# TIBCO ActiveMatrix BusinessWorks™ Plug-in for EJB User's Guide

*Software Release 6.1.1*
*April 2020*

## Important Information

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. Please 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.

# Contents

**TIBCO Documentation and Support Services . . . . . 5**

**Product Overview . . . . . 6**

EJB Overview . . . . . 7

**Getting Started . . . . . 8**

Creating a Project . . . . . 8

Connecting to an EJB Server . . . . . 9

Connecting to Apache OpenEJB Standalone Server . . . . . 10

Connecting to JBoss/WildFly Application Server . . . . . 10

Connecting to Oracle WebLogic Server . . . . . 11

Connecting to IBM WebSphere Application Server . . . . . 11

Creating an EJB Configuration Shared Resource . . . . . 13

Configuring EJB Client JAR Files . . . . . 14

Configuring a Process . . . . . 14

Importing CORBA-Related Classes . . . . . 15

Testing a Process . . . . . 15

Deploying an Application . . . . . 16

TIBCO Business Studio Overview . . . . . 17

**EJB Configuration Shared Resource . . . . . 18**

**EJB Palette . . . . . 21**

EJB2Home . . . . . 21

EJB2Remote . . . . . 24

EJB3Remote . . . . . 26

**Working with the Sample Project . . . . . 29**

Importing the Sample Project . . . . . 29

Running the Sample Project . . . . . 30

Configurations for basic_use_ejb2 . . . . . 31

Configurations for basic_use_ejb3 . . . . . 31

Configurations for ejb_with_java_invoke . . . . . 32

Configurations for ejb3_with_JavaToXml . . . . . 33

Configurations for remote_object_release . . . . . 33

Configurations for stateless_remote_cache . . . . . 34

**Managing Logs . . . . . 36**

Log Levels . . . . . 36

Setting Up Log Levels . . . . . 36

Exporting Logs to a File . . . . . 37

**Error Codes . . . . . 38**

# TIBCO Documentation and Support Services

### How to Access TIBCO Documentation

Documentation for TIBCO products is available on the TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product. To access the latest documentation, visit https://docs.tibco.com.

### Product-Specific Documentation

Documentation for TIBCO ActiveMatrix BusinessWorks™ Plug-in for EJB is available on the TIBCO ActiveMatrix BusinessWorks™ Plug-in for EJB Product Documentation page.

The following documents for this product can be found on the TIBCO Documentation site:

- *TIBCO ActiveMatrix BusinessWorks Plug-in for EJB Installation*
- *TIBCO ActiveMatrix BusinessWorks Plug-in for EJB User's Guide*
- *TIBCO ActiveMatrix BusinessWorks Plug-in for EJB Release Notes*

### How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

- For an overview of TIBCO Support, visit http://www.tibco.com/services/support.
- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support portal at https://support.tibco.com.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to https://support.tibco.com. If you do not have a user name, you can request one by clicking Register on the website.

### How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the TIBCO Ideas Portal. For a free registration, go to https://community.tibco.com.

# Product Overview

You can use TIBCO ActiveMatrix BusinessWorks™ Plug-in for EJB to connect to J2EE-compliant application servers and invoke Enterprise JavaBeans (EJB) components, or enterprise beans on the servers.

TIBCO ActiveMatrix BusinessWorks™ is a leading integration platform that can integrate a wide variety of technologies and systems within enterprise and on cloud. TIBCO ActiveMatrix BusinessWorks includes an Eclipse-based graphical user interface (GUI) provided by TIBCO Business Studio™ for design, testing, and deployment. If you are not familiar with TIBCO ActiveMatrix BusinessWorks, see the TIBCO ActiveMatrix BusinessWorks documentation for more details.

TIBCO ActiveMatrix BusinessWorks Plug-in for EJB plugs into TIBCO ActiveMatrix BusinessWorks, which connects TIBCO ActiveMatrix BusinessWorks with EJB containers.

TIBCO ActiveMatrix BusinessWorks supports plug-ins to extend the palette functionality. After installing the plug-in, an EJB Configuration shared resource and an EJB palette become available in the TIBCO Business Studio. You can add the plug-in activities to the BusinessWorks process you are designing, and integrate them with the BusinessWorks process.

At run time, the plug-in activities are performed as part of the BusinessWorks process execution. Each plug-in consists of activities which share common functionality and properties.

The following three activities are included in the EJB palette:

- EJB2Home activity: you can use this activity to retrieve EJB home object and create EJB remote object for EJB 2.x.
- EJB2Remote activity: you can use this activity to invoke the EJB remote method which is deployed on the EJB server for EJB 2.x.
- EJB3Remote activity: you can use this activity to get EJB remote object and invoke the remote method which is deployed on the EJB server for EJB 3.x.

### Integrating with JMS

To handle the message-driven beans, you must integrate the EJB palette of this plug-in with the JMS palette of TIBCO ActiveMatrix BusinessWorks. When you design the BusinessWorks processes with this plug-in, only session beans and entity beans are supported.

### Sending or Receiving Java Objects

Occasionally input parameters or return values for enterprise beans are Java objects. To send or receive a Java object to or from an EJB, use the Java palette to create the object within the process definition. For more information on how to work with Java objects in process definitions, see *TIBCO ActiveMatrix BusinessWorks Bindings and Palettes Reference*.

Examples:

- To call a remote method that requires a Java object as an input parameter, create an object with the Java Invoke activity.
- To use a Java object received as a return value from an EJB, parse an object to the Java Invoke activity.

# EJB Overview

An EJB component is a software component that encapsulates the business logic of an application.

EJB technology is the server-side component architecture for Java Platform, Enterprise Edition (Java EE). EJB technology enables rapid and simplified development of distributed, transactional, secure, and portable applications based on Java technology.

You can use EJB Palette of TIBCO ActiveMatrix BusinessWorks Plug-in for EJB to connect to J2EE-compliant application servers and invoke EJB components or enterprise beans on the servers.

The EJB specification includes support for Java Transaction API (JTA) UserTransactions. TIBCO ActiveMatrix BusinessWorks provides support for these transactions and you can call enterprise beans within a client-managed transaction. See *TIBCO ActiveMatrix BusinessWorks Application Development* for more information about transactions.

## Accessing an EJB

The EJB standard defines mechanisms for a client machine to access an EJB 2.x or an EJB 3.x entity:

- To access an EJB 2.x entity, you have to obtain a reference to a JNDI server, and perform a JNDI lookup operation to obtain a reference to a home object and also a reference to a remote object from the home object. Then you can invoke methods on the remote object.
- To access an EJB 3.x entity, you have to obtain a reference to a JNDI server, and perform a JNDI lookup operation to obtain a reference to a remote object. Then you can invoke methods on the remote object.

# Getting Started

This tutorial is designed for beginners who want to use TIBCO ActiveMatrix BusinessWorks Plug-in for EJB in TIBCO Business Studio.

### Prerequisites

Ensure your EJB application server is running and EJBs are deployed on the EJB application server before using the plug-in.

All the operations are performed in TIBCO Business Studio. See TIBCO Business Studio Overview to get familiar with TIBCO Business Studio.

A basic procedure of using TIBCO ActiveMatrix BusinessWorks Plug-in for EJB includes the following steps:

1. Creating a Project
2. Connecting to an EJB Server
3. Creating an EJB Configuration Shared Resource
4. Configuring EJB Client JAR Files
5. Configuring a Process
6. Testing a Process
7. Deploying an Application

## Creating a Project

The first task of using the plug-in is creating a project. After creating a project, you can add resources and processes.

An Eclipse project is an application module configured for TIBCO ActiveMatrix BusinessWorks. An application module is the smallest unit of resources that is named, versioned, and packaged as part of an application.

### Procedure

1. Start TIBCO Business Studio by using one of the following ways:
   - Microsoft Windows: click **Start** > **All Programs** > **TIBCO** > ***TIBCO_HOME*** > **TIBCO Business Studio *version_number*** > **Studio for Designers**.
   - Mac OS and Linux: run the TIBCO Business Studio executable file located in the `TIBCO_HOME/studio/version_number/eclipse` directory.
2. From the menu, click **File** > **New** > **BusinessWorks Resources** to open the BusinessWorks Resource wizard.
3. In the "Select a wizard" dialog, click **BusinessWorks Application Module** and click **Next** to open the New BusinessWorks Application Module wizard.
4. In the Project dialog, configure the project that you want to create:
   a) In the **Project name** field, enter a project name.
   b) By default, the created project is located in the workspace current in use. If you do not want to use the default location for the project, clear the **Use default location** check box and click **Browse** to select a new location.
   c) Use the default version of the application module, or enter a new version in the **Version** field.
   d) Keep the **Create empty process** and **Create Application** check boxes selected to automatically create an empty process and an application when creating the project.

