TIBCO BusinessWorks™ Container Edition

Binding and Palette Reference

Version 2.10.0 | December 2024

Document Updated: February 2025

Contents

Binding

Bindings, which are well known external communication gateways, establish connections between Service Oriented Architecture (SOA) services and their consumers.

The two types of binding components are:

- **Service Binding**: used to create and expose a service to the external world. The service can contain one or more operations. Once exposed, clients can consume the service.

- **Reference Binding**: used to create a client that can connect and communicate to an external service.

TIBCO BusinessWorks™ Container Edition supports REST as well as SOAP bindings.

For more information about REST bindings, see the *TIBCO BusinessWorks Container Edition REST Reference*.

REST Binding

REST Binding provides external connectivity for REST over HTTP. You can specify custom HTTP headers and parameters using REST binding. It supports GET, PUT, POST, and DELETE HTTP methods. It also supports JSON, XML, and plain text message types.

Binding

This section has the following fields.

Field	Description
Resource	The name of the resource.
Resource Service Path	Specify the path of the Service Resource.
HTTP	The name of the HTTP Connector.

Field	Description
Connector Name	**Tip:** Click on the **HTTP Connector Name** field to display details about the HTTP Connector resource.
Request Client Format	The type of request message format. The two available request message type options are: **Body** and **Form**. • **Body**: Provides two request body format option: JSON and XML. Both or any one of these options can be used. • **Form**: This option can be used to inject the parameters of a Web form into a RESTful web service.
Reply Client Format	The type of reply message format. The two available reply message format options are: JSON and XML.

Operations

This section shows the following details.

Field	Description
Name	The name of the HTTP method used, for example, GET, PUT, POST, and DELETE.
Nickname	The specified name of the service, for example, getBooks.

Operation Details

This section shows the following details.

Field	Description
Summary	This tab has the following available options: • **Summary**: The summary of the REST resource

Field	Description

	• **HTTP Method**: Displays the HTTP Method specified in the **Operations** section. These are the available HTTP methods:
	   ○ POST
	   ○ GET
	   ○ PUT
	   ○ DELETE

Request	Specifies the resource created, using the POST method. **Note:** Request and Response depend on the method selected.

Response	This tab has the following available options:
	
	• **Use HTTP Headers**: Selecting this check box includes the REST Web service application (or client) within the HTTP headers and body of a request, all of the parameters, context, and data required by the server-side component to generate a response.
	
	• **Use Custom Status Line**: You can specify a custom status line (status code and reason phrase) to the outgoing message. The codes used must be defined in the configuration under the **Response Status** tab.
	
	• **Response with Status Code Only**: The operation returns a status code as response, when this check box is selected. Message body is not required. For example, using a POST operation returns a **201** status code which means "**Created**" and responds with the resource URL.
	
	• **Resource Schema**: Displays the schema selected. This option is not available when the **Use Custom Status Line** and **Response with Status Code Only** check boxes are selected. These are the available options:
	   ○ String
	   ○ Integer
	   ○ Boolean
	   ○ **XSD element**: Selecting this option to either select the XSD schema element available under the **Schemas** folder of your

Field	Description
	project or a create new XML schema resource. Click **Create New Schema** to a create new XML schema resource using the Simplified Schema Editor wizard.
Response Status	Specifies the response code for the operation and the code message. For example, code **201** means **Created** or code **503** means **Service Unavailable**. You can also add your own custom code and reason phrase.

Parameters

This section shows the following details.

Parameter Name	Type	Required
Parameter name of the operation used	The parameter type. It can be any one of the following: • String • Integer • Boolean	Whether this parameter is required. The available options are **Yes** or **No**.

Working with Path and Query Parameters

TIBCO BusinessWorks Container Edition REST binding supports the use of `path` parameters for the resource service path (for example, `/books/{isbn}` instead of `/books}` in the REST service. The user must specify every parameter by enclosing it in brackets `{ }`. For example, `/book/{isbn}` specifies the user can have `isbn` as a parameter. The client invokes this service using the URL `http:/<host>:<port>/book/<isbn>` where `<isbn>` is an actual value. Using the path parameters combined with static URL, for example, `/book/{isbn}/events`, is also supported.

> ❗ **Important:** The Query and Path parameter names should not be same as the Form parameters name. For example, if Form parameter contains name as one of its parameter, the following path and query parameters are not allowed and throws an exception:
>
> ```
> /resource/{name}
>
>  /resource?name={name}
> ```

All the parameters defined in the resource service path are made available to the user as an **Input** for every operation. Refer to *Support for Path and Query Parameters*. See the following image for the `/book/{isbn}`) example to see the **Input** for the GET operation.
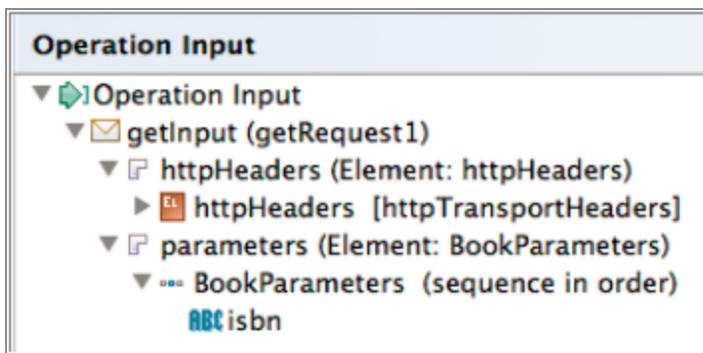
About Swagger UI

Using the **Swagger UI** you can visualize RESTful services. It specifies the format (URL, method, and representation) to describe REST the web services.

> ℹ️ **Note:** Authentication support has been added to the REST binding. Hence, the Swagger UI displays Basic Auth **Username** and **Password** fields, regardless of whether the **Authenticate** check box is selected or not selected in REST binding.

Partial Response in REST

Using the **Partial Response** feature in REST, you can retrieve only the data you need, instead of bulk data as a response. You can also request only those fields that are required as part of the response.

For example, in the Swagger UI to select the immediate children nodes of an object, specify the required field names separated by comma such as ISBN, author, bookName, and so on.

The partial response feature uses a fields query parameter.

> **ℹ Note:** The fields keyword is reserved to be used internally by TIBCO BusinessWorks Container Edition. Adding the fields keyword as a query parameter automatically triggers support for partial responses.

Partial responses do not support selecting particular objects from an array. For example, using the fields keyword for /books/isbn, the isbn's of all the books are returned.

Format: The values of the fields keyword can be a comma separated value of fields of the response message. Fields can be specified directly, for example, using `isbn` or hierarchy `/book/isbn`.

Rest Service Wizard

Rest Service Wizard is used to create a new REST resource or add REST services to an existing resource in TIBCO BusinessWorks Container Edition.

The Rest Service Wizard has the following fields:

Field	Description
Resource Name	The name of the REST resource.
Summary	The summary or description of the REST resource.
Resource Service Path	The relative path for this REST service resource.
	If an application contains multiple REST bindings, ensure that the location of the path parameters is unique for each REST binding.
	For example, one REST binding is using the paths `/book/{isbn}` and another REST binding is using the path `/book/{authorid}`. Since `{isbn}` and `{authorid}` are defined at the same location in the URI , one of these services do not function correctly.
	In addition to path parameters, the path in a REST binding can also contain query parameters. For example,
	`/resource/path/{pathparam}? query={queryparam}` or

Field	Description
	`/resource/path/{pathparam}?{ queryparam}`
Resource Definition	The XSD schema element to be used for creating the REST resource.
	You can also use this to create the input and output of each operation defined. You can override this on the next screen if required, for each operation.
Operations	These are the HTTP methods implemented by this REST service.
	Currently POST, GET, PUT, PATCH, DELETE, OPTIONS, HEAD, and custom methods are supported for users to implement.
	Note: You can add custom operations by clicking on the **Add Custom Operation** button.
Implementation Data	The implementation data field can be **Structured** or **Opaque**.
	• **Structured**: The XSD element structure is preserved for the input and output of every operation. You need not manually parse the payload to generate the actual element to be used in the process.
	• **Opaque**: Use this mode to apply the pass through mechanism. You get a `messageBody` element in the input or output of every operation and then you must use either parse activities for JSON or XML to get a structured output for the payload.

Note: To add additional services to a process, click **Create A Rest Service** on the top left of the process canvas.

> **(i)** **Note:** Re-creating a component containing a REST binding after deleting the component is not supported. To add the REST binding, in the process editor, right click on the service without a binding. Go to **Components > [componentware] > Create REST Binding**. A binding is created for the service. The binding has to be re-configured as the previous configurations are lost.

REST Reference

Create a REST Reference binding to consume a REST endpoint. You cannot edit the REST Reference binding configuration.

Before you begin

Swagger 2.0-compliant REST API documents must be imported into the TIBCO BusinessWorks Container Edition project's Service Descriptors folder. This will give you the ability to expand and collapse endpoints, operations, parameters and response codes in the Project Explorer view.

To consume a REST API that exists in the Service Descriptor of the project, do the following:

Procedure

1. Expand the Swagger 2.0 files in the Service Descriptors special folder to view the endpoints, operations, parameters, response codes.

2. Drag and drop an endpoint on the right side of the canvas to create a REST Reference Binding.

    This will create a cloud shaped icon with a right facing arrow. The cloud is an indication that it is a REST Reference whereas the arrow within the cloud indicates that it is a binding. Since the binding is within a cloud, it is an indication that it is a REST binding. You cannot convert a REST binding to a SOAP binding or vice-versa.

    > **(i)** **Note:** When you create a REST reference for the service, ensure to edit the **Default Host** field in the HTTP Client Resource to reflect the actual host name. By default, the **Default Host** field is set to localhost.

3. Drag and drop an operation from the Reference Binding on to the canvas.

This creates an Invoke activity which is pre-configured to invoke the operation. It also creates an HTTP Client Shared Resource with the host name and port number. The configuration for these entities is copied from the Swagger document from which you created the Reference Binding. The Reference consists of the name of the API as well as the operations it supports.

When invoking a POST or PUT method, you must provide the request string in the **Input** tab. To do so, click the column next to **item** under **postRequest** in XPath Expression and provide the request string in the drop-down box.

4. Test the configured process using the TIBCO Business Studio for BusinessWorks Debugger.

SOAP Binding

You can use SOAP binding to create service providers and service consumers, based on the SOAP protocol. TIBCO BusinessWorks Container Edition supports SOAP 1.1 and SOAP 1.2 protocol. Messages created by SOAP binding can either use HTTP or JMS transport for communication. Depending upon whether you want to create a service provider or a service consumer, there are two types of SOAP bindings: SOAP Service Binding and SOAP Reference Binding.

Terminology

TIBCO BusinessWorks™ Container Edition uses specific terminology to identify the purpose of the message, that is exchanged between a service provider and a service consumer.

Term	Description
ServiceRequest	A request message received by a service provider
ServiceReply	The response message sent by a service provider
ReferenceRequest	A request message generated by a service consumer
ReferenceReply	A response message received by a service consumer

SOAP Headers

The following table describes the terminology used for different configurations, that can be used to create a SOAP Header.

Term	Description
Described	Described denotes that the WSDL message definition used to create a SOAP header is available in the concrete WSDL.
Undescribed	Undescribed denotes that the WSDL message definition used to create a SOAP header is not available in the concrete WSDL.
Bound	Specifies that the SOAP header definition is the part of the same WSDL message, that is also used to create a SOAP body. In such cases, a WSDL message contains multiple parts. Some parts are defined to describe SOAP headers and others are defined to describe the SOAP body.
Unbound	Specifies that the SOAP header definition is not a part of the same WSDL message, that is also used to create a SOAP body

> **Note:** Based on the definitions in the SOAP Headers table, SOAP headers can be categorized into any of the following three types of configurations:
>
> - Described bound headers
> - Described unbound headers
> - Undescribed unbound headers

SOAP Attachments

Refer to the following table for SOAP attachment types and their relevant description.

SOAP Attachment Types	Description
Described Bound Attachments	In this configuration, a part of the input or output WSDL message of type `base64binary` is configured as an attachment. In a concrete WSDL, the attachment is described as a mime part of the multipart message.
Undescribed Unbound Attachments	None of the parts of the WSDL input or output message is configured to be an attachment. The concrete WSDL does not indicate whether an attachment is in the input or output message. Attachment mapping is done at the Binding Configuration under Context Mapping (Request/Response)

Generating Concrete WSDL

You can create the concrete WSDL using Concrete WSDL Generation Wizard and using web browser.

