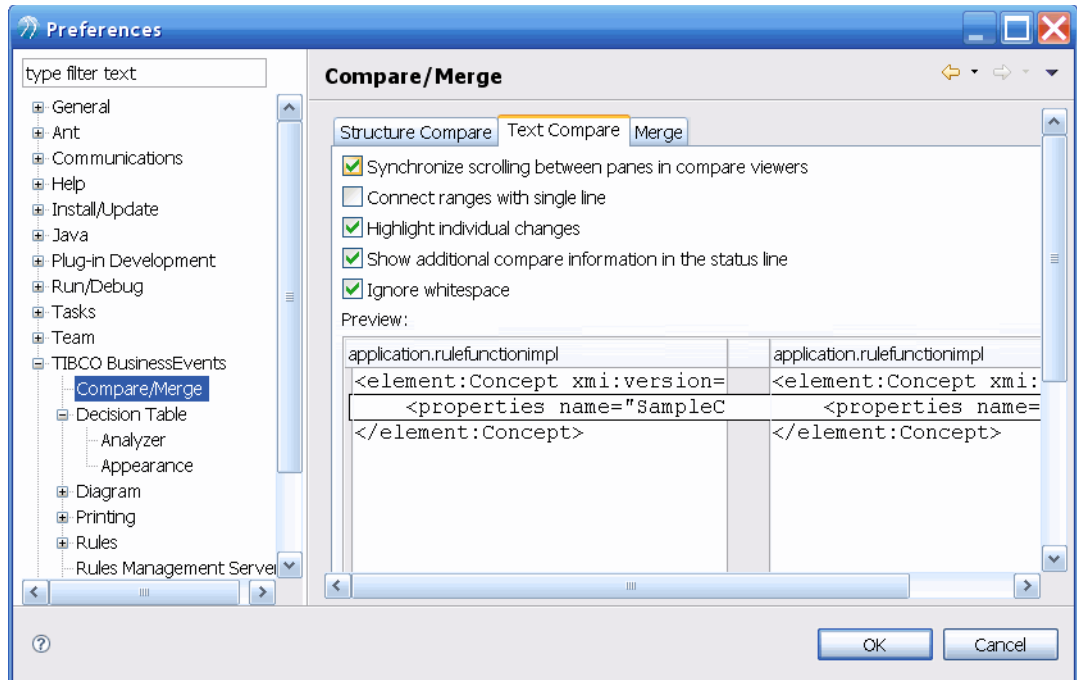
Compare/Merge Preferences

This tabbed dialog is in the TIBCO BusinessEvents preferences. It applies to decision tables.



In the Structure Compare tab (shown above), set preferences for initial settings and colors. The effect of your choices is shown in the lower panel.

In the Text Compare tab (shown below), you can set preferences for text. The effect of your choices is shown in the lower panel.



The Merge tab has one option: Allow columns to be automatically merged.

Diagram Preferences

The diagram preferences apply to the following types of diagrams:

- Concept
- Dependency
- Event
- Project
- Sequence
- State Model (This option is available only in the TIBCO BusinessEvents Data Modeling add-on.)

Preferences set using a diagram’s palette are applicable only to the displayed diagram.

Preferences set using the Preferences dialog define the default preferences for diagrams of that type.

Preferences for specific diagrams override global preferences, where the same settings exist.

Table 49 Global Diagram Preferences

Option	Description
Reset Tool After Changes	If enabled, resets the tool to the Select tool after you add a node to the diagram. Default: Enabled.
Auto Hide Scrollbars	If enabled, then when the complete contents can display in the space available, scrollbars are hidden. If not enabled, scrollbars remain. Default: Auto hide scrollbars is enabled.
Show Tooltips	If enabled, you can see the tooltips when you hover the mouse on an element in the diagram. Default: Show Tooltips is enabled.
Link Types	Shows the links as straight lines or curved lines. Default: Straight.

