

TIBCO BusinessEvents™

Getting Started

Software Release 3.0.1

November 2008

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 LICENSE.PDF) 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.

TIB, TIBCO, TIBCO Software, TIBCO Adapter, Predictive Business, Information Bus, The Power of Now, The Power to Predict, TIBCO BusinessEvents, TIBCO ActiveMatrix BusinessWorks, TIBCO Rendezvous, TIBCO Enterprise Message Service, TIBCO PortalBuilder, TIBCO Administrator, TIBCO Runtime Agent, TIBCO General Interface, and TIBCO Hawk are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

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

Excerpts from Oracle Coherence documentation are included with permission from Oracle and/or its affiliates. Copyright © 2000, 2006 Oracle and/or its affiliates. All rights reserved.

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 © 2004-2008 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

| | |
|--|------------|
| Figures | vii |
| Preface | ix |
| Enterprise Suite and Inference Edition Features | x |
| Related Documentation | xi |
| TIBCO BusinessEvents Documentation | xi |
| Other TIBCO Product Documentation | xii |
| Typographical Conventions | xiii |
| How to Contact TIBCO Support | xvi |
| Chapter 1 Introduction | 1 |
| Overview | 2 |
| Example Projects Provide Additional Learning | 2 |
| Skills Required | 2 |
| Tutorial Scenario | 3 |
| Chapter 2 Project Design Tutorial | 5 |
| Fraud Detection Runtime Flow | 6 |
| Create the FraudDetection Project | 8 |
| Create a Rendezvous Channel and a Destination | 11 |
| Define the Debit Event | 13 |
| Validate the Project and Set a Default Event for a Destination | 15 |
| Define the Account Concept | 17 |
| Create the FraudCriteria Scorecard | 19 |
| Configure the InitializeAccount Rule Function | 20 |
| Optional Exercise: Create a Concept Instance Using a Standard Function | 23 |
| Configure the ProcessDebits Rule Set and ApplyDebit Rule | 25 |
| Configure the FraudDetection Rule | 28 |
| Configure the CheckNegativeBalance Rule | 30 |
| Validate the Project, Configure and Build the Archive | 31 |
| Configure an ActiveMatrix BusinessWorks Process for Testing | 34 |
| Deploy the Project | 38 |

| | |
|--|-----------|
| Test the Application | 39 |
| Chapter 3 Cache Object Management Tutorial | 41 |
| Caching and Multi-Engine Overview | 42 |
| JMS Server Required | 43 |
| Create a JMS Channel and Destination | 44 |
| Delete or Disable the Rendezvous Channel and its Destination | 47 |
| Check for Existence of Accounts Before Creating | 49 |
| Configure Cache Cluster Properties | 51 |
| Configure Cache Server Nodes | 54 |
| Configure Inference Agents | 55 |
| Deploy the Agents and Cache Servers | 58 |
| Troubleshooting | 59 |
| Test the Application | 60 |
| Chapter 4 Backing Store Tutorial | 63 |
| Prepare the Database | 64 |
| Add a JDBC Connection Resource | 67 |
| Configure the Backing Store Properties | 70 |
| Deploy and Test the Application | 72 |
| Troubleshooting | 72 |
| Reset the Backing Store Tutorial | 73 |
| Appendix A Getting Started With TIBCO Designer | 75 |
| Welcome to TIBCO Designer | 76 |
| Starting TIBCO Designer | 77 |
| Startup Options | 77 |
| TIBCO Designer Administration | 80 |
| TIBCO Designer Interface Overview | 82 |
| Main Window | 82 |
| Project Panel | 83 |
| Palette Panel | 85 |
| Design Panel | 85 |
| Configuration Panel | 86 |
| Working With Palettes | 87 |
| Current Selection and Palette Panel Display | 87 |
| Using the Palette Browser | 88 |
| Showing and Hiding Palettes | 89 |
| Choosing Palette Mode or Non-palette Mode | 90 |

| | |
|---|------------|
| Displaying Palettes in a Separate Window | 91 |
| Working With User Palettes | 91 |
| Customizing the Display | 93 |
| Choosing Panel Layout | 93 |
| Accessing Documentation | 95 |
| Appendix B Managing Projects and Resources | 97 |
| Overview of Projects | 98 |
| Project Structure | 98 |
| Using a Version Control System | 99 |
| Project Templates | 99 |
| Creating Projects | 100 |
| Validating Projects | 101 |
| Saving Projects | 102 |
| Opening and Reopening Projects | 104 |
| Adding Resources To Your Project | 105 |
| If Adding a Resource Results in an Error | 106 |
| Working With Global Variables | 107 |
| Global Variables Editor | 107 |
| Deleting Projects | 111 |
| Tips and Tricks for Working With Projects | 112 |
| Index | 113 |

Figures

| | | |
|-----------|--|-----|
| Figure 1 | TIBCO Designer startup panel | 77 |
| Figure 2 | Startup panel Administration options | 80 |
| Figure 3 | The TIBCO Designer window. | 82 |
| Figure 4 | Project tree in the project panel | 84 |
| Figure 5 | Global variables in project panel | 84 |
| Figure 6 | Palette panel changes depending on current selection | 87 |
| Figure 7 | Palette mode vs. non-palette mode | 90 |
| Figure 8 | The three-panel view | 93 |
| Figure 9 | Three-panel view with palettes selected | 94 |
| Figure 10 | Layout options | 94 |
| Figure 11 | Adding a resource to a project | 105 |

Preface

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

BusinessEvents
The Power to Predict™

Topics

- *[Enterprise Suite and Inference Edition Features, page x](#)*
- *[Related Documentation, page xi](#)*
- *[Typographical Conventions, page xiii](#)*
- *[How to Contact TIBCO Support, page xvi](#)*

Enterprise Suite and Inference Edition Features

BusinessEvents is available in the Inference Edition and in the Enterprise Suite. The components available in each option are listed below.

Inference Edition and Enterprise Suite

Inference Edition provides inferencing features and comprises the following components (also included in Enterprise Suite):

- Server—The BusinessEvents runtime engine.
- Workbench—A TIBCO Designer[™] palette of BusinessEvents resources.
- TIBCO ActiveMatrix BusinessWorks 5.x Plug-in—A TIBCO Designer palette of activities that enables communication between BusinessEvents and ActiveMatrix BusinessWorks. (When you select this option, BusinessEvents Workbench and Server are also automatically selected.)
- Documentation—TIBCO BusinessEvents documentation. The `doc` folder contains an HTML and a PDF folder. If you do not install documentation, this folder is not included in the installation.

Enterprise Suite Only

All of the above components plus the following:

- Decision Manager application—A business user rule-building application.



The Decision Manager application is available only on Windows.

- Rules Management Server—A rules server for the Decision Manager application.
- Query—A language and set of functions for querying cache data.
- Database Concepts—A utility for creating concepts from database metadata, with functions for updating the associated database tables or views.
- State Modeler—A component that enables you to model the life cycle of concept instances.

Related Documentation

This section lists documentation resources you may find useful.

TIBCO BusinessEvents Documentation

- *TIBCO BusinessEvents Installation*: Read this manual for instructions on site preparation and installation.
- *TIBCO BusinessEvents Getting Started*: After the product is installed, use this manual to learn the basics of BusinessEvents. This guide provides step-by-step instructions to implement an example project and also explains the main ideas so you gain understanding as well as practical knowledge.
- *TIBCO BusinessEvents User's Guide*: Read this manual for instructions on using TIBCO BusinessEvents to create, manage, and monitor complex event processing projects.
- *TIBCO BusinessEvents Decision Manager*: This manual explains how to use decision tables to create rules using a spreadsheet-like interface, as well as how to administer the Rules Management Server.
- *TIBCO BusinessEvents Language Reference*: This manual provides reference and usage information for the BusinessEvents rule language and the BusinessEvents query language.
- *TIBCO BusinessEvents Cache Configuration Guide*: This online reference is available from the HTML documentation interface. It provides configuration details for cache-based object management. Cache-based object management is explained in *TIBCO BusinessEvents User's Guide*.
- *TIBCO BusinessEvents Java API Reference*: This online reference is available from the HTML documentation interface. It provides the Javadoc-based documentation for the BusinessEvents API.
- *TIBCO BusinessEvents Functions Reference*: This online reference is available from the HTML documentation interface. It provides a listing of all functions provided with BusinessEvents, showing the same details as the tooltips available in the TIBCO Designer rule editor interface.
- *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.

Other TIBCO Product Documentation

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

- TIBCO ActiveMatrix BusinessWorks™
- TIBCO Rendezvous®
- TIBCO Enterprise Message Service™
- TIBCO Designer™
- TIBCO Hawk™
- TIBCO Runtime Agent™

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

| Convention | Use |
|-------------------------------------|---|
| <i>TIBCO_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 C:\tibco.</p> <p>Other TIBCO products are installed into an installation environment. Incompatible products and multiple instances of the same product are installed into different installation environments. The directory into which such products are installed 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 C:\tibco.</p> <p>TIBCO BusinessEvents installs into a version-specific directory within <i>TIBCO_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 C:\tibco\be\3.0.</p> |
| code font | <p>Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example:</p> <p>Use MyCommand to start the foo process.</p> |
| bold code font | <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 admin. • 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, MyCommand is enabled: MyCommand [enable disable] |

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> |
|  | <p>The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.</p> |
|  | <p>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.</p> |
|  | <p>The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.</p> |

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 param1 and param2, or the pair param3 and param4.</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command requires two parameters. The first parameter can be either param1 or param2 and the second can be either param3 or param4:</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 param1. You can optionally include param2 as the second parameter. And the last parameter is either param3 or param4.</p> <pre>MyCommand param1 [param2] {param3 param4}</pre> |

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.

Chapter 1 **Introduction**

This brief chapter provides an overview of the tutorial provided in this guide.

Topics

- [Overview, page 2](#)
- [Tutorial Scenario, page 3](#)

Overview

This guide contains three tutorials based on one simplified business scenario. The tutorials provides step-by-step instructions and also explain the main ideas so you gain understanding as well as practical knowledge. References to related information are provided so you can jump to the main documentation on any topic to learn more.

The tutorial does not use any state modeler features, so it can be used with both the TIBCO BusinessEvents Inference Edition or the TIBCO BusinessEvents Enterprise Suite software.

Chapter 2, Project Design Tutorial, on page 5 This tutorial shows you how to configure a BusinessEvents project, run it at the command line, and test its behavior. Using a simple scenario, this tutorial focuses on inferencing features, which are central to BusinessEvents. See the section, Understanding Conflict Resolution and Run to Completion Cycles in *TIBCO BusinessEvents User's Guide*.

Chapter 3, Cache Object Management Tutorial, on page 41 This tutorial is based on the [Project Design Tutorial](#) and shows you how to implement cache-based object management. The tutorial shows you how to configure and deploy two cache server nodes and three engines running inference agents, to see how multi-engine, load balancing and fault tolerance features work. Requires use of TIBCO Enterprise Message Service software for point-to-point messaging.

Chapter 4, Backing Store Tutorial, on page 63 This tutorial shows you how to add backing store support to the project you configured in the [Cache Object Management Tutorial](#). Adding a backing store enables the deployed application to recover from total system failure. This tutorial requires use of Oracle 10G software (see the product readme for version information) or the Express Edition, which you can download from the Oracle web site.

Example Projects Provide Additional Learning

After you have completed the tutorial, you can explore the examples in the *BE_HOME/Examples* directory. These examples demonstrate specific techniques that you can apply in your work. For example, you can learn how to implement cache object management with a backing store.

Skills Required

The tutorial is written for users with little or no familiarity with TIBCO products. Readers should have some familiarity with Java programming.

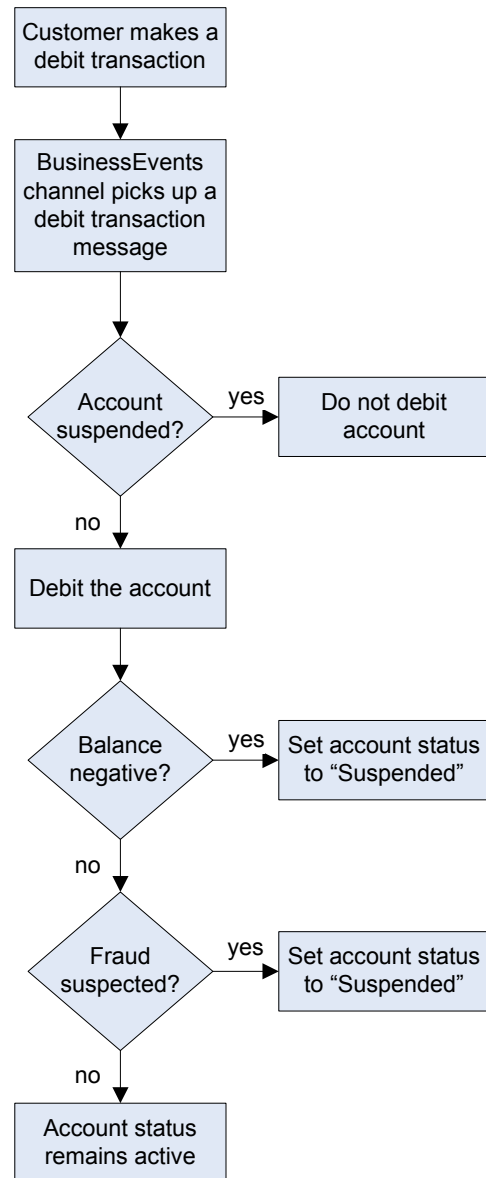
Tutorial Scenario

The tutorial is built on a simplified fraud detection scenario and decision making flow, illustrated by the following flow chart.

To establish whether fraud is suspected, the runtime engine correlates the frequency of and amount of debits in a rolling time window, and flags accounts that satisfy both of the following criteria:

- The account incurs more than three debit transactions in a two minute period.
- The sum of the debits that occurred in the two minute period totals more than 80% of the average monthly balance of the account.

For the purpose of the tutorial, messages arrive from a ActiveMatrix BusinessWorks process running in test mode. The event correlation is performed using simple BusinessEvents rules. Suspicious accounts are simply set to "Suspended." Actions are printed to the monitor so you can see the project in action.



Chapter 2

Project Design Tutorial

This tutorial takes you from start to finish through all the steps of configuring, building, deploying, and testing a BusinessEvents project. The emphasis is on basic project design, with deploytime activities limited to the basic actions required to test a design.

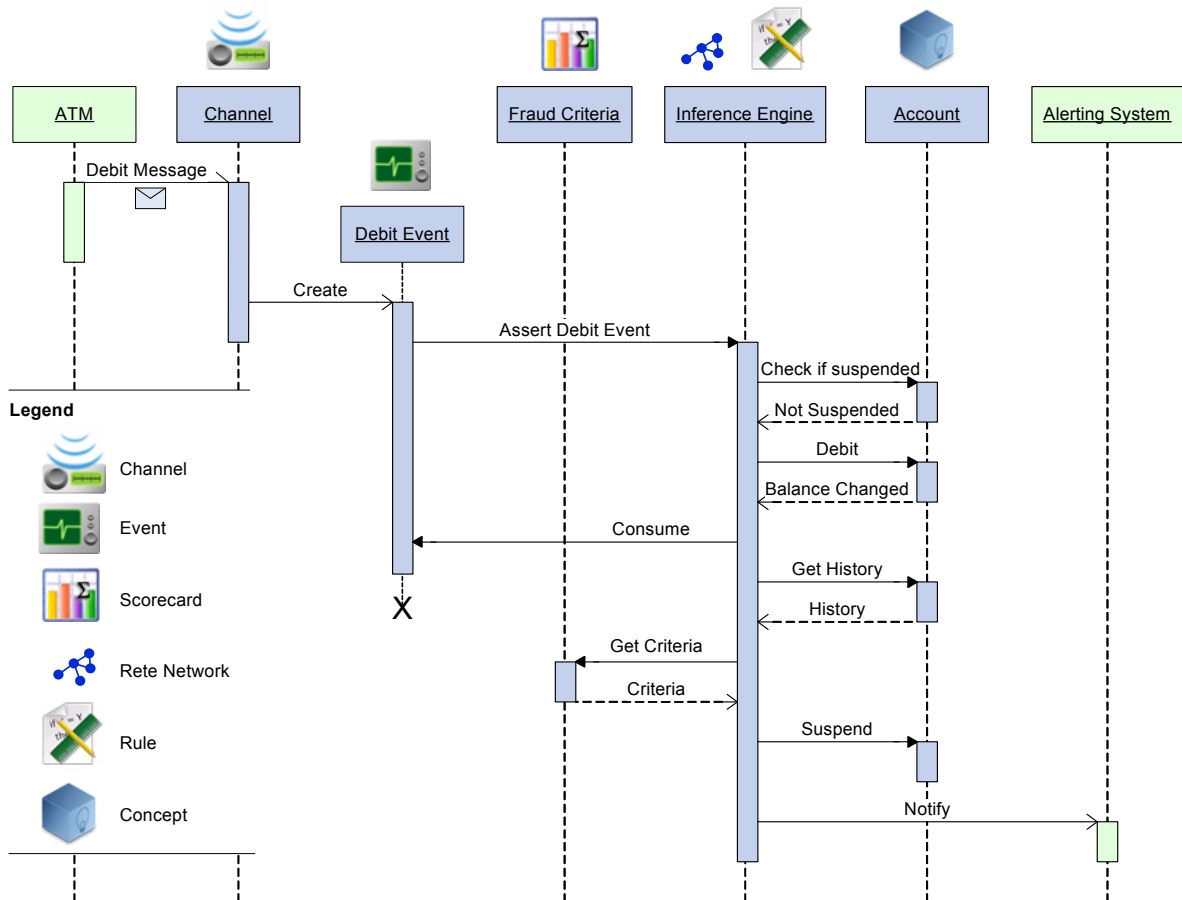
Topics

- *[Fraud Detection Runtime Flow, page 6](#)*
- *[Create the FraudDetection Project, page 8](#)*
- *[Create a Rendezvous Channel and a Destination, page 11](#)*
- *[Define the Debit Event, page 13](#)*
- *[Validate the Project and Set a Default Event for a Destination, page 15](#)*
- *[Define the Account Concept, page 17](#)*
- *[Create the FraudCriteria Scorecard, page 19](#)*
- *[Configure the InitializeAccount Rule Function, page 20](#)*
- *[Configure the ProcessDebits Rule Set and ApplyDebit Rule, page 25](#)*
- *[Configure the FraudDetection Rule, page 28](#)*
- *[Configure the CheckNegativeBalance Rule, page 30](#)*
- *[Validate the Project, Configure and Build the Archive, page 31](#)*
- *[Configure an ActiveMatrix BusinessWorks Process for Testing, page 34](#)*
- *[Deploy the Project, page 38](#)*
- *[Test the Application, page 39](#)*

Fraud Detection Runtime Flow

The sequence diagram below summarizes what happens when you exercise the completed tutorial project. You will learn more the details as you complete the tutorial steps.

In summary: A debit message for a certain account arrives through a BusinessEvents channel. As a result, a Debit event is created and asserted into working memory, which triggers a rule to apply the debit to the account. The debit, combined with recent debits for that account, triggers the fraud detection rule to set the account status to "Suspended" and send out an alert. See [Tutorial Scenario on page 3](#) for details about what causes the status to become Suspended.



Reading the above diagram from left to right, you can see that the following occurs:

1. A debit message arriving through a BusinessEvents *channel* is transformed into a debit *event*. (At design time you create an event type for this purpose, with the appropriate properties.) The debit event instance is then *asserted into working memory*. This means that it is added to the *Rete network*, an in-memory network of objects based on the Rete algorithm which enables fast matching of facts with rule dependencies.
2. The presence of this new event in the working memory causes the inference engine to check for rules that are designed to be triggered when this event is asserted.
3. A Debit rule checks if the account status is "Suspended" and if not, debit the account balance by the debit amount. The rule then consumes the event. It is important to consume events when they are no longer needed so that they don't trigger rules to fire when the conditions are met again.
4. A FraudDetection rule checks the recent account activity, stored as history of the Account concept, and the fraud detection criteria stored in a scorecard. The rule calculates if the frequency and amount of the debits signals fraud.
5. A CheckNegativeBalance rule checks the account balance after a debit and if it is negative (less than zero) the rule sets the account status to "Suspended." This rule is not triggered in the example shown, because the balance has not fallen below zero.
6. In the example shown above, fraud is detected. The rule sets the Account status to Suspended and it sends out an alert. In a real world scenario, this alert would go to the desktop of an employee for follow up. To keep the tutorial simple, the rule just prints a message to the console.

Create the FraudDetection Project

In this task, you start TIBCO Designer and create an empty project.

Learning Points

What is TIBCO Designer? TIBCO Designer is a project-building user interface used by many TIBCO products, including BusinessEvents. A basic guide to the use of TIBCO Designer is provided in [Appendix A, Getting Started With TIBCO Designer, on page 75](#).


How should I organize project folders? Use folders to organize your project components in any way you like. Example projects are kept simple and use folder names such as Concepts, Events, Rules. More complex projects might use a deeper folder hierarchy with names that relate to the purpose or contents of the folders.

What are palettes? TIBCO Designer organizes resources used to build a project into palettes. Palettes are provided for each product you install. Because ActiveMatrix BusinessWorks is provided with BusinessEvents for use in development (but not production), you see all the ActiveMatrix BusinessWorks palettes, as well as the BusinessEvents ones.

BusinessEvents adds two palettes to those available for TIBCO Designer and TIBCO ActiveMatrix BusinessWorks:

- The BusinessEvents Activities palette is used for ActiveMatrix BusinessWorks integration projects.
- The BusinessEvents Workbench palette provides the resources you need to build a BusinessEvents project. Click the **Palettes** tab (on the left), then select the **BusinessEvents Workbench** palette to see the resources.

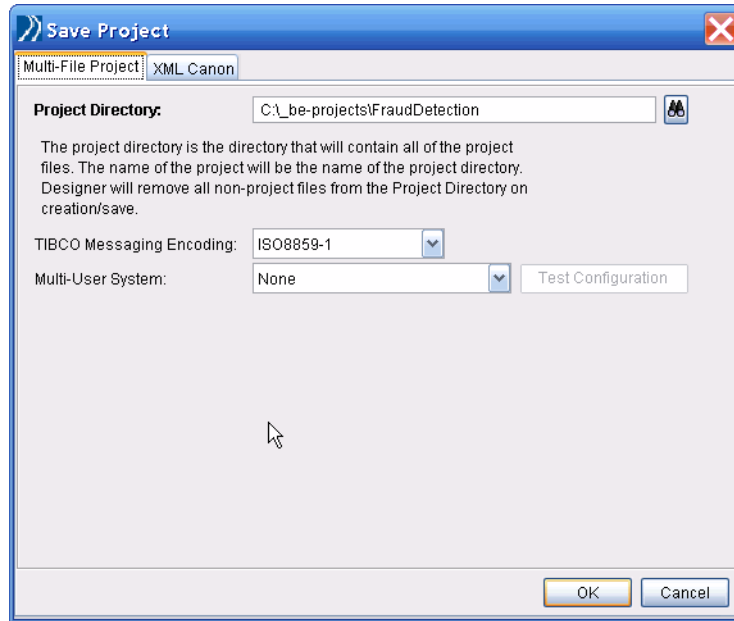



If the palette names are not visible, click the Switch Palette Modes button () to display them.