Generate Concrete WSDL Using Wizard

To generate the WSDL:

Procedure
1. Click **Generate Concrete WSDL**.
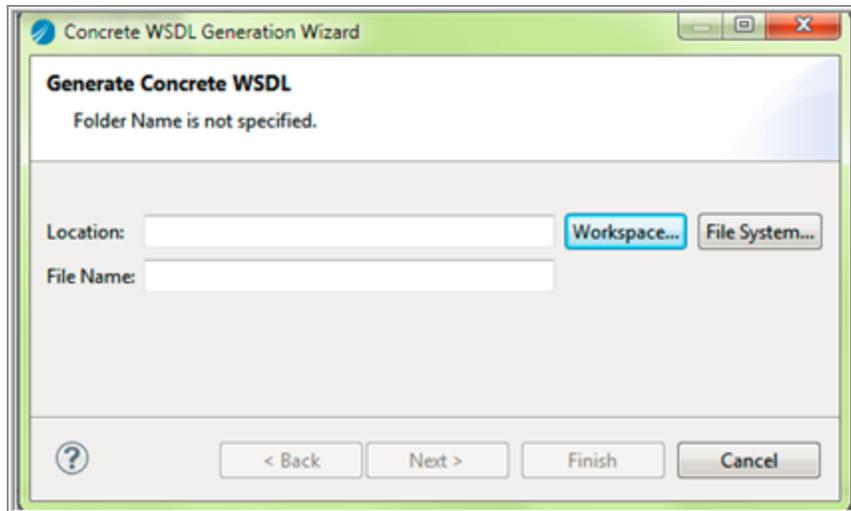
2. Click the **Workspace** tab to create the WSDL in your workspace and specify a location where WSDL is to be created.

3. To select a WSDL location:

    a. Click **Workspace** for a workspace location. Recommended: Select the **Service Descriptor** folder to contain all project-related workspace WSDLs.

    b. Click **File System** to save the concrete WSDL on a non-workspace disk location.

    c. Click **Finish** to generate a WSDL with default settings.

    The **File Name** field shows the name of the WSDL, for example `BooksService_gen.wsdl`. Override it if necessary.

4. Click **Next**.

5. Select the **Embed Abstract WSDL** and **Embed Schema** checkboxes.

    By default the **Embed Abstract WSDL** and **Embed Schema** checkboxes are selected.

    With **Embed Abstract WSDL** checkbox selected, the concrete WSDL generated has an abstract WSDL content in it.

    The namespace of the concrete WSDL is the same as that of an abstract WSDL. If the checkbox is not selected, then the namespace of the concrete WSDL is different than that of the abstract WSDL.

    When you select the **Embed Schema** checkbox, the concrete WSDL generated has the schema definition (mentioned in the abstract WSDL) in it. A schema without a target

namespace is not supported.

> **Note:** To avoid any namespace resolution error in the application module: If the concrete WSDL is placed in a workspace location (as mentioned in the earlier step), clear the **Embed Abstract WSDL** checkbox.

6. For advance configuration of the concrete WSDL such as **Namespace URI**, **Service Name**, **Host**, and **Port** (for SOAP over HTTP), click the **Advance** tab. Override them as per requirement.

7. Click **Finish**.

Generate Concrete WSDL Using Browser

The following steps are applicable only for SOAP over HTTP service.

Before you begin

SOAP service must be running.

Procedure

1. Run `lendpoints` command on console to get the list of endpoint URLs.

2. Copy the endpoint URL obtained in the browser. Append `?wsdl` string and hit enter. This generates and downloads concrete WSDL on your local disk. Since TIBCO BusinessWorks Container Edition 2.4.0 , the concrete WSDL is downloaded on your local disk. Use a text or an XML editor to view the concrete WSDL.

SOAP Service Binding

Use SOAP service binding to create a service provider.

Binding

This section has the following fields.

Field	Description
Name	The name of the service binding.
Description	A short description of the service getting exposed.
Target Namespace	The **Target Namespace** of the service.
Expose Security Context	Selecting this option exposes the transport level security context in the message header, and places the information from the user's security context (either authentication or SSL certificate information) into the Context or SecurityContext output element. **Note:** This is a fairly expensive operation and SSL certificates can consume memory resources. Select this option only if you require information from the user's security context for later use in your process.

Generate Concrete WSDL

The SOAP service binding helps you to create the concrete WSDL using Concrete WSDL Generation Wizard. For more information, see Generating Concrete WSDL.

SOAP Default Configuration

This section has the following fields.

Field	Description
SOAP Version	The version of the SOAP specification: 1.1 or 1.2. The default is 1.1.
Style	The default SOAP binding style for all operations. Specify either **Document** or **RPC** style.
Encoding	Encoding is literal.
Attachment Style	Select the default attachment style for all the binding operations. Can be overidden at the operations level for each operation.

Field	Description
	The following attachment styles are available: • SwA - Supports both, Bound and Unbound Attachments • MTOM

Persistence Configuration

This section has the following fields.

Field	Literal Value/Module Property	Description
Persistence Type	None	Select either the **NONE** or **File** (**Directory**, **Create Directories**, and **Threshold Size** fields) persistence configuration types.
Directory	Yes	This is the directory where attachments are serialized.
Create Directories	Yes	Creates new directories if not present on the disk.
Threshold Size	Yes	The maximum size of the attachment that can be kept in memory. Attachments larger than the specified size are written to the disk. The file name generated for serialization is random except when the Content-disposition header is available for the attachment. Here, the name is the same as the content-disposition header value. Configure the size to **Bytes**, **KB** or **MB**.

Threading Policy Details

The following are the details for the SOAP Service Binding threading policy.

Activity Type	Multi-Threaded?	Default Thread Count	Thread Count Configuration
Transport: HTTP			
ProcessStarter	Yes	• Minimum thread pool threads = 10 • Maximum thread pool threads = 75	It can be configured on the **HTTP Connector** shared resource using the following fields: • Minimum QTP threads • Maximum QTP threads

Add Binding

Click the **Add Binding** icon to add a binding.

Delete Binding

Click the **Delete Binding** icon to delete a binding.

Fault Operation Configuration

Fault Operation Configuration enables you to configure fault messages.

Message Part Configuration

This section has the following fields.

Field	Description
Part Name	Specifies the part name of the WSDL message.
Part Type	The body of the SOAP message.

Unbound Header Configuration

You can configure SOAP Headers in this section. The WSDL message part, other than the input WSDL message, can be configured as a header here. The WSDL messages can be in an external WSDL.

Configurations done here will appear in the Concrete WSDL in the SOAP header element.

Add Unbound Header

Use the **Add** icon [icon] to select and add this configuration. Click the icon and select the WSDL, the message and then the part.

This section has the following fields:

Field	Description
Message	Specifies the part name of the WSDL message as configured in the WSDL.
Part	You can configure the WSDL message part as any of the following: • Header - Configures the WSDL part as a SOAP Header • Body - Configures the WSDL part to SOAP Body • Attachment -Configures the WSDL part as a SwA attachment.

Delete Unbound Header

You can delete an unbound header. Select the required header and click the [icon] icon.

SOAP Reference Binding

Using SOAP reference binding you can create a service consumer using SOAP protocol.

> ℹ️ **Note:** It is recommended to use the HTTP Client Shared Resource when configuring SOAP Reference Binding. Response received for a transport communication error might be different if compared with response received using the HTTP Client shared resource.

Binding

This section has the following fields.

Field	Description
Name	The name of the reference binding.
Description	A short description of the HTTP reference binding.
WSDL Service	Specifies the service name and its target namespace.
WSDL Port	Specifies the binding implemented by the service. If more than one implementation is available in concrete WSDL, select an appropriate implementation.
SOAP Version	The version of the SOAP specification: 1.1 or 1.2.
Attachment Style	The attachment style for the incoming and outgoing messages. The following attachment styles are available: • SwA • MTOM
Enable DefaultFaultElement	Select the checkbox to enable the fault `DefaultFaultElement` on the **Invoke** activity. By default the checkbox is not selected.
Ignore mustUnderstand	Select this checkbox to ignore the value of the mustUnderstand attribute, in the incoming request, when the attribute is set to true. By default the checkbox is not selected.

Field	Description
Enable CustomFaultElement	Select this checkbox to add the `faultCode`, `faultString`, and `faultActor` attributes in the custom fault element on the **Invoke** activity. By default the checkbox is not selected.

Transport Configuration

This section has the following fields.

Field	Literal Value/Module Property	Description
Transport Type	None	The transport over which the SOAP message is sent. HTTP and JMS transports are supported.
Application for HTTP Transport		
Default Host/Port	None	Specifies the default host and port configuration available in the concrete WSDL.
HTTP Client Name	Yes	Name of the HTTP Client resource. **Note:** • Click the ⬚ icon to select a shared resource for the HTTP client. • Apache Axis2 library is upgraded. So, the SOAP Reference Binding supports Apache HttpComponents implementation library only, instead of Apache Commons library.
Endpoint URI	Yes	The endpoint of the service through which SOAP clients can access the service.

Field	Literal Value/Module Property	Description
Application for JMS Transport		
JMS Connection Name	None	Name of the JMS Connector resource
JMS Destination	Yes	Specifies the name of the destination for the incoming JMS messages. The syntax of the destination name is specific to the JMS provider you are using.
Messaging Style	None	Specifies the style of the incoming messages on the specified destination. It can be **Generic**, **Queue**, or **Topic**.
Message Type	None	Specifies the type of the incoming messages on the specified destination. It can be either **Text Message** or **Bytes Message**.
Delivery Mode	No	The delivery mode of messages. • **Persistent Messages** are stored and forwarded. • **Non-Persistent Messages** are not stored and can be lost due to failure. The default is **Persistent**.
Expiration	Yes	The length of time a message can remain active. 0 means that the message does not expire.
Priority	Yes	The priority of the message. Priority is a value from 0-9. Higher numbers signify a higher priority (that is, 9 is a higher priority than 8). The default is 4.
JMS Application	None	The user-defined JMS Properties required to be available in the incoming message.

Field	Literal Value/Module Property	Description
Properties		
JMS Application Properties (Response)	None	The user-defined JMS Properties required to be sent in the outgoing message.

Persistence Configuration

This section has the following fields.

Field	Literal Value/Module Property	Description
Persistence Type	None	Select any one from the available persistence configuration types. • None • File Selecting the **File** option displays the **Directory**, **Create Directories**, and **Threshold Size** fields.
Directory	Yes	The directory to write attachments that exceed the specified threshold size.
Create Directories	Yes	Creates new directories if not already present on the disk.
Threshold Size	Yes	The maximum size (in bytes) of an incoming message attachment that can be kept in memory. Attachments larger than the specified size are written to a file in the specified directory.

Field	Literal Value/Module Property	Description
		The file name is the output, so that subsequent activities in the process can access the file and read its contents.

Threading Policy Details

The following are the details about the **SOAP Reference Binding** threading policy.

Activity Type	Multi-Threaded?	Default Thread Count	Thread Count Configuration
Transport: HTTP			
Asynchronous	Yes	• Core pool size = 5 • Max pool size =10	You can either configure it by using the **Thread Pool** shared resource or by providing it in the `config.ini` using the following properties: • `com.tibco.bw.palette.http.requestresponse.threadpool.coresize` • `com.tibco.bw.palette.http.requestresponse.threadpool.maxsize`

Fault Operation Configuration

Fault Operation Configuration enables you to configure fault messages.

Message Part Configuration

This section has the following fields.

Field	Description
Part Name	Specifies the part name of the WSDL message.
Part Type	The body of the SOAP message.

Undescribed Header Configuration

Configure SOAP Headers in this section. WSDL message's part, other than the input WSDL message, can be configured as SOAP header here. The WSDL messages can be in an external WSDL file.

Add Undescribed Header

Use the **Add** icon to select and add this configuration. Click the icon and select the WSDL, the message, and then the part.

This section has the following fields:

Field	Description
Header Name	Specifies the message name.
Header Part	Specifies the message part.
Cardinality	Configure the cardinality of the configured SOAP header as one of the following: • Optional(?) - The SOAP header is optional.

Field	Description
	• Required - The Soap Header is required and must be supplied when the process is called.
	• Repeating (*) - Zero or more SOAP headers must be supplied are present when tha process is called.

Delete Undescribed Header

You can delete an undescribed header. Select the required header and click the icon.

Basic Activities Palette

Basic activities are a simple means of interacting with a service, manipulating the passing data, or handling exceptions. They also contain Loops that are Groups with Conditions, which follow a pattern at run time such as, initialize the loop, update the loop at each iteration, and test conditions for the loop to stop iterating.