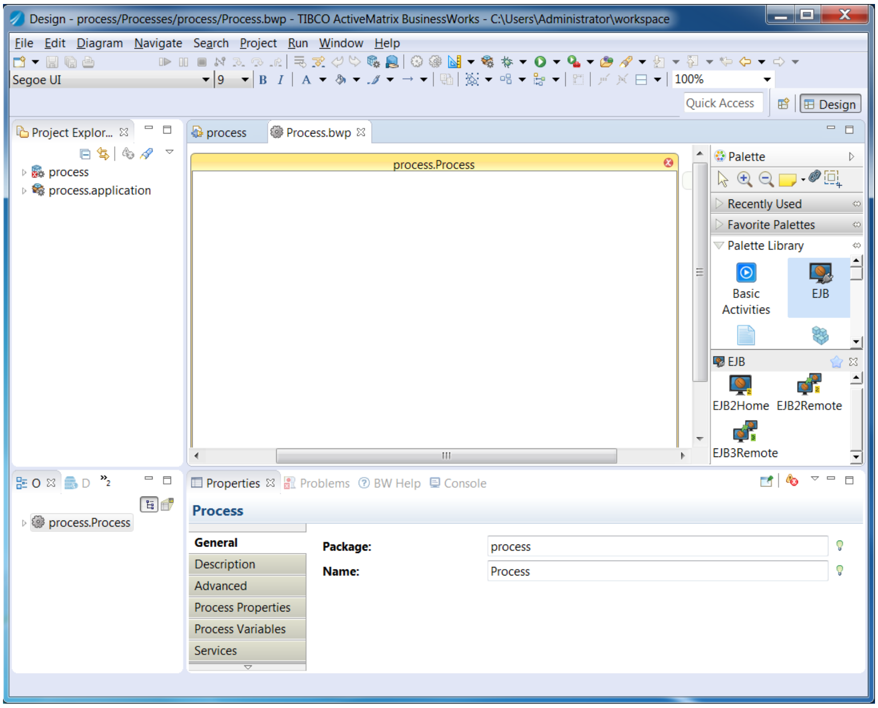
e) Select the **Use Java configuration** check box if you want to create a Java module.

A Java module provides the Java tooling capabilities.

f) Click **Finish** to create the project.

### Result

The project with the specified settings is displayed in the Project Explorer view.



Design - process/Processes/process/Process.bwp - TIBCO ActiveMatrix BusinessWorks - C:\Users\Administrator\workspace

Package: process

Name: Process

## Connecting to an EJB Server

You have to connect to an EJB server before invoking any EJB by using TIBCO ActiveMatrix BusinessWorks Plug-in for EJB.

TIBCO ActiveMatrix BusinessWorks Plug-in for EJB supports the following application server vendors:

- Apache OpenEJB Standalone Server
- JBoss Application Server
- JBoss Enterprise Application Server and WildFly Server
- Oracle WebLogic Server
- IBM WebSphere Application Server

For the list of supported server versions, see the Readme file.

Depending on your application server vendor, perform the following tasks to connect to the EJB server:

- Connecting to Apache OpenEJB Standalone Server
- Connecting to JBoss/WildFly Application Server
- Connecting to Oracle WebLogic Server
- Connecting to IBM WebSphere Application Server

## Connecting to Apache OpenEJB Standalone Server

After creating a project, you have to create a connection between the plug-in and an EJB server.

### Prerequisites

Ensure that you have created a project, as described in Creating a Project.

### Procedure

1. Copy all `.jar` files into a local directory.

   The `.jar` files are located in the *`OPENEJB_HOME`*`/lib` directory, where *`OPENEJB_HOME`* is the directory that your Apache OpenEJB Standalone Server is installed into.

   For more information about the server JAR files, see related documents from your EJB server vendor.
2. Expand the created project in the Project Explorer view, and drag the local directory to the **lib** folder.
3. In the File and Folder Operation dialog, click **Link to files and folders** and then click **OK**.

> If you want to deploy the application to TIBCO Enterprise Administrator, click **Copy files and folders**.

## Connecting to JBoss/WildFly Application Server

After creating a project, you have to create a connection between the plug-in and an EJB server.

### Prerequisites

Ensure that you have created a project, as described in Creating a Project.

### Procedure

1. Copy all `.jar` files into a local directory.

   The `.jar` files are located in the *`JBoss_Home`*`/bin/client` directory, where *`JBoss_Home`* is the directory that your JBoss Enterprise Application Server and WildFly Server are installed into.

   For more information about the server JAR files, see related documents from your EJB server vendor.
2. In this local directory, create `.properties` files and add contents to each `.properties` file. Create `.properties` files based on the version of your JBoss Application Server.

   Create a `.properties` file with the file name `jndi.properties` and add the following contents to the file:

   ```
   jboss.naming.client.ejb.context=true
   java.naming.security.principal=remote://<JBoss7_IP_Address>:<Port>
   java.naming.factory.initial=org.jboss.naming.remote.client.
   InitialContextFactory
   java.naming.security.principal=<username>
   java.naming.security.credentials=<password>
   ```

   Create another `.properties` file with the file name `jboss-ejb-client.properties` and add the following contents to the file:

   ```
   remote.connectionprovider.create.options.org.xnio.Options.SS
   L_ENABLED=false
   remote.connections=default
   remote.connection.default.host=<JBoss7_IP_Address>
   remote.connection.default.port=<Port>
   remote.connection.default.connect.options.org.xnio.Options.S
   ```

```
ASL_POLICY_NOANONYMOUS=false
remote.connection.default.username=<username>
remote.connection.default.password=<password>
```

3. Expand the created project in the Project Explorer view, and drag the local directory to the **lib** folder.
4. In the File and Folder Operation dialog, click **Link to files and folders** and click **OK**.

   > If you want to deploy the application to TIBCO® Enterprise Administrator, click **Copy files and folders**.

5. From the menu, click **Window** > **Open Perspective** > **Other** to open the Open Perspective dialog, click **Java**, and then click **OK**.
6. In the Package Explorer view, expand **META-INF**, right-click **MANIFEST.MF**; then click **Open With** > **Plug-in Manifest Editor**.
7. In the **Runtime** tab, click **Add** in the Classpath panel.
8. In the Jar Selection window, click **lib** > **the local directory name** and press Enter.

## Connecting to Oracle WebLogic Server

After creating a project, you have to create a connection between the plug-in and an EJB server.

### Prerequisites

Ensure that you have created a project, as described in Creating a Project.

### Procedure

1. Copy all `.jar` files into a local directory.

   The `.jar` files are located in the *`WebLogic_Home`*`/server/lib` directory, where *`WebLogic_Home`* is the directory that your Oracle WebLogic Server is installed into.

   For more information about the server JAR files, see related documents from your EJB server vendor.
2. Expand the created project in the Project Explorer view, and drag the local directory to the **lib** folder.
3. In the File and Folder Operation dialog, click **Link to files and folders** and then click **OK**.

   > If you want to deploy the application to TIBCO Enterprise Administrator, click **Copy files and folders**.

## Connecting to IBM WebSphere Application Server

After creating a project, you have to create a connection between the plug-in and an EJB server.

### Prerequisites

Ensure that you have created a project, as described in Creating a Project.

### Procedure

1. Copy all `.jar` files into a local directory.

   The `.jar` files are located in the *`WebSphere_Home`*`/AppServer/runtimes` directory, where *`WebSphere_Home`* is the directory that your IBM WebSphere Application Server is installed into.

   For more information about the server JAR files, see related documents from your EJB server vendor.

2. Copy the following files from IBM WebSphere Application Server to the local directory:

    - *WebSphere_Home*/AppServer/profiles/AppSrv01/properties/ssl.client.props
    - *WebSphere_Home*/AppServer/profiles/AppSrv01/properties/sas.client.props

3. On a command line, navigate to the *TIBCO_HOME*/tibcojre/*version_number*/bin directory, and then type the following command: keytool.exe.

    For example:

    ```
    keytool.exe -genkey -v alias test -keystore D:/key.jks
    -storepass password
    ```

    The keystore files, key.jks and trust.jks, are created after this step.

4. Modify the ssl.client.props file to customize your environment.
    You can obtain the file from the WebSphere Application Server installation.

    ```
    com.ibm.ssl.protocol=SSL
    com.ibm.ssl.trustManager=SunX509
    com.ibm.ssl.keyManager=SunX509
    com.ibm.ssl.contextProvider=SunJSSE
    com.ibm.ssl.keyStoreType=JKS
    com.ibm.ssl.keyStoreProvider=SUN
    com.ibm.ssl.keyStore=/home/user1/etc/key.jks
    com.ibm.ssl.trustStoreType=JKS
    com.ibm.ssl.trustStoreProvider=SUN
    com.ibm.ssl.trustStore=/home/user1/etc/trust.jks
    ```

    For more information, see http://www-01.ibm.com/support/knowledgecenter/SS7JFU_8.0.0/com.ibm.websphere.express.doc/info/exp/ae/tcli_ejbthinclient.html.

5. Create a .properties file with the file name jndi.properties in the local directory and add the following JVM parameters:

    - com.ibm.SSL.ConfigURL: references a file URL that points to the ssl.client.props file.
    - com.ibm.CORBA.ConfigURL: references a file URL that points to the sas.client.props file.
    - com.ibm.CORBA.Debug.Output: assigns a value of NUL.

    For example:

    ```
    -com.ibm.SSL.ConfigURL="file:///home/user1/ssl.client.props"
    -com.ibm.CORBA.ConfigURL="file:///home/user1/sas.client.props"
    -com.ibm.CORBA.Debug.Output=NUL
    ```

6. Expand the created project in the Project Explorer view, and drag the local directory to the **lib** folder.

7. Click **Link to files and folders** in the prompted **File and Folder Operation** dialog and press Enter.

    If you want to deploy the application to TIBCO Enterprise Administrator, click **Copy files and folders**.

8. Click **Run** > **Run Configurations**. In the Run Configurations dialog, click the **Arguments** tab and add the following JVM parameters in the **VM arguments**.

    - Dcom.ibm.SSL.ConfigURL: references a file URL that points to the ssl.client.props file.
    - Dcom.ibm.CORBA.ConfigURL: references a file URL that points to the sas.client.props file.
    - Dcom.ibm.CORBA.Debug.Output: assigns a value of NUL.