How do I choose a palette resource? You can choose a resource by dragging its icon from the palette panel to the design panel, or you can right-click and choose the resource from menus. The palettes and resources you see depend on the context. Only those that present valid choices for your current context are shown.

Task A Create the Fraud Detection Project

1. Click **Start > Programs > TIBCO > TIBCO Designer version > Designer version**.
2. From the Project menu, select **New Project > New Empty Project**. You see the Save Project dialog:



3. In the Multi-File Project tab of the Save Project dialog box, click the browse button () and create a directory in the location where you want to save your project files. Name the directory **FraudDetection**.
4. In the TIBCO Message Encoding field, select **ISO8859-1** if it is not already selected.

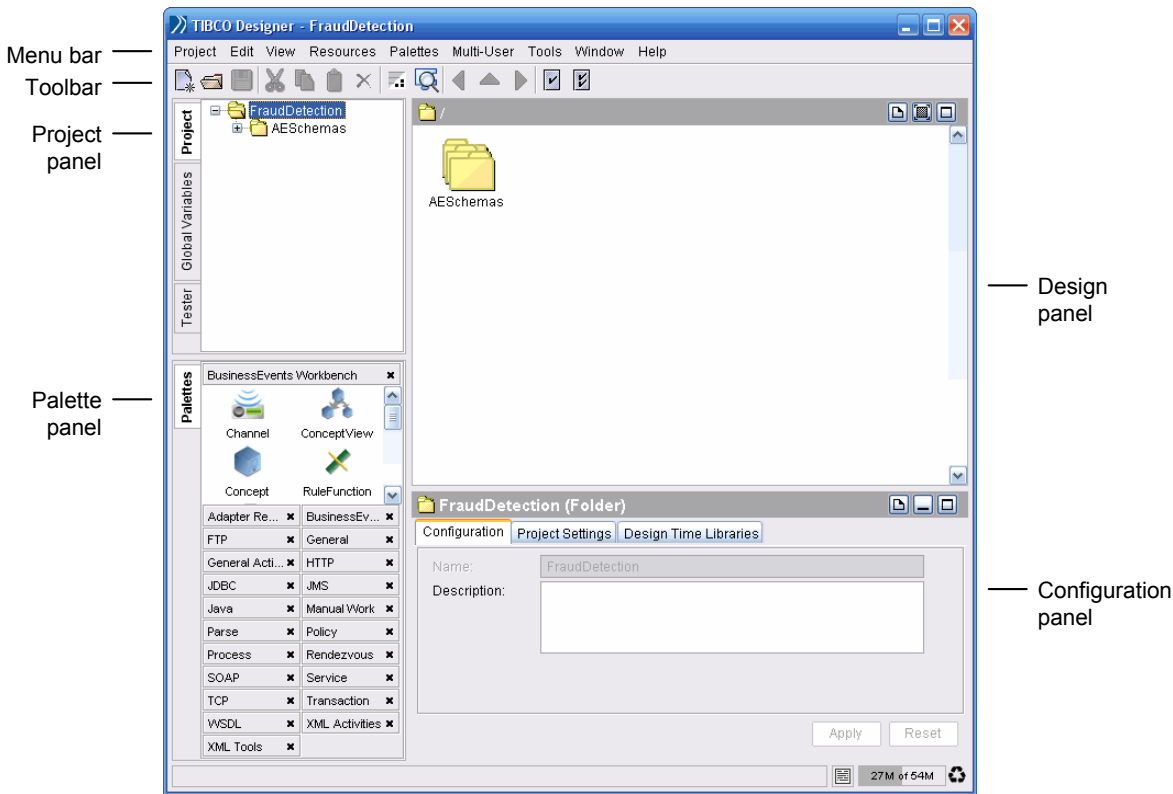


Encoding In order to deploy a BusinessEvents project as an application in a TIBCO Administrator domain, the project encoding must match the domain encoding. By default, TIBCO Administrator domains use ISO8859-1 encoding.

5. Click **OK**.

You see the TIBCO Designer user interface. In the Project tab, the root folder has the same name as the directory you created, and one subfolder called AESchemas is created automatically.

Your display looks similar to this (you may be using a different number of panels). Names of the panels shown below are used throughout documentation. BusinessEvents Workbench palette has been selected:



Summary and Next Steps

You have created a new empty project in TIBCO Designer. The process is the same for all TIBCO products that use the TIBCO Designer interface. Use of a common project-building tool shortens your learning curve and enables you to integrate different TIBCO products to create useful applications.

Next you will begin to define your BusinessEvents project by building a channel for information to enter the system, and a destination for the system to listen to.

The order in which you build up the project is not important. For example, you might decide that first you want to define all the objects (ontology) for the project.

Create a Rendezvous Channel and a Destination

In this task you configure a Rendezvous channel with one destination, the DebitTransaction destination, to listen for account debit messages.

Learning Points

What are channels and destinations? Messages enter and leave the system through channels. You create destinations within a channel to define the message sources and sinks. Events are created using data in incoming messages, and outgoing messages are created using data from events. Later in the tutorial, you will set up the relationship between the DebitTransaction destination and an event type.

Note that in this tutorial outbound messages are simply sent to the console, so there are no outbound destinations.

How are channels and destinations created? You create channels and destinations at design-time, as explained below. When you are planning BusinessEvents projects, you would consider the incoming and outgoing messages for your project, and then define the channels, destinations, and the corresponding event types.

More Information

Chapter 2, Working With Channels and Destinations in *TIBCO BusinessEvents User's Guide*.

Task B Create a Rendezvous Channel and Destination

1. Select the project (root) folder. In the design panel, right-click and select **New Folder**. A folder icon appears in the design panel.
2. Click the folder icon to select it, and in the Configuration tab, name the folder **Channels**. Click **Apply**.



Apply and Save Clicking Apply saves changes in memory but not on disk. If you want to close the project and open it at a later time, remember to select Project > Save (or click the Save button). You are also prompted to save unsaved work when you select Project > Exit.

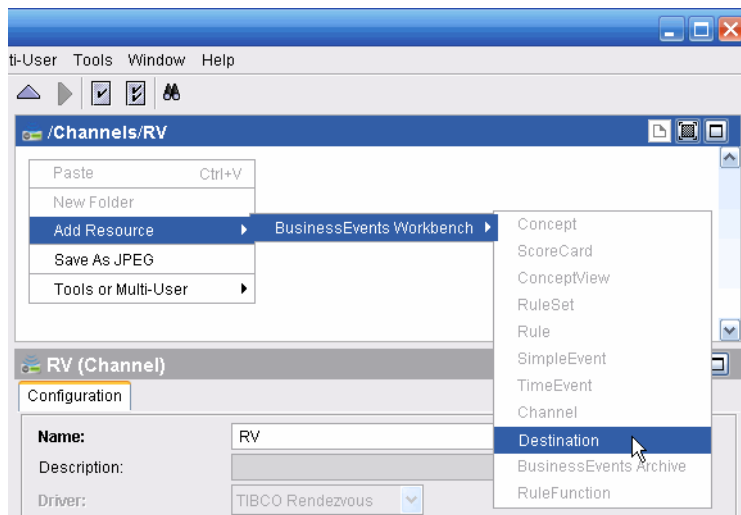
3. Open the Channels folder, right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > Channel**.

4. In the Configuration tab, name the channel **RV**, and from the Method of Configuration drop-down list, select **Properties**. Leave the Service, Network, and Daemon fields blank so that default values are used. Click **Apply**.



Resource names and directory names in the path to a resource can't be any of the keywords or other words listed in the section Keywords and other Reserved Words in *TIBCO BusinessEvents Language Reference*, and they can't use spaces.

5. Double-click the **RV** channel in the design panel. It opens like a folder so you can add multiple destinations within it.
6. Right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > Destination**.



Notice that all irrelevant resources are dimmed. Within a channel, the only type of resource you can add is a Destination resource.

7. Name the destination **DebitTransaction**.
8. In the Subject field, type **BE.DEBIT.TXN**. Click **Apply**. The destination listens for Rendezvous messages with this subject.
9. Click **Apply** and save the project.

Summary and Next Steps

Now you have built a channel and a destination within that channel to listen for messages. Later in the tutorial, you will create a ActiveMatrix BusinessWorks process and publish messages that will be picked up by this destination so you can see the project in action.

The next step is to create an event.

Define the Debit Event

In this task, you begin to build the project ontology by defining an event.

Learning Points

What is an event? The term *event* is overloaded: it means an activity that happens, and the definition of an object that represents the activity in BusinessEvents (also known as an event type), and also an instance of that event definition.

How are events (event instances) created? Event types are created at design time. Event instances are created using data in incoming messages. When a destination receives a message, it creates an event to hold the information from the message. Events can also be created by rules. Events are automatically asserted into working memory, where their presence will trigger rules (if all rule conditions are met). Outgoing messages are created using events that are sent to destinations.

What is an event payload? Just as messages have properties and a message body, events can have properties and payloads. The payload is optional. It is used to hold more complex data. The Debit event does not use a payload. To learn about payloads, see Chapter 3, Working With Simple Events of *TIBCO BusinessEvents User's Guide*.

What is a default destination? Events of a certain event type are often sent to the same destination. To simplify the process of sending those events, you can specify a default destination in the event type. You can send an event to the default destination of its event type using the `Event.sendEvent()` function. (As needed, you can send an event to a specific destination using the `Event.routeTo()` function which lets you specify the target destination.)

When you test the project (later), you will configure a Send Event activity, which will make use of the default destination.

More Information


Chapter 3, Working With Simple Events, Chapter 4, Working With Time Events, Chapter 5, Working With Advisory Events, in *TIBCO BusinessEvents User's Guide*.

Task C Define the Debit Event

1. In the FraudDetection project folder, create a folder and name it **Events**.
2. Open the Events folder, right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > SimpleEvent** and name the event **Debit**.



To open a folder, either click the folder in the project tree or double click the folder icon in the design panel.

3. In the Default Destination field, click the browse button () and in the Select a Resource dialog, select **/Channels/RV.channel/DebitTransaction**. Click **OK** to dismiss that dialog, then click **Apply**.
4. Select the **Properties** tab for Debit, add the following properties and click **Apply** and **Save**.

| Name | Type |
|-----------------|--------|
| TransactionType | String |
| AccountId | String |
| Amount | double |



Remember to save the project frequently.

Summary and Next Steps

Now you have defined an event type. When you configure events in your working projects, you will examine the messages that you want to listen for, and configure the event characteristics accordingly.


Next you will set Debit as the default event for the DebitTransaction destination—so that a Debit event instance will be created and asserted into working memory whenever a message arrives at the DebitTransaction destination.


Validate the Project and Set a Default Event for a Destination

In this task you learn how to define a default event type for a destination, and what that means. You also learn to validate your project.


Learning Points

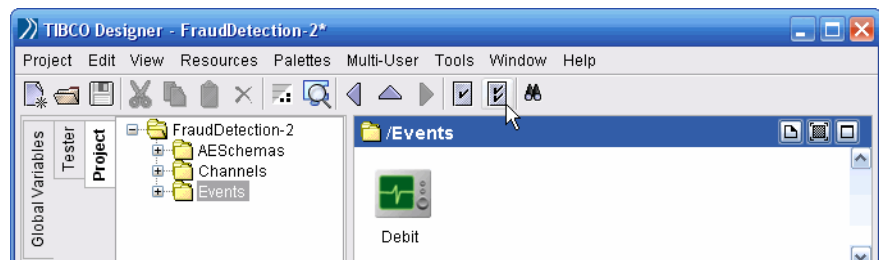
What is a default event? The default event configured for a destination is used to hold information transferred from an incoming message, when no event type is specified in the message. The event is then asserted into working memory, where it may trigger rules.


What is validation? The Validate All () feature checks the whole project, looking for inconsistencies and incompletely configured resources. If there are errors, the validation feature makes it easy to correct them. If you rename a resource, for example, BusinessEvents can correct all references to the resource, except those in text areas such as the rule editor.

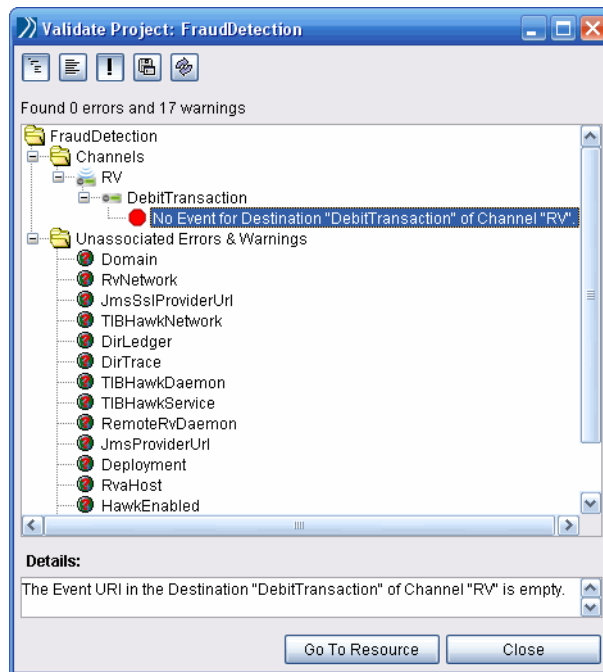
The Validate Resource feature () checks the validity of the currently displayed resource, and resources referenced in the currently displayed resource. However, it does not check whether the current resource is correctly referenced elsewhere in the project.

Task D Validate the Project and Set a Default Event for a Destination

1. Click the Validate All () button:




You see the Validate Project dialog, showing the following warning. (If you don't see any the warnings, click the Show Warnings () button):



Warnings provide information and might not indicate a problem. For example, many global variables are provided by default, and you may not need to use any of them.

When you run this validation check, you see a warning that the DebitTransaction destination does not have a default event. Providing one is the next step in the tutorial.

For many projects, default event and default destination settings are useful, but they are not required for all projects.

2. In the Validate Project dialog, click the text beside the red circle (directly below DebitTransaction, and then click the **Go To Resource** button. The Configuration tab for the DebitTransaction destination displays.
3. In the Default Event field, click the browse button () and select **Debit**, then click **Apply** and **Save**.

Summary and Next Steps

Now instances of the Debit event type will be created and asserted into working memory whenever a message arrives at the DebitTransaction destination.

Next you will continue to configure the project ontology by defining a concept.

Define the Account Concept

In this task, you define the Account concept, which holds basic information about an account: an ID, a balance, an average monthly balance, and an account status. You also learn some useful information about concepts and how they are used.

Learning Points

What is a concept? A concept type is a definition of a set of properties that represent the data fields of an entity. Concept types are like Java classes, and concept instances are like Java objects.

How are concept instances created? Concept instances are created by rules and rule functions. For example, when you start the runtime engine, an instance of the Account concept is created by the InitializeAccounts rule function.

How can I manage concept instances? Instances of concepts and events (also known as "facts" and "entities") can be persisted in various ways, as determined by the business need. See *TIBCO BusinessEvents User's Guide* and provided examples for more details.

How is history tracked? When the History setting for a concept property is 0 (zero) the current value is stored without a date-time stamp. When the history setting is 1, the current value is stored, along with the date and time the value was added or changed. When the history value is greater than 1, BusinessEvents tracks changes to property values up to the specified number (using a ring buffer). The Policy setting additionally determines what values are recorded, all values or only changes to the prior value.

For the Debits property, BusinessEvents records "All Values," that is, it 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—a person can debit the account twice by the same amount. For a property such as "address" you might want to track only changes to the value.

More Information

Chapter 6, Working With Concepts, in *TIBCO BusinessEvents User's Guide*.

Task E Define the Account Concept

1. In the FraudDetection project folder, create a folder and name it **Concepts**.
2. Open the Concepts folder, right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > Concept**. (You can also select a resource by dragging it from its palette into the design panel.)
3. A concept icon appears in the design panel. In the Configuration tab, name the concept **Account** and click **Apply**.

- Select the **Properties** tab for the Account concept, and create the following properties. Click the plus button (+) to add lines:

| Name | Type | Policy | History |
|-------------------|--------|--------------|---------|
| Balance | double | Changes Only | 1 |
| Debits | double | All Values | 5 |
| Status | String | Changes Only | 1 |
| AvgMonthlyBalance | double | Changes Only | 1 |

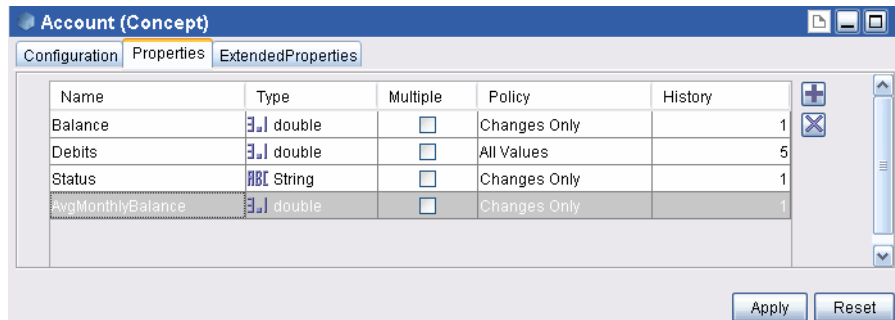


Concept Attributes and Concept Relationships

Where is the account ID?— in the @extId attribute Concepts, events, and scorecards have some built-in attributes, in addition to the properties you define here. The attribute `extId`, referenced as `@extId`, will hold the account ID.

Concept Relationships Concepts can have containment and reference relationships to other concepts. You set them up at the dialog where you selected the Double data type. Concepts can also inherit from other concepts. that relationship is set up on the Configuration tab. See Chapter 6, Working With Concepts in *TIBCO BusinessEvents User's Guide* for more details.

- The Properties tab now looks like this:



- Click **Apply**, then **Save** the project.

Summary and Next Steps

You have defined a concept type to hold information about bank accounts. The last step in building the ontology of your project is to set up a scorecard to hold fraud detection criteria that are used in rules.

Create the FraudCriteria Scorecard

In this task, you finish building the project ontology by creating a scorecard.

Learning Points

What is a scorecard? A scorecard is a special type of concept. A scorecard serves as a static variable. You can use a scorecard resource to track key performance indicators or any other information.

The FraudCriteria scorecard will store the criteria used to determine fraud, not any specific data about customer accounts. In this example, you will use this scorecard in rules.

How are scorecards created? Unlike concepts, there is only one instance of a scorecard. 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 (rule session). This tutorial uses one inference agent. However, in the next tutorial you will deploy multiple agents in a group, and each has its own instance of the scorecard.

Task F Create the FraudCriteria Scorecard

1. At the project level, right-click in the design panel and select **Add Resource > BusinessEvents Workbench > ScoreCard**. (There is only one scorecard in this project, so we don't need a folder for it).
2. Name the scorecard **FraudCriteria** and click **Apply**.
3. Select the **Properties** tab for **FraudCriteria** and create the following properties:

| Name | Type | Policy | History |
|----------------|--------|--------------|---------|
| interval | long | Changes Only | 0 |
| num_txns | int | Changes Only | 0 |
| debits_percent | double | Changes Only | 0 |

4. Click **Apply** and save the project.

Summary and Next Steps

You have now set up the channel for the project, and its ontology, that is, the definitions of all the entity types (events, concepts, and scorecards) that are needed to store information (facts) about possible fraud detection.

Next, you will configure a rule function that sets values for the scorecard and creates an instance of the Account concept.

Configure the InitializeAccount Rule Function

In this task, you configure a rule function that you will later set to execute at startup. The InitializeAccount rule function sets values for the FraudCriteria scorecard properties. It also creates an instance of the Account concept.

Learning Points

In the InitializeAccount rule function you use an ontology function to create a concept instance.

What is a rule function? A rule function is a function you write in the BusinessEvents rule language.

How are rule functions created and used? You write rule functions using the rule editor. You can use rule functions in rules and other rule functions, in event preprocessors, and as startup or shutdown functions for an agent.

What is an ontology function? An ontology function is automatically created for each concept and event in your project. These are constructor functions. Another type of ontology function enables you to create and schedule a time event.

What other types of functions are there? BusinessEvents provides a large library of functions. You can also add custom Java functions. See Chapter 4, Creating Custom Functions in *TIBCO BusinessEvents Language Reference* for details.

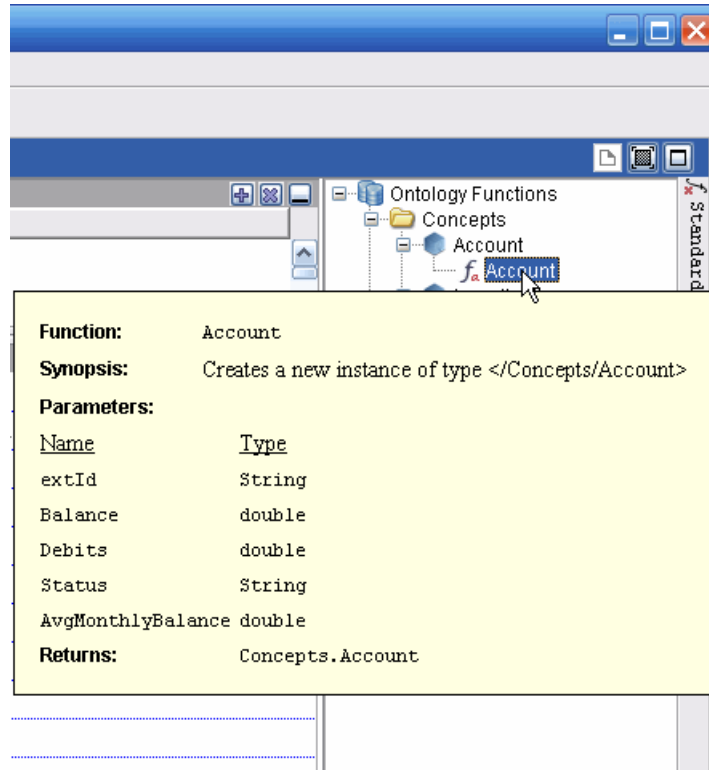
Task G Configure the InitializeAccount Rule Function

1. At the project level, create a **Rules** folder.
2. Open the Rules folder, right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > RuleFunction**.
3. Name the rule function **InitializeAccounts** and click **Apply**.
4. In the design panel, double-click **InitializeAccounts** to open it.
You see the rule function editor.
5. Add the following lines to provide values to the FraudCriteria scorecard (and to comment your code):

```
//Intialize scorecard variables
FraudCriteria.debits_percent = .8;
FraudCriteria.interval      = 120*1000; /* 120 seconds */
FraudCriteria.num_txns = 3;
```

Notice that when you type the period after `FraudCriteria`, a list of its properties pops up so you can select a property.