The available types of loops are:

- Local Transaction
- For Each
- Iterate
- Repeat
- Repeat On Error
- Scope
- While

Critical Section

Critical Section groups are used to synchronize process instances. At run time, multiple process instances can be executed concurrently. You can use **Critical Section** groups to synchronize the set of actions performed in the process across multiple process instances. Only one process instance can execute the **Critical Section** group and its contents at any specified time. Other concurrently running process instances that are associated with the corresponding **Critical Section** group wait at the start of the **Critical Section** group, until the process instance that is currently executing complete the **Critical Section** group.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the group in the process.
Group Type	The type of the group. The default is **Critical Section**.
Timeout (msec)	The maximum time (in milliseconds) that a process instance waits at the start of the critical section group while another process instance is executing the critical section. When a timeout occurs, the process instance that is waiting to execute the critical section fails with an Activity Timeout Exception. The timeout field can be used to avoid deadlock situations.
Shared Lock	Select this checkbox to synchronize multiple critical section groups that use the same shared variable as a lock.
Shared Variable Type	This field appears only when the **Shared Lock** checkbox is selected. The **Shared Lock** option can be defined using a **Module Shared Variable** or a **Job Shared Variable**. When a module shared variable is used against job shared variable to define **Shared Lock**, it has different implications. • **Module Shared Variable**: provides a shared lock to synchronize process instances from different processes that are within a module. Using a module shared variable, you can synchronize multiple **Critical Section** groups that can be part of different process instances. These **Critical Section** groups can be in the same process or in another process, provided all processes are contained within a module. • **Job Shared Variable**: provides a shared lock to synchronize within a job. Using a job shared variable, you can synchronize multiple **Critical Section** groups that are part of the same job. These **Critical Section** groups can be in the same process or in an another process, provided all process instances are part of the same job.
Shared Variable	This field appears only when the **Shared Lock** checkbox is selected. Name of the **Module Shared Variable** or **Job Shared Variable**.

Description

Provide a short description for the Critical Section group.

Variables

The **Variables** tab provides a way to declare variables that are only limited to the scope of this group.

You can select from the following available **Variable Type** options:

- String

- Integer

- Decimal

- Boolean

- Date&Time

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
ActivityTimedOutException	A timeout has been reached.

Local Transaction

Use **Local Transaction** group for multiple activities to be part of a local transaction. A local transaction group ensures that all participants of the transaction are rolled back together. A single resource performs all its transaction work.

- JMS

- JDBC

A **Local Transaction** group has to be further configured as a specific type of a **Local Transaction** or **Transaction Transport**. The **Transaction Transport** type determines the activities that can participate in this local transaction group. For example, "JDBC" transaction transport only provides transaction support for certain JDBC activities.

If a **Call Process** activity is used within the local transaction group, then the direct subprocesses called are also part of the local transaction.

For more information about **Call Process** activity, see Call Process.

General

The **General** tab has the following fields.

Name	Description
Name	The name to be displayed as the label for the activity in the process.
Transaction Transport	The transport type. Select from the following available transaction transport types: • **None** • **JMS**: To configure a JMS local transaction, select **JMS** as the transaction type of the group. • **JDBC**: To configure a JDBC transaction, select **JDBC** as the transaction type of the group.

JMS

JMS activities participate in a transaction using the JMS local transaction. A JMS session, when specified as transacted, supports a single series of JMS operations. The local transaction group enlists a set of produced messages and a set of consumed messages into a unit of work. JMS local transactions use the underlying JMS sessions to group together all the activities that are part of the transaction group into one unit of work.

All activities using the same JMS connection are part of one transaction. If a transaction group contains activities that use more than one JMS connection, for example, a **JMS Send Message** activity using connection 1 and another **JMS Send Message** activity using connection 2, there are two underlying transactions created.

Furthermore, activities must use the same messaging styles to be in the same transaction.

Possible deadlock situation: A deadlock situation arises if you have **JMS Send Message** and **Get JMS Queue Message** activities performing JMS operations on the same queue. A message not sent out until a commit is performed, keeps **Get JMS Queue Message** activity waiting indefinitely.

> ❗ **Important:** It is recommended not to have a design that sends and receives messages on the same destination using the same session in a single transaction.

When a transaction commits, its input is acknowledged and the associated output is sent. When a transaction rollback is done, all the produced messages (output stream) are destroyed and the consumed messages (input stream) are automatically recovered. You can use the JMS sender activities with the JMS local transaction. At run time, the underlying JMS activities use the same transacted JMS session to provide transaction semantics for messages sent and received by the JMS activities.

For more information about JMS palette and activities, see JMS Palette.

JDBC

Using the JDBC transaction and multiple JDBC activities, you can access the same database connection to participate in a transaction. Only JDBC activities that use the same JDBC Connection participate in this transaction type. However, other activities can be part of the transaction group. If the transaction commits, all JDBC activities using the same JDBC connection in the transaction group commit. If the transaction rolls back, all JDBC activities using the same JDBC connection in the transaction group roll back.

The transaction group commits automatically, if all activities in the group complete the transaction and a non-error transition is taken out of the transaction group. If any error occurs while processing the activities in the group, including errors in non-JDBC activities, it rolls back the transaction and returns an error. Individual JDBC activities can override the default transaction behavior and commit separately.

For more information about JDBC palette and activities, see JDBC Palette.

For Each

The **For Each** group is a loop element that can be used for iterating through array elements. The **For Each** activity is very useful to determine these array elements

dynamically at run time.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Group Type	The type of group. The default is **For Each**.
Counter Name	The name of the counter or Index variable. **Note:** Select the **Global** check box to make the index variable available for the activities outside the loop.
Start Counter Value	The start value for the counter ($index) which is specified through an XPath expression. This can be a literal value.
Final Counter Value	The final value of the counter which is set using an XPath expression. This value is obtained by the count of the number of elements of a specific type in the request message.
Accumulate Output	Select this check box to accumulate the output of one of the activities in a group. The output of the selected activity accumulates each time the activity is executed.
Output Activity	Select the activity from the group of activities for an output reference when executing the loop.
Output Name	The name of the list of the accumulated output of the selected activity.
Reset Variables	The variable to be reset during each iteration of the **For Each** group. Click the **Add** icon, and select the variable to be reset. To delete the variable, click the variable in the **Reset Variables** field. The

Field	Description
Remove	**Remove** icon with a tooltip 'Remove *<variable name>*' is displayed. Click the **Remove** icon to remove the variable from the **Reset Variables** field.

Description

Provide a short description for the For Each activity.

Variables

A group of shared variables. Depending on the group selected in **Group Type**, **Container** displays the group. For example, if the **For Each** group is selected in the **Group Type** field, it displays **For Each** in **Container**. You can add multiple complex or simple types variables.

You can select from the following available **Variable Type** options:

- String

- Integer

- Decimal

- Boolean

- Date&Time

Iterate

An **Iterate** group repeats the series of grouped activities once for every item in an existing sequence or list. The list can be items of any datatype. The loop executes for the number of iterations specified.
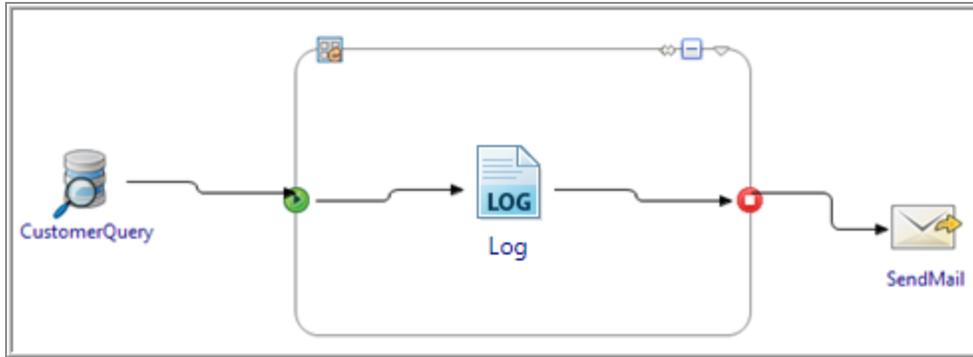
This process performs the following operations:

1. A **JDBC Query** activity (CustomerQuery) is used to query a database and populate a list of customer records. The customer records are then passed to a group containing the **Log** activity.

2. The **Log** activity writes the name and address of each customer to a log and appends to the log as each record is written.

3. The group iterates once for every customer record returned by the **CustomerQuery** activity.

4. The process then reads the log that was written for its data to be available to the process, and then transitions to a **SendMail** activity to send the contents of the customer list through email.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Group Type	The type of group. The default is **Iterate**.
Index Name	The index name.
	Note: Select the **Global** checkbox to make the index variable available for the activities outside the loop.

Field	Description
Variable List	This field is an XPath expression. XPath is used to specify the schema element you want to refer to. Click the **Edit XPath Expression** icon to specify a value for this field in the XPath Builder wizard. You can use a simple expression containing a complete list, or a more complex one and only process certain items in the list.
Iteration Element	Provide a name for a process variable containing the current iteration element in this field. **Note:** Select the **Global** check box to make the **Iteration Element** available for the activities outside the loop. For more information, see Call Process
Accumulate Output	Select this check box to accumulate the output of one of the activities in a group. The output of the selected activity is accumulated each time the activity is executed.
Output Activity	Select the activity from the group of activities for an output reference when executing the loop.
Output Name	The name of the list of the accumulated output of the selected activity.
Reset Variables	The variable to be reset during each iteration of the **Iterate** group. Click the **Add** icon, and select the variable to be reset. To delete the variable, click the variable in the **Reset Variables** field. The **Remove** icon with a tooltip 'Remove *<variable name>*' is displayed. Click the **Remove** icon to remove the variable from the **Reset Variables** field.

Description

Provide a short description for the Iterate activity.

Variables

A group of shared variables. Depending on the group selected in **Group Type**, **Container** displays the group. For example, if the **Iterate** group is selected in the **Group Type** field, it displays **Iterate** in **Container**. You can add multiple variables of complex or simple type.

You can select from the following available **Variable Type** options:

- String

- Integer

- Decimal

- Boolean

- Date&Time

Repeat

The **Repeat** group activity is responsible for repeating the series of grouped activities until the specified condition evaluates to true. The activities are always executed once before checking whether the condition is `true`. After executing the series of activities, the initial evaluation of the condition is done, and it does not repeat when the condition evaluates as `true`. The **Repeat** activity evaluates the loop's conditional statement to determine whether or not more iterations of the loop are required.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Group Type	The type of group. The default is **Repeat**.
Index Name	Use this field to specify a name for the index variable.

Field	Description
	Note: Select the **Global** checkbox to make the index variable available for the activities outside the loop.
Condition	The condition (`$index`) is specified through an XPath expression which can be a literal value. Click the **Edit XPath Expression** button to set an XPath expression.
Accumulate Output	Select this checkbox to accumulate the output of one of the activities in a group. The output of the selected activity is accumulated each time the activity is executed.
Output Activity	Select the activity from the group of activities for an output reference when executing the loop.
Output Name	The name of the list of the accumulated output of the selected activity.
Reset Variables	The variable to be reset during each iteration of the **Repeat** group. Click the **Add** icon, and select the variable to be reset. To delete the variable, click the variable in the **Reset Variables** field. The **Remove** icon with a tooltip 'Remove *<variable name>*' is displayed. Click the **Remove** icon to remove the variable from the **Reset Variables** field.

Description

Provide a short description for the Repeat group.

Variables

A group of shared variables. The **Container** displays the group. This depends on the group selected in **Group Type**. For example, if the **Repeat** group is selected in the **Group Type** field, it displays **Repeat** in **Container**. You can add multiple variables of complex or simple type.

You can select from the following available **Variable Type** options:

- String

- Integer

- Decimal

- Boolean

- Date&Time

Repeat on Error

Using the **Repeat on Error** group, you can repeat a series of activities when an unmanageable error occurs. The activities in this group are executed once. If there are no unmanageable errors, the loop terminates. If an error occurs for which there is no error transition, the condition of the loop is evaluated. If the condition is `true`, the loop terminates. If the condition is `false`, the loop repeats until there is no error or the condition is `true`.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Group Type	The type of group. The default is **Repeat on Error**.
Index Name	Use this field to specify a name for the index variable.
Stop Repeat Condition	The condition specifies that the activities inside the **Repeat On Error** are performed repeatedly until the condition evaluates to `false`.

Description

Provide a short description for the Repeat on Error group.

Variables

A group of shared variables. Depending on the group selected in the **Group Type**, the **Container** displays the group. For example, if **Repeat on Error** group is selected in the **Group Type** field, it displays **RepeatOnError** in **Container**. You can add multiple variables of complex or simple type.