Table 49 Global Diagram Preferences (Cont'd)

Option	Description
Run Layout On Changes	<p>After adding or deleting a node or an edge, refreshes the layout of the diagram. The layout options are None, Incremental, and Full.</p> <p>Default: None.</p>
Animation	<p>You can specify how a diagram behaves while it is changing from one layout option to another. Animation options allow you to see the nodes and edges moving from one arrangement to another. If you disable animation, the display simply switches from one kind of layout to another.</p> <p>Fade: When enabled, the diagram transitions from faded out to fully colored state. The transition completes in the duration specified in the Layout duration option.</p> <p>Interpolation: When enabled, the nodes and edges appear to move from one layout to another, when you select a different layout. The transition completes in the duration specified in the Layout duration option.</p> <p>Viewport Change: Enables the interpolation animation when the viewport changes as a result of zooming. If enabled, the zoom change is reflected gradually with animation.</p> <p>By default, animation is enabled along with Fade, Interpolation and Viewport Change.</p> <p>Duration: The time taken by animation to complete.</p> <p>Layout: Time, in milliseconds, for the layout change to complete. Applicable to the interpolation and fade options.</p> <p>Default for Layout: 500</p> <p>Viewport Change: Time, in milliseconds, for the viewport change to complete.</p> <p>Default for Viewport Change: 1000</p>
Opaque Movement	<p>Interactive Zoom Sensitivity specifies the sensitivity of the mouse in interactive zooming.</p> <p>Pan Sensitivity specifies the sensitivity of the mouse in panning.</p> <p>Default: Opaque Movement is enabled, Interactive Zoom Sensitivity is 200.0, and Pan Sensitivity is 1.0.</p>
Magnify Tool	<p>The size the zoom window, and the level of zoom.</p> <p>Default: Window Size is 250, and Zoom Level is 3.</p>

The preferences for all types of diagrams are listed in [Table 50 on page 454](#). The preferences are organized alphabetically and not grouped according to the diagram types.

Table 50 Specific Diagram Preferences (Alphabetical) (Sheet 1 of 3)

Option	Description
Cluster Layout Style	Sets whether to show Selected Entity Project diagrams in a circular or symmetric clustered layout. Default: Circular.
Create view when analyzing	If enabled, creates a Selected Entity Project diagram when you choose to run the project analyzer. Default: Disabled.
Dependency Levels	Sets how many levels of dependencies to view in a dependency diagram - One, two, or all. Default: One.
Filter Options	Sets whether to show various project resources in a Selected Entity Project diagram: Show Concepts, Show Events, Show Decision Tables, Show Domain Model, Show State Machines, Show Archives, Show Rules, Show Rule functions, Show Scorecard, Show Channels, Show Scope Links, Show Usage Links, Show Archived Destinations, Show Archived Rules, Show Archived Rules (All), Show Rules in Folders, Show Tooltips, Group Concepts, Group Events, Group Rules, and Group Rule Functions. Default: All enabled except for Archived Rules, Group Concepts, Group Events, Group Rules, and Group Rule Functions.
Fix node and edge labels	Keeps the labels of the nodes and edges with their graphics. Default: Enabled.
Grid	Sets the display either to without grid, or grid with lines, or grid with points. Default: Lines.
Layout Style	Sets either the orthogonal or hierarchical layout. Default: Orthogonal.

Table 50 Specific Diagram Preferences (Alphabetical) (Sheet 2 of 3)

Option	Description
Layout Quality	<p>Options are Draft (lowest quality), Medium, and Proof (highest quality). It takes longer to generate a higher quality diagram than a lower quality diagram. A different algorithm is used in each case.</p> <p>Default: Draft.</p>
Link Routing	<p>Determines how links are routed when the hierarchical layout is used:</p> <p>Orthogonal—Routes the links using horizontal and vertical straight line segments (that is, using right angles)</p> <p>Polyline—Routes the links using straight line segments with arbitrary angles</p> <p>Overlapping lines are more likely with orthogonal routing than with polyline routing. With polyline routing the routing algorithm adds path nodes as needed to avoid overlapping lines.</p> <p>Note that the line segments can be joined using straight lines or curves in both cases.</p> <p>Default: Orthogonal.</p>
Link Routing— Fix Node Positions	<p>This setting affects link routing behavior for all layout options.</p> <p>If enabled, node positions do not change when you use the link routing feature.</p> <p>If disabled, link routing changes node positions as needed for clarity.</p> <p>Default: Disabled.</p>
Link Routing— Fix Node Sizes	<p>This setting affects link routing behavior for all layout options except hierarchical layouts.</p> <p>If enabled, node sizes do not change when you use the link routing feature.</p> <p>If disabled, link routing changes node sizes as needed for clarity.</p> <p>Default: Disabled.</p>
Orientation	<p>Defines the general direction in which the links display, to reflect hierarchical relationships between the entities. TIBCO BusinessEvents diagrams are not particularly hierarchical, but setting this option defines the general direction of the layout.</p> <p>Options are: Top to Bottom, Bottom to Top, Left to Right, and Right to Left.</p> <p>Default: Top to Bottom.</p>