6. On the right side of the user interface, click the vertical Ontology tab and expand the list under Ontology Functions. You can make the function area larger to see the list more easily. You see the same folder structure as you used to build the ontology.
7. Hover the cursor over the Account ontology function to see a tooltip showing its properties:



8. Drag the ontology function corresponding to the Account concept onto the design panel. The following signature is provided for you:

```
Concepts.Account.Account(/*extId String *//*Balance double
*//*Debits double *//*Status String *//*AvgMonthlyBalance
double */)
```

9. Configure the signature so that you provide the following initial values for each property.

```
extID: ActA
Balance: 20000.0
```

Debits: 0
 Status: **Normal**
 AvgMonthlyBalance: **10000.0**

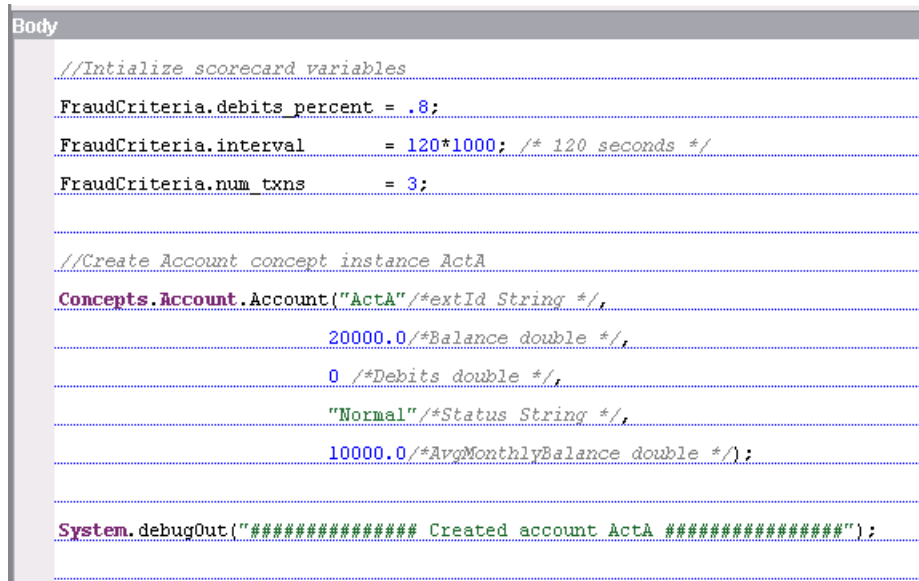
Document your code with a comment. Your display should look similar to the following:

```
//Create Account concept instance ActA
Concepts.Account.Account("ActA" /*extId String */,
                        20000.0 /*Balance double */,
                        0 /*Debits double */,
                        "Normal" /*Status String */,
                        10000.0 /*AvgMonthlyBalance double */);
```

10. Now enter the following. Notice again, that when you type the period after System you can select debugOut from a list. The hash marks, by the way, are just to make the message more visible when printed to the console.

```
System.debugOut("#####Created account ActA
#####");
```

The completed values look like this in the editor:



The screenshot shows a code editor with a tab labeled 'Body'. The code is as follows:

```
//Intialize scorecard variables
FraudCriteria.debits_percent = .8;
FraudCriteria.interval      = 120*1000; /* 120 seconds */
FraudCriteria.num_txns      = 3;

//Create Account concept instance ActA
Concepts.Account.Account("ActA" /*extId String */,
                        20000.0 /*Balance double */,
                        0 /*Debits double */,
                        "Normal" /*Status String */,
                        10000.0 /*AvgMonthlyBalance double */);

System.debugOut("##### Created account ActA #####");
```

11. Click **Apply**, then save the project.

Optional Exercise: Create a Concept Instance Using a Standard Function

Another way to create an instance of a concept is to use a standard function called `Instance.createInstance()`. If you want to learn about this feature, complete this exercise to create a second instance of the Account concept.

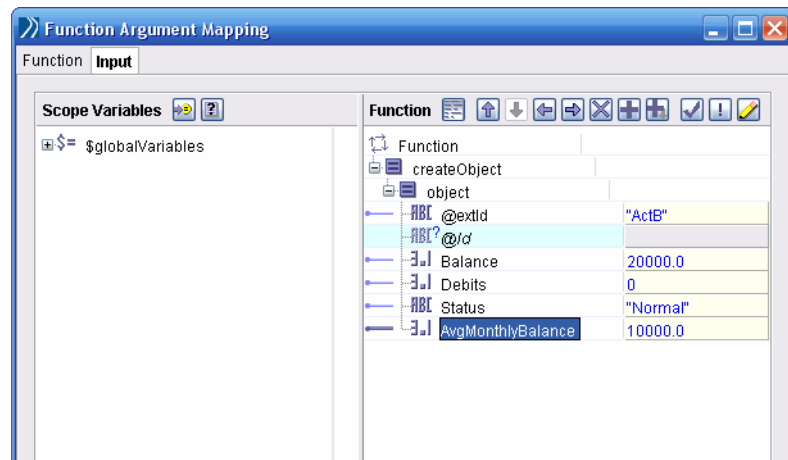
1. Put the cursor on a new line in the rule editor and click the vertical Standard tab (on the right side of the TIBCO Designer window) to view the library of standard functions.
2. Expand the list so you can see `Instance.createInstance()`.

Note that the function name has a small "m" to the left of the "f" icon. The "m" indicates a *mapper* function.

3. Drag the `createInstance` function to the design panel and type an open parenthesis at the end of the name. A link to the XSLT mapper appears:

```
Instance.createInstance(<<xslt-template>>
```

4. Click in the link to display the Function Argument Mapping dialog.
5. First you'll select the type of concept you want to create an instance of. In the Entity Path field of the Function tab, browse to and select the concept Account, then click **Apply**.
6. Click the Input tab to see the Scope Variables and Function panels. Expand the tree in the Function panel and enter values for ActB. Except for the account ID (@extId), you can use the same values you entered for ActA. Your display looks like this:



You can also use global variables to set initial values at runtime. To use a global variable, you expand the Scope Variables panel and drag the cursor from the variable name to the property or attribute name.

7. Click **Apply** then click **OK** to dismiss the window. Close the parentheses and end the line with a semicolon. The finished line looks like this in the rule editor (with additional colors and formatting):

```
Instance.createInstance("/Concepts/Account");
```

You can click inside the parentheses to view or change the details, as desired.

8. Comment your code for readability. Create a line above the line you added and type:

```
//Create Account concept instance ActB using XSLT Mapper
```

9. So that you know when your second account is created at runtime, type the following:

```
System.debugOut("##### Also created account actB");
```

Summary and Next Steps

You have configured a rule function that will initialize an account (optionally two accounts) at runtime. Next you will configure rules that take action on arrival of debits, depending on whether the fraud criteria are met or not.

Configure the ProcessDebits Rule Set and ApplyDebit Rule

In this task you create a rule set called ProcessDebits, and in it, create a rule called ApplyDebit. This rule fires when certain conditions are met.

When a Debit event is asserted into working memory, the engine checks the ApplyDebit rule because it has a Debit event in its scope. The engine checks the rule conditions. If working memory contains an Account concept instance whose ID matches the ID in the debit, the rule is eligible to execute. When the rule executes, if the account status is not “Suspended,” the rule action debits the account instance by the amount specified in the Debit event.

Learning Points

What is a rule set? A rule set is a collection of rules. Rules must belong to a rule set so that when you build the project EAR for deployment, you can select which rule sets to include, and you can select different rule sets to be used in different rule sessions (inference agents). Rule functions, on the other hand, do not belong to rule sets. A rule function is available in all rule sessions (inference agents).

What are rules and how are they created? Rules define actions to take when certain conditions are met. Rules are written in the BusinessEvents rule language, which is similar to the Java language. Rules are declarative and are generally narrow in scope. The rule editor UI has three panels: the Declaration panel, the Conditions panel, and the Actions panel, which you will use in the next task.

How are rules used at runtime? The rule engine checks all changes and additions to working memory and evaluates or reevaluates rules, using their Declaration and Conditions as appropriate. Eligible rules are added to the rule agenda.

What is the rule agenda A rule fires when it is at the top of the agenda. The engine determines the order of firing using rule declarations and conditions, and each rule’s priority. As the contents of working memory change, the engine reevaluates rules and removes any that are no longer eligible to fire. See Understanding Conflict Resolution and Run to Completion Cycles in *TIBCO BusinessEvents User’s Guide* for details.

Task H Configure the ProcessDebits Ruleset and ApplyDebit Rule

1. Open the Rules folder, right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > RuleSet**. Name the rule set **ProcessDebits**.
2. With the rule set open, right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > Rule**. Name the rule **ApplyDebit**.
3. In the Configuration tab, set the Priority field to **1** and click **Apply**.

The **Priority** setting is used by the runtime engine when determining the order in which rules are fired. Those 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.

- 4. Double-click the **ApplyDebit** rule icon to open the rule editor.
- 5. Drag the Debit event from the project tree into the Declaration panel. Similarly drag the Account concept into the Declaration panel.

The Declaration panel now looks like this:

| Declaration | |
|---|---------|
| Term | Alias |
|  /Events/Debit | debit |
|  /Concepts/Account | account |

Declaration The Declaration provides the scope of the rule. It lists all the entity types to be used in the rule, and their aliases. By default the alias is set to the entity name, but you can change it as desired.

- 6. In the Conditions panel, type the following:

```
//Checks whether the extId of an Account instance in working memory
//matches the incoming event's account ID
account@extId == debit.AccountId;
```

Notice that when you type the At sign (@), a pick list of concept attributes appears. Attributes are built-in. You can't add or remove attributes.

- 7. In the Actions panel, just below the Conditions panel, type these statements (one line break below is added for clarity):

```
//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+ ">");
}
```

When the rule engine executes the above actions, it first checks whether the matching account is suspended. If it is not suspended, the engine performs the debit. In either case, it prints information to the console.

8. It is important to consume (delete) the event after it has served its purpose. At the end of the Action statements, type the following:

```
Event.consumeEvent(debit);
```

9. Click **Apply**, then save the project. The completed rule looks like this:



If you do not consume events when their work is done, their presence (along with other conditions) might trigger rules erroneously.

The screenshot displays the Rule Editor interface for the 'ApplyDebit' rule. The top section, titled '/Rules/ProcessDebits/ApplyDebit', contains a 'Declaration' table with two entries: '/Events/Debit' with alias 'debit' and '/Concepts/Account' with alias 'account'. Below this is the 'Conditions' section, which includes a comment '//Checks whether the extId of an Account instance in working memory' and '//matches the incoming event's account ID', followed by the condition `account@extId == debit.AccountId;`. The 'Actions' section contains a comment '//If Account Status is not Suspended, debits the account' and a code block that checks if the account status is not 'Suspended'. If true, it debits the account by updating `account.Debits`, `account.Balance`, and includes debug output statements. If false, it outputs a message indicating the account is suspended. Finally, it calls `Event.consumeEvent(debit);`. The bottom section, 'ApplyDebit (Rule) Configuration', shows the rule's name as 'ApplyDebit', a description field, and a priority of '1 (Highest)'. 'Apply' and 'Reset' buttons are at the bottom right.

```

//Checks whether the extId of an Account instance in working memory
//matches the incoming event's account ID
account@extId == debit.AccountId;

//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);

```

ApplyDebit (Rule) Configuration

Name: ApplyDebit

Description:

Priority: 1 (Highest)

Apply Reset

Summary and Next Steps

In this task you have seen how to configure rules using the rule editor. The next task continues the activity of rule building. You will define the rule that implements the fraud detection tests.

Configure the FraudDetection Rule

Finally, you have arrived at the fraud detection rule itself. In the FraudDetection rule you will use a few standard functions, along with the values in an Account concept instance, and the FraudCriteria scorecard to calculate whether recent activity indicates possible fraud.

Task I Configure the FraudDetection Rule

This task provides summary instructions for actions you have already learned. If you need a reminder, refer to [Task H](#) for details.

1. Add another rule to the ProcessDebits rule set and name it **FraudDetection**.
2. Open the rule editor for the **FraudDetection** rule.
3. In the **Declaration** area, declare the **Account** concept.
In the Alias column, the alias is automatically set to account. The alias is used to refer to this concept in the rule action area.
4. In the top right of the user interface, click **Standard > Temporal > History** and then hover the cursor over howMany to see the signature of this standard function and understand the purpose of its parameters.
5. Drag the howMany() function into the Conditions area. The characters Temporal.History.howMany appear.
6. Complete typing the first condition as shown below. You can drag the DateTime functions onto the rule editor, or type the names. Add an annotation to make the rule easier to read.

```
// 1. Checks the number of debits in the verification interval
Temporal.History.howMany(account.Debits,
    DateTime.getTimeInMillis(DateTime.now())-FraudCriteria.interval,
    DateTime.getTimeInMillis(DateTime.now()),
    true)
> FraudCriteria.num_txns;
```

The above condition checks whether the number of debits in the specified interval prior to the current time is greater than the specified number of debits. The interval and the number of debits are set in the FraudCriteria scorecard.

7. Similarly, enter the second condition:

```
// 2. Checks the percentage of the average balance that was debited in the
verification interval
Temporal.Numeric.addAllHistoryDouble(account.Debits,
    DateTime.getTimeInMillis(DateTime.now())-FraudCriteria.interval)
    > FraudCriteria.debits_percent*account.AvgMonthlyBalance;
```

This condition checks whether the sum of all debits in the verification interval is greater than the specified percentage of the account average monthly balance. The specified percentage is set in the FraudCriteria scorecard. The average monthly balance, for the purposes of this tutorial, is set in the Account instance created by the InitializeAccounts rule function.

Note that a rule can specify multiple conditions and all conditions in a rule must be met, before the action is done. That is, each condition is joined by an implied "AND" operator.

8. In the Actions area, enter the following:

```
account.Status="Suspended";
System.debugOut("##### Account Id <"+account.extId+"> STATUS set to
<Suspended>. Fraud suspected.");
```

When an account debit event arrives, and the account fails the verification tests set in the conditions, the status is set to Suspended, and a message prints to the console.

9. Click **Apply** and save the project.

**Summary and
Next Steps**

You will define one more rule, to set the account status to Suspended if the account balance goes negative.

Configure the CheckNegativeBalance Rule

This rule simply suspends accounts whose balance goes negative, for obvious business reasons. When an account balance changes, this rule checks whether the the new balance is less than zero. If it is, the account is suspended.

Task J Configure the CheckNegativeBalance Rule

This task provides summary instructions for actions you have already learned. If you need a reminder, refer to [Task H](#) for details.

1. Add another rule to the ProcessDebits rule set and name it **CheckNegativeBalance**.
2. Open the rule editor for the **CheckNegativeBalance** rule.
3. In the **Declaration** area, declare the **Account** concept.
4. In the **Conditions** area, type the following:

```
//Checks that the balance is less than zero
account.Balance < 0;
//Checks that Account status is not set to Suspended
account.Status!="Suspended";
```

5. In the **Actions** area, type the following:

```
account.Status="Suspended";
System.debugOut("##### Account ID <"+account.extId+"> STATUS set to
<Suspended>. Balance <"+account.Balance+"> is less than zero");
```

6. Click **Apply** and save the project.

Summary and Next Steps

You have now configured the project's ontology and rules. Now you are ready to configure and build the archive, ready for deployment.

Validate the Project, Configure and Build the Archive

Learning Points

In this task you turn your attention to deployment configuration. You configure an Enterprise Archive Resource (EAR) and a BusinessEvents Archive Resource (BAR).

What is a BusinessEvents Archive (BAR)? A BAR contains all the deploytime data for one agent in a BusinessEvents project.



Multiple Agents (Rule Sessions) in an EAR Each BAR in an EAR deploys as one inference agent. You would configure each BAR as appropriate for each inference agent's needs.

What do you configure in a BAR? When you configure an inference agent's BAR for deployment, you can make various choices based on the resources in the project. If there are multiple BARs you can configure each one differently, according to need. Configuration choices determine the following:

- Which rule sets to deploy.
- Which destinations to listen to.
- Which rule functions to use (if any) as event preprocessors, and what threading to use.
- Which rule functions (if any) to execute at startup or shutdown (or both). See the topic *Working With Startup and Shutdown Rule Functions* in *TIBCO BusinessEvents User's Guide*.
- What kind of object management to implement. Object management is a large topic and the most commonly used option is covered in [Chapter 3, Cache Object Management Tutorial, on page 41](#). For this tutorial, use the default option, In Memory, which means that objects are kept in memory only and are not persisted when the engine stops.

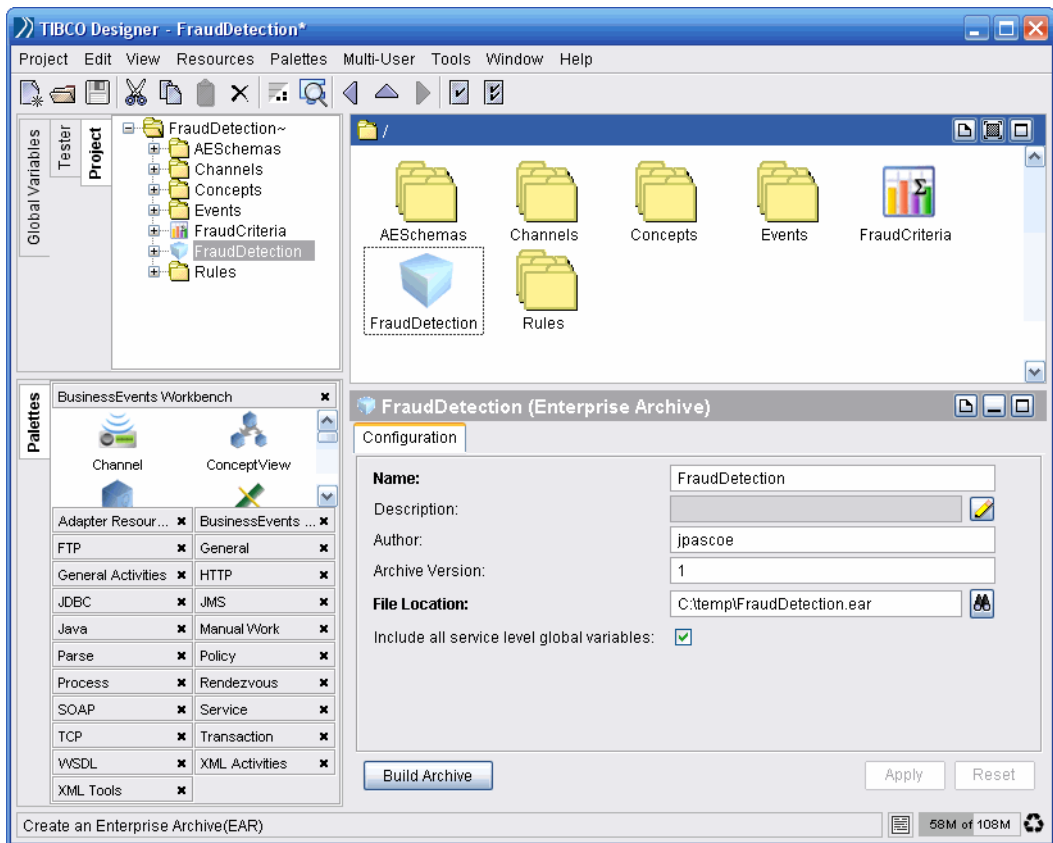
What is an event preprocessor? A preprocessor is a rule function that processes incoming messages before BusinessEvents transforms them into Events. For example, a preprocessor might filter the messages so that only certain ones are used as events. Preprocessors are multi-threaded and you can choose from various threading and queue options, as appropriate to handle the work load. By default the threading uses the system-wide shared queue and threads. See the topic *Working With Event Preprocessors* in *TIBCO BusinessEvents User's Guide*.

Task K Validate the Project, Configure and Build the Archive

1. As you did in [Task D, Validate the Project and Set a Default Event for a Destination, on page 15](#), click the Validate All () button and validate the

project to ensure that you haven't introduced any errors since your last check. Remember that warnings do not necessarily indicate a problem. Correct any issues before continuing.

- Click the root folder to display the top level folders in the design panel, then right-click in the design panel, and select **Add Resource > General > Enterprise Archive**. BusinessEvents names it automatically based on the project name, FraudDetection.
- In the File Location field, specify the path to the EAR file and its name, for example, C:\temp\FraudDetection.ear. Click **Apply**. Your display should look similar to this:



- Double-click the FraudDetection enterprise archive resource to open it.

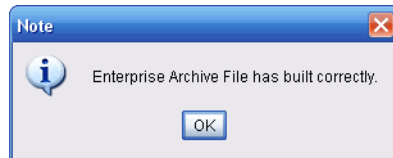
A Shared Archive (SAR) resource is added automatically. It contains resources that are available to the deployed application, such as a custom cache configuration file or a JDBC connection resource.

5. Right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > BusinessEvents Archive**.
6. Name the BusinessEvents Archive resource (BAR) **FraudDetection**. In the Type field, select Inference, and click **Apply**.
7. Select the **Input Destinations** tab. In the row for the URI `/Channels/RV.channel/DebitTransaction`, check the Enable checkbox. After deployment, BusinessEvents listens for messages from this destination and transforms them into events.

Another way to enable the DebitTransaction destination is to select Defaults in the Listener Set area. This option is useful if the project has several destinations and you just want to enable all the default destinations.

BusinessEvents checks the Default column if the default event for that destination is used in a rule set selected for deployment on the RuleSets tab.

8. Now configure the InitializeAccount rule function to execute on startup. Select the **Startup/Shutdown** tab and in the Startup Actions area, click the Add (+) button. In the Select a Resource dialog, select **InitializeAccount**. Click **OK** to dismiss the dialog, then click **Apply** and save the project.
(In the Object Management tab, use the default setting, In Memory.)
9. Click the FraudDetection EAR resource and in the Configuration tab, click **Build Archive**. You may be prompted to save the project. You should see the message "Enterprise Archive File has built correctly":



Summary and Next Steps

Before you deploy the EAR file, you will set up a simple Rendezvous publisher, using a ActiveMatrix BusinessWorks process, to provide test data to the application.

Configure an ActiveMatrix BusinessWorks Process for Testing

Learning Points

You will use the ActiveMatrix BusinessWorks process you configure here to send Rendezvous messages that trigger the fraud detection rule behavior, when conditions are met.



BusinessEvents customers can download and install ActiveMatrix BusinessWorks for use at design-time. If you want to use ActiveMatrix BusinessWorks at runtime, you must purchase the fully licensed product.




Integration projects Projects that include BusinessEvents functionality can also include functionality provided by other TIBCO products, resulting in rich interactions with other enterprise software. Integration with ActiveMatrix BusinessWorks is documented in Chapter 11, Out-of-Process ActiveMatrix BusinessWorks Integration and Chapter 12, In-Process ActiveMatrix BusinessWorks Integration in *TIBCO BusinessEvents User's Guide*.

What is the Send Event activity? The Send Event activity sends messages formatted so that BusinessEvents can transform the received message into an event of the type you select in the activity. You can only use this activity in BusinessEvents projects that include the type of event that you want to send.