You can select from the following available **Variable Type** options:

- String

- Integer

- Decimal

- Boolean

- Date&Time

Scope

The **Scope** group comprises a collection of nested activities. Scope group is shared by all the nested activities. You can use a Scope to contain a unit of work, by making it easy to manage, execute, and revert. For example, if a customer cancels an online order, the money must be returned and the online order canceled, without affecting other orders. You can use a Scope to manage these activities as a unit.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Group Type	The type of group. The default is **Scope**.

Description

Provide a short description for the Scope group.

Variables

A group of shared variables. Depending on the group selected in the **Group Type**, the **Container** displays the group. For example, if the **Scope** group is selected in the **Group Type** field, it displays **Scope** in **Container**. You can add multiple variables of complex or simple type.

You can select from the following available Variable Type options:

- String
- Integer
- Decimal
- Boolean
- Date&Time

While

The **While** activity repeats the series of grouped activities if the specified condition evaluates as `true`. The condition is evaluated when the group is enters. If the condition evaluates to `false`, the activities within the group are not executed.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Group Type	The type of group. The default is **While**.
Index Name	Use this field to specify a name for the index variable. **Note:** Select the **Global** checkbox to make the index variable available for the activities outside the loop.

Field	Description
Condition	The condition ($index) is specified through an XPath expression which can be a literal value. Click the **Edit XPath Expression** button to set an XPath expression.
Accumulate Output	Select this checkbox to accumulate the output of one of the activities in a group. The output of the selected activity is accumulated each time the activity is executed.
Output Activity	Select the activity from the group of activities for an output reference when executing the loop.
Output Name	The name of the list of the accumulated output of the selected activity.
Reset Variables	The variable to be reset during each iteration of the **While** group. Click the **Add** icon, and select the variable to be reset. To delete the variable, click the variable in the **Reset Variables** field. The **Remove** icon with a tooltip 'Remove *<variable name>*' is displayed. Click the **Remove** icon to remove the variable from the **Reset Variables** field.

Description

Provide a short description for the While group.

Variables

A group of shared variables. Depending on the group selected in the **Group Type**, **Container** displays the group. For example, if the **Scope** group is selected in the **Group Type** field, it displays **Scope** in **Container**. You can add multiple variables of complex or simple type.

You can select from the following available **Variable Type** options:

- String
- Integer

- Decimal

- Boolean

- Date&Time

Constructor

You must use the constructor for each class to instantiate that class.

Compensate

Compensate activity is executed only from the compensation handlers when a fault is encountered outside a scope (this scope has compensation handler defined). This activity triggers compensation handler for a scope present only on the same level. All activities present in scope get executed successfully.

The compensation handlers are defined per scope. The compensation handlers of scopes that are directly enclosed by the scope and contain **Compensate** activity, can be executed. The handlers execute in reverse order, so the handler of the last completed scope is executed first. The compensation handlers can only be defined on scope level.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.

Description

Provide a short description for the Compensate activity.

Empty

An **Empty** activity denotes an activity with no action to be performed. You can specify the **Name** and **Description**, but there is no input or output for this activity.

This activity is useful if you want to join multiple transition flows. For example, there are multiple transitions out of an activity and each transition takes a different path in the process. In this scenario you can create a transition from the activity at the end of each path to an **Empty** activity to resume a single flow of execution in the process.

Exit

Exit activity is a synchronous activity that cancels any asynchronous activities that are in a waiting state and terminate the process instance. You can specify the **Name** and **Description**, but there is no input or output for this activity.

Get Context

The **Get Context** activity retrieves the value of the specified context parameter. This is useful if your process requires some context information from a request or a response.

The **Get Context** activity must be placed after the **Invoke** or the **Receive** activity.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Use Shared Context	Select the checkbox to use a job shared variable to share the context.
Shared Context	This field is displayed when the **Use Shared Context** field is selected. Select the **Go to** icon to locate the required job shared variable.

Field	Description
Direction	This field is displayed when the **Use Shared Context** field is clear. Select from the following available options to specify from where the context information should be fetched:
	• **Receive Service**: pulls context information from a service request. Selecting this option displays the **Service** field.
	• **Return from Reference**: pulls context information from a reference response. Selecting this option displays the **Reference** field.
	• **Fault from Reference**: pulls context information from a reference fault response.
Service/Reference	Select the targeted portType and operation from where the context information is to be picked.
Fault	This field is displayed when **Fault from Reference** option is selected in the **Direction** field.

Description

Provide a short description for the Get Context activity.

Output

The following is the output for the activity.

Output Item	Datatype	Description
<schema>	varies	The value of the context resource specified in the **Direction** field on the **General** tab is the output for this activity. The schema specified for the context resource determines the schema for the output.

Invoke

Invoke activity is an asynchronous activity that calls an operation of a process reference. The process reference can be configured to invoke another process (subprocess) or invoke an external service through a binding such as SOAP.

> ℹ **Note:** The **Call Process** activity serves a similar purpose as the **Invoke** activity. The **Call Process** activity only calls non-WSDL-based subprocesses, or direct subprocesses, and the **Invoke** activity only calls WSDL-based subprocesses, or service subprocesses.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Service	Select the service for which to specify an operation.

Description

Provide a short description for the Invoke activity.

Advanced

The **Advanced** tab has the following field.

Input item	Literal Value/Module Property	Description
Activity Timeout (seconds)	Yes	The **Invoke** activity timeout in seconds. You can either select a preset timeout (Default Timeout, 30, 60,or 120 seconds) or specify a custom timeout in seconds.

Input item	Literal Value/Module Property	Description
		Important: If the Invoke activity is configured to be **Default Timeout**, then the timeout value equates to "3" minutes by default. However this default value (3 minutes) can be modified to be a different value by setting the property `bw.engine.activity.async.waitTime` in `BW_JAVA_OPTS` environment variable. The value for the property should be specified in milliseconds. For example: – `Dbw.engine.activity.async.waitTime=<timeout in milliseconds>` **Note:** When the **Invoke** activity timeouts, the in-line subprocess instance called by the **Invoke** activity can be cancelled before the subprocess instance completes. Hence, the business logic in the cancelled process instance may not be executed to its entirety.

Input

The following is the input for the activity.

Input item	Datatype	Description
input message	complex	The input message of the operation. This element contains all the input message parts for the operation.

Output

The following is the output for the activity.

Output item	Datatype	Description
output message	complex	The output message for the operation. This element contains all the output message parts for the operation.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
ActivityTimedOutException	A timeout has been reached.

> **Note:** When used to invoke REST service with REST reference binding, it contains additional elements such as clientFault and serverFault.

Invoke for SOAP with HTTP

When used to invoke SOAP service, it displays the faults defined in the WSDL file along with the following exceptions:

Fault	Generated When..
HTTPClientException	The HTTP server responds with a message containing the 4XX status code.
HTTPServerException	The HTTP server responds with a message containing the 5XX status code.
HTTPCommunicationException	An HTTP exception occurred while executing the specified method, or when trying to read the response.

To send HTTP responses logging on the wire, set the following logger in the `logback.xml` file:

```
<logger name="org.apache.http">
<level value="DEBUG"/>
</logger>
```

Rethrow

Rethrow activity is used within an exception-handling routine to throw the caught error again. This is useful if you want to perform some processing within an error-handling routine, and also send it to the next higher scope.

Receive

Receive activity initiates the process execution based on the receipt of a request message for an operation of the process service. The **Receive** activity can be configured to act as a process starter or a signal-in activity.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Create Instance	Select this checkbox if you want to start a process.
Service	Select the process service and the operation from which to receive the request.

Description

Provide a short description for the Receive activity.

Output Editor

The schema of the process variable must match the schema of the output of the activity shown on the **Output** tab.

Output

Output Item	Datatype	Description
output	varies	The output of this activity is the incoming message from the calling client or service. The structure of the message is determined by the schema of the input message for the selected operation on the **General** tab.

Reply

Reply activity is a synchronous activity that sends a message in response to a request message that was received for an operation of a process service. This activity can be used to send a reply or a fault.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity.
Service	Select the process and the operations for which to send a response.

Field	Description
Reply With	Select from the following available options to send the reply message: • Output Message • Undeclared Fault - This field does not appear when the value in the **Service** field is not provided.

Description

Provide a short description for the Reply activity.

Input Editor

The schema of the process variable must match the schema of the input for the activity shown on the **Input** tab.

Input

Input Item	Datatype	Description
input	varies	The reply message to send. The schema depends on the output message of the operation for this reply.

Set Context

The **Set Context** activity can be used to set the value of a context information resource.

The **Set Context** activity must be placed before the **Invoke** or the **Receive** activity.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Direction	Select from the following available options to specify where the context information should be supplied: • **Invoke Reference**: Selecting this option displays the **Reference** tab. • **Reply to Service**: sends context information to a service. Selecting this option displays the **Service** field. • **Reply to Service with Fault**: sends context information for a service fault reply.
Use Shared Context	Select the check box to use a job shared variable to share the context.
Shared Context	This field is displayed when the **Use Shared Context** field is selected. Select the **Go to** icon to locate the required job shared variable.
Service/Reference	Select the targeted portType and operation where you want to supply the context information.
Fault	This field is displayed when the **Reply to Service with Fault** option is selected in the **Direction** field.

Description

Provide a short description for the activity.

Input

The following is the input for the Set Context activity.

Input Item	Datatype	Description
input message	complex	The input message of the operation. This element contains all input message parts for the operation.

Output

Output Item	Datatype	Description
<schema>	varies	The value of the context resource specified in the **Direction** field on the **General** tab is the output for this activity. The schema specified for the context resource determines the schema for the output.

Set EPR

Set EPR activity is a dynamic alternative to the static service element defined in the WSDL.This activity determines which service to be invoked in an application.To use an incoming message, define the EPR schema as a part of the message in the WSDL.

An endpoint reference (EPR) combines web services elements that define the address for a resource in a SOAP header. If you use an incoming message, an EPR schema should be defined as a part of the message in WSDL.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity.
Dynamic Reference	The name of the dynamic target interface for which this task is supplying the service name and namespace

> ℹ️ **Note:** The service name must be unique for every application.

Description

Provide a short description for the Set EPR activity.

Input

The following is the input for the activity.

Input Item	Description
SimpleEndpointReference	Contains endpoint address information for the destination of the outgoing message

Output

The following is the output for the activity.

Output Item	Description
EndpointReference	Contains the following elements: • Address: Endpoint address • Interface: The name of the dynamic target interface for which this task is supplying the service name and namespace

Throw

The **Throw** activity throws the specified fault and the control is passed to any error-handling routine defined to handle the error. This activity is useful in a group or in a called process. You can use this activity to catch and raise your own error conditions. The **Throw** activity is useful if an error in the business logic occurs during processing.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Default Fault	This field is displayed when you use the **Throw** activity in a service process or a service subprocess. Select this check box if you want the default message text and the error code for the error message. This is reflected in the **Input Editor** and **Input** for this activity.
Fault Schema	This field is displayed when you use the **Throw** activity in a direct subprocess. Select a fault schema to send to the parent process. In a direct subprocess, fault schemas are defined on the **Fault Editor** tab of the **End** activity. If you select the **- Default -** option, the $_error process variable is propagated.

Description

Provide a short description for the Throw activity.

Input Editor

Selecting the **Default Fault** check box displays the **message** and the **msgCode** elements in the **Input Editor**.

Input

The following is the input for the activity.

Input Item	Datatype	Description
message	string	The text of the generated error message.
msgCode	string	The error code of the error message. This is used as an identifier for the error for the applications to test for an error.

Fault

This activity does not produce any fault. Click on this tab to view fault schemas specified on the **Input** tab.

Start

The **Start** activity is the first activity, and acts as the starter activity, in a direct subprocess definition. A direct subprocess can be called from a parent process, and the **Start** activity is used to define the input expected by the process.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.

Description

Provide a short description for the Start activity.

Output Editor

Use the **Output Editor** tab to define data for the subprocess to use as input. Any process that calls this process definition must supply the data specified on the **Output Editor** tab. You can define your own datatype on this tab, and can reference an XML schema stored in the project. Once defined, the data specified on the **Output Editor** tab becomes the output schema of the **Start** activity. This data then becomes available to other activities in the process definition.

> ✓ **Tip:** From the Select Schema Element Declaration window, select **Include WSLD Inline Schemas** to display inline schemas from the WSDLs in the module.

Output

The output for the activity is defined by the specified data elements on the **Output Editor** tab.