Table 50 Specific Diagram Preferences (Alphabetical) (Sheet 3 of 3)

Option	Description
Orthogonal Fix Node Sizes	<p>This setting affects link routing behavior for the hierarchical layout option only.</p> <p>If enabled, node sizes do not change when you use the link routing feature.</p> <p>If disabled, link routing changes node sizes as needed for clarity.</p> <p>Default: Disabled.</p>
Run analysis when creating view	<p>Runs the Project Analyzer when creating a Selected Entity Project diagram.</p> <p>Default: Enabled.</p>
Run fast layout for large diagrams	<p>Only for Selected Entity Project diagrams. If enabled, TIBCO BusinessEvents filters out certain properties before generating a large Selected Entity Project diagram to make it look simpler.</p> <p>Default: Disabled.</p>
Show all properties in Concept Node	<p>If enabled, a diagram shows all properties of a concept node, instead of showing only four default properties.</p> <p>Default: Disabled.</p>
Show all properties in Event Node	<p>If enabled, a diagram shows all properties of an event node, instead of showing only four default properties.</p> <p>Default: Disabled.</p>
Show Catalog Functions	<p>Sets whether to show catalog functions in a Sequence diagram.</p> <p>Default: Enabled.</p>
Show Catalog Function Return Links	<p>Sets whether to show return links of catalog functions in a Sequence diagram.</p> <p>Default: Disabled.</p>
Show Expanded Names	<p>Sets whether to show expanded names in a Sequence diagram.</p> <p>Default: Enabled.</p>
Snap to grid	<p>If you move nodes in a diagram and the grid is shown, the nodes snap to the grid lines if Snap to grid is enabled.</p> <p>Default: Disabled.</p>
Undirected layout	<p>Sets no orientation for the diagram.</p> <p>Default: Disabled.</p>

Tester Preferences

The Tester preferences section lets you configure the tester behavior. See [Chapter 23, Testing and Debugging Projects, on page 399](#).

Table 51 Reference to Tester Preferences

Option	Description
Input Directory	The directory in which test data files you create are stored. Default: /TestData
Output Directory	The directory in which the result files are stored. Default: /TestData/<Projectname>/<Processing Unit name>
No of WM Objects	Maximum number of objects in working memory using a preference. Default: 50
Auto scroll Modified Result Tables	If enabled, you see scroll bars for the cells in Result Test Data after clicking Fit Content. These scroll bars move together for both the After and Before sections. Default: Enabled

When you test a project, tester shows the values changed while running the engine and the instances created in Result Test Data. To set the text color preferences for background and foreground of the modified values, go to **Window > Preferences > TIBCO BusinessEvents > Tester > Appearance**.

Table 52 Reference to Tester Appearance Preferences

Option	Description
Background of the modified value	The background color of the changed values. Click the color box, which opens a color palette. Select the new color from it. Default: Light Pink
Foreground of the modified value	The text color of the changed values. Click the color box, which opens a color palette. Select the new color from it. Default: Blue

Table 52 Reference to Tester Appearance Preferences

Option	Description (Cont'd)
Font for the modified value	The font in which the modified value appears. Click Change... to change the font. Default: Tahoma Regular 11

Appendix A **Handling Null Properties**

This appendix explains three cases for special handling of null concept property values.