    For example:

    ```
    -Dcom.ibm.SSL.ConfigURL="file:///home/user1/ssl.client.props"
    -Dcom.ibm.CORBA.ConfigURL="file:///home/user1/sas.client.props"
    -Dcom.ibm.CORBA.Debug.Output=NUL
    ```

If you want to deploy the application to TIBCO Enterprise Administrator, navigate to the *TIBCO_HOME*/bw/6.3/bin directory, and add the following content to the bwcommon.tra file:

```
java.extended.properties=-Dcom.ibm.SSL.ConfigURL="file:///home/user1/
ssl.client.props" -Dcom.ibm.CORBA.ConfigURL="file:///home/user1/
sas.client.props" -Dcom.ibm.CORBA.Debug.Output=NUL
```

# Creating an EJB Configuration Shared Resource

An EJB Configuration shared resource is necessary to specify the configuration for the JNDI server before running any EJB activity.

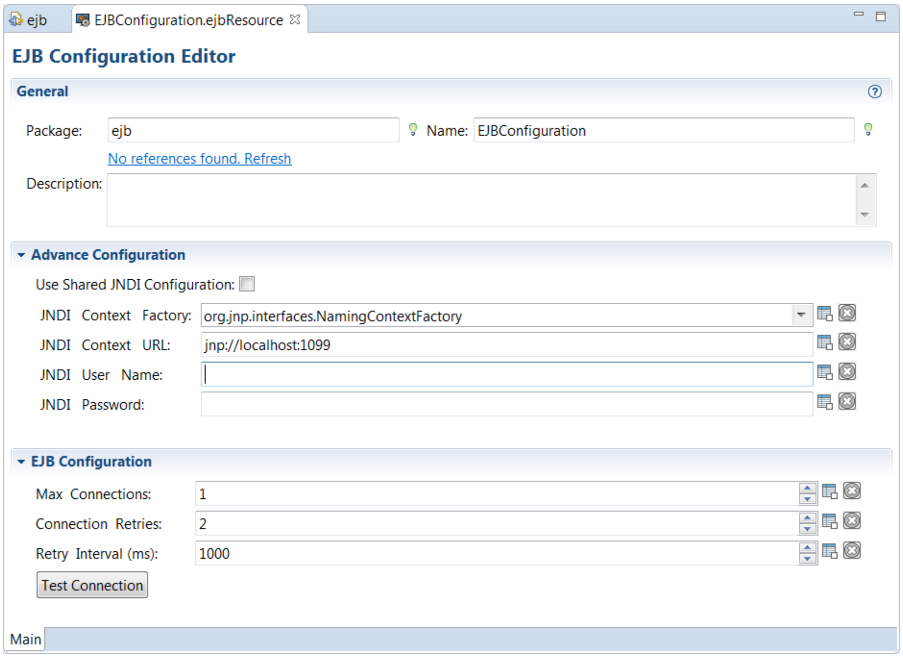
## Prerequisites

The EJB Configuration shared resource is available at the **Resources** level. Ensure that you have created a project, as described in Creating a Project.

Ensure that you have already connected to an EJB server, as described in Connecting to an EJB Server.

## Procedure

1. Expand the created project in the Project Explorer view.
2. Right-click the **Resources** folder and click **New** > **EJB Configuration** .
3. In the EJB Configuration wizard, the resource folder, package name, and resource name of the EJB configuration are provided by default. If you do not want to use the default configurations, change them accordingly. Click **Finish** to open the EJB Configuration Editor panel.
4. Configure the EJB Configuration shared resource in the displayed editor.



EJB Configuration Editor

General

Package: ejb
Name: EJBConfiguration
No references found. Refresh
Description:

Advance Configuration

Use Shared JNDI Configuration: □
JNDI Context Factory: org.jnp.interfaces.NamingContextFactory
JNDI Context URL: jnp://localhost:1099
JNDI User Name:
JNDI Password:

EJB Configuration

Max Connections: 1
Connection Retries: 2
Retry Interval (ms): 1000

5. Click **Test Connection** to verify the configuration, as described in EJB Configuration Shared Resource.

## Configuring EJB Client JAR Files

The EJB client JAR files are necessary for the plug-in to be used as the client view of EJB.

### Prerequisites

Ensure that you have created a project, as described in Creating a Project.

Ensure that you have connected to an EJB server, as described in Connecting to an EJB Server.

### Procedure

1. Copy the EJB client JAR files to a local directory.
2. Expand the created project in the Project Explorer view, and drag the local directory to the **lib** folder.
3. Click **Link to files and folders** in the File and Folder Operation dialog and press Enter.

   If you want to deploy the application to TIBCO Enterprise Administrator, click **Copy files and folders**.

## Configuring a Process

After creating a project, an empty process is created. You can add activities to the empty process to complete a task. For example, create an EJB 2.x remote object.

When you configure a process, you might have to import the CORBA-related classes into the **MANIFEST.MF** file of the project. For details, see Importing CORBA-Related Classes.

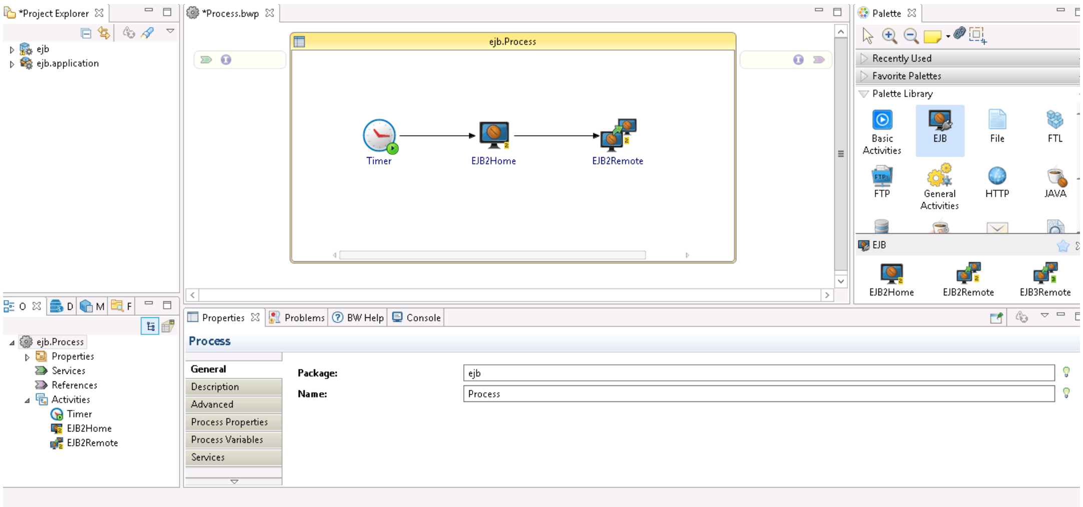
### Prerequisites

Ensure that you have finished the following tasks before configuring a process:

- Creating a Project
- Connecting to an EJB Server
- Creating an EJB Configuration Shared Resource
- Configuring EJB Client JAR Files

### Procedure

1. In the Project Explorer view, click the created project and open the empty process from the **Processes** folder.
2. Select activities from the Palette view and drop them in the Process editor.
   For example, select and drop the Timer activity from the General Activities palette and the EJB2Home and EJB2Remote activities from the EJB palette.



Package: ejb

Name: Process

3. Drag the icon to create a transition between the added activities.
4. Configure the added activities, as described in EJB Palette.

> An EJB Configuration shared resource is required when you configure the EJB activities. See Creating an EJB Configuration Shared Resource for more details on how to create the EJB Configuration shared resource.

5. Click **File** > **Save** to save the project.

## Importing CORBA-Related Classes

When you configure a process to invoke the Container Managed Persistence (CMP) entity bean of EJB 2.0 on IBM WebSphere Application Server, if you add the EJB client JAR files to the **lib** folder of a project, the class space of the project might be inconsistent with the added EJB client JAR files.

To prevent this inconsistency, you must import the CORBA-related classes to the **MANIFEST.MF** file of the project.

### Procedure

1. From the menu, click **Window** > **Open Perspective** > **Other** to open the Open Perspective dialog, click **Java**, and then click **OK**.
2. In the Package Explorer view, expand **META-INF**, right-click **MANIFEST.MF**; then click **Open With** > **Plug-in Manifest Editor**.
3. Click the **Dependencies** tab on the right panel and click **Add** in the Imported Packages pane to add the CORBA-related classes.

## Testing a Process

After configuring a process, you can test the process to check whether the process completes your task.

### Prerequisites

Ensure that you have configured a process, as described in Configuring a Process.

### Procedure

1. On the toolbar, click **Debug** > **Debug Configurations**.

2. Click **BusinessWorks Application** > **BWApplication** in the left panel.

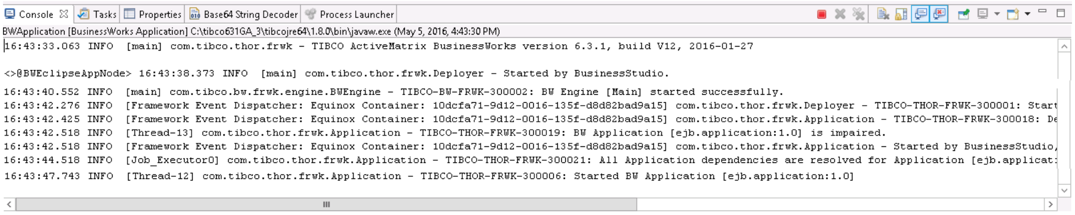
   By default, all the applications in the current workspace are selected in the **Applications** tab. Ensure that only the application you want to debug is selected in the **Applications** tab in the right panel.

3. Click the **Advanced** tab and click **Browse** to locate the `logback` file.

   By default, the log file is located in the `*TIBCO_HOME*/bw/*version_number*/config/design/logback` directory and error logs are captured.

   See Managing Logs for more details.