End

The **End** activity is the last activity in a direct subprocess definition. When a direct subprocess is called from another process, the direct subprocess provides output data to the calling process. You can map data from other activities in the process to an output schema specified on the **End** activity. This becomes the output of the process.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.

Description

Provide a short description for the End activity.

Input Editor

Use the **Input Editor** tab to define ouput data for the subprocess. Any process that calls this process definition receives this data after the process call completes.

You can define your own datatype on this tab, and reference an XML schema stored in the project. Once defined, the data specified on the **Input Editor** tab becomes the input schema of the **End** activity. You can then map data from other activities in the process to the **End** activity's input, and this becomes the output of the process when the process completes.

> ✓ **Tip:** From the Select Schema Element Declaration window, select **Include WSLD Inline Schemas** to display inline schemas from the WSDLs in the module.

Input

The input for the activity is defined by the specified data elements on the **Input** tab.

Fault Editor

Use the **Fault Editor** tab to specify possible faults for the direct subprocess to throw.

Engine Command

The **Engine Command** activity allows you to retrieve statistics and information about process definitions, process instances, and activities for the application and the appnode that is executing the command. This activity also lets you perform job maintenance, such as suspending and resuming the process instances.

Some commands require memory and processor overhead for gathering statistics for retrieving information on the current state of the process. Because of the performance implications, certain instrumentation is disabled by default.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as label of the activity in the process definition.
Command To Execute	The command to execute.
Force	Select this checkbox to stop an application forcefully.
Process Definition Name	Certain commands accept a process definition name. This field allows you to specify the process definition name that applies to the selected command. This field only appears when the one of the following commands is selected in the Command to Execute field: GetActivityStats

Field	Description

- GetProcessDefinitionStats
- GetProcessInstanceInfo
- ResumeProcessStarter
- SuspendProcessStarter

> **Note:** If the Process Definition name is not defined in either the **General** tab or in the **Input** field, the command is executed for all processes in the application.

Description

Provide a short description of the End Command activity.

Input

The input for the activity is the following.

Field	Description
Name	The input for this activity varies depending upon the command selected in the Command to Execute field on the Configuration tab. For more information about the input for each command, see Commands.

Output

The output for the activity is the following.

Field	Description
Name	The output for this activity varies depending upon the command selected in the Command to Execute field on the Configuration tab. For more information about the input for each command, see Commands.

Commands

The following commands can be executed by the Engine Command activity.

GetActivityStats

Retrieves information about the activities that have been executed for a given process definition from the time the process instrumentation statistics have been turned on. The process instrumentation statistics can be turned on in any of the following ways:

- using the StartStatsCollection command
- from the command line using the **enablestats** command

The activity information is cumulative. A single activity name represents all executions of that activity.

The ExecutionTime computation for the **Call Process** activity includes the sum of the execution times for all activities in the called process, not just the execution time for the **Call Process** activity itself.

> **Note:** If the **Process Definition Name** is not defined in either the **General** tab or in the **Input** field, the command is executed for all processes in the application.

Input Item	Datatype	Description
ProcessDefinitionName	string	Name of the process definition. Specifying a value for this element overrides the process definition specified on the **General** tab.

Output Item	Datatype	Description
ProcessDefinitionName	string	Name of the process definition.
ActivityName	string	Name of the activity.

Output Item	Datatype	Description
ExecutionCount	long	Number of times the activity has been executed.
ElapsedTime	long	Elapsed time of an activity is the time difference (in milliseconds) between start time and end time of the activity. Between the start and end time, control may get switched with other activities from the other jobs. This is the time taken to execute an activity plus all the delays in acquiring resources like engine threads, JDBC connections, network, and so on. The elapsed time includes the execution time plus time taken for evaluating all the forward transitions from that particular activity and getting the next activity ready to execute, which includes executing its input mapping if all dependencies are met.
ExecutionTime	long	The Execution Time for an activity is the actual time (in milliseconds) required by the activity to complete while using the engine thread.

Output Item	Datatype	Description
		Asynchronous activities may use other threads not included in this time.
ErrorCount	long	Total number of executions of the activity that have returned an error.
LastReturnCode	string	Status code returned by most recent execution of this activity.
MinElapsedTime	long	Elapsed clock time (in milliseconds) of the activity execution that has completed in the shortest amount of elapsed time.
MaxElapsedTime	long	Elapsed clock time (in milliseconds) of the activity execution that has completed in the longest amount of elapsed time.
MinExecutionTime	long	Execution time (in milliseconds) of the activity execution that has completed in the shortest amount of execution time.
MaxExecutionTime	long	Execution time (in milliseconds) of the activity execution that has completed in the longest amount of execution time.

Output Item	Datatype	Description
MostRecentElapsedTime	long	Elapsed clock time (in milliseconds) of the most recently completed activity execution.
MostRecentExecutionTime	long	Execution time (in milliseconds) of the most recently completed activity execution.
ExecutionCountSinceReset	long	Number of times the activity has been executed since the last reset of the statistics.

GetProcessDefinitionStats

Retrieves information about process definitions. The process instrumentation statistics can be turned on in any of the following ways:

- using the StartStatsCollection command

- from the command line using the **enablestats** command

> **Note:** If the **Process Definition Name** is not defined in either the **General** tab or in the **Input** field, the command is executed for all processes in the application.

Input Item	Datatype	Description
ProcessDefinitionName	string	Name of the process definition. Specifying a value for this element overrides the process definition specified on the **General** tab.

Output Item	Datatype	Description
ProcessDefinitionName	string	Name of the process definition.
NumberCreated	long	Number of process instances created for this process definition.
NumberSuspended	long	Number of times process instances have been suspended.
NumberAborted	long	Number of times process instances have been aborted.
NumberCompleted	long	Number of process instances that have successfully completed.
TotalExecutionTime	long	Total execution time (in milliseconds) for all successfully completed process instances.
AverageExecutionTime	decimal	Average execution time (in milliseconds) for all successfully completed process instances.
TotalElapsedTime	long	Total elapsed time (in milliseconds) for all successfully completed process instances.
AverageElapsedTime	decimal	Average elapsed clock time (in milliseconds) for all successfully completed process instances.
MinElapsedTime	long	Elapsed clock time (in milliseconds) of the process instance that has completed in the shortest amount of elapsed time.
MaxElapsedTime	long	Elapsed clock time (in milliseconds) of the process instance that has completed in the longest amount of elapsed time.
MinExecutionTime	long	Execution time (in milliseconds) of the

Output Item	Datatype	Description
		process instance that has completed in the shortest amount of execution time.
MaxExecutionTime	long	Execution time (in milliseconds) of the process instance that has completed in the longest amount of execution time.
MostRecentExecutionTime	long	Execution time (in milliseconds) of the most recently completed process instance.
MostRecentElapsedTime	long	Elapsed clock time (in milliseconds) of the most recently completed process instance.
NumberCompletedSinceReset	long	Number of process instances that have completed since the last reset of the statistics.

GetProcessInstanceInfo

The values of the input elements for this command specify for which process instances you want to retrieve information. If you specify more than one input element, the values are treated as an AND condition. For example, If you specify the process definition name and the minimum duration, information for process instances for the specified process definition that meet the minimum duration time is returned.

Note: If the Process Definition Name is not defined in either the **General** tab or in the **Input** field, the command is executed for all processes in the application.

Input Item	Datatype	Description
ProcessId	string	ID for the process instance.
ProcessDefinitionName	string	Name of the process definition. Specifying a value for this element overrides the process definition

Input Item	Datatype	Description
		specified on the **General** tab.
MinimumDuration	string	Minimum time (in milliseconds) in elapsed clock time since the process instance started. All process instances with greater elapsed times are retrieved.

Output Item	Datatype	Description
ProcessId	string	ID for the process instance.
ProcessDefinitionName	string	Name of the process definition.
Status	string	Status of the process.
StartTime	string	Time (in milliseconds) when the process instance started.
ElapsedTimeSinceStarted	long	Elapsed clock time (in milliseconds) since the process instance started.
MainProcessName	string	Name of the main process definition. If the process is a regular process, or a spawned subprocess, the name of the process is returned. If the process is an in-line subprocess, the name of the root process that initiated the subprocess call is returned.
ParentProcessName	string	Name of the parent process. If the process is a regular process, or a spawned subprocess, the parent process name is null. If the process is an inline-subprocess, the name of the immediate parent process is returned.
SubProcessName	string	Name of the subprocess. If the process is a regular process, or a spawned subprocess, the subprocess name is null. If the process is an

Output Item	Datatype	Description
		inline-subprocess, the name of the subprocess is returned.

GetProcessStarterStats

Retrieves information about either active or inactive process starters. The information is cumulative. A single process starter name represents all executions of that process starter.

Output Item	Datatype	Description
ProcessName	String	Name of the process definition.
NumberCreated	integer	Number of process instances created by this process starter.
NumberCompleted	integer	Number of process instances that have completed.
NumberRunning	integer	Number of process instances currently executing.
NumberCreatedPerHour	integer	Number of process instances per hour created by this process starter.
StartTime	String	Time at which the process starter was started.
ElapsedTimeSinceStarted	integer	Elapsed clock time since the process starter was started.
ProcessStarterName	String	Name of the process starter. It is of the form `processName+ProcessStarterName`
Status	String	Status of the process starter. The status can be CANNOT_START, STARTED, or STOPPED.
RestartedFromCheckpoint	boolean	True if the process was restarted from a checkpoint.

GetRecoverableProcesses

Retrieves the process instances that can be recovered. Use the returned process ID in the RestartRecoverableProcess or RemoveRecoverableProces commands.

Output Item	Datatype	Description
ProcessId	string	ID for the process instance.
ProcessDefinitionName	string	Name of the process definition.
Status	string	Status of the process.

KillProcessInstance

Kills the specified process instance. The process instance is stopped immediately and permanently removed from the engine.

Input Item	Datatype	Description
ProcessId	string	Process ID of the process instance you want to kill.

RemoveRecoverableProcess

Removes the specified recoverable process instance from the list of potential recoverable processes. After executing this command, the checkpoint data of the specified process instance is removed and the process instance can no longer be recovered.

Input Item	Datatype	Description
ProcessId	string	Process ID of the checkpointed process instance you want to remove. Obtain the process ID of the recoverable process with the GetRecoverableProcesses command.

RestartRecoverableProcess

Restarts the specified recoverable process instance.

Input Item	Datatype	Description
ProcessId	string	Process ID of the checkpointed process instance you want to restart. Obtain the process ID of the recoverable process with the GetRecoverableProcesses command.

ResumeProcessInstance

Resumes the specified process instance.

Input Item	Datatype	Description
ProcessId	string	Process ID of the process instance you want to resume.

ResumeProcessStarter

Resumes the specified process starter.

> **Note:** If the **Process Definition Name** is not defined in either the **General** tab or in the **Input** field, the command is executed for all process starters in the application.

Input Item	Datatype	Description
ProcessDefinitionName	string	The name of the process definition whose process starter you want to resume. Specifying a value for this element overrides the process definition specified on the **General** tab.

StartStatsCollector

Enables process instrumentation statistics for the application in the current appnode.

StopStatsCollector

Disables process instrumentation statistics for the application in the current appnode.

StopApplication

Gracefully stops the application.

SuspendProcessInstance

Suspends the specified process instance.

Input Item	Datatype	Description
ProcessId	string	Process ID of the process instance you want to suspend.

SuspendProcessStarter

Suspends the specified process starter.

> **ⓘ** **Note:** If the **Process Definition Name** is not defined in either the **General** tab or in the **Input** field, the command is executed for all process starters in the application.

Input Item	Datatype	Description
ProcessDefinitionName	string	The name of the process definition whose process starter you want to suspend. Specifying a value for this element overrides the process definition specified on the **General** tab.

File Palette

The File palette is used to read, write, delete, or create files. You can use the process starter present in this palette to poll for files and start a process based on the available file.

Copy File

The **Copy File** is a synchronous activity that you can use to copy files and directories to a new location.

General

The **General** tab has the following fields.

Field	Literal Value/ Process Property/Module Property	Description
Name	None	The name to be displayed as the label of the activity in the process.
From Filename	Yes	The path and name of the file or directory to copy. For directories, you must specify an absolute path. You may use wildcards when specifying files within a directory. For example, to copy a directory, specify `c:\myDirectory`. To copy all text files in a directory, specify `c:\myDirectory\*.txt`.
To Filename	Yes	The destination for the copy operation. This must be an absolute path.