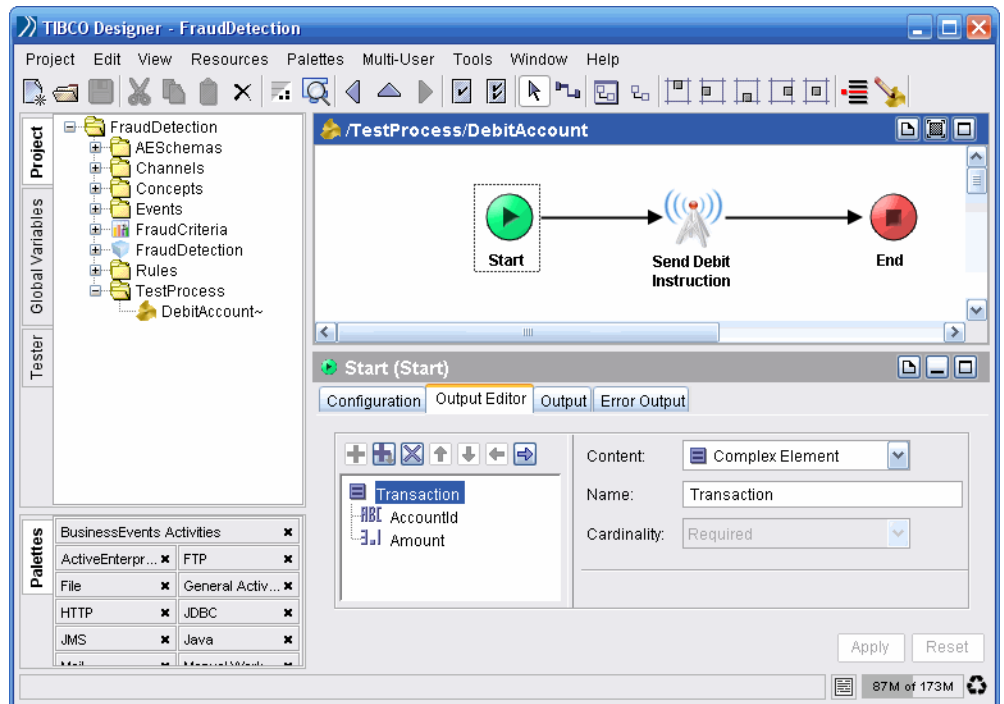
Task L Configure a ActiveMatrix BusinessWorks Process for Testing


For more information about configuring ActiveMatrix BusinessWorks processes, see *TIBCO BusinessWorks Process Design Guide*.

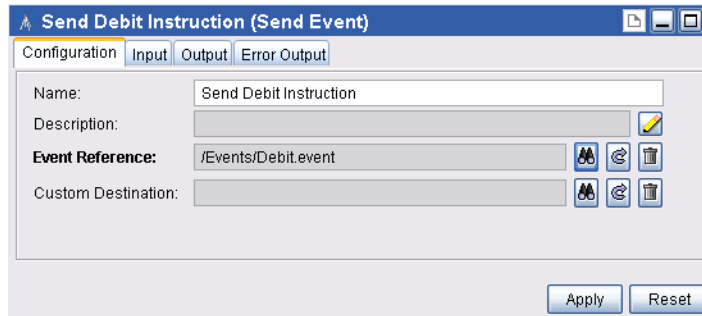
1. Add a new folder as a child of the root (project) folder and name it **TestProcess**.
2. Open the TestProcess folder. Right-click in the design panel, and select **Add Resource > Process > Process Definition**. Name the process **DebitAccount** and click **Apply**.
3. In the design panel, double-click the **DebitAccount** process to open it.
You see Start and End icons.
4. From the BusinessEvents Activities palette, add a **Send Event** resource to the design panel and drop it between the Start and End activity icons. Name the Send Event resource **Send Debit Instruction**.

5. Connect the three activities using transitions:
 - a. Click the **Create Transition** icon () on the TIBCO Designer toolbar.
 - b. Position the cursor over the Start activity and drag the mouse to the Send Event activity. You see a dotted arrow connecting the two activities.
 - c. Similarly, position the cursor over the Send Event activity and drag the mouse to the End activity. When focus leaves the dotted arrows they become solid arrows.
 - d. Click the **Pointer Tool** icon () on the TIBCO Designer toolbar.
6. Click the **Start** activity and define a simple output schema:
 - a. Select the **Output Editor** tab, click the add () button to add an element, and name the element **Transaction**.
 - b. With the Transaction element selected, click the add button two more times. Two parameters are added to the Transaction element.
 - c. Select each parameter in turn and configure it. Define the parameters as **AccountId**, of type String, and **Amount**, of type decimal. Click **Apply**.

The display now looks like the following (showing Amount as an example):



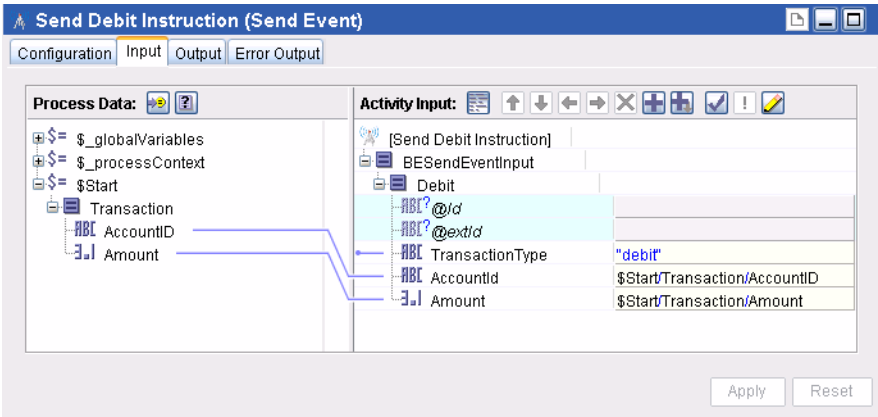
7. In the design panel, click the **Send Debit Instruction** activity. In the Event Reference field, click the browse button () and select the **Debit** event type. Click **Apply**.



Earlier, you set DebitTransaction as the default destination for the Debit event. Because of this connection, the Send Event activity can send messages to that destination without additional configuration. When the messages arrive, BusinessEvents will transform them into Debit events.

8. Next, map the process data from the Start activity to the parameters of the Send Debit Instruction (Send Event) activity:
 - a. Select the Input tab for Send Debit Instruction.
 - b. In the Process Data panel, fully expand the \$Start variable and in the Activity Input panel, fully expand the BESendEventInput elements.
 - c. Drag the cursor from **AccountId** in the Process Data panel to **AccountId** in the Activity Input panel. You see a connecting line.
 - d. In the same way, connect the **Amount** parameter in the Process Data panel to the **Amount** parameter in the Activity Input panel.
 - e. In the Activity Input panel, to the right of the TransactionType parameter name, type "**debit**" (including the quotes). You see a short connecting line within the panel beginning with a dot, indicating that the value is set within the activity and is not mapped.
 - f. Click **Apply** and save the project.

The completed configuration looks like the following:



Summary and Next Steps You've created a simple mechanism for sending test data into the running engine. Next you will deploy the project.

Deploy the Project

Learning Points

You are now ready to deploy this project. For testing purposes, you can simply start the engine at the command line. In production scenarios it is usual to deploy to a TIBCO Administrator domain (see Chapter 28, *Deploying a TIBCO BusinessEvents Project of TIBCO BusinessEvents User's Guide* for more details).

Task M Deploy the Project and Start the BusinessEvents Engine

1. Open a command window and navigate to `BE_HOME\bin`, for example:

```
c:\tibco\be\3.0\bin.
```

2. Enter a command using the following format:

```
be-engine -n enginename filepath\earfilename
```

For example:

```
be-engine -n acme-engine C:\BEProjects\FraudDetection.ear
```

The `-n` option is optional. It lets you assign a user-friendly name to the engine. A separate log file using the name is created as well.

You see engine startup messages in the console, beginning like the following:

```
Using property file: C:\tibco\be\version\bin\be-engine.tra
*****
Using arguments :-n acme-engine C:\BEProjects\FraudDetection.ear
```

You see various initialization messages, then you see the message you configured in the startup rule function (see [Configure the InitializeAccount Rule Function on page 20](#)):

```
##### Created account ActA
```

If you completed the optional exercise ([Optional Exercise: Create a Concept Instance Using a Standard Function on page 23](#)) you see the additional message:

```
##### Also created account ActB
```

Summary and Next Steps

Next you will send data into the engine and see the results.

Test the Application

Learning Points


Now that the engine is running, you will use the ActiveMatrix BusinessWorks to simulate data coming into the application from account activity. You can use this testing technique on other projects too.

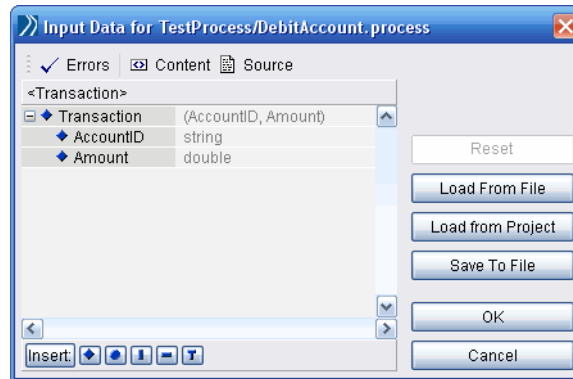
For more details on defining ActiveMatrix BusinessWorks processes see *TIBCO BusinessWorks Process Design Guide*.

Task N Test the Application

1. With the project open in TIBCO Designer, open the **DebitAccount** process (so you can see the activities within it), and then select the **Tester** tab (on the left).

Now you have entered the testing environment. Entering the testing environment starts a TIBCO ActiveMatrix BusinessWorks engine.

2. Click the **Supply Input Data to Starter** () button. You see the Input Data dialog.




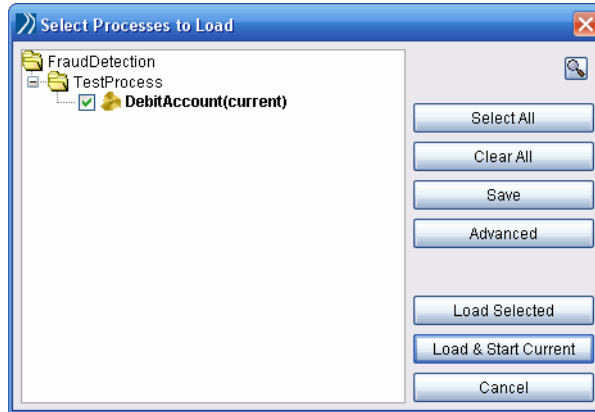
3. Double click in the **AccountID** field and enter **ActA**.



If you used a different value for the concept instance (in the section [Configure the InitializeAccount Rule Function on page 20](#)) use that value here. The account ID is shown in bold below:

```
Concepts.Account.Account("ActA" /*extId String */,
                          20000.0 /*Balance double */,
                          0 /*Debits double */,
                          "Normal" /*Status String */,
                          10000.0 /*AvgMonthlyBalance double */);
```

4. Double-click in the **Amount** field, and enter **3000**. This amount will be debited each time you run a job. Click **OK** to save and dismiss the window.
5. Position the command window so you will be able to see the messages.
6. Click the **Start Testing Viewed Process** () button. You see the Select Processes to Load dialog:




7. Select **DebitAccount(Current)** (if it is not selected) and click **Load and Start Current**.

In the command window look for the following message:

```
[FraudDetection] ##### Debiting account <ActA> by
<3000.0> dollars. New balance is <17000.0>
```

The project is set up to "suspend" accounts that incur more than three debit transactions over a two-minute rolling window, where the sum of the debits totals more than 80% of the average monthly balance of the account. The average monthly balance is set to \$10,000. So the third debit of \$3,000 within two minutes triggers the rule.

8. Run more jobs to trigger the FraudDetection rule action. Run each job as follows:
 - a. In the test panel click the TestProcess/DebitAccount .process process instance.
 - b. Click the Create a Job () button (on the left). This action creates and runs the job.

When the rule conditions are met, you see a message like the following:

```
[FraudDetection] ##### Account id <ActA> STATUS set to <Suspended>
```

Congratulations—You have completed the tutorial!

This tutorial shows you how to add caching functionality, and how to deploy multiple nodes and agents to perform different tasks in the runtime environment such as storing cache data, and running inference agents for load balancing and fault tolerance

This tutorial uses the fraud detection project configured in [Chapter 2, Project Design Tutorial, on page 5](#).

Topics

- [Caching and Multi-Engine Overview, page 42](#)
- [Create a JMS Channel and Destination, page 44](#)
- [Delete or Disable the Rendezvous Channel and its Destination, page 47](#)
- [Check for Existence of Accounts Before Creating, page 49](#)
- [Configure Cache Cluster Properties, page 51](#)
- [Configure Cache Server Nodes, page 54](#)
- [Configure Inference Agents, page 55](#)
- [Deploy the Agents and Cache Servers, page 58](#)
- [Test the Application, page 60](#)

Caching and Multi-Engine Overview

This tutorial builds on the [Project Design Tutorial](#), to explore common object management and deploy-time options.

Object management (OM) Refers to various ways that BusinessEvents can manage the state of ontology object instances created by each inference agent.

Cache-based OM When you use cache-based object management, object data is kept in memory caches, with (optional but recommended) redundant storage of each object for reliability and high availability. Within a cache cluster, nodes deployed as cache servers manage the data and handle recovery. Cache data is shared by all agents in the cluster.

All the provided caching schemes use a distributed cache and are configured for production as shipped. In a distributed cache, cached object data is partitioned between the nodes (JVMs) in the cache cluster for efficient use of memory. No two nodes are responsible for the same item of data. You can configure one or more backups of each object to be kept on different nodes to provide reliability.

Multi-engine features With cache OM you can use multi-engine features. Multiple inference agents deployed in the same cache cluster share the cache data and can run concurrently. They can be differently configured agents, or instances of the same agents (known as an *agent group*).

Load balancing and fault tolerance Deploying an agent group enables you to use load balancing or fault tolerance or both. When you deploy more inference agent instances than the specified number of active agents, the inactive inference agents are automatically used for fault tolerance. Inactive agents maintain a passive Rete network and do not listen to events from channels.



Load balancing is used with point-to-point messaging such as JMS queues. (Broadcast messages are received by all active agents in the cluster.) See Load Balancing and Fault Tolerance Between Inference Agents in *TIBCO BusinessEvents User's Guide*.

To explore these features you will make some minor modifications to the fraud detection project, and then deploy five nodes: three inference agents to provide load balancing and fault tolerance, and two cache servers, providing fault tolerant data storage. You will then exercise the project to see the effect of the cache, load balancing, and fault tolerance.

JMS Server Required

Load balancing requires use of point-to-point messaging using a queue. The tutorial provides step-by-step instructions for connecting to the TIBCO Enterprise Message Service™ server, running on the default port. If you use a different JMS provider, adapt the instructions accordingly.

Create a JMS Channel and Destination

In this exercise you will add a JMS Connection resource and a JMS channel with one destination. You will configure the project to use this destination instead of the Rendezvous destination used in the project design tutorial ([Chapter 2, Project Design Tutorial, on page 5](#)).

Learning Points

Load Balancing You can configure your project to enable load balancing of messages in a queue among active members of an agent group. Load balancing can't be done using the Rendezvous channel already configured for the Fraud Detection project because it broadcasts messages, and all the agents would pick up the same messages. Instead you must use point to point messaging using queues. When an agent retrieves a message from the queue it sends back an acknowledgement to the server. That message then leaves the queue. Each message is processed by only one of the agents.

Shared Resources In [Task B, Create a Rendezvous Channel and Destination, on page 11](#), you configured a Rendezvous channel using properties. To demonstrate another configuration method, you will configure the JMS channel using a shared resource. Generally, you would use a shared resource if you wanted to reuse the resource in different places in your project.

More Information

For more on JMS channels, read Chapter 2, Working With Channels and Destinations in *TIBCO BusinessEvents User's Guide*.

Task A Start the JMS Server and Create a JMS Connection

Start the JMS Server

1. Start the TIBCO Enterprise Message Service™ server:

Start > All Programs > TIBCO > TIBCO EMS *VersionNo* > Start EMS Server

Create JMS Connection

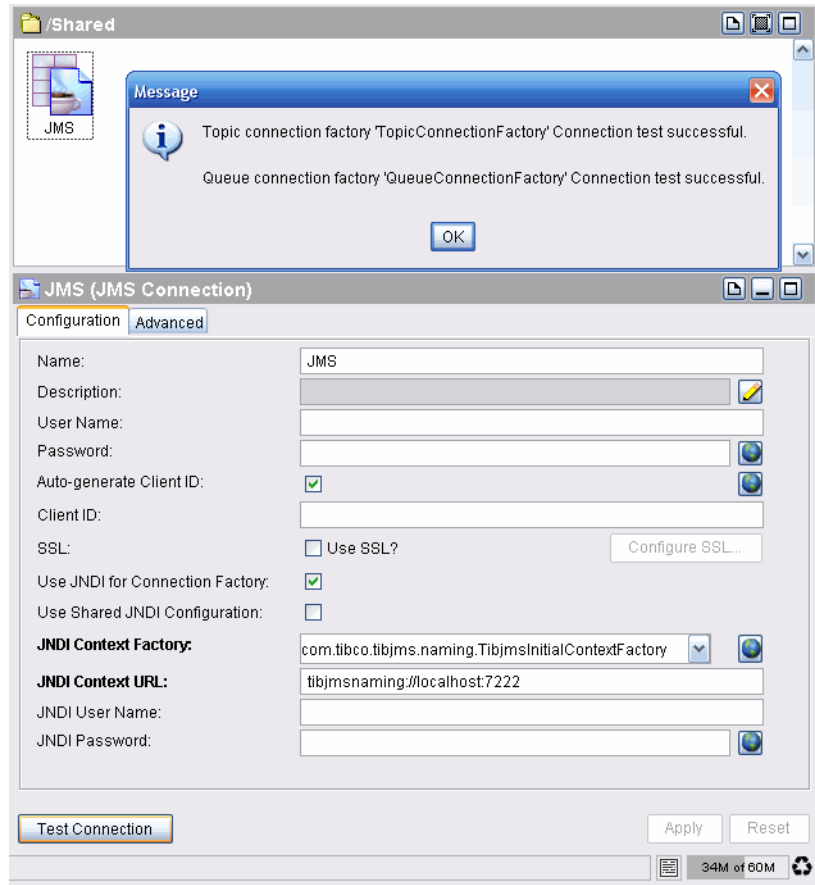
2. Start TIBCO Designer (Click **Start > Programs > TIBCO > TIBCO Designer version > Designer version.**)
3. From the Project menu, select **Open Existing Project** and open the FraudDetection project you configured using [Chapter 2, Project Design Tutorial, on page 5](#).

If you don't want to configure the project yourself, you can use the configured example, located here:

`BE_HOME/examples/FraudDetectionwithCache/FDCache`

4. Select **Project > Save As** and save the project as **FDCache**. The project name becomes a directory name in the location you specify. All files are copied to this directory.


5. In the project root, create a folder called **Shared**. This folder is not required but is just one way to organize your project.
6. Open the Shared folder and in the design panel, right-click and select **Add Resource > JMS > JMS Connection**.
7. Name the connection **JMS** and click **Apply**. No other configuration is required as default values are used in this tutorial.
8. Click **Test Connection**. You see a message that the connection is successful:

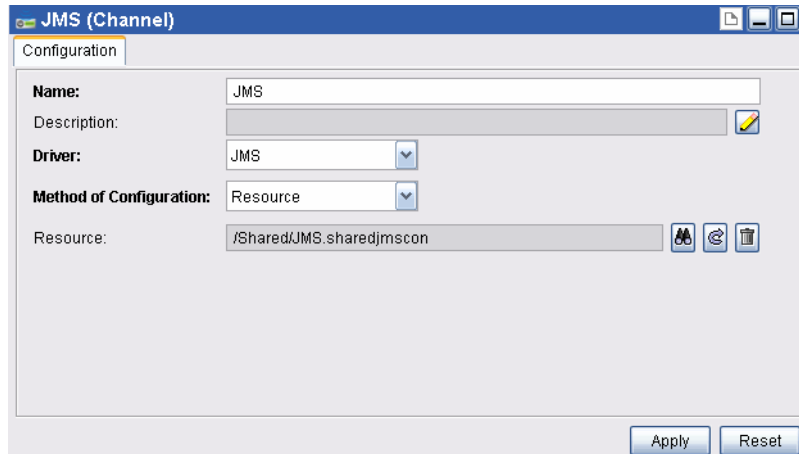


9. If the connection test fails, check that the Enterprise Message Service server is running. When the connection test is successful, click **OK**.


Task B Create a JMS Channel and Destination

1. Open the Channels folder, right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > Channel**.

2. In the Configuration tab, name the channel **JMS**, choose the **JMS** driver, and from the Method of Configuration drop-down list, select **Resource**.
3. Click the browse button () and select the JMS Connection resource you configured in [Task B](#). Then click **Apply**.




The screenshot shows the 'JMS (Channel)' configuration window. The 'Configuration' tab is active. The 'Name' field is set to 'JMS'. The 'Driver' dropdown is set to 'JMS'. The 'Method of Configuration' dropdown is set to 'Resource'. The 'Resource' field contains the path '/Shared/JMS.sharedjmscon'. There are 'Apply' and 'Reset' buttons at the bottom right.

4. Open the **JMS** channel, right-click in the design panel, and select **Add Resource > BusinessEvents Workbench > Destination**.
5. Name the destination **DebitTransactionJMS**.
6. In the Default Event field, click the browse button () and select **Debit**.
7. In the (Queue) Name field, type **DEBITTXN**. Click **Apply**. The destination listens to a queue with this name.
8. Click **Apply** and **Save** the project.

Task C Change the Debit Event's Default Destination

Now you will change the default destination for the Debit event from the Rendezvous destination to the JMS destination. See [Task D, Validate the Project and Set a Default Event for a Destination, on page 15](#) if you want a reminder about the purpose of default events.

1. Open the Debit event.
2. In the Default Destination field, click the browse button () and select **/Channels/JMS.channel/DebitTransactionJMS**, then click **Apply** and **Save** the project.

Delete or Disable the Rendezvous Channel and its Destination

The Rendezvous destination is not used in the caching tutorial. Its existence won't affect any functionality. However, it's a good idea to remove unused resources.

Alternatively, you can disable the Rendezvous destination in the BAR resource Input Destinations tab, if you want to use it again for some other reason.

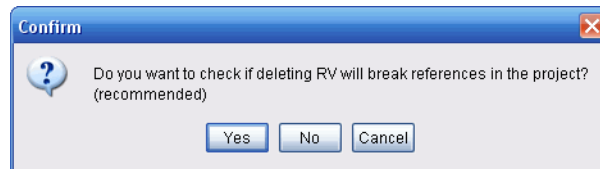
Learning Points

In the BAR resource, you can select what rulesets to deploy, what destinations (listeners) to enable, and you can specify startup and shutdown actions (among other configuration options). Thus from one project you can deploy multiple EARs each with differently configured BAR files.

Task D Delete the Rendezvous Channel

Note: Choose either [Task D](#) or [Task E](#).

1. In the project panel, right-click the RV channel entry and select **Delete**. At the confirmation dialog ("Are you sure you want to delete?") click **Yes**.
2. You see the following dialog offering to check for potential problems before you actually delete the resource (If you deviated from the steps you may not see this message. If you don't see the dialog, just read through the steps):



For learning purposes, click **Yes**. You see the References dialog showing the project path to the FraudDetection BAR resource.

3. Click the project path in the References dialog to jump to the BAR resource. Select the Input Destinations tab. You can see that the RV destination is enabled.

In a real-world situation, checking references can reveal additional work you must do, or a reason to cancel the deletion. However, you can go ahead and delete the channel in this case.

4. In the References dialog (which remains open) click **OK**. The RV channel is now deleted.
5. Open the Input Destination tab of the BAR resource, you see that only the JMS destination is listed.

The JMS destination is enabled by default because it has a default event.

Task E Disable the Rendezvous Channel

Note: Choose either [Task D](#) or [Task E](#).

1. Open the Input Destination tab of the BAR resource and select the **Custom** listener set.
2. In the row for the Rendezvous destination, uncheck the Enable checkbox.
3. Click **Apply** and **Save** the project.

Check for Existence of Accounts Before Creating

In the basic Fraud Detection project (see [Chapter 2, Project Design Tutorial, on page 5](#)), one or optionally two Account instances are created on startup using startup rule functions.