4. Click **Debug** to test the process in the selected application.
   TIBCO Business Studio changes to the Debug perspective. The debug information is displayed in the Console view.

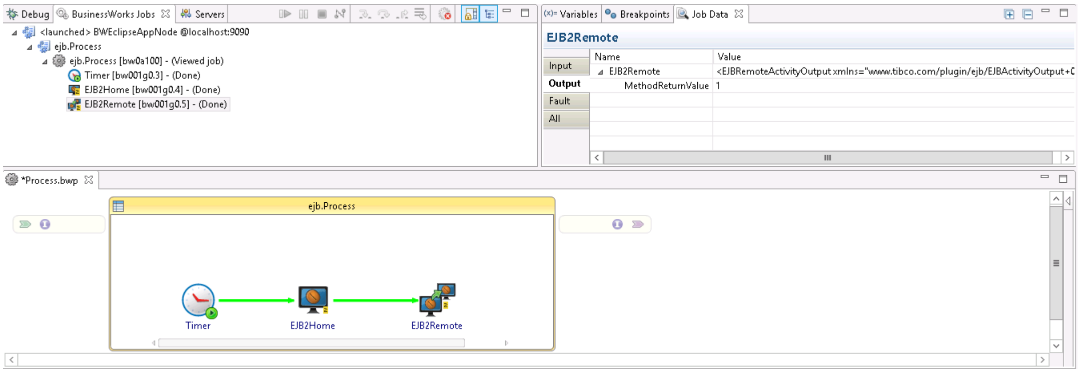


```
BWApplication [BusinessWorks Application] C:\tibco631GA_3\tibcojre64\1.8.0\bin\javaw.exe (May 5, 2016, 4:43:30 PM)
16:43:33.063 INFO  [main] com.tibco.thor.frwk - TIBCO ActiveMatrix BusinessWorks version 6.3.1, build V12, 2016-01-27

<>@BWEclipseAppNode> 16:43:38.373 INFO  [main] com.tibco.thor.frwk.Deployer - Started by BusinessStudio.
16:43:40.552 INFO  [main] com.tibco.bw.frwk.engine.BWEngine - TIBCO-BW-FRWK-300002: BW Engine [Main] started successfully.
16:43:42.276 INFO  [Framework Event Dispatcher: Equinox Container: 10dcfa71-9d12-0016-135f-d8d82bad9a15] com.tibco.thor.frwk.Deployer - TIBCO-THOR-FRWK-300001: Start
16:43:42.425 INFO  [Framework Event Dispatcher: Equinox Container: 10dcfa71-9d12-0016-135f-d8d82bad9a15] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300018: De
16:43:42.518 INFO  [Thread-13] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300019: BW Application [ejb.application:1.0] is impaired.
16:43:42.518 INFO  [Framework Event Dispatcher: Equinox Container: 10dcfa71-9d12-0016-135f-d8d82bad9a15] com.tibco.thor.frwk.Application - Started by BusinessStudio,
16:43:44.518 INFO  [Job_Executor0] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300021: All Application dependencies are resolved for Application [ejb.applicat:
16:43:47.743 INFO  [Thread-12] com.tibco.thor.frwk.Application - TIBCO-THOR-FRWK-300006: Started BW Application [ejb.application:1.0]
```

5. In the **Debug** tab, expand the running process and click an activity.

6. In the upper-right corner, click the **Job Data** tab, and then click the **Output** tab to check the activity output.



```
<launched> BWEclipseAppNode @localhost:9090
  ejb.Process
    ejb.Process [bw0a100] - (Viewed job)
      Timer [bw001g0.3] - (Done)
      EJB2Home [bw001g0.4] - (Done)
      EJB2Remote [bw001g0.5] - (Done)
```

EJB2Remote

Input | Output | Fault | All

| Name | Value |
|---|---|
| EJB2Remote | <EJBRemoteActivityOutput xmlns="www.tibco.com/plugin/ejb/EJBActivityOutput+C |
| MethodReturnValue | 1 |

## Deploying an Application

After testing, if the configured process works as expected, you can deploy the application that contains the configured process into a runtime environment, and then use the **`bwadmin`** utility to manage the deployed application.

Before deploying an application, you must generate an application archive, which is an enterprise archive (EAR) file that is created in TIBCO Business Studio.

Deploying an application involves the following tasks:

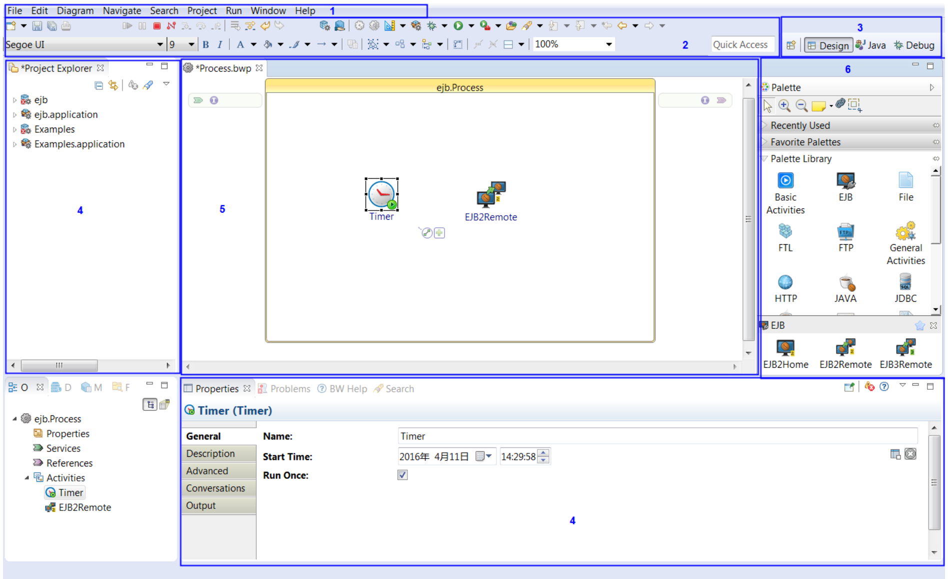
1. Uploading an application archive
2. Deploying an application archive
3. Starting an application

See *TIBCO ActiveMatrix BusinessWorks Administration* for more details on how to deploy an application.

# TIBCO Business Studio Overview

TIBCO Business Studio is an Eclipse-based integration development environment that is used to design, develop, and test ActiveMatrix BusinessWorks applications.

TIBCO Business Studio provides a workbench in which you can create, manage, and navigate resources in your workspace. A *workspace* is the central location on your machine where all data files are stored.



Name: Timer

Start Time: 2016年 4月11日 14:29:58

Run Once: ☑

The workbench consists of the following elements:

1. **Menu**: contains menu items such as File, Edit, Diagram, Navigate, Search, Project, Run, Window, and Help.

2. **Toolbar**: contains buttons for frequently used commands such as New, Save, Enable/Disable Business Studio Capabilities, Create a new BusinessWorks Application Module, Create a new BusinessWorks Shared Module, Debug, Run, and so on.

3. **Perspective**: contains an initial set and layout of views that are required to perform a certain task. TIBCO Business Studio launches the Modeling perspective by default. You can change the perspective from the menu **Window** > **Open Perspective** > ***Perspective_Name***.

4. **View**: displays resources. For example, the Project Explorer view displays the ActiveMatrix BusinessWorks applications, modules, and other resources in your workspace, and the Properties view displays the properties for the selected resource. You can open a view from the menu **Window** > **Show View** > ***View_Name***.

5. **Editor**: provides a canvas to configure, edit, or browse a resource. Double-click a resource in a view to open the appropriate editor for the selected resource. For example, double-click an ActiveMatrix BusinessWorks process (`MortgageAppConsumer.bwp`) in the Project Explorer view to open the process in the editor.

6. **Palette**: contains a set of widgets and a palette library. A *palette* groups activities that perform similar tasks, and provides quick access to activities when configuring a process.

# EJB Configuration Shared Resource

You can use an EJB Configuration shared resource to specify the connection configuration for a JNDI server.

### General

In the General panel, you can specify the package that stores the EJB Configuration shared resource and the shared resource name.

The following table lists the configurations in the General panel of the EJB Configuration shared resource:

| Field | Module Property? | Description |
|---|---|---|
| **Package** | No | The name of the package where the new shared resource is added. |
| **Name** | No | The name to be displayed as the label for the shared resource in the process. |
| **Description** | No | A short description for this shared resource. |

### Advance Configuration

The following table lists the configurations in the Advance Configuration panel of the EJB Configuration shared resource:

| Field | Module Property? | Description |
|---|---|---|
| **Use Shared JNDI Configuration** | No | If this check box is selected, the **JNDI Configuration** field is displayed. You can use it to choose a JNDI Configuration resource.<br>If this check box is cleared, the **JNDI Context Factory**, **JNDI Context URL**, **JNDI User Name**, and **JNDI Password** fields are displayed. |
| **JNDI Configuration** | No | Specifies a JNDI Configuration resource.<br>This field is displayed only when the **Use Shared JNDI Configuration** check box is selected. |

| Field | Module Property? | Description |
|---|---|---|
| **JNDI Context Factory** | Yes | Select an initial context factory supplied by each supported application server vendor from the list. Different initial context factories other than the default options are available in this list:<br>- **org.apache.openejb.client.RemoteInitialContextFactory** Apache OpenEJB Standalone Server<br>- **org.jboss.naming.remote.client.InitialContextFactory** JBoss Enterprise Application Server and WildFly Server<br>- **weblogic.jndi.WLInitialContextFactory** Oracle WebLogic Server<br>- **com.ibm.websphere.naming.WsnInitialContextFactory** IBM WebSphere Application Server<br>- **Others (Unsupported)** For other application servers<br>This field is displayed only when the **Use Shared JNDI Configuration** check box is cleared. |
| **JNDI Context URL** | Yes | Specifies the URL of the server.<br>See related JNDI provider documentation for the syntax of the URL.<br>This field is displayed only when the **Use Shared JNDI Configuration** check box is cleared. |
| **JNDI User Name** | Yes | Specifies the user name to be used when logging in to the JNDI server. If the JNDI provider does not require access control, this field can be empty.<br>This field is displayed only when the **Use Shared JNDI Configuration** check box is cleared. |
| **JNDI Password** | Yes | Specifies the password to be used when logging in to the JNDI server. If the JNDI provider does not require access control, this field can be empty.<br>This field is displayed only when the **Use Shared JNDI Configuration** check box is cleared. |