Field	Literal Value/ Process Property/Module Property	Description
		Do not use wildcard characters in this element.
Overwrite	Literal Value	Select this checkbox to overwrite the existing file with the same name, if it exists.
		The activity raises an exception if the specified file or directory exists and this checkbox is not selected.
Create Non-Existing Directories	Literal Value	When this checkbox is selected, the activity creates all directories in the specified path, if they do not already exist.
		If this checkbox is not selected with non-existing one or more directories in the specified path, it displays an exception.
Include Sub-Directories	Literal Value	Includes all sub-directories in the source directory, when the source to copy is a directory.

Description

Provide a short description for the Copy File activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
fromFileName	string	The path and name of the file or directory to copy.
		For directories, specify an absolute path. You may use wildcards when specifying files within a directory. For example:

Input Item	Datatype	Description
		• To copy a directory, specify `c:\myDirectory`. • To copy all text files in a directory, specify `c:\myDirectory\*.txt`.
toFileName	string	The destination for the copy operation. This must be an absolute path and wildcards cannot be used in this element.

> **Note:** The **Input** field always takes precedence for all **File** palette activities with respect to the field equivalent on the **General** tab.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
FileNotFoundException	The file does not exist. The source file or the destination is not found and the **Create Non-Existing Directories** checkbox is not selected on the **General** tab.
IllegalCopyException	The destination exists but its type is different than the type of the specified source. For example, in the activity input, *fromFileName* is a directory and *toFileName* is a file.
FileAlreadyExistsException	The file already exists at the destination where you are attempting to copy the file and the **Overwrite** field is not selected on the **General** tab.
FileIOException	There is an input/output error. For example, the file or directory already exists and it is write-protected.

Remove File

The **Remove File** activity is a synchronous activity that removes the specified files from the directory. If the specified directory is not empty, it generates an exception.

General

The **General** tab has the following fields.

Field	Literal Value/ Process Property/ Module Property	Description
Name	None	The name of the file to be displayed as the label of the activity in the process.
FileName	Yes	The path and name of the file to be removed from the directory.
Include Timestamp	None	Select the checkbox to display timestamp, in addition to the date.

Description

Provide a short description of the Remove File activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
fileName	string	The path and name of the file to be removed from the directory.

Output

The following is the output for the activity.

Output Item	Datatype	Description
fileInfo	complex	This element contains the **fullName**, **fileName**, **location**, **type**, **readProtected**, **writeProtected**, **size**, and **lastModified** data.
fullName	string	The name of the file (or empty directory) and the file path information.
fileName	string	The name of the file (or empty directory) without file path information.
location	string	The path to the file or the empty directory.
ConfiguredFileName	string	An optional element. This element is not populated by this activity.
type	string	The file type.
readProtected	boolean	Signifies whether the file or (or empty directory) is protected from reading
writeProtected	boolean	Signifies whether the file or (or empty directory) is protected from writing
size	integer	The size of file in bytes.
lastModified	string	The timestamp indicating when the file was last modified.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
FileNotFoundException	The file does not exist.
FileIOException	An input/output exception occurs, when trying to remove the file or empty directory.

Create File

The **Create File** is a synchronous activity that creates a new file or directory with the specified name at specified location.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Filename	Yes	The path and name of the file to create. Select the **Is a Directory** checkbox on the **General** tab to specify the name of the directory to create.
Overwrite	None	Overwrites the existing file with the same name, if it exists. The activity throws an exception, if the specified file exists and this checkbox is not selected.
Is a Directory	None	When this checkbox is selected, the activity creates a directory instead of a file.

Field	Literal Value/Process Property/Module Property	Description
Create Non-Existing Directories	None	When this checkbox is selected, the activity creates all directories in the specified path, if they do not already exist.
		If this checkbox is not selected and one or more directories in the specified path do not exist, it throws an exception.
Include Timestamp	None	Select the checkbox to display timestamp, in addition to the date.

Description

Provide a short description for the Create File activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
fileName	string	The path and name of the file to create.
		Select the **Is a Directory field** check box on the **General** tab to specify the name of the directory to create.

Output

The following is the output for the activity.

Output Item	Datatype	Description
fileInfo	complex	The element containing **fullName**, **fileName**,

Output Item	Datatype	Description
		location, **configuredFileName**, **type**, **readProtected**, **writeprotected**, **size**, and **lastModified**
fullName	string	The name of the file or directory, including the path information
fileName	string	The name of the file or directory without the path information
location	string	The path to the file or directory
configuredFileName	string	This element is optional and it is not populated by this activity
type	string	The type of the file
readProtected	boolean	Signifies whether the file or directory is protected from reading
writeProtected	boolean	Signifies whether the file or directory is protected from writing
size	integer	The size of the file (in bytes)
lastModified	string	The time stamp indicating when the file was last modified

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action, see the *TIBCO BusinessWorks™ Container Edition Error Codes* guide.

Fault	Generated When..
FileAlreadyExistsException	The file already exists at the destination where you are

Fault	Generated When..
	attempting to copy the file. The **Overwrite** checkbox is not selected on the **General** tab.
FileIOException	There is an Input/Output error. For example, the file or directory already exists and is write-protected.

File Poller

The **File Poller** activity is a process starter activity that polls for files or directories with the specified name and starts a process, when a change (creation, modification, or deletion) is detected.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Filename	Yes	The path and name of the file or directory to monitor. You can also use wildcard characters to monitor a directory for files that match the provided specification. For example, `C:\files\*.log` must match any change to a file in the files directory with an extension of `.log`.
Polling Interval(sec)	Yes	The polling interval (in seconds) to check for the specified file.

Field	Literal Value/Process Property/Module Property	Description
Include Existing Files	None	Check for the specific file among the files that existed before the process engine starts. • **When selected** - the existing file matching the specification in the **Filename** field creates a new process instance, when a process engine starts. • **When not selected** - any existing file matching the specification in the **Filename** field is ignored, unless there is a change in the file.
Exclude File Content	None	Specifies not to load the data from the file into this activity's output. If selected, the contents of the file are not available to the subsequent activities in the process.
Content as	Yes	The type of content in the file which can be either **Text** or **Binary**. This field is available only when the **Exclude File Content** checkbox is not selected.
Encoding	Yes	The character encoding for the file (when **Text** is specified in the **Content as** field). This field is visible only when **Content as** value is set to **Text**.
Include Timestamp	None	Select the checkbox to display timestamp, in addition to the date.

Description

Provide a short description of the File Poller activity.

Advanced

The **Advanced** tab has the following fields.

Field	Description
Sequence Key	This field can contain an XPath expression that specifies which processes should run in order. Process instances with sequencing keys that evaluate to the same value are executed sequentially in the order the process instance was created.
Custom Job Id	This field can contain an XPath expression that specifies a custom ID for the process instance.
Poll for Create Events	Select this checkbox to start a new process when a file or directory matching the specification in the **Filename** field is created. If not selected, the newly created files or directories are ignored.
Poll for Modify Events	Select this checkbox to start a new process when a file or directory matching the specification in the **Filename** field is modified. If not selected, the modifications to the files or directories are ignored.
Poll for Delete Events	Select this checkbox to start a new process when a file or directory matching the specification in the **Filename** field is deleted. If not selected, deletions of files or directories are ignored.
Include Sub-Directories	Select this checkbox to enable monitoring of the events that occur in subdirectories of the specified directory. If not selected, only the directory is monitored and subdirectories are ignored.
Mode	Select the type of polling to perform from the following modes in the drop-down list: • **Only Files**: only tracks the specified changes for files. • **Only Directories**: only tracks the specified changes for directories. • **Files and Directories**: tracks the specified changes for both files and directories.
Sort by	Specifies the order in which the files should be processed, if more than one file is detected. You can sort by **File Name**, **Date Modified**, or **None**.

Field	Description
	The files are processed as the polling interval occurs. Hence, the **Sort By** field applies to all files that have changed since the last polling interval. The field is useful for processing the files in a specific order when the process engine starts and the **Include Existing Files** field is selected (then all existing files are processed in the specified order). **Note:** The **File Poller** activity starts process instances based on the specified **Sort By** order, but the process instances may not complete in the same order. If you want to guarantee that processes start and complete in a specified order, specify Max Jobs = 1, and set the engine persistence mode to group or datastore.
Sort Order	Specifies whether the order specified in the **Sort By** field is **Ascending** or **Descending**. This field is only applicable if the **Filename** or **Date Modified** is selected in the **Sort By** field.

Output

The following is the output for the activity.

Output item	Datatype	Description
action	string	The occurred event to trigger the **File Poller** activity. The possible values are: `create`, `remove`, or `modify`.
timeOccured	integer	The time when the **File Poller** activity detected the change in the file. The time is defined as the amount of time in milliseconds. For example, since midnight, January 1, 2013 UTC.
fileInfo	complex	This element contains the **fileName**, **location**, **type**, **readProtected**, **writeProtected**, and **size** data.
fullName	string	The name of the file, including the path information.

Output item	Datatype	Description
fileName	string	The name of the file without the path information.
location	string	The path to the file.
configuredFileName	string	The name of the file or directory this activity is configured to operate on.
type	string	The file type.
readProtected	boolean	Signifies whether the file or directory is protected from reading
writeProtected	boolean	Signifies whether the file or directory is protected from writing
size	integer	The size of the file in bytes.
lastModified	string	The timestamp indicating when the file was last modified.
fileContent	complex	The element containing the **textContent** and **binaryContent** data.
textContent	string	The content of the file (text files), if the **Exclude File Contents** check box is not selected on the **General** tab.
binaryContent	binary	The content of the file (binary files), if the **Exclude File Contents** check box is not selected on the **General** tab. Only available only if **type** is set to binary.
encoding	string	The character encoding for the file. This field is present only if **Content as** field is set the value as **Text**.

List Files

The **List Files** activity is a synchronous activity that returns information about files or directories, or a listing of all the files in the specified directory.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Filename	Yes	The path and name of the file or directory to monitor. You can also use wildcard characters to monitor a directory for files that match the provided specification. For example, `C:\files\*.log` must match any change to a file in the files directory with an extension of `.log`.
Mode	None	The type of listing you want to retrieve. You can select from the following options: • **Only Files** • **Only Directories** • **Files and Directories**
Include Timestamp	None	Select the checkbox to display timestamp, in addition to the date.

Description

Provide a short description for the List Files activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
fileName	string	The path and name of the directory with files or subdirectories to list. Providing a file or directory name returns the information about that file or directory. You can use wildcards to list files or directories that match the provided specification. For example: • `C:\files\*.log` lists any file in the files directory that have a `.log` extension. • `C:\files\mylogs\*.*` lists all files in the directory `mylogs`.
recursive	boolean	Enabling this to true, lists all the files and directories from the same directory and sub-directory. By default, it is set to `false`.

Output

The following is the output of the activity.

Output Item	Datatype	Description
fileInfo	complex	Element containing the **fullName**, **fileName**, **location**, **configuredFileName**, **type**, **readProtected**, **writeProtected,size**, and **lastModified** data. This is a repeating element. Each item in the element is a file or directory included in the specified directory matching the provided criteria.
fullName	string	The name of the file, including the path information.
fileName	complex	The name of the file without the path information.
location	string	The name of the file or directory this activity is configured to operate on.

Output Item	Datatype	Description
configuredFileName	string	The path to the file.
type	string	The type of the file.
readProtected	boolean	Signifies whether the file or directory is protected from reading
writeProtected	boolean	Signifies whether the file or directory is protected from writing
size	integer	The size of the file in bytes.
lastModified	string	The timestamp indicating when the file was last modified. You can compare values in this element using the *compare-dateTime()* XPath function. For example, `compare-dateTime($List-Files/ListFilesActivityOutput/files/fileInfo[fileName="myFile.txt"]/lastModified,current-dateTime())`

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
FileNotFoundException	The file does not exist and the specified path is either incorrect or not found.

Read File

The **Read File** activity is a synchronous activity that is used to read a file and place its contents into the output of the activity.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Filename	Yes	The name of the file.
Exclude File Content	None	Specifies not to load the data from this file into the output of the activity. If selected, the contents of this file are not available to the subsequent activities in the process.
Read as	Literal Value	This field is available only if the **Exclude File Content** checkbox is not selected. The type of the content in the file. It can be either **Text** or **Binary**.
Include Timestamp	None	Select the checkbox to display timestamp, in addition to the date.

Description

Provide a short description for the Read File activity.

Input

The following is the input of the activity.

Input Item	Datatype	Description
fileName	string	The name and path of the file to read.
encoding	string	The character encoding of the text files. This element is available only when the **Read as** field on the **General** tab is set to **Text**. You can specify the encoding names supported by Java in this element. If this element is not specified, the default encoding of the Java Virtual Machine used by the process engine is used.