Topics

- [Handling Null Concept Property Values, page 460](#)

Handling Null Concept Property Values

This appendix explains three cases for special handling of null concept property values:

- [Enabling Use of the Nillable Attribute on page 460](#)
- [Enabling Null Property Values to Appear When Serializing Concepts to XML on page 460](#)
- [Enabling and Setting Special Treatment of Numeric Null Values on page 462](#)

This section also provides related procedures and a reference to the properties used ([Table 53, Properties for Null Property Handling, on page 464](#)).

Enabling Use of the Nillable Attribute

The presence of the `xsd:nillable` attribute in an XSD element means that the corresponding element in the XML file permits null values.

Setting `tibco.be.schema.nil.attrs=true` in `studio.tra` causes the `xsd:nillable` attribute ("`xsd:nillable=true`") to be set on all elements in the TIBCO BusinessEvents concept XSD. When an element in the XML file generated using that XSD has a null value, the `xsi:nil="true"` attribute is set on that element.

When set to false, the `xsd:nillable` attribute is not added and the corresponding XML file does not treat empty elements as null values.

In the absence of the `xsd:nillable` attribute in the XSD element, a corresponding empty element in the XML file is assumed to have a value. Elements that have no value are treated as empty strings (" ").



Effect on schema generation tool The setting for this property affects the concept XSD files generated using the Generate Schema utility. See [Exporting \(Generating\) Concept and Event Schema \(XSD\) Files on page 7](#) for details.

Enabling Null Property Values to Appear When Serializing Concepts to XML

By default concept properties with null values are excluded when concept objects (instances) are serialized to XML. You can override this behavior.

Setting the following property to false in the `studio.tra` file causes properties with null values to be included in the XML representation of a concept:

```
tibco.be.schema.exclude.null.props=false
```


Examples of Nillable Attribute and Null Properties Settings

These examples illustrate the effect of the following properties on concept serialization:

```
tibco.be.schema.nil.attrs
tibco.be.schema.exclude.null.props
```

If Null Properties are Excluded

```
tibco.be.schema.nil.attrs= true or false
tibco.be.schema.exclude.null.props=true
```

Suppose a Customer concept instance has no value for its CustomerName property. By default, the CustomerName property is excluded from the XML output. The output might look like the following:

```
<CustomerID>111</CustomerID>
<Country>Japan</Country>
<City>Tokyo</City>
```

If null properties are excluded when concepts are serialized, the `tibco.be.schema.nil.attrs` property has no effect on concept serialization.

If Null Properties are Included and the Nillable Attribute is Set

```
tibco.be.schema.nil.attrs=true
tibco.be.schema.exclude.null.props=false
```

The output for the Customer concept instance shown above would be as follows, where there is no value for the `CustomerName` element in the concept instance:

```
<CustomerID>111</CustomerID>
<Country>Japan</Country>
<City>Tokyo</City>
<CustomerName
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:nil="true"/>
```

If Null Properties are Included and the Nillable Attribute is not Set

```
tibco.be.schema.nil.attrs=false
tibco.be.schema.exclude.null.props=false
```

In this case, each null property is considered to be an empty string, and is represented, for example, as follows:

```
<CustomerName/>
```

Enabling and Setting Special Treatment of Numeric Null Values

If you enable null values to be output to XML (see [Enabling Null Property Values to Appear When Serializing Concepts to XML on page 460](#)), then you may also want to configure additional properties for defining how to treat null values for numeric types, as explained in this section.

TIBCO BusinessEvents does not implicitly support null values for numeric types. This can lead to interoperability issues when working with external sources, such as databases, which do permit blank (null) values.

To address such issues, you can enable special treatment of numeric null values at design time. At runtime, TIBCO BusinessEvents then uses a special numeric value for each numeric datatype to represent a null value. Default special values are provided and you can override the defaults at runtime (see [Table 53, Properties for Null Property Handling, on page 464](#)).

The special numeric values that indicate null are used in TIBCO BusinessEvents when serializing and deserializing a concept to and from its XML representation, and when performing various operations on database concepts and the database tables to which they are linked.