### EJB Configuration

The following table lists the configurations in the EJB Configuration panel of the EJB Configuration shared resource:

| Field | Module Property? | Description |
|---|---|---|
| **Max Connections** | Yes | The maximum number of naming contexts that are created and cached in the connection pool.<br>See Pooling and Caching for more information.<br>If this field is set to zero, the naming context is not cached, and a new context is created for each lookup operation. |
| **Connection Retries** | Yes | The maximum number of attempts can be made to connect to the application server, or to create the naming context.<br>If this field is set to zero, only one attempt can be made to establish a connection. |
| **Retry Interval (ms)** | Yes | The time interval, in milliseconds, of making each connection attempt to the application server, or to create the naming context.<br>If this field is set to zero, the new connection attempt is started immediately. |
| **Test Connection** | No | Used to test the connection configuration for the JNDI server. |

### Pooling and Caching

Creating an InitialContext class requires a large amount of overhead. Therefore, contexts can be cached and placed in a pool to improve performance over time. Define the **Max Connections** field to specify the maximum number of InitialContext classes that the plug-in creates at any given time.

The InitialContext class is created when you start a process during initializing the EJB2Home activity or EJB3Remote activity, and all created contexts are placed into the pool. If you want to use a context, fetch an existing context from the pool to obtain a reference to the EJB 2.x home object or EJB 3.x remote object. After the EJB 2.x home object or EJB 3.x remote object is obtained, the context is released back into the pool. If all contexts are being used, TIBCO ActiveMatrix BusinessWorks Plug-in for EJB blocks any new requests until a context is freed from the pool.

All contexts are cached for reuse by subsequent process instances. If you specify zero in the **Max Connections** field, contexts are not cached, and each request creates a new InitialContext class.

If a context becomes stale (for example, the server is restarted), TIBCO ActiveMatrix BusinessWorks Plug-in for EJB attempts to create a new context to replace the stale context in the pool. Use the **Connection Retries** field to define the maximum number of attempts to reestablish the context.

# EJB Palette

A palette groups the activities that connect the same external applications together. An EJB palette is added after installing TIBCO ActiveMatrix BusinessWorks Plug-in for EJB.

The EJB palette consists of three activities:

- EJB2Home

  This activity connects to an EJB server for EJB 2.x (specified by an EJB Configuration shared resource, as described in EJB Configuration Shared Resource), performs a JNDI lookup operation to obtain a reference to a home object, and obtains a reference to a remote object from the home object. You can use this activity to invoke any method defined by the home object.

- EJB2Remote

  This activity invokes remote methods on the remote object obtained by the EJB2Home activity from an EJB server for EJB 2.x. This activity must be placed after an EJB2Home activity in a process definition.

- EJB3Remote

  This activity connects to an EJB server for EJB 3.x (specified by an EJB Configuration shared resource), performs a JNDI lookup operation to obtain a reference to a remote object, and invokes remote methods.

## EJB2Home

The EJB2Home activity connects to an EJB server for EJB 2.x, performs a JNDI lookup operation to obtain a reference to a home object, and obtains a reference to a remote object from the home object. You can use this activity to invoke any method defined by the home object.

You can also use this activity to invoke a method on the EJB home. After this activity is performed, you can use the EJB2Remote activity to invoke methods on the remote object without performing additional remote lookups.

It is optional to cache the home object and the stateless remote objects to reuse them across process instances. However, this improves the performance because further remote lookups are not necessary.

### General

In the **General** tab, you can establish a connection to an EJB server, and specify the JNDI server for the EJB.

| Field | Module Property? | Description |
|---|---|---|
| **Name** | No | The name to be displayed as the label for the activity in the process. |
| **EJB Configuration** | Yes | An EJB Configuration shared resource defines a set of relationships and their participating entities.<br>Click 🔍 to select an EJB Configuration shared resource.<br>If no matching EJB Configuration shared resource is found, click **Create Shared Resource** to create one. For more details, see Creating an EJB Configuration Shared Resource. |

| Field | Module Property? | Description |
|---|---|---|
| **JNDI Name** | Yes | The name registered with the JNDI server for the EJB. |
| **Home Interface Class** | No | The name of the home interface for the EJB. The home interface extends javax.ejb.EJBHome.<br>Click 🔍 to display the available classes for enterprise beans. |
| **Home Interface Method** | No | The method of the home interface to call. The list provides the methods contained in the selected interface class in the **Home Interface Class** field. |

### Description

In the **Description** tab, you can enter a short description for the EJB2Home activity.

### Advanced

In the **Advanced** tab, you can specify the objects that whether you want to cache:

| Field | Module Property? | Description |
|---|---|---|
| **Cache Home Object** | No | Specifies whether the home object is to be cached for use by process instances.<br>• If this check box is cleared, each process instance performs a JNDI lookup to obtain the home object reference.<br>• If this check box is selected, process instances reuse the cached home object reference.<br>Choose to cache the home or remote object improves performance over time. However, the home or remote object might become stale. That is, the object changes on the EJB server, and the cached object no longer matches the object on the server.<br>If the EJB2Home activity encounters a stale home object in the cache, the activity attempts to re-create the home object. The **Connection Retries** field of the EJB Configuration shared resource specifies the number of retries that the EJB2Home activity performs before failing to re-create the home object. If the EJB2Home activity is successful in re-creating the home object, the activity succeeds and a new home object is placed in the cache. If the activity cannot re-create the home object after the maximum number of retries, the activity fails and takes the error transition. |

| Field | Module Property? | Description |
|---|---|---|
| **Cache Stateless Remote Object** | No | Specifies whether the remote object is to be cached for use by process instances. The remote object is only cached for stateless session enterprise beans.<br>• If this check box is cleared, each process instance obtains a new remote object reference.<br>• If this check box is selected, process instances reuse the cached remote object reference.<br>If the process instance encounters a stale cached remote object, the object is removed from the cache, an error is returned, and the activity fails. However, subsequent process instances attempt to re-create the remote object. |

### Input

The following table lists the input element in the **Input** tab of the EJB2Home activity:

| Input Item | Data Type | Description |
|---|---|---|
| `MethodParameters` | Complex | An object containing the parameters required by the home interface method. This item is displayed only if the method selected in the **General** tab requires parameters. |

### Output

In the **Output** tab, you can find the search results.

The following table lists the output element in the **Output** tab of the EJB2Home activity:

| Output Item | Data Type | Description |
|---|---|---|
| `MethodReturnValue` | Complex | The value returned by the home interface method. This item is displayed only if the home interface method selected in the **General** tab returns a value. |

### Fault

In the **Fault** tab, you can find the error code and error message of the EJB2Home activity. See Error Codes for more detailed explanation of errors.

The following table lists error schema elements in the **Fault** tab of the EJB2Home activity:

| Error Schema Element | Data Type | Description |
|---|---|---|
| errorCode | String | Displays the error code returned by the plug-in. |
| errorMessage | String | Displays the error message returned by the plug-in. |
| errorStackTrace | String | Displays the complete stack trace that causes the exception. |

| Error Schema Element | Data Type | Description |
|---|---|---|
| exceptionClass Name | String | Displays the class name of the root exception that causes the exception. |

# EJB2Remote

The EJB2Remote activity invokes remote methods on the remote object from an EJB server for EJB 2.x. This activity must be placed after an EJB2Home activity in a process definition.

## General

In the **General** tab, you can establish a connection to an EJB server, and specify the remote interface and call a method.

| Field | Module Property? | Description |
|---|---|---|
| **Name** | No | The name to be displayed as the label for the activity in the process. |
| **Home Activity** | No | The EJB2Home activity in the process definition that obtained the reference to the remote object.<br>The list only supplies EJB2Home activities that obtain references to remote objects. EJB2Home activities that do not create remote objects are not listed. |
| **Remote Interface Class** | No | The name of the remote interface (the interface that extends javax.ejb.EJBObject) obtained by the EJB2Home activity. This field is populated automatically based on the selected EJB2Home activity. |
| **Remote Interface Method** | No | The method of the remote interface to be called. The list provides a list of public methods contained in the remote interface class. |

## Description

In the **Description** tab, you can enter a short description for the EJB2Remote activity.

## Advanced

In the **Advanced** tab, you can specify the remote objects that you want to release:

| Field | Module Property? | Description |
|---|---|---|
| **Release Remote Object** | No | If this check box is selected, the remote object is released after this activity is completed. If this check box is cleared, the remote object is used by subsequent EJB2Remote activities within this process definition.<br>This check box is only meaningful for stateful session and entity beans. Cached remote stateless session beans are never released when this check box is selected. This check box is useful if you want to release the memory used by the stateful session and entity bean before continuing with the remainder of the process definition.<br>If you invoke a method on a stateful session or entity bean, select this check box only when no subsequent activities in the process definition invoke the remote object. If the object is released and a subsequent invocation is made, an error is returned. |

### Input

The following table lists the input element in the **Input** tab of the EJB2Remote activity:

| Input Item | Data Type | Description |
|---|---|---|
| `MethodParameters` | Complex | An object containing the parameters required by the remote interface method. This item is displayed only if the method selected in the **General** tab requires parameters. |

### Output

In the **Output** tab, you can find the search results.