Output

The following is the output of the activity.

Output Item	Datatype	Description
fileInfo	complex	This element contains **fullName**, **fileName**, **location**, **type**, **readProtected**, **writeProtected**, **size**, and **lastModified** data.
fullName	string	The name of the file with the path information.
fileName	string	The name of the file without the path information.
location	string	The path to the file.
configuredFileName	string	An optional element. It is not populated by this activity.
type	string	The file type.
readProtected	boolean	Signifies whether the file or directory is protected from reading
writeProtected	boolean	Signifies whether the file or directory is protected from writing

Output Item	Datatype	Description
size	integer	The size of the file in bytes.
lastModified	string	The timestamp indicating when the file was last modified.
fileContent	complex	The element containing the textContent data.
textContent	string	The content of the file (text files), if the **Exclude File Content** check box is not selected on the **General** tab.
binaryContent	binary	The content of the file (binary files), if the **Exclude File Content** check box is not selected on the **General** tab. Only available when **Content as** is set to **Binary**.
encoding	string	The character encoding for the file when the value specified in the **Read as** field in the **General** tab is **Text**.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
FileNotFoundException	The file does not exist.
UnsupportedEncodingException	The content of the text file is invalid, and the content of the file is read into the process data.
FileIOException	An input/output exception occurred when trying to read the file.

Rename File

The **Rename File** activity is a synchronous activity that is used to rename or move files. This activity can also rename directories. However, you cannot use this activity to move the directory to a new location.

General

The **General** tab consists of the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as a label of the activity in the process.
From Filename	Yes	The path and name of the file to rename or move, or the path and name of the directory to rename. The value in this element must be an absolute path.
To Filename	Yes	The new name and location of the file or directory. The files can be moved to a new location, but the directory location remains unchanged. The value of this element must be an absolute path.
Overwrite	None	Select this checkbox to overwrite the existing file with the same name when renaming or moving. If not selected, the activity raises an exception, if the specified file or directory exists.
Create Non-Existing Directories	None	When this checkbox is selected, the activity creates all directories in the specified path, if they do not already exist. If not selected and one or more directories in the specified path do not exist, it throws an exception.

Field	Literal Value/Process Property/Module Property	Description
Include Timestamp	None	Select the checkbox to display timestamp, in addition to the date.

Description

Provide a short description for the Rename File activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
fromFileName	string	The path and name of the file to rename or move, or the path and name of the directory to rename. The value in this element must be an absolute path.
toFileName	string	The new name and location of the file or directory. The files can be moved to a new location, but the directory location remains unchanged. The value of this element must be an absolute path.

Output

The following is the output for the activity.

Output Item	Datatype	Description
fileInfo	complex	This element contains the **fullName**, **fileName**, **location**, **type**, **readProtected**, **writeProtected**, **size**,

Output Item	Datatype	Description
		and **lastModified** data.
fullName	string	The name of the file (or directory) including the path information.
fileName	string	The name of the file (or directory) without the path information.
location	string	The path to the file or the directory.
configuredFileName	string	An optional element. This element is not populated by this activity.
type	string	The file type.
readProtected	boolean	Signifies whether the file or directory is protected from reading
writeProtected	boolean	Signifies whether the file or directory is protected from writing
size	integer	The size of file in bytes.
lastModified	string	The timestamp indicating when the file was last modified.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
FileNotFoundException	The file does not exist.

Fault	Generated When..
IllegalRenameException	The destination file exists and its type is different than the type of the source file.
FileAlreadyExistsException	The file already exists.
FileIOException	An input/output exception occurred when trying to rename the file or directory. This exception is also encountered while attempting to move a directory.

Write File

The **Write File** activity is a synchronous activity that writes content to the specified file.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Filename	Yes	The path and name of the file. Wildcards are not permitted in this field.
Append	None	Appends the contents to an existing file. Not selecting this checkbox overwrites the file.
Write as	Literal Value	Specify whether **Text** or **Binary**.

Field	Literal Value/Process Property/Module Property	Description
Create Non-Existing Directories	None	When selected, the activity creates all directories in the specified path, if they do not already exist. If this checkbox is not selected with one or more non-existing directories in the specified path, it throws an exception.
Compress	None	This field specifies whether to compress the output file using GZip format. Specify **None** for no compression or **GZip** for a compressed output file. When you specify **GZip** for this field, rename the file to use the `.gz` suffix and use gunzip to decompress the file.
Include Timestamp	None	Select the checkbox to display timestamp, in addition to the date.

Description

Provide a short description for the Write File activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
fileName	string	The path and name of the file. Wildcards are not permitted in this field.
textContent	string	The contents of the file (text files). This field is present when **Write as** is set to **Text**. When **Write as** is set to

Input Item	Datatype	Description
		Binary, this field is replaced by the field **binaryContent**.
addLineSeparator	boolean	This specifies whether to add a carriage return after each input line. This field is present when the value of the **Write as** field on the **General** tab is set to **Text**.
encoding	string	The character encoding for text files. This element is available only when **Text** is specified in the **Write as** field on the **General** tab. You can specify encoding names supported by Java in this element. If this element is not specified, the default encoding of Java Virtual Machine is used.

Output

The following is the output for the activity.

Output Item	Datatype	Description
fileInfo	complex	This element contains the **fileName**, **location**, **type**, **readProtected**, **writeProtected**, and **size** data.
fullName	string	The name of the file, including the path information.
fileName	string	The name of the file without the path information.
location	string	The path to the file.
configuredFileName	string	An optional element. This element is not populated by this activity.
type	string	The file type.
readProtected	boolean	Signifies whether the file or directory is protected

Output Item	Datatype	Description
		from reading
writeProtected	boolean	Signifies whether the file or directory is protected from writing
size	integer	The size of the file in bytes.
lastModified	string	The timestamp indicating when the file was last modified.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
FileNotFoundException	The file does not exist.
UnsupportedEncodingException	The encoding is not valid.
FileIOException	An input/output exception occurs when trying to write to the file.

FTP Palette

The **FTP** palette is used to issue FTP commands.

FTP Change Default Directory

FTP Change Default Directory is a synchronous activity that changes the current default directory on the remote machine to the specified directory path.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, port, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this command. The FTP Connection remains open, if this check box in not selected. As a result, all the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), share the FTP session until the connection is closed.

Description

Provide a short description of the FTP Change Default Directory activity.

Input

The **Input** tab has the following fields.

Input Item	Datatype	Description
Host	string	The name of the FTP server.
		This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for incoming FTP requests on the FTP server.
		This overrides the value specified in the **FTP Connection** property.
Username	string	The username for logging into the FTP server.
		This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server.
		This overrides the value specified in the **FTP Connection** property.
NewDefaultDir	string	The path to the new default directory on the FTP server.
Timeout	integer	The amount of time to wait for the FTP server to respond. This timeout is used, when the FTP server places the FTP connection used by this activity into a waiting state.
		This overrides the value specified in the **FTP Connection** property.

Output

This activity produces no output.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred while executing the FTP command.
ConnectionErrorException	An error occured while connecting to the FTP server.

FTP Delete File

FTP Delete File is a synchronous activity that issues an FTP `delete` or `mdelete` command to delete one or more files from the remote server.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.

Field	Literal Value/Module Property	Description
Quit (post-command)	None	Select this check box to specify that the FTP Connection should be closed after completing this command. The FTP connection remains open, if this check box in not selected. As a result all the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), share the FTP session until the connection is closed.

Description

Provide a short description of the FTP Delete File activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server.

Input Item	Datatype	Description
		This overrides the value specified in the **FTP Connection** property.
RemoteDirectory	string	An optional element. The path to the directory containing the files to be deleted.
		This element is concatenated with the RemoteFileName element to provide the complete path to the files to be deleted.
RemoteFileName	string	The path and name of the file to delete on the remote server. This field permits wildcards (? for any one character, * for one or more characters).
		You can specify an absolute or relative path for the remote file name. Relative paths are relative to the path specified in the RemoteDirectory element or in the user's home directory, if RemoteDirectory is null.
Timeout	integer	This timeout is used, when the FTP server places the FTP Connection used by this activity, into a waiting state.

Output

This activity produces no output.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionErrorException	An error occurred when connecting to the FTP server.

FTP Dir

FTP Dir is a synchronous activity that provides a sequential list of files in the specified directory of the FTP server.

General

The **Genaral** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this command. The FTP connection remains open, if this check box in not selected. As a result, all FTP activities that specify the same FTP session (that is, the same host, port, username, and password), share the FTP session until the connection is closed.
NLST	None	Selecting this check box specifies to use the NLST command instead of the DIR command. The DIR command produces verbose output including the file size, modification time, and permissions. NLST produces only a list of file names.

Description

Provide a short description of the FTP Dir activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for the incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
DirParameters	string	The command parameters to pass to the `DIR` or `NLST` command.
Directory	string	An optional parameter. The path to the directory in which you want to list the files. If null, the current remote directory is listed.
Timeout	integer	The amount of time to wait for the FTP server to respond. This timeout is used, when the FTP server places the FTP connection used by this activity into a waiting state.

Input Item	Datatype	Description
		This overrides the value specified in the **FTP Connection** property.

Output

The following is the output of the activity.

Output Item	Datatype	Description
ItemCount	integer	The number of entries returned from the remote directory listing.
DirectoryItems	string	A repeating element containing the list of items returned.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred while executing the FTP command.
ConnectionErrorException	An error occured while connecting to the FTP server.

FTP Get

FTP Get is a synchronous activity that issues an FTP `get` or `mget` command to the specified server. The content of the remote files can be placed in the output of the activity or written directly to the local storage. If you select to place the contents of a remote file into the output of the activity, you can only retrieve one remote file (FTP `get` command) and the file's content is stored in memory as part of the output of the activity. If you write the

contents of the retrieved files to the local storage, you can retrieve one or more files (FTP `get` or `mget` commands).

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this command. The FTP connection remains open, if this check box in not selected. All the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), can share the FTP session until the connection is closed.
Use Process Data	None	Specifies to write contents of the file retrieved from the FTP server to the process data. The output of the activity includes an element that contains the remote file's contents, and the data can be used by subsequent activities in the process.

Field	Literal Value/Module Property/Process Property	Description
		Caution: Do not use this option if you are retrieving multiple files. You can only write one files's contents to this output of the activity. Selecting this check box with wildcards specified in the **RemoteFileName** input element returns an error.
Binary	None	Specifies whether to retrieve the file content in binary or ASCII mode (selected signifies binary). Using ASCII mode automatically handles the differences in line endings between the FTP server and the machine executing the FTP command. The ASCII mode implies text data. This text data can be encoded in a character encoding other than ASCII (such as UTF-8 or Latin-1).

Description

Provide a short description of the FTP Get activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
RemoteFileName	string	The path and name of the file to retrieve from the remote server. Wildcards (? for any one character, * for one or more characters) are permitted in this field. You can specify an absolute or relative path for the remote file name. Relative paths are relative to the path

Input Item	Datatype	Description
		specified in the **RemoteDirectory** element or the current directory, if **RemoteDirectory** is `null`. **Note:** Do not specify wildcards for this element, if **Use Process Data** check box is selected.
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
Username	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Timeout	integer	The amount of time to wait for the FTP server to respond. This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state.

Output

The following is the output for this activity.

Output Item	Datatype	Description
FileTransferred	complex	When one or more remote files are retrieved and written to the local storage, this element contains a list of the retrieved remote filenames. This element contains the following two elements: • Name • NumOfBytes If an error occurs while retrieving the files, the activity output is not available. However, the **Fault** tab lists the successfully transferred files and the files that failed to be transferred.
Name	string	The name of the retrieved remote file.
NumOfBytes	integer	The number of bytes comprised in the associated file name.

Fault

The **Fault** tab lists the following possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
GetFilesException	An error occurred during the FTP get operation.
ConnectionErrorException	An error occurred when connecting to the FTP server.
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionTimeoutException	The connection to the FTP server is timed out before the command execution is complete.

FTP Get Default Directory

FTP Get Default Directory is a synchronous activity that retrieves the name of the current remote directory. The default remote directory is operating system dependent and determined by the remote FTP server. On UNIX systems, the default remote directory is usually the home directory of the user account that is used to establish an FTP connection.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP Connection should be closed after completing this command. The FTP connection remains open if this check box in not selected. All the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), can share the FTP session until the connection is closed.

Description

Provide a short description of the FTP Get Default Directory activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Timeout	integer	The amount of time to wait for the FTP server to respond. This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state. This overrides the value specified in the **FTP Connection** property.