The special numeric values that indicate null appear only in TIBCO BusinessEvents. The appropriate null value is used in the XML or database representation of the concept property.

Conversely, when deserializing or importing a concept instance, TIBCO BusinessEvents represents numeric null values using the special numeric values that indicate null in the concept instance.

To enable special treatment of numeric null values, set the following property in `studio.tra`:

```
tibco.be.schema.treat.null.values=true
```

At runtime you can override the default values using the following properties in the project CDD file at the cluster level:

```
tibco.be.property.int.null.value=value
tibco.be.property.long.null.value=value
tibco.be.property.double.null.value=value
```

Summary

Set the following properties in

`BE_HOME/studio/eclipse/configuration/studio.tra` as desired:

- To enable null property values to appear when serializing concepts to XML, add the following property and set it to false:

```
tibco.be.schema.exclude.null.props=false
```

- To enable use of the nillable attribute in the concept XSD, add the following property and set it to true:

```
tibco.be.schema.nil.attrs=true
```

- To enable special handling of null properties, add the following property and set it to true:

```
tibco.be.schema.treat.null.values=true
```

Also see [Setting Runtime Properties for Special Treatment of Null Values on page 463](#)

Setting Runtime Properties for Special Treatment of Null Values

If you have enabled special treatment of null numeric properties ([Enabling and Setting Special Treatment of Numeric Null Values on page 462](#)), you can override the default special numeric values that indicate numeric null values in TIBCO BusinessEvents as follows.

1. In TIBCO BusinessEvents Studio, open the CDD editor for the project and add the following properties at the cluster level properties sheet. Provide the special values as desired:

```
tibco.be.property.int.null.value=value
```

```
tibco.be.property.long.null.value=value
```

```
tibco.be.property.double.null.value=value
```

Choose values that will not be misinterpreted as literal values.

2. Save the file.
3. Deploy the project with the updated CDD file.

Property Reference for Null Property Handling

Set the following properties in the `studio.tra` file as needed to configure the output for your needs before you generate the EAR file.

Table 53 Properties for Null Property Handling

Property	Notes
Properties Set in <code>BE_HOME/studio/eclipse/configuration/studio.tra</code>	
<code>tibco.be.schema.nil.attrs</code>	<p>Setting this property to true causes the <code>xsd:nilable</code> attribute (<code>"xsd:nilable=true"</code>) to be set on all elements in the TIBCO BusinessEvents concept XSD.</p> <p>See Enabling Use of the Nillable Attribute on page 460</p> <p>Possible values are true and false.</p> <p>Default is false.</p>
<code>tibco.be.schema.exclude.null.props</code>	<p>When this property is set to true, null-valued concept properties are not output when the concept is serialized to XML.</p> <p>When set to false, null-valued concept properties are output to XML.</p> <p>See Enabling Null Property Values to Appear When Serializing Concepts to XML on page 460.</p> <p>Possible values are true and false.</p> <p>Default is true.</p>
<code>tibco.be.schema.treat.null.values</code>	<p>Setting this property to true causes TIBCO BusinessEvents to use special numeric values that indicate null for numeric datatypes. The special numeric values are set using the properties listed next.</p> <p>See Enabling and Setting Special Treatment of Numeric Null Values on page 462.</p> <p>Possible values are true and false.</p> <p>Default is false.</p>

Table 53 Properties for Null Property Handling (Cont'd)

Property	Notes
Properties Set in CDD at the cluster level	
<code>tibco.be.property.int.null.value</code> <code>tibco.be.property.long.null.value</code> <code>tibco.be.property.double.null.value</code>	<p>These properties define a special numeric value that indicates null. Use a value that will not be confused with an actual numeric value.</p> <p>These properties are used only if <code>tibco.be.schema.treat.null.values</code> is set to true.</p> <p>Default values for each numeric datatype are the following Java constants:</p> <pre>int: Integer.MIN_VALUE long: Long.MIN_VALUE double: Double.MIN_VALUE</pre> <p>For Integer and Long these constants represent the most negative value. For Double the constant represents smallest positive nonzero value (4.9e-324).</p>