The following table lists the output element in the **Output** tab of the EJB2Remote activity:

| Output Item | Data Type | Description |
|---|---|---|
| `MethodReturnValue` | Complex | The value returned by the remote interface method. This item is displayed only if the remote interface method selected in the **General** tab returns a value. |

### Fault

In the **Fault** tab, you can find the error code and error message of the EJB2Remote activity. See Error Codes for more detailed explanation of errors.

The following table lists error schema elements in the **Fault** tab of the EJB2Remote activity:

| Error Schema Element | Data Type | Description |
|---|---|---|
| errorCode | String | Displays the error code returned by the plug-in. |

| Error Schema Element | Data Type | Description |
|---|---|---|
| errorMessage | String | Displays the error message returned by the plug-in. |
| errorStackTrace | String | Displays the complete stack trace that causes the exception. |
| exceptionClass Name | String | Displays the class name of the root exception that causes the exception. |

## EJB3Remote

The EJB3Remote activity connects to an EJB server for EJB 3.x, performs a JNDI lookup operation to obtain a reference to the remote object, and invokes remote methods.

### General

In the **General** tab, you can establish a connection to an EJB server for EJB 3.x.

| Field | Module Property? | Description |
|---|---|---|
| **Name** | No | The name to be displayed as the label for the activity in the process. |
| **Use Cached Remote Object** | No | If this check box is selected, the **Cache From** field is displayed, and you can choose an existing remote object.<br>If this check box is cleared, the **EJB Configuration** and **JNDI Name** fields are displayed. |
| **Cache From** | No | The remote object that you want to reuse.<br>This field is displayed only when the **Use Cached Remote Object** check box is selected. |
| **EJB Configuration** | Yes | An EJB Configuration shared resource defines a set of relationships and their participating entities.<br>Click 🔍 to select an EJB Configuration shared resource.<br>This field is displayed only when the **Use Cached Remote Object** check box is cleared. |
| **JNDI Name** | Yes | The name registered with the JNDI server for EJB.<br>This field is displayed only when the **Use Cached Remote Object** check box is cleared. |
| **Remote Interface Class** | No | The name of the remote interface for EJB 3.x, which includes the annotation @Remote (javax.ejb.Remote).<br>Click 🔍 to display the available classes for enterprise beans. |

| Field | Module Property? | Description |
|---|---|---|
| **Remote Interface Method** | No | The remote interface method to be called. The list provides the methods contained in the selected interface class in the **Remote Interface Class** field. |

### Description

In the **Description** tab, you can enter a short description for the EJB3Remote activity.

### Advanced

In the **Advanced** tab, you can specify the remote objects that you want to release.

The following table lists the configurations in the **Advanced** tab of the EJB3Remote activity:

| Name | Module Property? | Description |
|---|---|---|
| **Release Remote** | No | If this check box is selected, the remote object is released after this activity is completed.<br>If this check box is cleared, the remote object is put in a cache pool after this activity is completed; therefore, you can reuse this remote object in the other EJB3Remote activities. |

### Input

The following table lists the input element in the **Input** tab of the EJB3Remote activity:

| Input Item | Data Type | Description |
|---|---|---|
| **MethodParameters** | Complex | An object containing the parameters required by the remote interface method. This item is displayed only if the method selected in the **General** tab requires parameters. |

### Output

The following table lists the output element in the **Output** tab of the EJB3Remote activity:

| Output Item | Data Type | Description |
|---|---|---|
| **`MethodReturnValue`** | Complex | The value returned by the remote interface method. This item is displayed only if the remote interface method selected in the **General** tab returns a value. |

### Fault

In the **Fault** tab, you can find the error code and error message of the EJB3Remote activity. See Error Codes for more detailed explanation of errors.

The following table lists error schema elements in the **Fault** tab of the EJB3Remote activity:

| Error Schema Element | Data Type | Description |
|---|---|---|
| errorCode | String | Displays the error code returned by the plug-in. |
| errorMessage | String | Displays the error message returned by the plug-in. |
| errorStackTrace | String | Displays the complete stack trace that causes the exception. |
| exceptionClass Name | String | Displays the class name of the root exception that causes the exception. |

# Working with the Sample Project

TIBCO ActiveMatrix BusinessWorks Plug-in for EJB packages a sample project with the installer. The sample project shows how the plug-in works.

After installing the plug-in, you can locate the sample project in the `*TIBCO_HOME*/bw/palettes/ejb/*version_number*/Samples` directory. This sample project contains six processes; each process corresponds to a task. This project uses JBoss Application Server as the EJB container.

- basic_use_ejb2.bwp
- basic_use_ejb3.bwp
- ejb_with_java_invoke.bwp
- ejb3_with_JavaToXml.bwp
- remote_object_release.bwp
- stateless_remote_cache.bwp

## Importing the Sample Project

Before running the project, you must import the sample project to TIBCO Business Studio.

### Prerequisites

- Ensure that you have started JBoss Application Server.
- Ensure that the `.jar` files, `ejb3server.jar` and `jboss7stateless.jar`, in the `*TIBCO_HOME*/bw/palettes/ejb/*version_number*/Samples/Examples/Examples/lib` directory, are deployed to your JBoss Application Server.

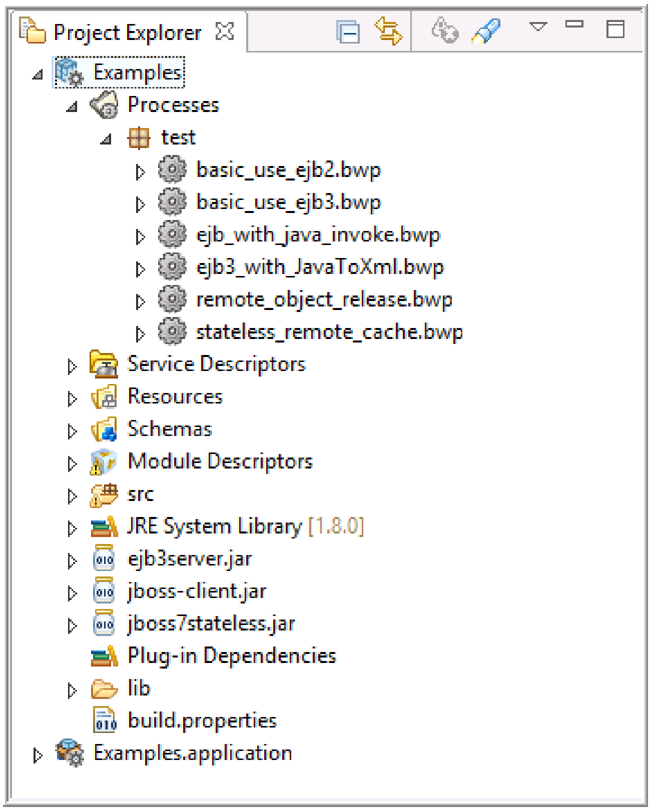
### Procedure

1. Start TIBCO Business Studio using one of the following ways:
   - Microsoft Windows: click **Start** > **All Programs** > **TIBCO** > ***TIBCO_HOME*** > **TIBCO Business Studio *version_number*** > **Studio for Designers**.
   - Mac OS and Linux: run the TIBCO Business Studio executable file located in the `*TIBCO_HOME*/studio/*version_number*/eclipse` directory.
2. From the menu, click **File** > **Import**.
3. In the Import dialog, expand the **General** folder and select the **Existing Studio Projects into Workspace** item. Click **Next.**
4. Click **Browse** next to the **Select root directory** field to locate the sample. Click **Finish.**

   The sample project is located in the `*TIBCO_HOME*/bw/palettes/ejb/*version_number*/Samples` directory.

### Result

The sample project is imported to TIBCO Business Studio.



## Running the Sample Project

You can run the sample project to see how TIBCO ActiveMatrix BusinessWorks Plug-in for EJB works.

### Prerequisites

Ensure that you have imported the sample project to TIBCO Business Studio, as described in Importing the Sample Project.

### Procedure

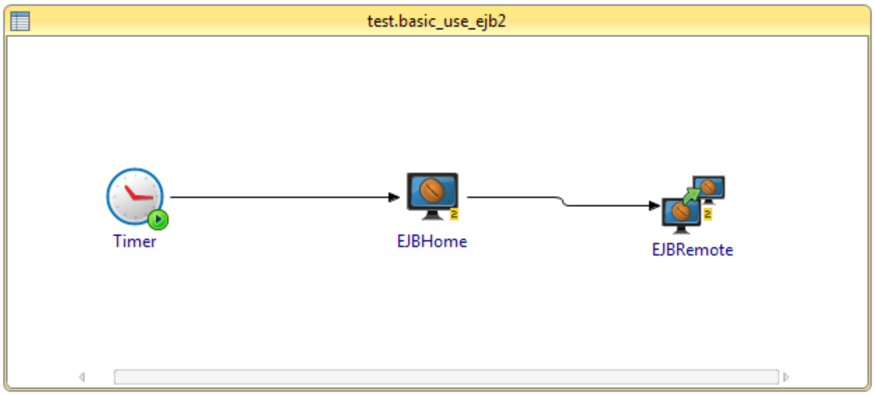
1. In the Project Explorer view, expand **lib**. Double-click **jboss-ejb-client.properties** and **jndi.properties** and update the contents according to your JBoss Application Server.
2. In the Project Explorer view, expand the **Resource** folder, and then expand the **test** shared resource.
3. Double-click **NewEJBResource.ejbResource** to edit the EJB connection, and then click **Test Connection** to validate your connection.
4. In the Project Explorer view, expand the **Module Descriptors** resource, and then double-click **Components**.
5. By default, all the processes are listed in the Components editor. In the Components editor, select the process that you do not want to run and click ✖.
6. On the toolbar, click the 💾 icon to save your changes.
7. From the menu, click **Run** > **Run Configurations** to run the selected process.
8. In the Run Configurations dialog, expand **BusinessWorks Application** and click **BWApplication**.
9. In the right panel, click the **Applications** tab, and select the check box next to **Examples.application**.