Startup rule functions execute on all active agents as they start up, unless you use application logic to perform a check before taking action.

Three instances of the agent will now be deployed in three engines, so you must ensure that the Account instance or instances are created only once (otherwise runtime exceptions will occur).

To do so, you will modify the `InitializeAccounts` rule to use a try catch block. The block will prevent concepts from being created if they already exist.

Learning Points

Caching and Engine Concurrency (multi-engine) The design of your project must take multi-engine features and cache object management into account.

How entities are shared between agents In a multi-engine configuration:

- All concept instances are shared between agents in an asynchronous manner.
- Event instances are *clustered* between agents—they are not shared. That is, each event instance is present on only one agent in a group.
- Scorecards are not shared between agents. Each inference agent maintains its own set of scorecards and the values in each agent can differ.

Avoiding concurrent changes As needed by the project, take care when making changes that affect the ontology instances to prevent another agent in the group from making concurrent changes. Read *Designing With Multiple Active Inference Agents* in *TIBCO BusinessEvents User's Guide* for more details.

Task F Add Checks in the InitializeAccounts Rule

1. With the `FDCache` project open in TIBCO Designer, open the **InitializeAccounts** rule function.
2. Just above the following comment:

```
//Create Account concept instance ActA
```

Add the following line:

```
try {
```

3. Just below the line:

```
System.debugOut("##### Created account ActA
#####");
```

Add the following lines:

```
} catch (Exception e1) {
// No action - the account is just not created if it exists.
}
```

4. Similarly wrap the lines that create ActB, if you created this optional account (see [Optional Exercise: Create a Concept Instance Using a Standard Function on page 23](#)). The complete section for ActB is shown below (formatted to fit):

```
try {
//Create Account concept instance ActB using XSLT Mapper
Instance.createInstance("/Concepts/Account");
System.debugOut(
"##### Created account ActB #####");
} catch (Exception e2) {
// No action - the account is just not created if it exists.
}
```

Note that you cannot copy the line beginning `Instance.createInstance` from the above code sample into the rule editor. If you have not created the account ActB, you can skip this step. If you want to add a second concept now, see [Optional Exercise: Create a Concept Instance Using a Standard Function on page 23](#) for instructions on adding a concept instance using the `Instance.createInstance()` function.

5. Click **Apply**.
6. Click the **FDCache** resource and select the Configuration tab.
7. In the **File Location** field, Specify a location and name for the EAR file. The tutorial uses `C:\temp\FDCache.ear`.
8. Click **Apply**. Click **Save**. Click **Build Archive**.

You should see the message "Enterprise Archive File has built correctly." If you do not, validate your project to identify and correct any errors and try again.

Configure Cache Cluster Properties

BusinessEvents ships with default values set for basic caching functionality, so all you have to do to enable caching in a project is give the cluster a name that is used in all nodes, and enable the multi-engine feature property.

Behind the simple setup is a set of configuration properties with default values. The Learning Points section below provides information on some configuration options that you don't have to set for the tutorial.

Learning Points

Node discovery The following default settings are used for multicast node discovery. Properties do not have to be added to the property files unless you use different values.

```
java.property.tangosol.coherence.clusteraddress=224.3.3.1
java.property.tangosol.coherence.clusterport=31337
```

To avoid possible collisions with other clusters running on your network, the tutorial uses a non-default cluster address. More cluster discovery settings are available for different situations, such as hosts with multiple NICs. Another method of discovering nodes, using well-known addresses for situations where use of multicast is not an option.

Cache cluster features Many characteristics of the cache cluster and caching scheme are specified by properties set in the engine properties file, such as the cluster name. More in depth customization is possible but is not often required and is not documented. Contact TIBCO support for assistance.

Multi-engine features Multiple engines can run concurrently against the same project ontology. You can use a mixture of differently configured agents and identically configured *agent groups* (see [Configure Inference Agents on page 55](#)). By default the multi-engine features are disabled, so that only the agents in one EAR file (that is, running in one JVM) are active at one time. To turn on multi-engine features, which offer load balancing (and implicit fault tolerance), you set the `be.engine.cluster.multiEngineOn` property to true.

Number of cache servers to start You must also set a property that defines the number of cache servers to start before starting any inference agents. The `be.engine.cluster.minCacheServers` property ensures that the cache is populated before inference agents begin processing events.

More Information

For details on other cluster-level properties not discussed in this tutorial (because their default values don't need to be changed) see Chapter 19, Configuring Cache Cluster Discovery and Chapter 20, Configuring Cache Cluster Settings of *TIBCO BusinessEvents User's Guide*.

Task G Configure Cluster Properties



Every node (agent and cache server) in the cluster uses the same set of cluster properties.

1. Open a copy of the provided engine properties file for editing:
`BE_HOME/bin/be-engine.tra`
2. Add the following set of properties to the file:

```
#### Cluster Level Properties

java.property.tangosol.coherence.cluster=AcmeCluster
java.property.tangosol.coherence.clusteraddress=224.3.3.11
java.property.tangosol.coherence.ttl=0
be.engine.cluster.multiEngineOn=true
be.engine.cluster.minCacheServers=2
```

The properties are briefly described in the table below.

3. Save the file in the `BE_HOME/bin` directory as **agent1.tra**.
You will copy and modify this file for each of the five engines you will start when testing the behavior of the deployed project.

| Property (Short name) | Description |
|-----------------------|--|
| cluster | Specifies the name of the cache cluster. The tutorial uses the name <code>AcmeCluster</code> . |
| clusteraddress | <p>Specifies the multicast IP address that the socket will listen to or publish on.</p> <p>Possible values are addresses between (and including) <code>224.0.0.0</code> and <code>239.255.255.255</code>. Default value is <code>224.3.3.1</code></p> <p>To avoid collisions with any other project deployed on your network, use a non-default address. The tutorial uses <code>224.3.3.11</code></p> |
| ttl | <p>Specifies the time-to-live setting for the multicast, that is, the maximum number of "hops" a packet can traverse. A hop is defined as a traversal from one network segment to another via a router.</p> <p>If you will start all nodes on one machine (which you can do for non-production situations), set the <code>ttl</code> to 0, to keep multicast packets from leaving the originating machine (on most operating systems).</p> |

| Property (Short name) | Description |
|-----------------------|---|
| multiEngineOn | <p>Set to true to enable multiple engines to be active at the same time.</p> <p>If set to false, only one engine is active at any time (as in earlier versions).</p> |
| minCacheServers | <p>Specifies the minimum number of storage-enabled nodes that must be active in the cluster when the system starts up before the following occur:</p> <ul style="list-style-type: none">• The other agents in the cluster start up.• Data is loaded from the backing store, if a backing store is configured. <p>In general, set to the number of cache servers in the deployment, in this case two.</p> |

Configure Cache Server Nodes

The purpose of cache servers is to store and serve cache data for the cluster. A cache server is a non-reasoning agent used as a storage node only. Storage nodes are responsible for object management. They participate in distribution, partitioning and storage of the objects in the cluster.

Learning Points

Local storage The local storage engine property is true by default, so you only have to set it when you want to disable cache data storage for a node. It is a best practice in production systems to use dedicated cache server nodes for cache data storage and to disable cache data storage in all other nodes.

Deploying a node as a cache server Cache server nodes are deployed using any EAR file that contains a BAR resource configured for an inference agent (or a query agent). All other agent-level properties in the TRA file are ignored when the following property is set: `be.engine.cacheServer=true`.

Number of cache servers to start At the cluster level you used the `be.engine.cluster.minCacheServers` property to define the number of cache servers that must be started before inference agents start. This means that you can start all the nodes in any order. The inference agents won't become active until the specified number of cache servers have started, ensuring that the agents can write to and read from the cache.

Task H Configure the Cache Server Property Files

1. Open the `BE_HOME/bin/agent1.tra` you configured in [Task G, Configure Cluster Properties, on page 52](#).
2. Add the following property so that the node deploys as a cache server:

```
#### Cache Server Property

be.engine.cacheServer=true
```

3. Save the file in the `BE_HOME/bin` directory, using the name `cs.tra`.

Configure Inference Agents

Inference agents are configured partly in TIBCO Designer using a BAR resource, and partly using engine property (TRA) files. You can use a supplementary property file for each agent to make command line startup easier.

Learning Points

Inference agents do the decisioning work. Each agent has a Rete network, for processing incoming events. You can think of an inference agent as a rule session, attached to the cluster.

Agent group An agent group is formed by deploying one agent multiple times, generally on different machines. Use of agent groups requires the multi-engine features to be enabled (see [Configure Cache Cluster Properties on page 51](#)).

Fault tolerance and load balancing are provided automatically when you use agent groups and multi-engine features. (Fault tolerance is also available without multi-engine). You can configure some agents to be inactive, available only for fault tolerance. However, all active agents load balance and if one agent fails, the others handle the load, so you may not need to keep an agent inactive.

To make an agent inactive, set the `Agent.AgentGroupName.maxActive` property to a lower number than the number of agents in the group. All deployed agents in addition to that number are inactive—which ones are inactive is specified by a priority number.

Agent group key (and scorecards) It is a good idea to identify each agent in a group uniquely. To do so you define an agent group key property and give each agent a unique value. For example, this property is required if your project uses scorecards. Scorecard values are not shared between agents in a group. When recovering from failure, the system needs to identify which scorecard instance belongs to which agent so that the correct value is restored.

Local storage For performance reasons, it is recommended that you disable local storage of cache data on inference agents and query agents. Instead use dedicated cache server nodes for storage of cache data. To disable local storage, set the following property set to false (it is true by default):

```
java.property.tangosol.coherence.distributed.localstorage=false
```

Task I Configure Inference Agent OM Settings in TIBCO Designer

1. As needed, open your caching project in TIBCO Designer.
2. Double-click the FraudDetection enterprise archive resource to open it and then double-click the FraudDetection BusinessEvents archive that is within it.

For details about basic configuration of the BAR, see [Validate the Project, Configure and Build the Archive on page 31](#).

3. In the Configuration tab, ensure that Type is set to **Inference**.

The other option, Query, is used when configuring a query agent (available in the TIBCO BusinessEvents Enterprise Suite software).

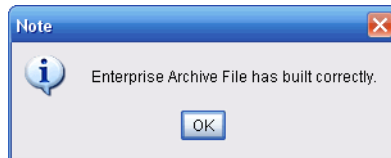
4. Select the **Object Management** tab and do the following
 - a. From the Type drop-down list, select **Cache**.
 - b. In the Agent Group Name field, type **FraudDetection**.

It is a recommended practice to use the BAR name as the agent group name.



Cache Modes In the Object Management tab, different *cache modes* can be set on individual entities, to tune memory use and performance. See Chapter 18, Understanding and Working With Cache Modes in *TIBCO BusinessEvents User's Guide* to explore cache mode features.

5. In the Configuration tab, click **Build Archive**. You may be prompted to save the project. You should see the message "Enterprise Archive File has built correctly":



Task J Configure the Inference Agent Engine Properties

In this step you will create three property files, one for each inference agent.

Preconfigured files are available in `BE_HOME/Examples/`

1. Open the `BE_HOME/bin/agent1.tra` you configured in [Task G, Configure Cluster Properties, on page 52](#).
2. Add the following properties:

```
#### Inference Agent 1 Properties
```

```
java.property.tangosol.coherence.distributed.localstorage=false
Agent.FraudDetection.maxActive=2
Agent.FraudDetection.priority=1
Agent.FraudDetection.key=agent1
```

Where `FraudDetection` is the agent group name you assigned in the BAR resource in TIBCO Designer (see [Task I, Configure Inference Agent OM Settings in TIBCO Designer, on page 55](#)).

3. Save the file in the *BE_HOME/bin* directory, using the name **agent1.tra**.
4. Modify the section title, and the priority and key properties as shown in the Inference Agent 2 Properties section below and save the file as *BE_HOME/bin/agent2.tra*:

```
####Inference Agent 2 Properties
```

```
java.property.tangosol.coherence.distributed.localstorage=false
Agent.FraudDetection.maxActive=2
Agent.FraudDetection.priority=2
Agent.FraudDetection.key=agent2
```

5. Modify the section as shown below and save the file as *BE_HOME/bin/agent3.tra*:

```
####Inference Agent 3 Properties
```

```
java.property.tangosol.coherence.distributed.localstorage=false
Agent.FraudDetection.maxActive=2
Agent.FraudDetection.priority=3
Agent.FraudDetection.key=agent3
```

Deploy the Agents and Cache Servers

It is helpful if you can size the command windows so that you can see all five windows at one time. First you will start the cache servers, as is common practice.

Learning Points

Cache servers start first It is important that cache servers start before inference agents. The order in which you start the engines is not important, as long as you have set the `be.engine.cluster.minCacheServers` property, which ensures cache servers start up first (see [Configure Cache Cluster Properties on page 51](#)).

Task K Start the BusinessEvents Engines and Cache Servers



Ensure that the TIBCO Enterprise Message Service server has started. One way to start the server is as follows:

Start > All Programs > TIBCO > TIBCO EMS *VersionNumber* > Start EMS Server.

1. Open two command windows and in each navigate to `BE_HOME/bin`. For example: `c:\tibco\be\3.0\bin`.
2. In each window enter the following command (modified as needed according to the location of your EAR file):

```
be-engine -n CS1 --propFile cs.tra c:\temp\FDCache.ear
be-engine -n CS2 --propFile cs.tra c:\temp\FDCache.ear
```



The `--propFile` argument is case sensitive.

The `-n` option is optional. It lets you assign a user-friendly name to the engine. The command windows and log files display the name you enter. A separate log file using the name is created as well.

You see many informational messages in the windows as the cache servers come up.

3. Open three more command windows in `BE_HOME/bin`, and in each window enter one of the following commands:

```
be-engine -n engine1 --propFile agent1.tra c:\temp\FDCache.ear
be-engine -n engine2 --propFile agent2.tra c:\temp\FDCache.ear
be-engine -n engine3 --propFile agent3.tra c:\temp\FDCache.ear
```

You see engine startup messages in the console, beginning like the following:

```
C:\tibco\be\3.0\bin>be-engine -n engine1 -p agent1.tra
c:\temp\FDCache.ear
Using property file: C:\tibco\be\3.0\bin\be-engine.tra
*****
TIBCO BusinessEvents Enterprise Suite 3.0.0.009 (2008-05-27)
Using arguments :-n engine1 -p agent1.tra c:\temp\FDCache.ear
Copyright- 2004-2008 TIBCO Software Inc. All rights reserved.
*****
Using property file: agent1.tra
```

You see various initialization messages, including:

```
Cluster AcmeCluster starting in MULTIPLE PRIMARY MODE
```

If you do not see the above, check that you configured the BusinessEvents archive resource Object Management tab to use cache OM.

On the engine that starts first, you see the message (or messages) you configured in the startup rule function (see [Configure the InitializeAccount Rule Function on page 20](#) and [Task F, Add Checks in the InitializeAccounts Rule, on page 49](#)):

```
##### Created account ActA
##### Also created account ActB
```

No other engine displays those messages because of the lock and the check for existence of this concept instance that you added in step [Task F, Add Checks in the InitializeAccounts Rule, on page 49](#).

Troubleshooting



If another user on the same network is using the same multicast properties as you, you will have to set a different cluster IP address property in all files.

Test the Application

Learning Points As in the section [Test the Application on page 39](#), while the nodes are running, you will use ActiveMatrix BusinessWorks to simulate data coming into the application from account activity. See that section for detailed instructions. It is assumed here that you are now familiar with the steps.

The process you created in [Configure an ActiveMatrix BusinessWorks Process for Testing on page 34](#) can be used again without modification.

Task L Test the Application

1. With the project open in TIBCO Designer, open the **DebitAccount** process and then select the **Tester** tab.
2. Click the **Supply Input Data to Starter** () button. In the AccountID field, enter ActA (or ActB) and in the Amount field enter a figure such as **3000**. This amount will be debited each time you run a job. Click **OK** to save and dismiss the window.
3. Position the command windows for all three inference agents so you will be able to see the messages. Remember that agent1 and agent2 are configured to be active while agent3 is inactive, for fault tolerance.
4. Click the **Start Testing Viewed Process** () button. At the Select Processes to Load dialog select **DebitAccount(Current)** (if it is not selected) and click **Load and Start Current**.

In the command window for engine1 (agent1), look for the following message:

```
[FraudDetection] ##### Debiting account <ActA> by
<3000.0> dollars. New balance is <17000.0>
```

5. Start JConsole, which is in the bin directory of your JDK installation (for example, C:/Program Files/Java/jdk1.6.0_07/bin/JConsole.exe) and expand the Coherence folders. For example, expand to:

```
Coherence > Cache > DistributedCache >
dist-unlimited-nobs$AcmeCluster$$be.gen.Concepts.Account > 1 >
back
```

Then click on **Attributes**. The panel on the right displays information.

Click **Refresh** periodically to refresh the display as you perform actions.

6. Run another job to again debit ActA. You see the message for this debit action in the command window for Engine2 (agent2).

Continue to run jobs. The JMS server sends messages to each active agent in a round-robin manner. If you created ActB you can send it test data, too.

Command windows for the active agents look similar to the following:

The image shows two screenshots of command windows. The top window is titled 'C:\WINDOWS\system32\cmd.exe - be-engine -n engine1 -p agent1.trac:\temp\FDCache.ear'. It displays a series of log messages from 2008 Sep 08 15:45:18 to 15:48:55. The messages include 'Agent Table Change', 'Verifying FT State', 'Debiting account <ActB> by \$3500.0', 'New balance: \$13000.0', and 'Saving Txn c om.tibco.ccp.runtime.service.om.coherence.cluster.agents.RtcTransaction@1aef798 to cache d ist-limited-nobs\$AcmeCluster\$FraudDetection\$AgentTxn-3'. The bottom window is titled 'C:\WINDOWS\system32\cmd.exe - be-engine -n engine2 -p agent2.trac:\temp\FDCache.ear'. It displays log messages from 2008 Sep 08 15:46:53 to 15:49:11. The messages include 'Debiting account <ActB> by \$3500.0', 'New balance: \$16500.0', 'Saving Txn c om.tibco.ccp.runtime.service.om.coherence.cluster.agents.RtcTransaction@1500efd to cache d ist-limited-nobs\$AcmeCluster\$FraudDetection\$AgentTxn-4', 'Debiting account <ActA> by \$3500.0', 'New balance: \$9500.0', and 'Account ID <ActA> STATUS set to <Suspended>. Fraud suspected.'.

```

C:\WINDOWS\system32\cmd.exe - be-engine -n engine1 -p agent1.trac:\temp\FDCache.ear
2008 Sep 08 15:45:18:404 GMT -? engine1 Info [AgentMgr-AcmeCluster-0] - Agent Table Chang
e, Verifying FT State
2008 Sep 08 15:45:18:576 GMT -? engine1 Info [AgentMgr-AcmeCluster-0] - Agent Table Chang
e, Verifying FT State
2008 Sep 08 15:45:18:591 GMT -? engine1 Info [AgentMgr-AcmeCluster-0] - Agent Table Chang
e, Verifying FT State
2008 Sep 08 15:45:18:607 GMT -? engine1 Info [AgentMgr-AcmeCluster-0] - Agent Table Chang
e, Verifying FT State
2008 Sep 08 15:47:06:678 GMT -? engine1 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### Debiting account <ActB> by $3500.0
2008 Sep 08 15:47:06:678 GMT -? engine1 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### New balance: $13000.0
2008 Sep 08 15:47:06:693 GMT -? engine1 Info [$defaultDesfination.Worker.0] - Saving Txn c
om.tibco.ccp.runtime.service.om.coherence.cluster.agents.RtcTransaction@1aef798 to cache d
ist-limited-nobs$AcmeCluster$FraudDetection$AgentTxn-3
2008 Sep 08 15:48:55:245 GMT -? engine1 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### Debiting account <ActA> by $3500.0
2008 Sep 08 15:48:55:245 GMT -? engine1 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### New balance: $13000.0
2008 Sep 08 15:48:55:261 GMT -? engine1 Info [$defaultDesfination.Worker.0] - Saving Txn c
om.tibco.ccp.runtime.service.om.coherence.cluster.agents.RtcTransaction@5facbd to cache di
st-limited-nobs$AcmeCluster$FraudDetection$AgentTxn-3

C:\WINDOWS\system32\cmd.exe - be-engine -n engine2 -p agent2.trac:\temp\FDCache.ear
ion] ##### Debiting account <ActB> by $3500.0
2008 Sep 08 15:46:53:461 GMT -? engine2 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### New balance: $16500.0
2008 Sep 08 15:46:53:493 GMT -? engine2 Info [$defaultDesfination.Worker.0] - Saving Txn c
om.tibco.ccp.runtime.service.om.coherence.cluster.agents.RtcTransaction@1500efd to cache d
ist-limited-nobs$AcmeCluster$FraudDetection$AgentTxn-4
2008 Sep 08 15:48:31:821 GMT -? engine2 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### Debiting account <ActA> by $3500.0
2008 Sep 08 15:48:31:836 GMT -? engine2 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### New balance: $16500.0
2008 Sep 08 15:48:31:836 GMT -? engine2 Info [$defaultDesfination.Worker.0] - Saving Txn c
om.tibco.ccp.runtime.service.om.coherence.cluster.agents.RtcTransaction@186b11c to cache d
ist-limited-nobs$AcmeCluster$FraudDetection$AgentTxn-4
2008 Sep 08 15:49:11:382 GMT -? engine2 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### Debiting account <ActA> by $3500.0
2008 Sep 08 15:49:11:382 GMT -? engine2 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### New balance: $9500.0
2008 Sep 08 15:49:11:413 GMT -? engine2 User [$defaultDesfination.Worker.0] - [FraudDetect
ion] ##### Account ID <ActA> STATUS set to <Suspended>. Fraud suspected.
2008 Sep 08 15:49:11:429 GMT -? engine2 Info [$defaultDesfination.Worker.0] - Saving Txn c
om.tibco.ccp.runtime.service.om.coherence.cluster.agents.RtcTransaction@556949 to cache di
st-limited-nobs$AcmeCluster$FraudDetection$AgentTxn-4
  
```

When the rule conditions are met, you see a message like the following:

```
[FraudDetection] ##### Account id <ActA> STATUS set to <Suspended>
```

7. To demonstrate failover to the inactive agent, press Ctrl-C to stop engine2. (It doesn't actually matter which active engine you stop). You see engine3 activating. Look for messages READY TO ACTIVATE then Activated.
8. Send some more events using the tester. Now you see the messages appear alternately in engine1 and engine3 command windows.
9. To demonstrate failback, start engine2 again. You see engine2 resume its active role, and in the command window for engine3, you see messages such as Setting Rule Session to INACTIVE and DeActivate.
10. To reset the test data, stop all nodes and start them again. Because cache data is all held in memory, stopping the system clears all data.

Congratulations—You have completed the caching tutorial!

Chapter 4 **Backing Store Tutorial**

In this tutorial you add a backing store to the project you prepared in [Chapter 3, Cache Object Management Tutorial](#), on page 41.

Topics

- *[Prepare the Database, page 64](#)*
- *[Add a JDBC Connection Resource, page 67](#)*
- *[Configure the Backing Store Properties, page 70](#)*
- *[Deploy and Test the Application, page 72](#)*
- *[Reset the Backing Store Tutorial, page 73](#)*

Prepare the Database

In this task you use provided scripts to set up the database schema. If you want to reset the tutorial, execute [Task C](#) and [Task D](#) again to recreate the empty tables.