Symbols

[@category 160](#)
[@closure 152, 281](#)
[@extId 143, 160, 170, 281](#)
[@id 143, 152, 160, 170, 281](#)
[@interval 152, 281, 281](#)
[@isSet 282](#)
[@isset 171](#)
[@length 282](#)
[@message 160](#)
[@parent 170, 282](#)
[@payload 143, 281](#)
[@scheduledTime 152, 281, 281](#)
[@ttl 143, 153, 281, 281, 281](#)
[@type 160](#)

A

[Accept Count 102](#)
[action-only functions 253](#)
[actions 221](#)
 [in rule editor 227](#)
[activities](#)
 [Input tab icons 301](#)
[activities palette 348, 365](#)
[addressing schema elements 336](#)
[advisory events 158, 158](#)
[AdvisoryEvent event type 286](#)
[Agent.AgentGroupName.checkDuplicates 143, 170](#)
[aliases 227](#)
[all values history policy 169](#)
[array indexes, start from zero or one 236, 284](#)
[arrays](#)
 [accessing and appending values 284](#)
 [indexes start from zero or one 236, 284](#)
 [primitive 276](#)
[attributes 281](#)
[AUTO-ACKNOWLEDGE 87](#)

B

[BE_HOME xxxii](#)
[be.engine.profile.*.duration 390](#)
[be.engine.profile.*.enable 388](#)
[be.engine.profile.*.file 389](#)
[be.engine.profile.*.level 390](#)
[be.engine.profile.BAR_Name.duration 389](#)
[be.engine.profile.BAR_Name.enable 388](#)
[be.engine.profile.BAR_Name.file 389](#)
[be.engine.profile.BAR_Name.level 390](#)
[be.engine.profile.delimiter 387](#)
[be.hawk.microagent.name 361](#)
[BEReceiveEventOutput 367, 371](#)
[BESendEventInput 368](#)
[BytesMessageSerializer 83](#)

C

[Candidate Event Key 370](#)
[changes from the previous release of TIBCO BusinessEvents Developer's Guide xxiv](#)
[changes only history policy 169](#)
[ChannelURI, variable 85](#)
[Ciphers \(SSL Server in HTTP channels\) 108](#)
[circular layout 441, 443](#)
[CLIENT_ACKNOWLEDGE 87, 89](#)
[ClientID \(channel property\) 75](#)
[Committing Project Artifacts to RMS at the Command Line 43](#)
[complex element 140](#)
[Compressible Mime Types 107](#)
[Compression 105](#)
[concept properties, accessing 283](#)
[concepts](#)
 [and ontology functions 250](#)
[conditions 221](#)
 [in rule editor 227](#)
[Connection Linger 104](#)
[Connection Timeout 102](#)
[constants in XPath 307](#)
[controlling start of state machines 168](#)

- custom functions
 - name overloading not supported [267](#)
 - return types supported [267](#)
 - static modifiers [267](#)
 - tool tips [270](#)
- Custom ID [367](#)
- customer support [xxxv](#)

D

- daemon (channel property) [75](#)
- date functions [248](#)
- DateTime functions [248](#)
- declaration [220](#)
 - rule editor [227](#)
- default events [62](#)
- Deployment category of advisory event [161](#)
- destinations
 - and channels [60, 76](#)
 - default destination for event [138](#)
- DestinationURI, variable [86](#)
- diagrams
 - layout menu options [441](#)
 - Link Navigator [441](#)
 - preferences [452](#)
- Document Page [104](#)
- Document Root [104](#)
- DUPS_OK_ACKNOWLEDGE [88](#)
- durable subscriber name, and special variables [85](#)
- DurableSubscriberName [85](#)

E

- Enable DNS Lookups [105](#)
- Engine (advisory) [160](#)
- Engine category of advisory event [161](#)
- engine functions [248](#)
- engine properties
 - See also Index of Engine Properties
- engine.primary.activated (advisory) [161](#)
- Engine.Profiler.startCollectingToFile() [391](#)