10. Click **Run** to run the process.

11. Click the ■ icon to stop the process.

## Configurations for basic_use_ejb2

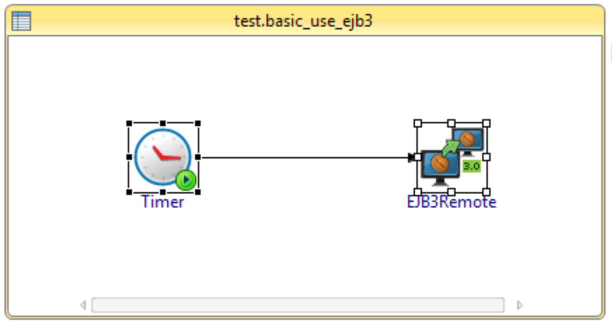
The basic_use_ejb2 process shows how to perform a lookup of an EJB 2.x home object, create a remote object, and invoke a remote method.



| Activity | Description |
|---|---|
| Timer | This activity starts the process at a specific time. |
| EJB2Home | This activity performs a JNDI lookup in the **NewEJBResource.ejbResource** EJB Configuration shared resource and creates an EJB 2.x remote object. |
| EJB2Remote | This activity invokes a remote method on the remote object that is created by the EJB2Home activity. |

## Configurations for basic_use_ejb3

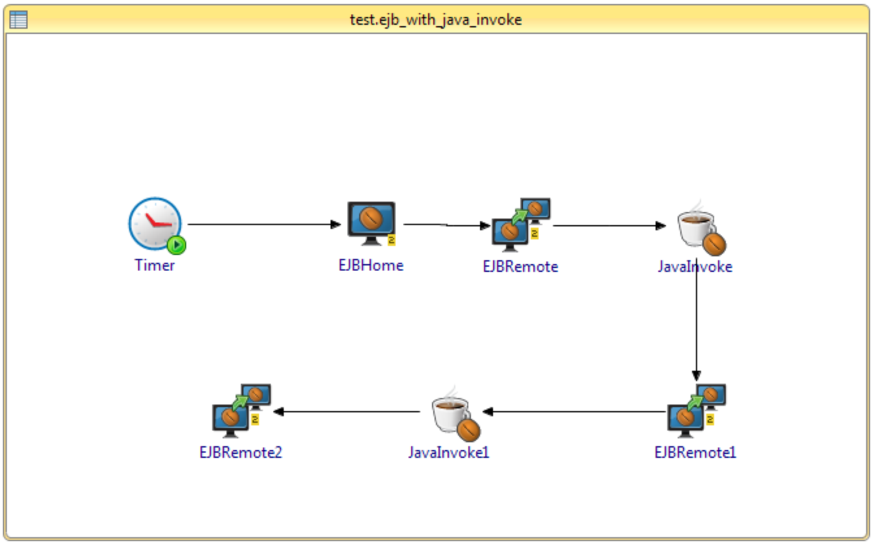
The basic_use_ejb3 process shows how to perform a lookup of an EJB 3.x remote object and invoke a remote method.



| Activity | Description |
|---|---|
| Timer | This activity starts the process at a specific time. |
| EJB3Remote | This activity performs a JNDI lookup in the **NewEJBResource.ejbResource** EJB Configuration shared resource and creates an EJB 3.x remote object, and then invokes a remote method. |

## Configurations for ejb_with_java_invoke

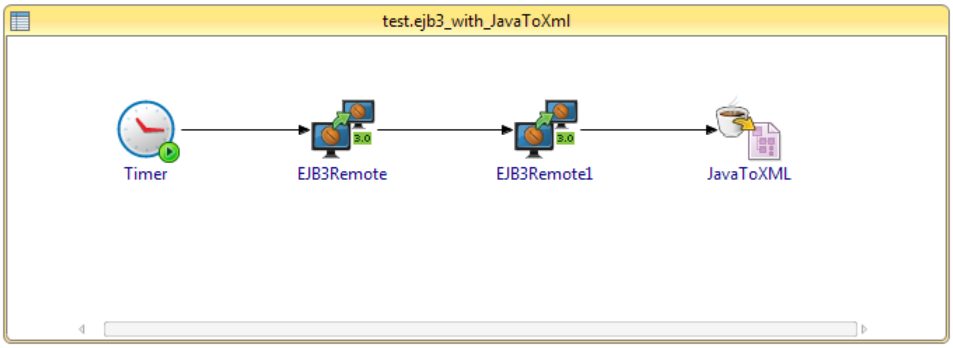
The ejb_with_java_invoke process shows how to use the EJB activities with the Java Invoke activities, to add, update, and delete data in a catalogue.



| Activity | Description |
|---|---|
| Timer | This activity starts the process at a specific time. |
| EJBHome | This activity performs a JNDI lookup in the **NewEJBResource.ejbResource** EJB Configuration shared resource and creates an EJB 2.x remote object. |
| EJBRemote | This activity invokes the addCatalogue() method on the remote object that is created by the EJBHome activity. As a result, a record is added to the catalogue. |
| JavaInvoke | This activity invokes a Java class method to check whether the catalogue record is added after the EJBRemote activity. |
| EJBRemote1 | This activity invokes the updateCataloguePrice() method on the remote object that is created by the EJBHome activity. As a result, a record is updated in the catalogue. |
| JavaInvoke1 | This activity invokes a Java class method to check whether the catalogue record is updated after the EJBRemote1 activity. |
| EJBRemote2 | This activity invokes the deleteCataloguePrice() method on the remote object that is created by the EJBHome activity. As a result, a record is deleted in the catalogue. |

## Configurations for ejb3_with_JavaToXml

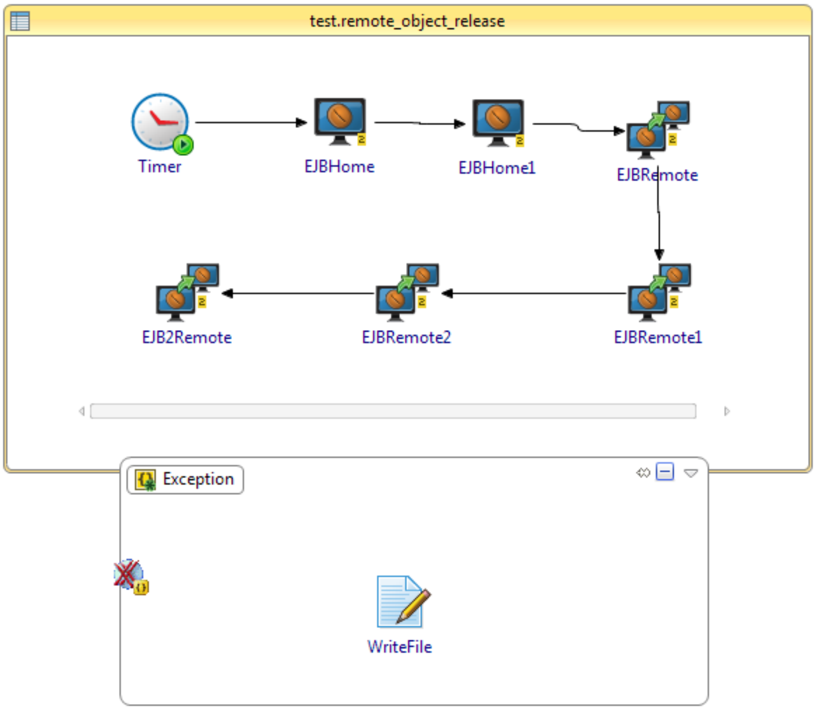
The ejb3_with_JavaToXml process shows how to use EJB activities with a JavaToXML activity to convert an EJB 3.x remote object to XML format.



| Activity | Description |
|---|---|
| Timer | This activity starts the process at a specific time. |
| EJB3Remote | This activity performs a JNDI lookup in the **NewEJBResource.ejbResource** EJB Configuration shared resource and creates an EJB 3.x remote object, and then invokes a remote method. |
| EJB3Remote1 | This activity receives the cached EJB 3.x object from the EJB3Remote activity and invokes another remote method. |
| Java To XML | This activity converts the data of the EJB 3.x object into an XML document. |

## Configurations for remote_object_release

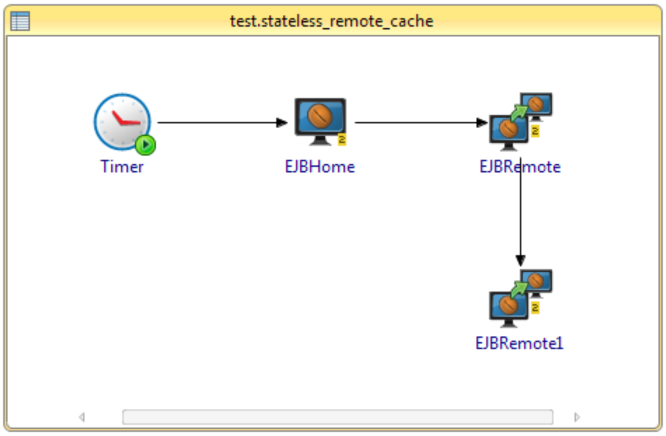
The remote_object_release process shows how to release EJB 2.x remote objects in one process instance.