Learning Points

Long Identifiers A known limitation in Oracle means that each identifier name cannot exceed 30 characters in length. The tutorial project does not have long identifiers. However, if your projects have long identifiers, you will perform an additional procedure to provide shorter aliases and then regenerate and run the scripts, as documented in the section Extra Procedure to Handle Long Identifier Names *TIBCO BusinessEvents User's Guide*.

After changing a project that uses a backing store If you change the project ontology you must update the database schema. A utility is available. However, it cannot handle all changes. See the section Updating an Existing Backing Store Database Schema in *TIBCO BusinessEvents User's Guide*.

Task A Ensure All Prerequisites are in Place

This tutorial assumes the following prerequisites are in place:

- Oracle 10G software. You can use the Express Edition, which you can download from the Oracle web site. (See the product readme file for specific version information.)
- The JDBC drivers file (for example, `ojdbc14.jar`), which you must copy to `BE_HOME/lib/ext`. The drivers are part of the Oracle Client software.
- `be-oracle.jar`, located in `BE_HOME/lib`, the default location.
- The EAR file from the [Cache Object Management Tutorial](#), `FDCache.ear`.

Task B Generate the SQL Scripts

1. Open a command window and navigate to `BE_HOME/bin`.
2. Run `be-oradeploy.exe` using the following command

```
be-oradeploy -o FDStore c:\temp\FDCache.ear
```

In the command window, you see various messages as generated scripts are created. The scripts contain various schema definition commands.

The following generated scripts appear in the `BE_HOME/bin` directory:

```
FDStore.sql
FDStore_cleanup.sql
FDStore_remove.sql.
```


FDStore.aliases



The aliases file is used to address a known limitation in Oracle: an identifier name cannot exceed 30 characters in length. If there are any long identifier names, they appear in the file. However, the fraud detection project has no long names. For details on how to handle long identifier names, see *TIBCO BusinessEvents User's Guide*, Chapter 24, Setting up a Backing Store Database, Task C, Shorten Long Names Using the Aliases File.

Task C Run the Initialize Database Script to Create the Oracle User

Running the provided `initialize_database.sql` script drops all existing backing store tables. It deletes the user (if one exists) before creating it again.

In the `initialize_database.sql` script, the Oracle user is set to `BE_USER`, with password `BE_USER`. You can edit the script to change these default settings, but the tutorial uses the default username and password.

1. Login to Oracle Server as the `system` user.

One way to do the above is to click **Start > All Programs > Oracle Database 10g Express Edition > Go To Database Home Page**. The database home page appears in a browser window. Type the user name **system** and the password you assigned when the software was installed.

2. Open a command window and navigate `BE_HOME/bin`.
3. Type **SQLPlus** then provide user name **system** and the password for that user.
4. Type `@initialize_database.sql` to run the provided script, `initialize_database.sql`. You see messages like the following:

```
DROP USER be_user CASCADE
      *
ERROR at line 1:
ORA-01918: user 'BE_USER' does not exist

User created.
Grant succeeded.
SQL>
```

Task D Login as the BusinessEvents Oracle User and Run SQL Scripts

In this step, you run scripts to create the database schema under the user you created. The schema combines the definitions in `base-types.sql`, `create-tables.sql`, and the generated `FDStore.sql` file.

Note that these scripts also perform cleanup before creating the schema. The first time you run the scripts, you see harmless error or warning messages because there is nothing to delete.

1. Login to the Oracle server as `BE_USER`, password `BE_USER` (The username and password in the script you ran in [Task C](#)).
2. Navigate to the location of the scripts and open an SQLPlus prompt.
3. Identify yourself as `be_user` with password `be_user`.
4. At the SQL prompt, type the following to run each script in turn:
 - a. `@base_types.sql`
 - b. `@create_tables.sql`
 - c. `@FDStore.sql`

Your database tables are now configured for use.

Add a JDBC Connection Resource

In this task, you will add a JDBC Connection resource to your project and configure it to connect to the backing store database. Details below explain how to connect to a local instance of Oracle 10g Express Edition database. Adapt the instructions as needed for your database product.



In a later task, you will specify the location of this resource, using the engine property `be.oracle.dburi.0`.

1. Start TIBCO Designer and from the Project menu, select **Open Existing Project**. Do one of the following:
 - Open the `FDCache` project you configured using [Chapter 3, Cache Object Management Tutorial, on page 41](#).
 - If you have not configured the caching project yourself, you can start with the configured caching example, located here:

`BE_HOME/examples/FraudDetectionwithCache/FDCache`

- If you don't want to configure the backing store project, you can view the configured backing store example, located here:

`BE_HOME/examples/FraudDetectionwithBackingStore/FDStore`

The rest of the tutorial assumes that the `FDCache` project is your starting point.

2. Select **Project > Save As**. Create a project directory called **FDStore** somewhere on your computer.

The project directory becomes the project name. All files are copied to this directory.

3. Open the **Shared** folder and add a JDBC Connection resource (from the JDBC palette).
4. In the drop-down list to the right of the JDBC Driver field, select **oracle.jdbc.driver.OracleDriver (thin)**. The driver appears in the JDBC Driver field and in the Database URL field the Database URL format appears as:

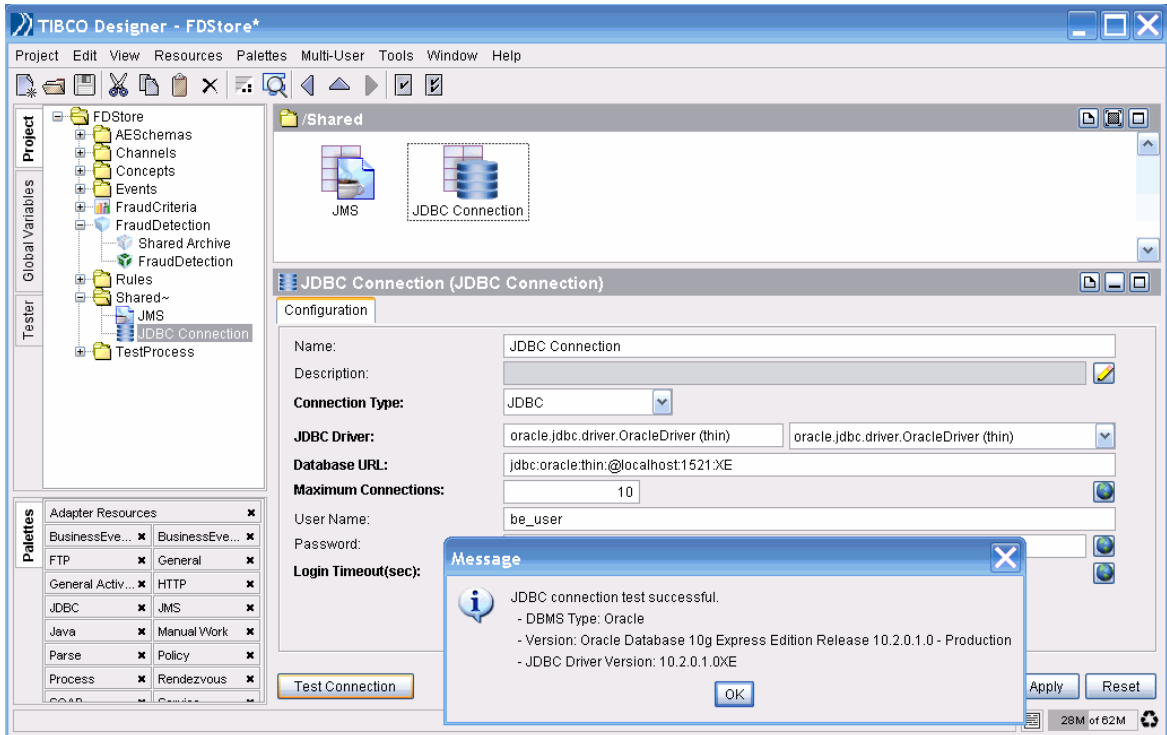
`jdbc:oracle:thin:@<host>:<port#>:<db_instancename>`

In the Database URL field, configure the provided format. For example:

`jdbc:oracle:thin:@localhost:1521:XE`

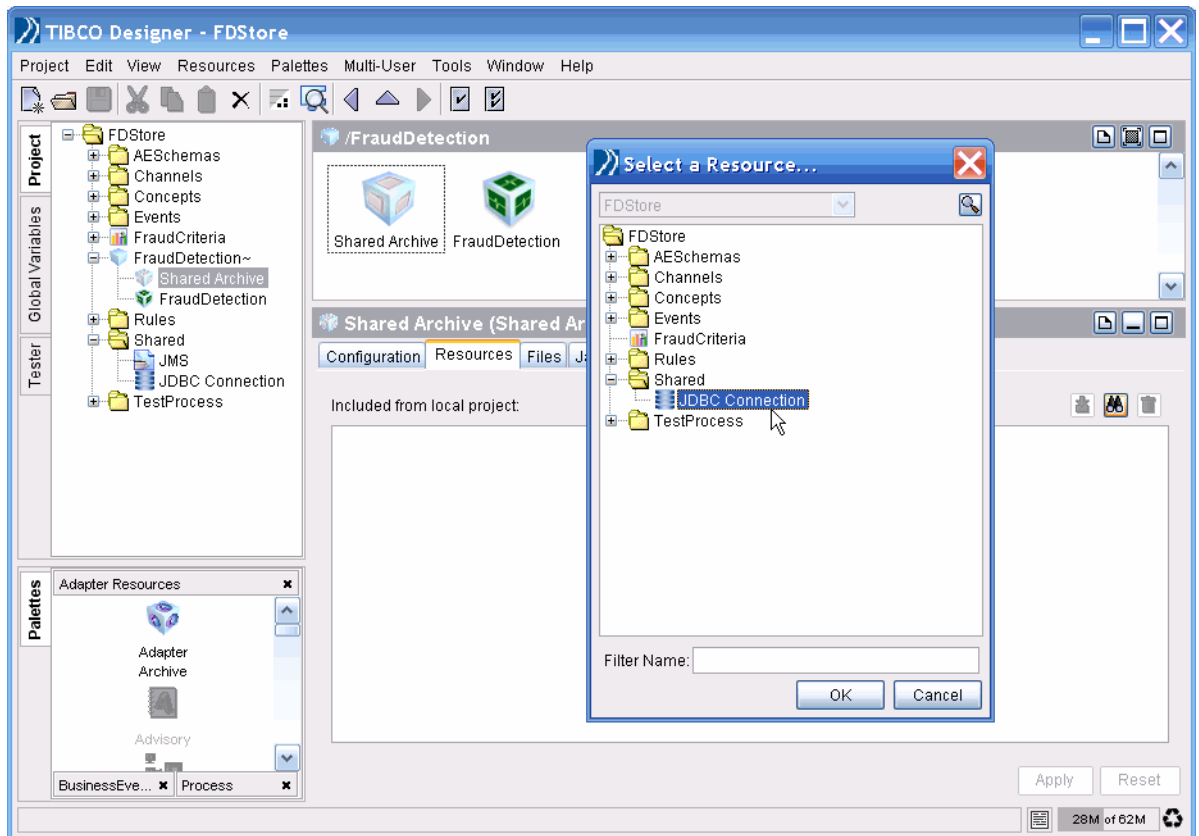
Where 1521 is the default port, and XE is the default instance name for Oracle Database 10g Express Edition. (The default instance name for Oracle Database 10g is ORCL).

5. In the User Name and Password fields, enter the username and password of the database user you created (see [Task C, Run the Initialize Database Script to Create the Oracle User, on page 65](#)). The username and password used in the tutorial are both `be_user`.
6. Make sure Oracle is up and running, then click **Test Connection**. If the details are correct, you see a success message.



7. Click **Apply**, then save the project.
8. You must add this resource to the shared archive. Select the Shared Archive resource (SAR) within the EAR resource. Select the Resources tab, browse to

the location of the JDBC Connection resource you just configured, and select it.



9. Click **Apply** then click **Save** to save the project. When you build the EAR file, the JDBC Connection resource is packaged into the shared archive (SAR) file.
10. Double-click the FraudDetection enterprise archive resource to open it and in the Configuration tab File Location field, change the EAR file name to **FDStore.ear** and save it somewhere on your computer, for example, C:\temp\FDStore.ear.
11. Click **Apply**, click **Save**, and then click the **Build Archive** button to build the EAR file.

Configure the Backing Store Properties

In this task, you add the same set of backing store related properties to each TRA file.

Learning Points Backing store related properties do the following:

- Enable backing store functionality
- Reference the JDBC Connection resource, using its project path
- Set various connection pool properties

Setting connection pool properties BusinessEvents performs backing store operations in bursts after each RTC. To tune your system, monitor the total number of threads, garbage collection, heap size, and so on, and adjust the pool size as needed. Using more connections can speed up recovery in the event of a failure, but uses more system resources.

Preloading options Some optional properties specify that at startup, some, or none of the objects in the backing store are loaded into the cache, thus controlling the size of the cache. Objects not loaded at startup are loaded on an as-needed basis, when needed during an RTC, when they are not found in the cache.

Limited Cache Size An optional pair of properties lets you set a limit to the size of the cache:

```
be.engine.cluster.isCacheLimited
java.property.be.engine.limited.cache.back.size.limit=10000
```

Entries above the limit are kept in the backing store. The default size is 10000. If you want to use the default, you only need to add the `isCacheLimited` property.

For more details, See Chapter 25, Project Configuration for Backing Store in *TIBCO BusinessEvents User's Guide*.

Task E Configure Backing Store Properties

1. In the `BE_HOME/bin` directory, open any one of the engine properties file you configured for the caching example: `cs.tra`, `agent1.tra`, `agent2.tra`, `agent3.tra`.
2. Add the following set of properties to all the TRA files.

```
#### Enable Backing Store Functionality

be.engine.cluster.hasBackingStore=true

#### Set JDBC Connection Resource Project Location
```

```
be.oracle.dburi.0 /Shared/JDBC Connection.sharedjdbc
```

```
#### Set Connection Pool Properties
```

```
be.oracle.dburi.pool.initial.0=2  
be.oracle.dburi.pool.min.0=2  
be.oracle.dburi.pool.max.0=5  
be.oracle.dburi.pool.enforce=true
```

(If you do not add `be.oracle.dburi.pool.enforce=true` then the pool properties are ignored even if present.

Deploy and Test the Application

Task F Deploy the Agents and Cache Servers



Ensure that the TIBCO Enterprise Message Service server has started. One way to start the server is as follows:

Start > All Programs > TIBCO > TIBCO EMS *VersionNumber* > Start EMS Server.

1. Open five command windows and in each navigate to *BE_HOME/bin*. For example: `c:\tibco\be\3.0\bin`.
2. In each window enter the one of the following commands (modified as needed according to the location of your EAR file):

```
be-engine -n CS1 --propFile cs.tra c:\temp\FDStore.ear
```

```
be-engine -n CS2 --propFile cs.tra c:\temp\FDStore.ear
```

```
be-engine -n engine1 --propFile agent1.tra c:\temp\FDStore.ear
```

```
be-engine -n engine2 --propFile agent2.tra c:\temp\FDStore.ear
```

```
be-engine -n engine3 --propFile agent3.tra c:\temp\FDStore.ear
```

For background information see [Deploy the Agents and Cache Servers on page 58](#).

Task G Test the Backing Store Functionality

Follow instructions in [Test the Application on page 60](#) to start the five engines and create some test data. Then try the following:

- Stop both cache servers. The data is no longer available to the application.
- Start both cache servers. The data is loaded from the backing store.
- Make more debits. The debits use the account balance that was loaded from the backing store.

You can also look at the Oracle database tables (not documented.)

Congratulations—You have completed the backing store tutorial!

Troubleshooting

See [Troubleshooting on page 72](#). Also, if the cluster fails to start correctly you may have to reset the database. See [Reset the Backing Store Tutorial on page 73](#).

Reset the Backing Store Tutorial

You can reset the tutorial for re-use. Different levels of reset are available as explained next:

| Do do this: | Do the following: |
|---|--|
| Remove the backing store data only. | Run <code>FDStore_cleanup.sql</code> . |
| Remove the database schema. | Run <code>FDStore_remove.sql</code> . Then run: <code>base_types.sql</code> <code>create_tables.sql</code> <code>FDStore.sql</code> As explained in Task C in the section Prepare the Database on page 64 . |
| Remove the database user and the database schema. | Run <code>initialize_database.sql</code> , and all other scripts as described in Task C in the section Prepare the Database on page 64 |

Appendix A **Getting Started With TIBCO Designer**

TIBCO Designer is an easy to use graphical user interface for creating integration projects.

This chapter and the next give an introduction to TIBCO Designer that is product independent. In this chapter, you learn about TIBCO Designer basics. In the next chapter, you learn about creating and managing projects and working with global variables.

These chapters are taken from *TIBCO Designer User's Guide*. The complete documentation, including *TIBCO Designer Palette Reference*, is available here on Windows:

Start > Programs > TIBCO > TIBCO Designer *version* > Designer Documentation

You can also access it here: *TIBCO_HOME\designer\version\doc*.

Topics

- [Welcome to TIBCO Designer, page 76](#)
- [Starting TIBCO Designer, page 77](#)
- [TIBCO Designer Administration, page 80](#)
- [TIBCO Designer Interface Overview, page 82](#)
- [Working With Palettes, page 87](#)
- [Customizing the Display, page 93](#)
- [Accessing Documentation, page 95](#)

Welcome to TIBCO Designer

TIBCO Designer allows you to easily create integration projects for your enterprise computing environment.

TIBCO Designer is available as a graphical user interface to different TIBCO products and is used by those products for configuration. Depending on the product you installed, you can, for example, use TIBCO Designer to create TIBCO BusinessWorks process definitions or create or modify adapter configurations.

- TIBCO BusinessWorks is a scalable, extensible, and easy to use integration platform that allows you to develop, deploy, and run integration projects. TIBCO BusinessWorks also includes an engine that executes the process, and a web-based GUI for monitoring and managing run-time components.
- Adapters allow you to configure the interface between an external system, such as an SAP R/3 application or a database, and the TIBCO ActiveEnterprise™ environment. Adapters are available as separate products.
- Custom adapters are created using the TIBCO Adapter SDK. You can prepare an adapter configuration for custom adapters using the *Adapter Resources* and *Adapter Schemas* palettes, which are discussed in *TIBCO Designer Palette Reference*.
- TIBCO BusinessWorks Collaborator™ gives companies the ability to coordinate business activities, measure their efficiency, and optimize them over time. The product facilitates complete visibility into business activities, along with the ability to collaborate on the modeling and modification of the rules and flows that define business those activities. TIBCO BusinessWorks Collaborator uses TIBCO Designer for configuration of FormFlows processes and for preparing Enterprise Archive files.

Starting TIBCO Designer

The following sections describe how to start TIBCO Designer and explain the options available once TIBCO Designer starts.

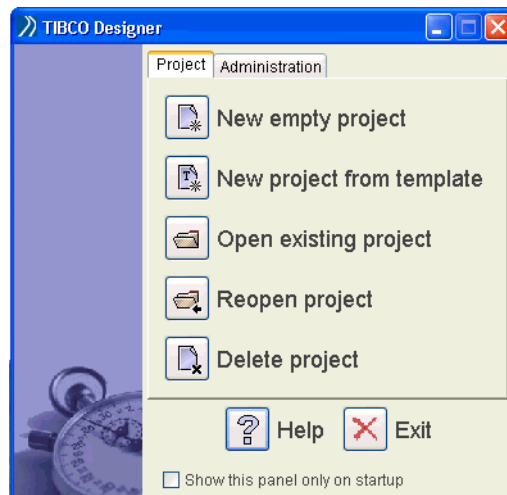
To Start TIBCO Designer

- Under Microsoft Windows:
Choose **Start > All Programs > TIBCO > TIBCO Designer *n.n* > Designer *n.n***
or
Invoke `install-path\tibco\designer\i.n.n\bin\designer`
- Under UNIX:
Invoke `install-path/tibco/designer/i.n.n/bin/designer`

Startup Options

When you launch TIBCO Designer, the startup panel is displayed:

Figure 1 TIBCO Designer startup panel



[Table 3](#) describes the startup options. [Table 4, Startup panel Administration options](#), describes the options available when you select the Administration tab.