- Engine.Profiler.stopCollecting() [391](#)
- EngineName, variable [85](#)
- engines
 - configuring for remote debugging [403](#)
- ENV_HOME [xxxii](#)
- errors [300, 300](#)
- errors in mappings [300](#)
- escape sequences [278](#)
- evaluation context [337](#)
- event functions [249](#)
- event preprocessors [243](#)
- event properties, accessing [285](#)
- event scheduler functions [155](#)
- Event.routeTo [138](#)
- Event.sendEvent [138](#)
- events [138](#)
 - advisory [158, 158](#)
 - and ontology functions [250](#)
 - default [62](#)
 - default destination [138](#)
 - inherits from, setting [137](#)
 - naming restrictions [139, 150](#)
 - payload parameters [140](#)
 - scheduling [155](#)
- Exception (advisory) [160](#)
- Exception category of advisory events [160](#)
- Exception function [249](#)
- Expiry Action [139](#)
- EXPLICIT_CLIENT_ACKNOWLEDGE [88](#)
- EXPLICIT_CLIENT_DUPS_OK_ACKNOWLEDGE [8](#)
- extended functions [251](#)
- extId
 - must be unique [143](#)

F

- File functions [249](#)

functions

- action only [253](#)
- extended (hidden) [251](#)
- mapper [234](#), [253](#)
- standard [248](#)
- temporal [254](#)
- tool tips for [252](#)
- types and usage [248](#)

functions documentation, accessing [xxxi](#)

G

global variables

- in the rule editor [20](#), [233](#)

H

Hawk.AMI.DisplayName [360](#)

hidden functions [251](#)

hierarchical layout [444](#)

hierarchical layout with normal routing [442](#)

hierarchical layout with orthogonal routing [441](#)

hints [299](#)

History [169](#)

HTTP Channel

- BUILT-IN server type settings [107](#)

- TOMCAT server type settings [104](#)

HTTP functions [249](#)

I

incremental layout [442](#), [444](#)

Inherits From [167](#)

inherits from event setting [137](#)

Input tab

- icons [301](#)

Insert Model Group Content [310](#), [310](#), [310](#)

Instance.startStateMachine() [168](#)

instances

- aliases for [227](#)

INVOKE BW PROCESS type of advisory event [160](#)

INVOKE BW PROCESS, type of BEBW advisory event [379](#)

IO functions [249](#)

isTransacted (channel property) [75](#)

J

Java Debug Interface (JDI) for remote debugging [404](#)

JMS

- channel naming restrictions [139](#), [150](#)

- converting messages to non-default events [62](#)

- durable subscriber name [85](#)

- header properties [139](#), [150](#)

JMSCorrelationID [93](#)

JMSDeliveryMode [92](#)

JMSDestination [92](#)

JMSExpiration [92](#)

JMSMessageID [93](#)

JMSPriority [92](#)

JMSRedelivered [93](#)

JMSReplyTo [93](#)

JMSTimestamp [93](#)

JMSType [93](#)

K

Key Manager Algorithm [108](#)

L

layout

- labeling [442](#)

layout menu [441](#)

layout options

- circular [443](#)
- hierarchical [444](#)
- incremental [444](#)
- orthogonal [443](#)
- symmetric [443](#)

Link Navigator [441](#)Link Routing [442](#)links in diagrams, view path of [441](#)

local channels

- maximum events in queue [79](#)

local variables [276](#)lockWM [377](#)**M**mapper functions [234](#), [253](#)

mapping

- addressing schema elements [336](#)
- Input tab icons [301](#)
- XPath operators and functions [341](#)

mappings [300](#)math functions [249](#)Max HTTP Header Size [106](#)Max HTTP Post Size [106](#)Max HTTP Save Post Size [106](#)Max KeepAlive Requests [105](#)Max Processors [103](#)Max Spare Threads [107](#)maximum (history parameter) [169](#)migrateCoherenceCalls [37](#)Min Spare Threads [107](#)Multiple (array) [169](#)**N**namespace [62](#)NCName datatype [120](#)network (channel property) [74](#)NO-ACKNOWLEDGE [89](#)number functions [249](#)**O**

ontology functions

- location of [250](#)
- purpose and types of [250](#)