| Activity | Description |
|---|---|
| Timer | This activity starts the process at a specific time. |
| EJBHome | This activity performs a JNDI lookup in the **NewEJBResource.ejbResource** EJB Configuration shared resource and creates an EJB 2.x remote object. |
| EJBHome1 | This activity performs a JNDI lookup in the **NewEJBResource.ejbResource** EJB Configuration shared resource and creates another EJB 2.x remote object. |
| EJBRemote | This activity invokes a remote method on the remote object that is created by the EJBHome activity. The **Release Remote Object** check box is selected in the **Advance** tab of this activity. |
| EJBRemote1 | This activity invokes a remote method on the remote object that is created by the EJBHome1 activity. The **Release Remote Object** check box is cleared in the **Advance** tab of this activity. |
| EJBRemote2 | This activity invokes a remote method on the remote object that is created by the EJBHome activity.<br>After the EJBRemote activity is completed and then the EJB 2.x object is released, a null pointer exception occurs. |
| EJB2Remote | This activity invokes a remote method on the remote object that is created by the EJBHome1 activity. |
| Write File | This activity writes the exception details to the specified file. |

## Configurations for stateless_remote_cache

The stateless_remote_cache process shows how to keep an EJB 2.x remote object in a cache when handling a stateless session bean.



| Activity | Description |
|---|---|
| Timer | This activity starts the process at a specific time. |

| Activity | Description |
|---|---|
| EJBHome | This activity uses the **NewEJBResource.ejbResource** EJB Configuration shared resource and performs a JNDI lookup of an EJB 2.x remote object. The **Cache Stateless Remote Object** check box is selected in the **Advance** tab of this activity. |
| EJBRemote | This activity invokes a remote method on the remote object that is created by the EJBHome activity. The **Release Remote Object** check box is selected in the **Advance** tab of this activity. |
| EJBRemote1 | This activity invokes a remote method on the remote object that is created by the EJBHome activity. |

# Managing Logs

When an error occurs, you can check logs to trace and troubleshoot the plug-in exceptions.

By default, error logs are displayed in the Console view when you run a process in debug mode. You can change the log level of the plug-in to trace different messages and export logs to a file. Different log levels correspond to different messages, as described in Log Levels.

## Log Levels

Different log levels include different information.

The plug-in supports the following log levels:

| Log Level | Description |
|---|---|
| Trace | Includes all information regarding the running process. |
| Debug | Indicates a developer-defined tracing message. |
| Info | Indicates normal plug-in operations. No action is required. A tracing message tagged with Info indicates that a significant processing step is reached, and logged for tracking or auditing purposes. Only info messages preceding a tracking identifier are considered as significant steps. |
| Warn | Indicates that an abnormal condition occurred. Processing continues, but for best practice, you can contact the administrator to investigate it. |
| Error | Indicates that an unrecoverable error occurred. Depending on the severity of the error, the plug-in might continue with the next operation or might stop. |

## Setting Up Log Levels

You can configure different log levels for the plug-in and plug-in activities to trace different messages.

If you do not configure any log levels, the plug-in uses the default log level of TIBCO ActiveMatrix BusinessWorks. The default log level is Error.

If neither the plug-in log nor the BusinessWorks log is configured in the `logback.xml` file, the error logs of the plug-in are displayed in the Console view by default.

If the plug-in log is not configured but the BusinessWorks log is configured in the `logback.xml` file, the configuration for the BusinessWorks log is implemented by the plug-in.

### Procedure

1. Navigate to the `*TIBCO_HOME*/bw/*version_number*/config/design/logback` directory and open the `logback.xml` file.
2. Add the following node in the **BusinessWorks Palette and Activity loggers** area to specify a log level for the plug-in:

```
<logger name="com.tibco.bw.palette.bwpluginejb.runtime">
   <level value="DEBUG"/>
</logger>
```

   The value of the `level` element can be `Error`, `Info`, or `Debug`.

If you set the log level to `Debug`, the input and output for the plug-in activities are also displayed in the Console view. See Log Levels for more details regarding each log level.

3. Optional: Add the following node in the **BusinessWorks Palette and Activity loggers** area to specify a log level for an activity:

```
<logger name="com.tibco.bw.palette.bwpluginejb.runtime.ActivityNameActivity">
   <level value="DEBUG"/>
</logger>
```

For example, add the following node to set the log level of the EJB2Home activity to `Debug`:

```
<logger name="com.tibco.bw.palette.bwpluginejb.runtime.EJBHomeActivity">
   <level value="DEBUG"/>
</logger>
```

The activities that are not configured with specific log levels use the log level configured.

4. Save the file.

## Exporting Logs to a File

You can update the `logback.xml` file to export plug-in logs to a file.

### Procedure

1. Navigate to the `*TIBCO_HOME*/bw/*version_number*/config/design/logback` directory and open the `logback.xml` file.

After deploying an application in TIBCO Enterprise Administrator, navigate to the `*TIBCO_HOME*/bw/domains/mydomain/appnodes/myspace/mynode` directory to find the `logback.xml` file.

2. Add the following node to specify the file where the log is exported:

```
<appender name="FILE" class="ch.qos.logback.core.FileAppender">
   <file>c:/bw6-bwpluginejb.log</file>
      <encoder>
        <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36}-%msg%n</pattern>
      </encoder>
</appender>
```

The value of the `file` element is the absolute path of the file that stores the exported log.

Add also the file name in the file path.

3. Add the following node to the `root` node at the bottom of the `logback.xml` file:

```
    <appender-ref ref="FILE" />
```

```
<root level="DEBUG">
   <appender-ref ref="STDOUT" />
   <appender-ref ref="FILE" />
</root>
```

4. Save the file.

# Error Codes

The following table lists error codes, detailed explanation of each error, where applicable, and ways to solve different errors.

| Error Code and Error Message | Role | Category | Description | Solution |
|---|---|---|---|---|
| TIBCO-BW-PALETTE-EJB-500001<br>`{0}` | errorRole | BW_Plugin | The general information of error messages. | No action. |
| TIBCO-BW-PALETTE-EJB-500002<br>`Class {0} cannot be loaded in app class path.` | errorRole | BW_Plugin | Fails to load the specified class in the app class path. | Check whether the `class .jar` file is added into the class path of this project. |
| TIBCO-BW-PALETTE-EJB-500003<br>`Exception occurred while {0}.` | errorRole | BW_Plugin | An exception occurs when initializing the JNDI server. | Check for the exception error message. |
| TIBCO-BW-PALETTE-EJB-500004<br>`Unknown exception occurred while {0}.` | errorRole | BW_Plugin | An unknown exception occurs. | Check for the exception error message. |
| TIBCO-BW-PALETTE-EJB-500005<br>`Method {0} cannot be found in class {1}.` | errorRole | BW_Plugin | Fails to find the specified method in the class. | Check whether the invoked method has been declared in the class file. |
| TIBCO-BW-PALETTE-EJB-500006<br>`Security exception while get method {0} from class {1}.` | errorRole | BW_Plugin | A security exception occurs while receiving the specified method from the class. | Check the permission for communication. |
| TIBCO-BW-PALETTE-EJB-500007<br>`Class cast exception occurred, the look up object cannot be cast to EJBHome object.` | errorRole | BW_Plugin | A class cast exception occurs while casting the lookup object to the EJB home object. | Check whether the JNDI name is correct, and the lookup result is not an EJB home reference. |

| Error Code and Error Message | Role | Category | Description | Solution |
|---|---|---|---|---|
| TIBCO-BW-PALETTE-EJB-500008<br>`IO exception occurred while {0}.` | errorRole | BW_Plugin | An IO exception occurs. | Check the IO channel. |
| TIBCO-BW-PALETTE-EJB-500009<br>`{0} {1}, Invalid beantype. Beantype= {2}.` | errorRole | BW_Plugin | An error occurs when the bean type is invalid. | Check in the EJB container whether the bean type has been specified to be one of EJB. |
| TIBCO-BW-PALETTE-EJB-500010<br>`{0} {1}, remote exception occurred while {2}.` | errorRole | BW_Plugin | A remote exception occurs. | Check the error message on the application server. |
| TIBCO-BW-PALETTE-EJB-500011<br>`{0} {1}, illegal access exception while invoke method {2} in class {3}.` | errorRole | BW_Plugin | An illegal access exception occurs while invoking the method in the specified class. | Check the access permission. |
| TIBCO-BW-PALETTE-EJB-500012<br>`{0} {1}, illegal argument exception while invoke method {2} in class {3}.` | errorRole | BW_Plugin | An illegal argument exception occurs while invoking the method in the specified class. | Specify the input parameters to match the requirements of the invoked method. |
| TIBCO-BW-PALETTE-EJB-500013<br>`{0} {1}, invocation target exception while invoke method {2} in class {3}.` | errorRole | BW_Plugin | An invocation target exception occurs while invoking the method in the specified class. | Check the error message on the application server. |
| TIBCO-BW-PALETTE-EJB-500014<br>`{0} {1}, remote object cannot be found in checked in object.` | errorRole | BW_Plugin | Fails to find the remote object in the checked-in object. | Check whether the JNDI name is correct and perform the lookup again. |

| Error Code and Error Message | Role | Category | Description | Solution |
|---|---|---|---|---|
| TIBCO-BW-PALETTE-EJB-500015<br>`{0}, checked in object cannot be found in process job resource.` | errorRole | BW_Plugin | Fails to find the checked-in object in the process job resource. | Perform the lookup again because the object has been released from the process context. |