Table 3 Startup panel Project options

| Option | Description |
|----------------------------------|---|
| New empty project | <p>Opens a new empty project in TIBCO Designer. An empty project includes the TIBCO Designer default palettes and their resources.</p> <p>By default when you open a new project, TIBCO Designer prompts you immediately where you want to save it. You may provide the location or click the Cancel button. If you do not want to see this dialog each time you create a new project, choose Edit > Preferences > General and unselect Show save dialog for new project.</p> |
| New project from template | <p>Opens a predefined project template. A project template is a pre-built project. It can contain folders, configured resources, and partially configured resources. Resources are the components of each project (see Resources on page 84).</p> <p>A project template can be preconfigured to include all the resources you may need for a certain type of project (for example, Web Services configuration). Using a template makes it possible to leverage an existing configuration when creating new projects.</p> <p>See Project Templates on page 99 for information on creating project templates.</p> |
| Open existing project | <p>Opens an existing project. See Opening and Reopening Projects on page 104.</p> |
| Reopen project | <p>Allows you to choose from a list of recently saved projects. TIBCO Designer may prompt for information, for example, a password.</p> |
| Delete project | <p>Allows you to delete a project. See Deleting Projects on page 111.</p> |

Table 3 Startup panel Project options

| Option | Description |
|--|--|
| Help | <p>Displays TIBCO Designer documentation. You may be prompted for your browser location if you are using TIBCO Designer for the first time. Information about browser locations on some operating systems is included in the prompt screen.</p> <p>You need to specify this path only once. After that, TIBCO Designer remembers the location even if you uninstall the current version and install a new version.</p> |
| Exit | Exits TIBCO Designer. |
| Show this panel only on startup | <p>If checked, the startup panel is only displayed during startup and closed after you've made your selection.</p> <p>If cleared, this panel reappears when no other TIBCO Designer windows are open. Leaving the panel on screen can be useful for project maintenance.</p> |

TIBCO Designer Administration

When you start TIBCO Designer, the startup panel allows you to open projects (see [Startup Options on page 77](#)). You can also use the startup panel to perform some TIBCO Designer administrative tasks. When you choose the Administration tab, the panel presents the choices shown in [Figure 2](#):

Figure 2 Startup panel Administration options

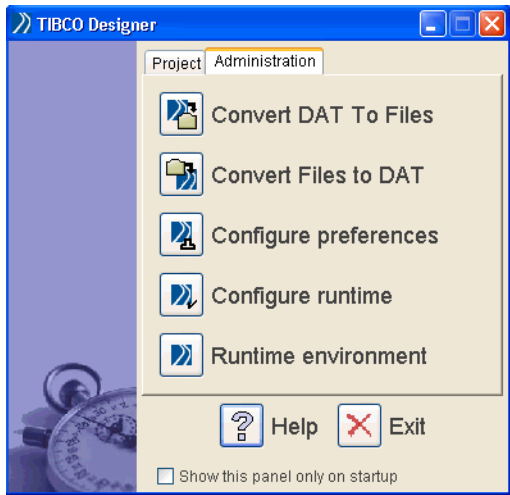


Table 4 Startup panel Administration options

| Option | Description |
|----------------------|---|
| Convert DAT to files | Displays a dialog that lets you specify the .dat file you want to convert and the folder for the multi-file project. Because a .dat file is a legacy format, you must convert it to a multi-file project before you can open a project from TIBCO Designer. |

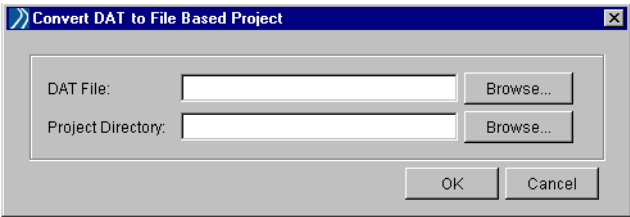


Table 4 Startup panel Administration options

| Option | Description |
|------------------------------|--|
| Convert files to DAT | Displays a dialog that lets you choose a project directory and a .dat file. When you specify both, TIBCO Designer converts the multi-file project in the directory to the .dat file. |
| Configure preferences | Displays the Preferences dialog. |
| Configure runtime | <p>Allows you to configure the TIBCO Designer runtime environment. You have these options:</p> <ul style="list-style-type: none"> Extended Class Path—classpath to be used by TIBCO Designer. You can specify file names or directories. If you specify directories, all .class, .zip and .jar files are loaded. The order in which the files are loaded depends on the file system. Palette Path—Location from which TIBCO Designer loads palettes. Maximum Heap Size—Maximum JVM heap size. User Directory—Default location for the application to store files. Command Line Arguments—Allows command line arguments to be passed to Designer. Currently -d (debug) is supported. If you specify -d, the log that is sent to the console becomes more detailed. |
| Runtime environment | <p>Displays TIBCO Designer runtime information. This information, which includes palette name and version information and Java property and value information, can be useful for debugging or during interaction with TIBCO Technical Support.</p> <p>Use the Export Runtime Settings command to create a file with all pertinent information.</p> |

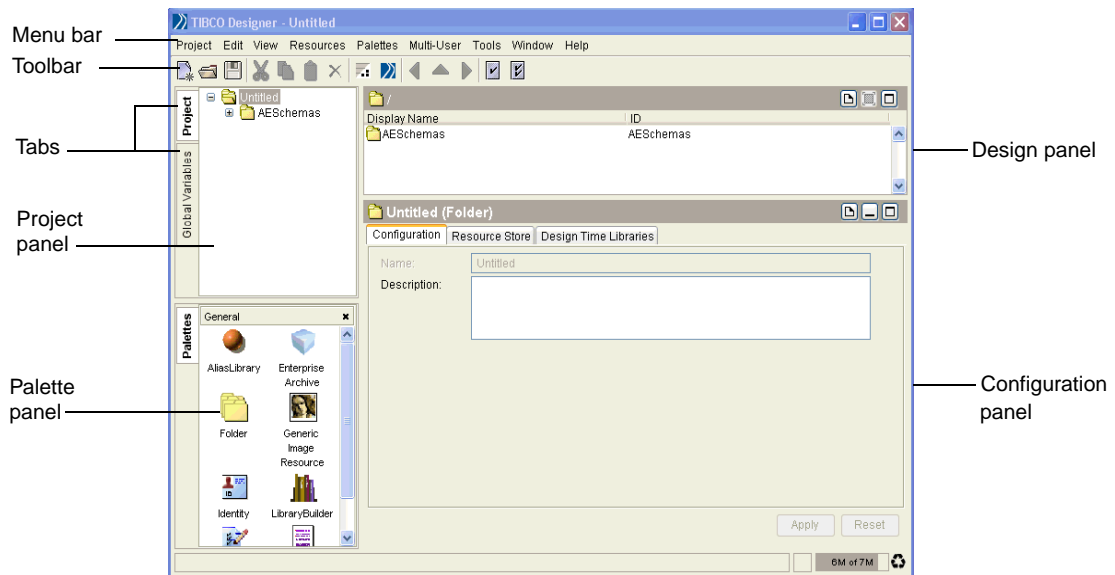
TIBCO Designer Interface Overview

The TIBCO Designer interface allows you to perform various functions. This section describes the TIBCO Designer main window and explains what you see in each of its panels.

Main Window

Figure 3 illustrates the TIBCO Designer window.

Figure 3 The TIBCO Designer window



The TIBCO Designer window has these components:

- Menu bar and Menus.
- Toolbar icons.
- Tabs in the leftmost area allow you to change what is displayed in the panel. See [Customizing the Display on page 93](#).



When something in the design panel or the configuration panel is selected, the panel is highlighted. This helps you see at one glance where the focus is.

- Four panels, which are (starting in the top left corner and continuing clockwise):
 - Project panel (can display the project tree or the project's global variables)
 - Design panel
 - Configuration panel
 - Palette panel



You can rearrange the panels and what they display. For example, the project panel and palettes can be combined to share one set of tabs. See [Customizing the Display on page 93](#) for more information.

The following sections explain the contents of each panel.

Project Panel

A *project* contains resources that implement the enterprise integration. This includes services (producers and consumers of information), any business logic that may be applied to that information, and deployment information.

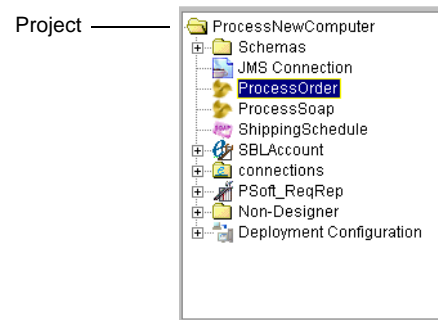
By default, the project panel allows you to view the [Project Display](#) or [Global Variables Display](#). When used in conjunction with other projects, the panel may be used for other purposes, for example, in conjunction with the TIBCO BusinessWorks tester.

Project Display

With the **Project** tab selected, the project panel displays the project tree. This includes the top-level (root) folder and the hierarchy of resources. The hierarchy of folders and resources corresponds to the hierarchy of folders and files in the project folder.

[Figure 4](#) illustrates an example project, *ProcessNewComputer*, in the project panel. Multiple TIBCO products were used to create the integration project: it contains two TIBCO BusinessWorks process definitions (*ProcessOrder* and *ProcessSoap*) and a Siebel adapter (*SBLAccount*).

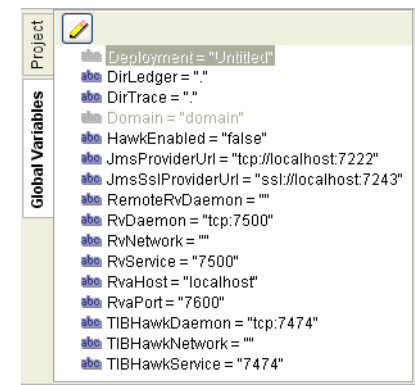
Figure 4 Project tree in the project panel



Global Variables Display

Global variables are associated with each project. To display them, click the Global Variables tab of the project panel. Clicking the pencil icon displays the global variable editor. See [Working With Global Variables on page 107](#) for more information.

Figure 5 Global variables in project panel



Resources

Resources are the components of a project. A simple TIBCO Designer resource corresponds to an object in a TIBCO application, such as an adapter configuration, an adapter service, a process definition, or an FTP activity. Resources can be complex and contain other resources, much like a folder can contain other folders on your computer's file system. For example, an adapter configuration may contain multiple folders with multiple publisher or subscriber service resources.

Each top-level resource (for example, each adapter configuration) corresponds to a file in the project's hierarchy of files in the project directory. This design allows developers to use a source control system and to check out only the top-level resources they are working with from a source control system, thus sharing their work.

Most resources have context-sensitive help available for the configuration of that resource. Right-click on the resource and choose **What Is This?** from the popup menu for more information on configuring the resource.



If TIBCO Designer cannot determine the type of a resource, it displays it as a special icon. This icon might indicate, for example, that the palette for that resource is not installed, that the file is not really part of your project, or that it has a misleading extension.

While you can place any file in a project, TIBCO Designer will display an error if it cannot map the file extension to a resource.

For example, source control systems may hide files from the user. If you open a Designer project that was under source control without the revision control system, these files will become visible and treated as "unknown" resources.

The icon at left may be different on Unix systems.

Palette Panel

Palettes organize resources and allow you to add them to your project. You select resources in the palette panel and drag and drop them into the design panel to add them to your project.

See [Working With Palettes on page 87](#) for more information.

Design Panel

The design panel displays the current resource selected in the project panel. For resources that contain other resources, the contents of the selected resource are shown in the design panel. For example, if you select a folder, its contents is displayed.

Configuration Panel

The configuration panel allows you to specify various configuration options for each resource. The type and the purpose of the selected resource determine the contents of the configuration panel. Usually there are one or more tabs in the configuration panel that allow you to access the various configuration options. The tabs organize the configuration options for the resource.

Click the help icon  in the top right corner of the configuration panel for online help on the current selection.

After you have added the configuration information, you must click the **Apply** button for each tab. If you decide you do not want to add the configuration information, click **Reset** before you apply any changes to return to the previous values for each field in the tab.

Working With Palettes

TIBCO Designer contains a small number of native palettes. In addition, each TIBCO application you install that uses Designer adds one or more palettes during installation.

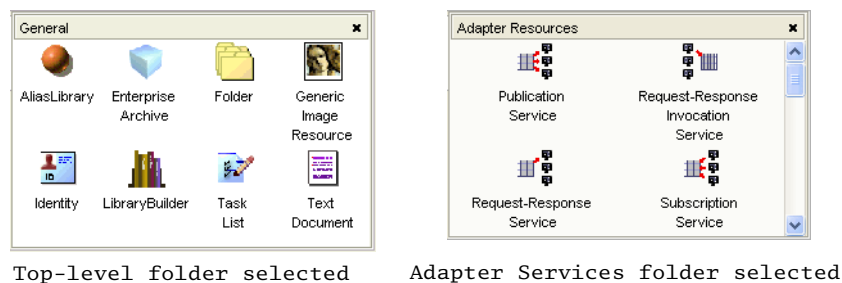
Which palettes are displayed depends on:

- the installed TIBCO products
- the resource selected in the project tree
- your preferences.

Current Selection and Palette Panel Display

When the default view is set as your view preference, the current selection in the project tree determines which palettes are displayed in the palette panel. You can change your view preferences to change what's displayed in the palette panel. See [Customizing the Display on page 93](#) for more information about how TIBCO Designer functions in palette mode.

Figure 6 Palette panel changes depending on current selection

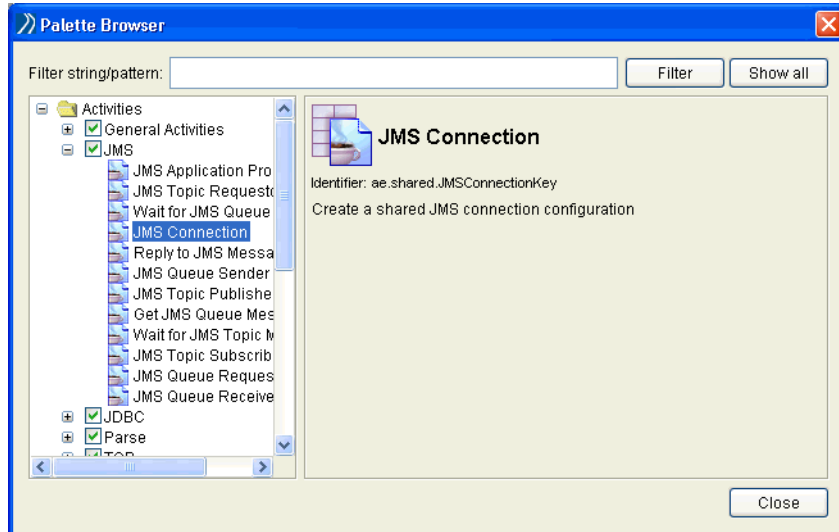


For example:

- Select the top-level project folder to see a palette for each adapter and some other palettes for general resources.
- Select the Adapter Services folder of an adapter in the project tree to see a palette of service resources. Drag any service resource into the design panel to add that resource to that adapter.

Using the Palette Browser

The palette browser allows you to manage palettes easily. You can explore palette groups, palettes and resource types, and enable or disable specific palettes to display in the palette panel. The browser includes a search filter that allows you to display only palettes that match a specific string. Click **Palettes > Browse** to display the palette browser.



The palette browser displays all available palettes grouped by palette groups in a hierarchy. Selecting a palette or resource in a palette displays its name, resource id and description in the right pane.

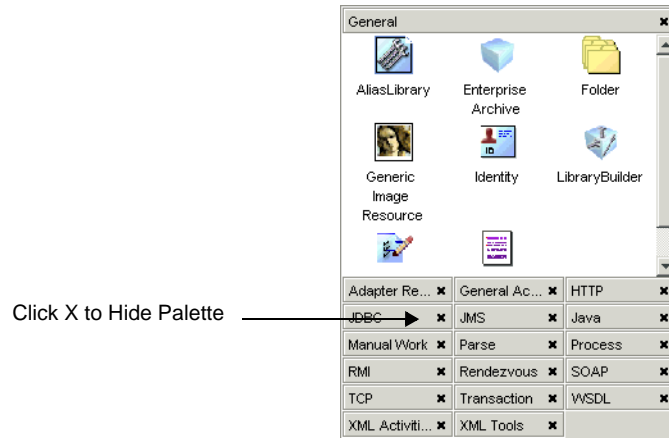
Clearing the check box next to a palette's name in the tree hides that palette in the palette panel. Selecting the check box shows that palette in the palette panel. You must close the palette browser to view the changes you have made.

By default, all palettes are visible in the palette browser. You can provide a filter in the **Filter string/pattern** field to limit the display. The filter uses regular expression matching. That is, if you enter JMS in the field, only palettes that have JMS in their name, description or containing palette are displayed. Matching is case insensitive. Entering a new expression and clicking **Filter** again performs a new search on all resource types and will not refine the current search.

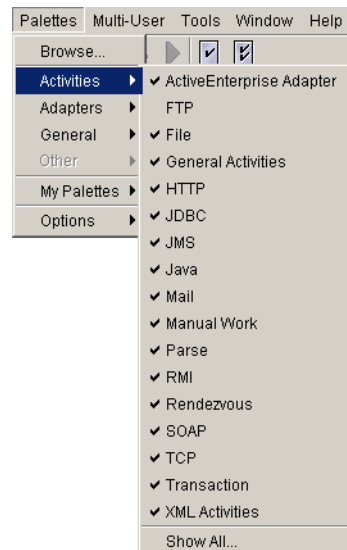
To clear the filter expression and display all palettes either click **Show all** or remove the expression from the entry box and click **Filter** again.

Showing and Hiding Palettes

Palettes can be hidden by clicking the **X** icon next to the palette name. For example, the following diagram shows the palette panel with a large number of palettes displayed. You can disappear a palette by clicking the **X** icon.



To redisplay a palette quickly, click the Palettes menu and select the palette to display. For example, the next diagram shows that the FTP palette is not visible in the palettes panel (an X does not display next to FTP). After you select FTP, its palette will display in the palette panel. Note that you can hide palettes using this menu. Each time you select a palette, it toggles between appearing and disappearing.



Choosing Palette Mode or Non-palette Mode

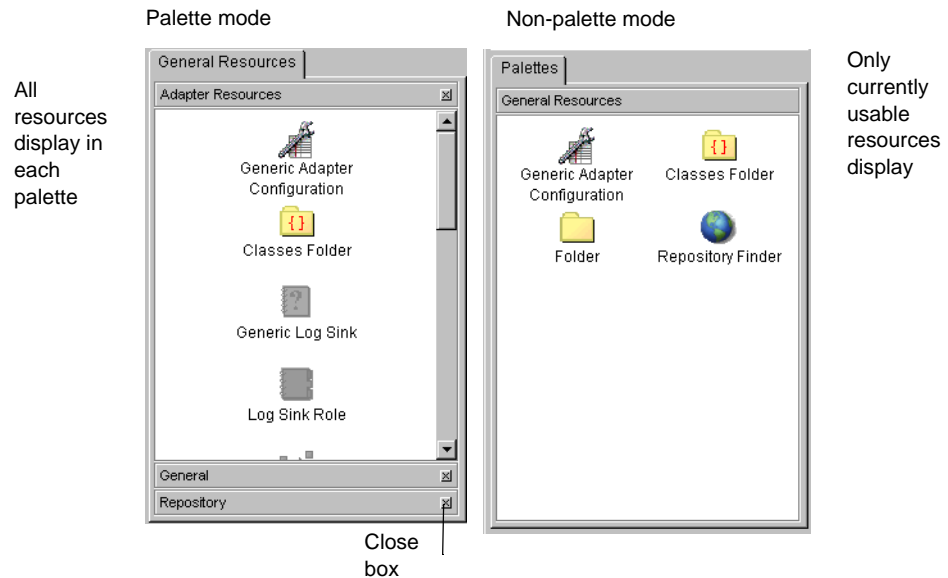
TIBCO Designer allows you to change the palette panel display to use palette mode or non-palette mode.

- In palette mode, each resource is shown in the palette it belongs to, and each palette shows all resources. In that case, unusable resources are grayed out.
- In non-palette mode, resources are displayed directly in the panel, and only currently usable resources are displayed.

While in palette mode, you can close individual palettes using the close marker (X) on the right. If you don't see close markers, choose **Palettes > Options > Show Close Boxes**. To hide close markers, choose **Palettes > Options > Hide Close Boxes**.

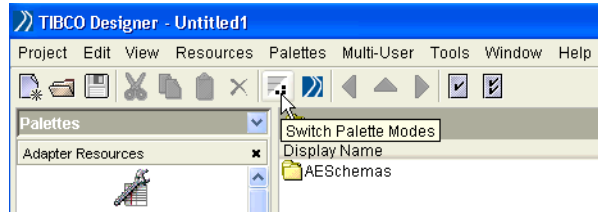
To redisplay a closed palette, choose **Palettes > Browse**, then locate and select the palette.

Figure 7 Palette mode vs. non-palette mode



To Switch Palette Modes

- Choose **Palettes>Options>Switch Palette Modes**.
- or
- Click the **Switch Palette Modes** button located in the tool bar.



Displaying Palettes in a Separate Window

You can display the palette panel in a separate window by choosing **Palettes > Options > Show Palettes in New Window**. You can also access this menu command from the right-button menu of any palette in the palette panel.

To restore the palette panel to its location in the main window, close the separate window in which the palette panel is displayed.

Working With User Palettes

User palettes allow you to save a collection of resources to a predefined location and either share it with other users or use it yourself at a later time.

To Create a User Palette

1. Choose **Palettes > My Palettes > New Palette**.
2. Specify the name of the palette.

When you later save your project, the palette is saved to the location specified by the `User Palette Directory` General Preference.

3. Drag resources from the project tree or from the design panel into the user palette.
4. When you save your project, the custom palette is then saved to the location specified by the `User palette directory` under the General tab in the Preferences dialog.

To Load a User Palette

1. Choose **Palettes > My Palettes > Reload Palettes**.

To Delete a User Palette

1. Make sure the palette is loaded.
2. Choose **Palettes > My Palettes > Delete Palettes**.

You are prompted for the palette(s) you want to delete.

Customizing the Display

You can customize how TIBCO Designer displays panels and palettes. This section gives an overview of the most frequently used display preferences.

Display preferences and other preferences are saved when you exit TIBCO Designer, even if you do not save your project. Display preferences are maintained for each user, even if that user completely uninstalls the product and installs a different version.

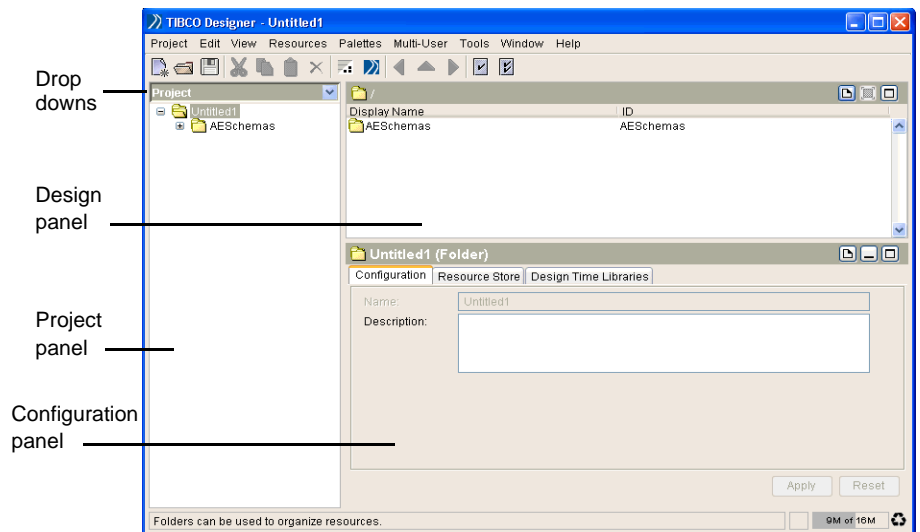


To return to the default settings, choose **Edit > Preferences** and click **Restore Default Settings**.

Choosing Panel Layout

If you prefer to view either the project tree or the palette panel, but not both in the left panel, you can set TIBCO Designer to display the three-panel view shown in [Figure 8](#). The view also uses drop downs, rather than tabs to access the project, global variables or palettes.

Figure 8 The three-panel view

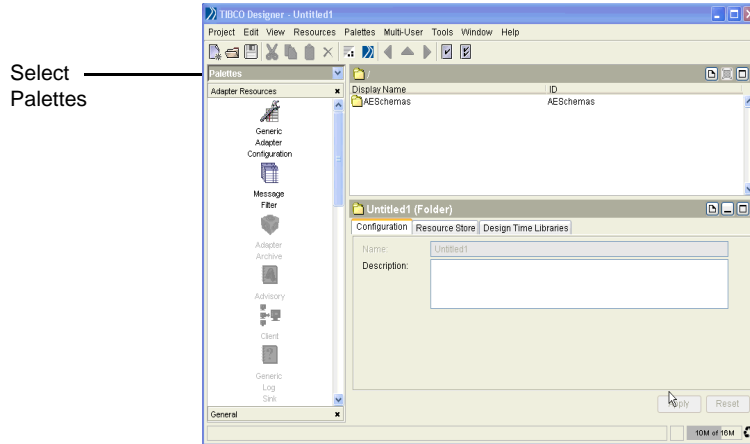


To Choose the Three-panel View

1. Choose **Edit > Preferences > View**.
2. Under **Layout**, select the appropriate icon and click **OK**.

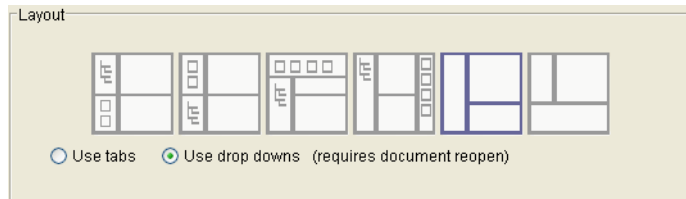
To navigate to palettes in this view, click the **Palettes** tab on the left (see [Figure 9](#)). The next diagram shows the results of this action.

Figure 9 Three-panel view with palettes selected




Additional layout options are available when you choose **Edit > Preferences > View**. If you select the **Use tabs** or **Use drop downs** option, you must close and reopen your project to make the change visible.