Orthogonal Layout [441](#)orthogonal layout [443](#)**P**

palettes

- TIBCO BusinessEvents activities [348](#), [365](#)

payload parameters

- cardinality [140](#)
- complex element [140](#)

payloads [140](#)

- all, parameter [141](#)
- attribute of type, parameter [141](#)
- choice, parameter [141](#)
- element of type, parameter [140](#)
- sequence, parameter [141](#)
- validation, parameter [142](#)
- XML element reference, parameter [140](#)
- XML group reference, parameter [141](#)

Policy (concept history) [169](#)preprocessors [243](#)

- options on failure [138](#)

primitive arrays [276](#)Priority [226](#)processTimeout [370](#)property arrays, index from zero or one [236](#), [284](#)property values, accessing [283](#), [283](#)Protocol (for HTTP channel) [106](#)Protocols (SSL Server in HTTP channels) [108](#)ProviderURL (channel property) [75](#)**Q**

queue

- local channel [79](#)

R

- remote debugging
 - configuring engine for [403](#)
 - Java Debug Interface and [404](#)
- RESTMessageSerializer [100](#)
- Restricted User Agents [107](#)
- Retry On Exception [138](#)
- Rule Debugger
 - configuring engine for remote debugging [403](#)
- rule editor
 - alias [227](#)
 - global variables in [20, 233](#)
- rule functions
 - ontology functions [250](#)
- rules
 - actions [227](#)
 - and advisory events [158](#)
 - and ontology functions [250](#)
 - conditions [227](#)
 - declaration [227](#)

S

- scheduling events [155](#)
- schema elements [298](#)
 - addressing [336](#)
- search predicates in XPath expressions [337](#)
- sending to another application [138](#)
- Sequencing Key [367](#)
- serializers [61](#)
- service (channel property) [74](#)
- SOAP functions [249](#)
- Socket Output Buffer Size [102](#)
- source control systems [5](#)
- specifying constants in XPath expressions [307](#)
- SSL Server
 - Ciphers [108](#)
 - Key Manager Algorithm [108](#)
 - Protocols [108](#)
 - Trust Manager Algorithm [108](#)
- Stale Connection Check [107](#)

- standard functions [248](#)
 - categories of [248](#)
- StartFileBasedProfiler() [392](#)
- state machines
 - controlling start of [168](#)
- StopFileBasedProfiler() [392](#)
- string functions [249](#)
- support, contacting [xxxv](#)
- symmetric layout [441, 443](#)
- system functions [249](#)

T

- TCP No Delay [103](#)
- technical support [xxxv](#)
- temporal functions [249](#)
 - arguments [254](#)
- TextMessageSerializer [84](#)
- TIBCO BusinessWorks
 - process definitions, and TIBCO BusinessEvents
 - activities [348, 365](#)
- TIBCO Rendezvous
 - converting messages to non-default events [62](#)
- TIBCO_HOME [xxxii](#)
- tibco.be.property.double.null.value [465](#)
- tibco.be.property.int.null.value [465](#)
- tibco.be.property.long.null.value [465](#)
- tibco.be.schema.exclude.null.props [464, 464](#)
- tibco.be.schema.nil.attrs [464](#)
- tibco.be.schema.treat.null.values [464](#)
- tibco.bwengine.name [360](#)
- time events
 - and ontology functions [250](#)
- Time to Live [138](#)
- timeouts
 - local channels [79](#)
- tool tips [270](#)
 - creating for custom functions [265](#)
 - turning off display [252](#)
- tool tips, for functions [252](#)
- Trust Manager Algorithm [108](#)

U

URI Encoding [104](#)

Use Body Encoding for URI [103](#)

Util functions [249](#)

V

validation of XML payload elements [142](#)

variables for use with durable subscriber name [85](#)

variables, local [276](#)

VRP (Virtual Rule Function) functions [249](#)

X

XML payloads element validation [142](#)

XPath [335](#)

- basics [336](#)

- editor [339](#)

- evaluation context [337](#)

- example [341](#)

- operators and functions [341](#)

- search predicates [337](#)

- specifying constants [307](#)

XPath functions [249](#)

XSLT statements [299](#), [300](#)

XSLT template [234](#), [253](#)