Figure 10 Layout options



Accessing Documentation

Documentation for TIBCO Designer is available in several ways:

- If you are using Microsoft Windows, you can use the **Start** menu to access TIBCO Designer documentation.
- From TIBCO Designer:
 - Click **Help > Designer Help** at any time to view this manual, which discusses how to use TIBCO Designer regardless of the application you are building.
 - Use **Help > Help For** to access product-specific documentation.
 - Right click on most resources and choose the **What Is This?** menu item to view specific help for that resource. If information is displayed in the Configuration panel, you can also click the Help icon  for online help.



When you invoke Designer help for the first time, you are prompted for the location of your web browser. You only need to supply this location once. Location information is included in the prompt.

Appendix B **Managing Projects and Resources**

Projects are the key organizational principle for the configuration information you specify with TIBCO Designer.

This chapter explains how to manage projects and the resources inside them.

Topics

- *Overview of Projects, page 98*
- *Creating Projects, page 100*
- *Validating Projects, page 101*
- *Saving Projects, page 102*
- *Opening and Reopening Projects, page 104*
- *Adding Resources To Your Project, page 105*
- *Working With Global Variables, page 107*
- *Deleting Projects, page 111*
- *Tips and Tricks for Working With Projects, page 112*

Overview of Projects

A *project* is a collection of resources, including, for example, adapter resources and process definitions. Together, these resources define the configuration of your integration project. In the TIBCO Designer window, a project is represented by the top-level (root) folder in the project panel. The top-level folder is initially named `Untitled` and is renamed to the name of the project when you save the project for the first time.

Each TIBCO Designer window contains only one project. If you open a second project, TIBCO Designer opens a new window.

When you close a project, the startup panel remains available for project maintenance or for opening a different project unless you selected the **Show this panel only on startup** check box on that window.

Project Structure

When you save a project, TIBCO Designer creates a hierarchy of folders and files in the location you choose for the project.



When you supply a project location, be sure no other files reside in that folder. TIBCO Designer removes any existing files before placing the project files into the folder.

When you create a multi-file project:

- There is one file per top-level resource. For TIBCO BusinessWorks, that may mean one file per process. For adapters, more resources may be considered part of a top-level resource.
- The project root directory identifies the project. The project root can be located anywhere in the file system and is determined when you first save the project. All components of a project are located under this common project root.
- Directories in the file system become folders in TIBCO Designer. However, not all folders in TIBCO Designer are directories in the file system:



- Folders created from a Folder resource in the General Palette (displays a multi-folder icon) become folders in the file system.



- Other folders, such as the Adapter Services folder inside an adapter configuration are logical folders. These folders only exist in memory in the resource that holds them. The actual data for these elements is stored in the file of the top-level resource. For example, the Adapter Services data are stored in the Adapter Configuration file.

- TIBCO Designer creates a file named `vcrepo.dat` in the project root directory when you first save the project. This file is used to store properties such as display name, TIBCO Rendezvous encoding, and description. This file can be used for identification in place of the project root directory and can be used as the repository locator string (`repoUrl`).



This file cannot be used for identification or as the `repoUrl` for TIBCO Adapters (including custom adapters based on TIBCO Adapter SDK).

Note that for ActiveEnterprise 5.2, the preferred method of deployment is to generate an EAR file and upload it into TIBCO Designer, not to run the adapter using a properties file.

Using a Version Control System

Multi-file projects support the use of different version control systems because they consist of separate files for each versionable component.

Once the root directory is defined under the control of a version control system, standard version control system tools can be used.

Project Templates

A project template is a pre-built project. It can contain folders for organization, configured resources, and partially configured resources. You can use a project template as the foundation for other projects similar in nature. Using a template, you can leverage your work when performing similar configurations.

Creating Projects

You create a new project using the startup panel when starting TIBCO Designer. You can also choose **Project > New Project** from the TIBCO Designer menu bar with TIBCO Designer already open. In that case, TIBCO Designer opens a new window for the new project.

TIBCO Designer allows you to create a project from scratch or to create a template-based project.

- **New Empty Project** — A new project contains a single AESchemas folder that will be used for adapter schema resources. See [Project Structure on page 98](#) for more information.
- **New Project from Template** — When you save a project as a template (**Project > Save As Template**), you can later load that template and customize it to create a new project. See [Project Templates on page 99](#).

When you create a new project, you are, by default, prompted immediately to save it. See [Saving Projects on page 102](#) for a discussion of the information you must supply.


Validating Projects

After you have created a project, you add resources to it and supply configuration information for your resources.


Before you prepare a project for deployment, it is critical that you validate it. TIBCO Designer includes reference-checking and other validation facilities that allow you to make sure a project is internally consistent. This is essential if you intend to run the project, or hand it to another user.

During validation, each resource always checks for broken references. Many resources have other resource-specific validation behavior.

To Validate a Resource

- Select the resource to be validated, then choose **Resources > Validate Resource** from the menu bar.
- With the resource selected, choose the Validate Resource icon. 

To Validate All Resources

- Choose **Project > Validate Project for deployment**.
- Click the Validate Project for Deployment icon. 

Note that TIBCO Designer handles references as strings. TIBCO Designer will help keep these references up to date, for example, when you move a resource to a different location. It is, however, possible to have "broken" references, for example, if you delete a resource and ignore the warnings displayed by TIBCO Designer. You can use the validation commands to find broken references.



By default, TIBCO Designer prompts whether you want to perform reference checking each time you perform an activity that might result in a broken reference (move, rename, and so forth). You can change the default behavior using the **Edit > Preferences > References** tab.

Saving Projects

When you save a project, you can save it as a multi-file project, or under XML Canon. This section explains how to save a multi-file project.

When you save a multi-file project, you have these choices:

- **Project Directory** — The directory that will contain the project files. Click **Browse** to select the directory.



Designer will remove any files in this directory when you save the project.

- **TIBCO Message Encoding** — Character Encoding used for the communication between TIBCO product components in this project at design time (debug mode), or if the project is running as a legacy local file-based project. The communication transport could be either TIBCO Rendezvous or TIBCO Enterprise Message Service. You have two choices:
 - **ISO8859-1 (Latin-1)**—Preferred encoding for projects that deal only with English and other Western European languages that belong to the ISO Latin-1 character set. If this encoding is used for languages that do not belong to the Latin-1 character set (such as Japanese, Arabic, etc.), data loss may result.
 - **UTF-8**—Preferred encoding for projects dealing with languages not belonging to the Latin-1 character set. This includes most languages except for English and other western European languages.




After deployment, the encoding setting of the TIBCO Administration Server will supersede this encoding. See the *TIBCO Administrator Server Configuration Guide* for more information.

- **Multi-User System** — Allows you to use a multi-user system such as file sharing, Perforce, or Visual SourceSafe.

After you have saved a project to a repository, you can select the project's **Project Settings** tab to:

- View information about the project. The information displayed depends on how the project was saved.
- View and change the project's messaging encoding for the data communication among the components in this project. This is only used in design mode, or when the project is running as a legacy local file-based project.

To Save a Project

1. In the main window, do one of the following:
 - Choose **Project > Save**.
 - Choose **Project > Save As** and specify the storage directory.
 - Click the **Save** icon .
2. In the dialog that is displayed, make sure the **Multi-File Project** tab is selected and provide the following information:
3. Click **OK**.

To Save a Project as a Template

1. Choose **Project > Save As Template**.
2. Provide the appropriate information, which is the same as discussed in [Saving Projects on page 102](#).

Opening and Reopening Projects

You can open a project in two ways:

- From the startup panel when you launch TIBCO Designer.
- Choose **Project > Open** from the TIBCO Designer main window if Designer is already open. In that case, TIBCO Designer will create a new window for your project.



If you need to open a .dat project, you must convert it first:

1. In the startup panel, click the **Administration** tab.
2. Choose **DAT to Files**.
3. Supply the name of the project directory when prompted.

You can then open the multi-file project from TIBCO Designer.

If you are opening a project under a revision control system, you need to provide the appropriate information.

You can reopen a project you opened recently in two ways:

- From the startup panel when you launch TIBCO Designer.
- Choose **Project > Reopen** from the TIBCO Designer main window if Designer is already open. In that case, TIBCO Designer will create a new window for your project.

Adding Resources To Your Project

Once you have created or opened a project, you can add resources to your project. To add a resource, you first select it in the palette panel, then drag and drop it into the design panel.

To Add a Resource to Your Project

1. Select the palette in which the resource can be found. For example, you find an adapter configuration resource in the palette named after the adapter.



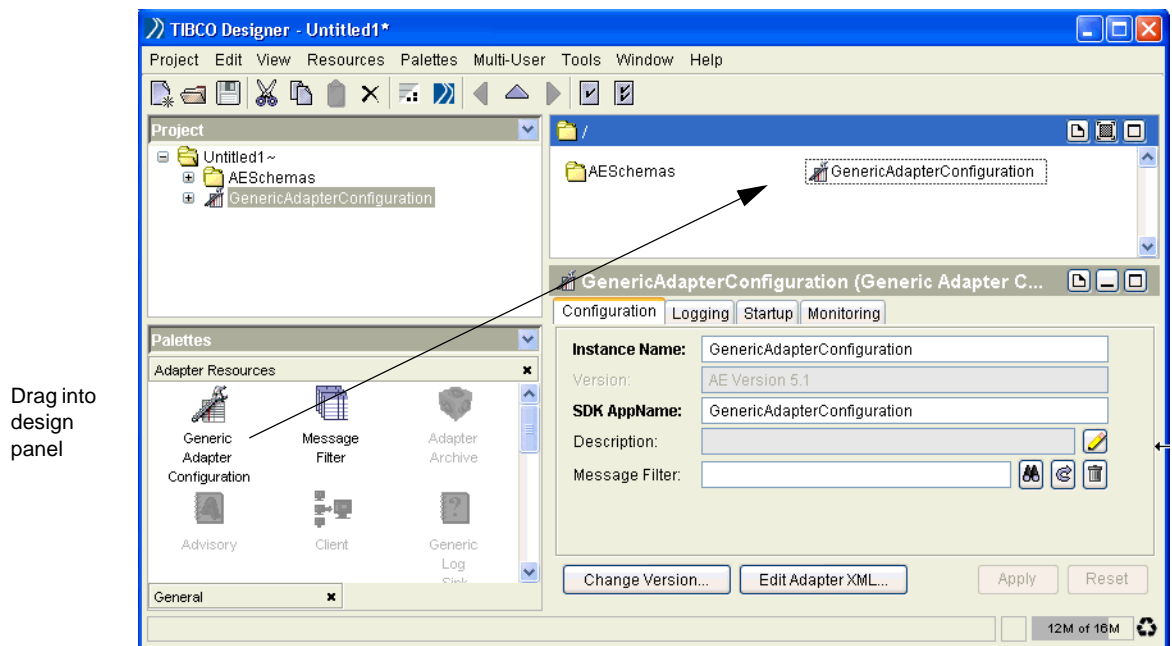
If the palettes are not visible in your palette panel, click the Switch Palette Modes icon.

2. Select a resource within the palette and drag and release it in the design panel.

The resource instance is displayed in the design panel and also added to your project tree. The configuration panel allows you to specify configuration information for the resource.

Figure 11 illustrates adding a resource to a project.

Figure 11 Adding a resource to a project





You can also add resources in other ways:

- In the palette panel, select the resource and choose **Add This To The Project** from the right-button menu.
- In the design panel, right click on an empty area (not on a resource) and select **Add Resource** from the right-button menu, then choose the appropriate submenu.
- You can enable resources to be added using double-clicks. Go to **Edit > Preferences** and select the **View** tab. Under **Palettes**, select **Initiate drag-n-drop through double-clicks**.

If Adding a Resource Results in an Error

Whether adding a resource is possible depends on what is currently displayed in the design panel.

If you try to add a resource that cannot be added to the current resource, an error results.

For example, if the root folder is displayed in the design panel, you can add an adapter instance. If any other resource is displayed, you cannot add the adapter instance.



Ideally, all resources that cannot be dragged into the design panel should be greyed out (palette mode) or not visible (non-palette mode). For some custom palettes that may not always be true.

Working With Global Variables

Global variables provide an easy way to set defaults for use throughout your project. There are several ways in which they can be used:

- Define a variable using TIBCO Designer, then override the value for individual applications at deployment time using TIBCO Administrator. You can also override values for predefined variables, unless the GUI does not allow you to make them settable later.
- Predefine a variable using TIBCO Designer, then override the value for individual services (for example, publication service or TIBCO BusinessWorks process) at deployment time using TIBCO Administrator. The values you specify are then used at runtime.

You can also override values for predefined variables, unless the GUI does not allow you to make them settable later.

- Predefine the variable using TIBCO Designer, then override it on the command line.

For example, you could assign the value 7474 to the predefined global variable `RvDaemon`. You can then use the variable in different sessions in your adapter. If you want to change the TIBCO Rendezvous daemon for your adapter, you can globally set it to a different value or override it from the command line.

When you want to use the global variable in the fields of a resource, enter the variable name surrounded by `%%` on both sides. Some fields in the configuration panel, such as user name and password fields, allow you to drag and drop global variables into the field.

When the project is deployed and the configured components run, all occurrences of the global variable name are replaced with the global variable value. For example, a global variable named `RvServiceTest` with a value of 7800 would be replaced with 7800.

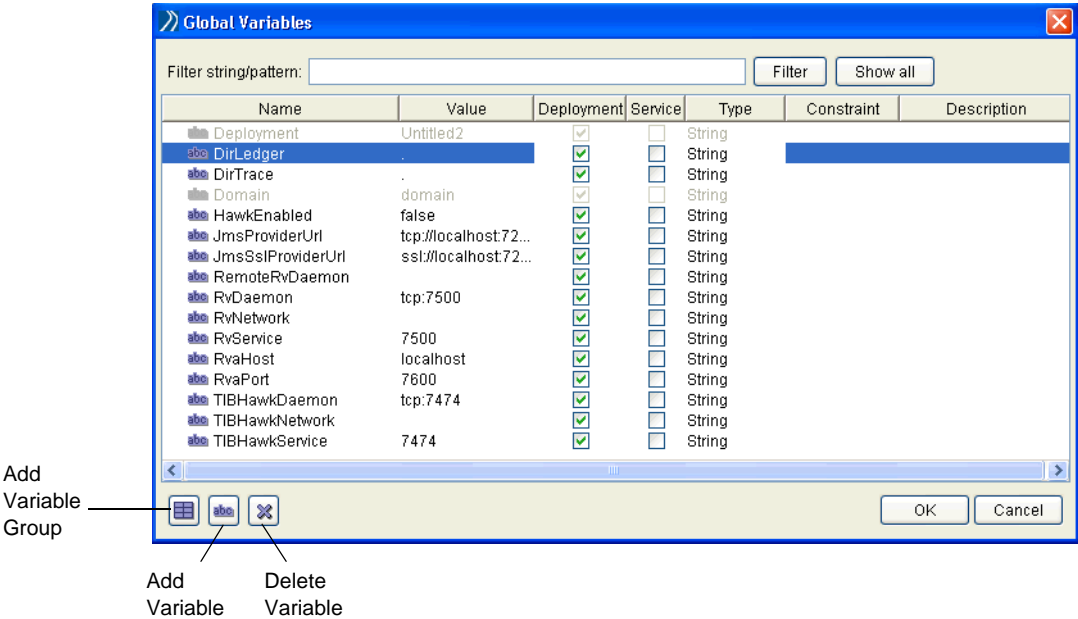
A number of global variables are predefined in TIBCO Designer. You can add definitions of any variables you need to the predefined variables.

Global Variables Editor

Use the global variables editor to create or modify global variables, mark variables as settable from TIBCO Administrator, and assign a type to a variable.

To display the global variables editor, in the project panel, select the **Global Variables** tab or drop down, then click the Open Advanced Editor (pen) icon. If you select a global variable, then click the pen icon, the editor opens with the variable selected.

The next diagram shows the global variables editor.



Adding a Global Variable or Global Variable Group

To add a global variable, select the create variable icon. A new line for the variable appears. Type a name for the variable by triple-clicking the name field.



The name MessageEncoding is reserved and cannot be used to name a global variable. A global variable of this name is created by Designer when generating an EAR file.

When creating a global variable group or variable, it is added to the bottom of the list. When you reopen the global variables editor, the group or variable displays alphabetically in the list.

Global variable groups are used for organizing variables. Variable groups are especially useful if multiple developers share a project using a version control system. To add a global variable group, select the add variable group icon.



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.

Using a Filter

By default, all global variables are visible in the editor. You can provide a filter in the **Filter** `string/pattern` field to limit the display. The filter uses regular expression matching. That is, if you enter `Rv` in the field, only variables that have `Rv` in their name are displayed. Matching is case insensitive. Entering a new expression and clicking **Filter** again performs a new search on all variables and will not refine the current search.

To clear the filter expression and display all variables either click **Show all** or remove the expression from the entry box and click **Filter** again.

Global Variable Attributes

To add or edit a name, value, constraint or description attribute, triple-click in the attribute field. The type attribute has a drop down menu that displays choices. Click in the type field to display the menu.

- **Name** — Provide a name for the variable.
- **Value** — Provide a value for the variable, depending on the type you select.
- **Deployment** — Select the deployment check box to make the variable visible and settable when deploying using TIBCO Administrator. If the check box is clear, the variable is not visible in TIBCO Administrator.
- **Service** — Indicates that the variable should be included when the `Include all service level global variables` option is selected when building the enterprise archive file. A variable that is settable on a per-service basis can be set for each adapter service or TIBCO BusinessWorks top-level process. Service level variables are only included in the service archive if the service uses them. This option is used for TIBCO adapter archives. TIBCO BusinessWorks includes all process-level variables regardless of usage.
- **Type** — Click in the field to select the variable type, `String`, `Integer`, `Boolean`, or `Password`. If `Password` is selected, the value you provide is obfuscated in the repository.
- **Constraint** — For `String` and `Integer` types, provide a range of allowed values. The constraint field for `Strings` is an enumeration, for example, `one , two ,`

three. The constraint field for Integers is for a range, for example, 1-100.

Note that constraints are currently not implemented in TIBCO Administrator.

- Description — Provide a description of the variable.

Deleting a Global Variable

Delete a global variable by selecting it and clicking the delete variable icon. You cannot delete a global variable that is in use. To find where global variables are used, click **Tools > Find Global Variable Usages**.

Deleting Projects

You do not delete projects from the TIBCO Designer main window but from the startup panel. See [Startup Options on page 77](#).

To Access the Startup Panel

- If TIBCO Designer isn't running, start it.
- If TIBCO Designer is running, close all windows. The startup panel is displayed unless `Show this panel only on startup` has been checked.

To Delete a Project

1. In the startup panel, make sure the Project tab is selected.
2. Click **Delete Project**.
3. In the panel that appears:
 - a. Specify the project directory
 - b. Specify a version control system if the project was used in conjunction with a version control system.
 - For File Sharing, any user with access to a project can delete the project.
 - For other version control systems you must make sure that both the (local) project directory and the directory you specify for the version control system are correct. You must also be sure to specify a user that has appropriate privileges for deleting the project.



You cannot delete projects based on XML Canon from TIBCO Designer. To delete such a project, you must use a WebDAV client.

Tips and Tricks for Working With Projects

This section contains additional information on using multi-file projects.

- **Use ASCII project names.** You must use an ASCII name for the project when saving the project from TIBCO Designer. Project names must not use the following characters: | / \ " ' : ?.
- **Avoid naming collision.** You cannot place a multi-file project and a single-file (.dat) project into the same directory.
- **Place schema in the AESchemas folder.** If you edit your project file in an XML editor and define schema outside the /AESchemas folder, the schema are placed in a directory called `__NON__DEFAULT__SCHEMA__FOLDER__` in `/tibco/public/<type>` where type is the kind of object (that is, class, scalar, union, and so forth).

It is cleaner to have schema files under /AESchemas. In addition, it is required you place schema files into /AESchemas if you want to edit your project using TIBCO Designer.

Note that while editing schema files is not prohibited, you do so at your own risk.

- **Consider using global variable groups.** Use global variable groups to allow multiple developers to work on global variables simultaneously. Each group has its own file in the multi-file project. See [Working With Global Variables on page 107](#) for more information.

Note, however, that an excessive amount of global variables (over 500) can lead to problems.

Index

A

account concept [17](#)
 ActiveMatrix BusinessWorks process for testing [34](#)
 add a jdbc connection resource [67](#)
 Adding
 a Resource to Your Project [105](#)
 Global Variable or Global Variable Group [108](#)
 Resources to a Project [105](#)
 applications
 testing (tutorial example) [39, 60](#)
 applydebit rule [25](#)

B

backing store properties [70](#)
 BE_HOME [xiii](#)

C

cache cluster properties [51](#)
 cache server nodes [54](#)
 caching overview [42](#)
 channels
 configuring (tutorial example) [11](#)
 check for existence of accounts before creating [49](#)
 checknegativebalance rule [30](#)
 Choosing
 Palette Mode or Non-palette Mode [90](#)
 Panel Layout [93](#)
 Three-panel View [94](#)
 Configuration Panel [86](#)
 configure and build the archive [31](#)
 create a concept instance using a standard function [23](#)
 create a JMSchannel and destination [44](#)

create a Rendezvous channel and a destination [11](#)
 create JMS connection [44](#)
 Creating
 a User Palette [91](#)
 Projects [100](#)
 customer support [xvi](#)
 Customizing the Display [93](#)

D

debit event [13](#)
 delete or disable the Rendezvous channel and its
 destination [47](#)
 Deleting
 a Project [111](#)
 a User Palette [92](#)
 Global Variable [110](#)
 Projects [111](#)
 deploy
 and test the application [72](#)
 the agents and cache servers [58](#)
 the project [38](#)
 Design Panel [85](#)
 destinations
 configuring (tutorial example) [11](#)
 Displaying Palettes in a Separate Window [91](#)

E

examples [2](#)

F

fraud detection runtime flow [6](#)
 fraudcriteria scorecard [19](#)
 frauddetection project [8](#)
 frauddetection rule [28](#)

G

Global Variables
 Attributes [109](#)
 Display [84](#)
 Editor [107](#)

H

Hiding Palettes [89](#)

I

inference agents [55](#)
 initializeaccount rule function [20](#)

J

JMS server required [43](#)

L

Loading a User Palette [92](#)

M

multi-engine overview [42](#)

O

Opening Projects [104](#)
 Overview of Projects [98](#)

P

Palette Panel [85](#)
 prepare the database [64](#)
 processdebits rule set [25](#)
 Project
 Display [83](#)
 Panel [83](#)
 Structure [98](#)
 Templates [99](#)

R

Reopening Projects [104](#)
 reset the backing store tutorial [73](#)
 Resources [84](#)

S

Saving
 a Project [103](#)
 a Project as a Template [103](#)
 Projects [102](#)
 set a default event for a destination [15](#)
 Showing Palettes [89](#)
 simple events
 configuring (tutorial example) [13](#)
 start the JMS server [44](#)

Starting TIBCO Designer 77
 Startup Options 77
 support, contacting xvi
 Switching Palette Modes 91

T

technical support xvi
 test the application 39
 testing applications 39, 60
 TIBCO Designer
 accessing documentation 95
 accessing startup panel 111
 TIBCO_HOME xiii
 Tips and Tricks
 for Working With Projects 112
 troubleshooting 59
 tutorial scenario 3

U

Using
 a Filter 109
 a Version Control System 99
 the Palette Browser 88

V

validate the project 15, 31
 Validating
 a Resource 101
 All Resources 101
 Projects 101

W

Working With
 Global Variables 107
 Palettes 87
 User Palettes 91