Output

The following is the output for the activity.

Output Item	Datatype	Description
CurrentDirectory	string	The directory currently set on the remote FTP server. By default, this directory is set by the FTP server. However, you can use the **FTP Quote** activity to issue an FTP command to change to another directory.

Fault

The **Fault** tab lists the following possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionErrorException	The connection to the FTP server is timed out before the command execution is complete.

FTP Make Remote Directory

FTP Make Remote Directory is a synchronous activity that creates the specified directory on the remote server.

General

The **Genera**l tab consists of the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this

Field	Literal Value/Module Property/Process Property	Description
		command.
		The FTP connection remains open, if you do not select this check box. All the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), can share the FTP session until the connection is closed.

Description

Provide a short description of the FTP Make Remote Directory activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for the incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server.

Input Item	Datatype	Description
		This overrides the value specified in the **FTP Connection** property.
RemoteSITECommand	string	An optional element. This is a system-specific command to execute before creating the directory.
ParentRemoteDirectory	string	An optional element. The path to the remote directory to create. This element is concatenated with the **RemoteDirName** element to provide the complete path to the directory to create.
RemoteDirName	string	The name of the remote directory to create. You can optionally include the complete path of the directory to create.
Timeout	integer	The amount of time to wait for the FTP server to respond. This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state. This overrides the value specified in the **FTP Connection** property.

Output

This activity produces no output.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionErrorException	An error occurred when connecting to the FTP server.

FTP Put

FTP Put is a synchronous activity that issues an FTP `put` or `mput` command to the specified server. You can use the process data as the file content to send to the remote server or send the files to the local disk storage. Select **Use Process Data** to place only one file on the remote server (FTP `put` command). To use locally stored files for placing more than one files on the remote server (FTP `mput` command).

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this command. The FTP connection remains open, if you do not select this check box. All the FTP activities that specify the same FTP session (that is, the same host,

Field	Literal Value/Module Property/Process Property	Description
		port, username, and password), can share the FTP session until the connection is closed.
Binary	None	Specifies whether to send the file content in binary or ASCII mode (selected signifies binary).
		When ASCII mode is used, the differences in line-endings between the FTP server and the machine executing the FTP command are automatically handled. This mode also implies text data, and that data can be encoded in a character encoding other than ASCII (such as UTF-8 or Latin-1).
Overwrite Existing File	None	Selecting this check box overwrites any existing files on the FTP server, if the FTP command sends files with the same names.
		If this check box is not selected, any attempt to send a file to the remote server with the same name as an existing file, results in an exception.
Append	None	This element specifies whether to append the data to an existing file. Select this check box to append the data to the file, if the specified file exists on the FTP server.
		If not selected, the file is overwritten if it exists on the FTP server.
Timeout (msec)	None	The amount of time to wait (in milliseconds) for the FTP server to respond.
		This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state.
		This overrides the value specified in the **FTP Connection** property.

Description

Provide a short description of the FTP Put activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
RemoteFileName	string	The path and name of the file to place onto the remote server.
		Wildcards are not permitted in this field.
Host	string	The name of the FTP server.
		This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for the incoming FTP requests on the FTP server.
		This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server.
		This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP Server.
		This overrides the value specified in the **FTP Connection** property.
Encoding	string	Encoding of the data to be put on the FTP server. This field is present when the **Use Process Data** check box is selected on the **General** tab.
Timeout	integer	The amount of time to wait for the FTP server to respond.

Input Item	Datatype	Description
		This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state. This overrides the value specified in the **FTP Connection** property.

Output

The following is the output for this activity.

Output Item	Datatype	Description
FileTransferred	string	A repeating string element containing a list of names of files that were transferred. This element contains the following two elements: • Name • NumOfBytes If an error occurs while transferring files, the activity output is not available. However, the **Fault** tab lists the successfully transferred files and the files that failed to be transferred.
Name	string	The name of the retrieved remote file.
NumOfBytes	integer	The number of bytes comprised in the associated file name.

Fault

The **Fault** tab lists the following possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
PutFilesException	An error occurred during the FTP Put operation.
ConnectionErrorException	An error occurred when connecting to the FTP server.
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionTimeoutException	The connection to the FTP server is timed out before the command execution is complete.

FTP Quote

FTP Quote is a synchronous activity that sends an arbitrary command to the FTP server. FTP commands vary by their operating system, FTP version, and configuration. Before using this activity, check for the availability of the commands on the remote server.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as label of the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this command.

Field	Literal Value/Module Property/Process Property	Description
		The FTP connection remains open, if you do not select this check box. All the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), can share the FTP session until the connection is closed.
Has Socket Data	None	If selected, the FTP server returns the data as a result of executing the command.

Description

Provide a short description of the FTP Quote activity.

Input

The following is the input for this activity.

Input Item	Datatype	Description
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for the incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.

Input Item	Datatype	Description
Password	string	The password for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
RemoteSITECommand	string	Optional. The system-specific command to be executed before the executing the specified command.
ValidReturnCodes	string	The optional list of valid return codes to validate the reply. Codes are three-digit numbers. To specify multiple codes, separate each code with a comma or a space. If the remote server returns a code not specified in this list, the activity raises an exception.
RemoteCommand	string	The command to be executed on the remote FTP server.
Timeout	integer	The amount of time to wait for the FTP server to respond. This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state. This overrides the value specified in the **FTP Connection** property.

Output

The following is the output for this activity.

Output Item	Datatype	Description
CommandItems	string	Each element in this repeating element is a returned value from the command executed on the remote server.

Fault

The **Fault** tab lists the following possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionErrorException	An error occurred when connecting to the FTP server.

FTP Remove Remote Directory

FTP Remove Remote Directory is a synchronous activity that deletes the specified directory from the remote FTP server.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as label of the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this command.

Field	Literal Value/Module Property/Process Property	Description
		The FTP connection remains open, if you do not select this check box. All the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), can share the FTP session until the connection is closed.

Description

Provide a short description of the FTP Remove Remote Directory activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for the incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server. This overrides the value specified in the **FTP**

Input Item	Datatype	Description
		Connection property.
ParentRemoteDirectory	string	An optional element. The path to the remote directory you want to remove.
		This element is concatenated with the **RemoteDirName** element to provide the complete path to the directory to remove.
RemoteDirName	string	The name of the remote directory you want to remove.
		You can optionally include the complete path of the directory to remove.
Timeout	integer	The amount of time to wait for the FTP server to respond.
		This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state.
		This overrides the value specified in the **FTP Connection** property.

Output

This activity produces no output.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionErrorException	An error occurred when connecting to the FTP server.

FTP Rename File

FTP Rename File is a synchronous activity that renames the specified file on the remote FTP server.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as label of the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this command. The FTP connection remains open, if you do not select this check box. All the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), can share the FTP session until the connection is closed.

Description

Provide a short description of the FTP Rename File activity.

Input

The following is the input for this activity.

Input Item	Datatype	Description
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for the incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
OldRemoteDirectory	string	An optional element. The path to the remote file you want to rename. This element is concatenated with the **OldRemoteFileName** element to provide the complete path to the file to rename.
OldRemoteFileName	string	The name of the remote file you want to rename. You can optionally include the complete path of the file to rename.

Input Item	Datatype	Description
NewRemoteDirectory	string	An optional element. The new path for the file to rename. This element is concatenated with the **NewRemoteFileName** element to provide the complete path to the file to rename.
NewRemoteFileName	string	The new name for the remote file. You can optionally include the complete path of the file.
Timeout	integer	The amount of time to wait for the FTP server to respond. This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state. This overrides the value specified in the **FTP Connection** property.

Output

This activity produces no output.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionErrorException	An error occurred when connecting to the FTP server.

FTP SYS Type

FTP Sys Type is a synchronous activity that retrieves the operating system of the FTP server.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as label of the activity in the process.
FTP Connection	Yes	The path to the shared configuration resource containing the user information such as host name, username, and password.
Quit (post-command)	None	Select this check box to specify that the FTP connection should be closed after completing this command. The FTP connection remains open, if you do not select this check box. All the FTP activities that specify the same FTP session (that is, the same host, port, username, and password), can share the FTP session until the connection is closed.

Description

Provide a short description of the FTP SYS Type activity.

Input

The following is the input for this activity.

Input Item	Datatype	Description
Host	string	The name of the FTP server. This overrides the value specified in the **FTP Connection** property.
Port	integer	The port number for the incoming FTP requests on the FTP server. This overrides the value specified in the **FTP Connection** property.
UserName	string	The username for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Password	string	The password for logging into the FTP server. This overrides the value specified in the **FTP Connection** property.
Timeout	string	The amount of time to wait for the FTP server to respond. This timeout is used when the FTP server places the FTP connection used by this activity into a waiting state. This overrides the value specified in the **FTP Connection** property.

Output

The following is the output for this activity.

Output Item	Datatype	Description
SystemType	string	The FTP server type.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionException	An error occurred when executing the FTP command.
ConnectionErrorException	An error occurred when connecting to the FTP server.

General Activities

General Activities palette comprises various general purpose activities and process starters. For example, **Timer**, a process starter activity, is used to start a process at a specific time.

Assign

Assign is a synchronous activity. You can use this activity to assign a value to a process-defined variable. The **Assign** activity is used to assign a value to the process-defined as well as scope-defined variables. Ensure to set all the essential values when using the **Assign** activity to set a process variable.

When the **Assign** activity is used inside the **For Each**, **Iterate**, **Repeat** or **While** activities, and the **Accumulate Output** checkbox is selected on the **General** tab of these activities, the **Output Activity** field does not list the **Assign** activity.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Process Variable to Set	A drop-down list of the available user-defined process variables for the process. Select any user-defined process variable to modify.

Description

Provide a short description of the Assign activity.

Input Item	Description
\<schema\>	The specified data schema for the user-defined process variable selected on the **General** tab.

Output

The schema specified for the process variable determines the schema for the output.

Call Process

The **Call Process** activity calls and executes an existing direct subprocess. Input data specified on the called subprocess is used by the **Call Process** activity.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Process Name	The subprocess you want to call. Click the **Look Up** 🔍 icon or the **Go To** icon to select a direct subprocess.

Field	Description
	Tip: An alternative way to call a subprocess contained in the same project, enter the name of the package and the name of the subprocess, separated by a period, in this field. In this example, *<PackageA>* is the name of the package, and *<Process1>* is the name of the subprocess. *<PackageA>.<SubProcess1>* To call a subprocess in a separate project, enter the name of the project, the name of the package, and the name of the subprocess, separated by periods, in this field. In this example, the name of the project is*<ProjectB>*, the name of the package is *<PackageB>*, and the name of the subprocess is *<SubProcess2>*. *<ProjectB>.<PackageB>.<SubProcess2>*
Process Name Dynamic override	Configuring this field is optional. Use this field to dynamically determine the subprocess to call when the process instance is running. Click the **Edit XPath Expression** 📝 icon to specify a value for this field in the XPath Builder wizard. For more information, see Dynamically Determining the Process to Call.
Spawn	Optional. If you select the **Spawn** checkbox, the parent process does not wait for the called subprocess to complete, the subprocess is executed in a separate process instance, and the parent process cannot access the output of the called process.
Custom Icon	Provide the location of icon image file. Click the **Choose a custom icon** 🔍 button to select the icon image file from your current workspace. Click the **Import an icon from the file system** ➕ button to select an icon image from your file system. Select the **Clear Value** ⊗ button to remove the field value. **Important:** TIBCO recommends to use image file of size 48 by 48 pixels and of the form `.png`, `.jpg`, or `.jpeg`.

Description

Provide a short description of the Call Process activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
input	varies	The input to the called process is defined in the **Output Editor** tab of the **Start** activity of the called process.

Output

The following is the output for the activity.

Output Item	Datatype	Description
output	varies	The output of the called process is defined in the **Input Editor** tab of the **End** activity of the called process.

Fault

The **Fault** tab lists faults that the called subprocess might throw with a **Throw** activity. To deal with these faults, add a **Catch Specific**, or **Catch All** block to the **Call Process** activity.

Dynamically Determining the Process to Call

You can use the **Process Name Dynamic Override** field on the **General** tab to specify an XPath expression that determines which direct subprocess to call. This is useful for calling a different direct subprocess depending upon the value of the XPath expression. For example, you have an incoming HTTP request that contains an attribute named orderAmount

For all incoming orders over $10,000, you want to call a direct subprocess that includes activities in the Manual Work palette to obtain the proper approvals. For orders under

$10,000, you want to call a direct subprocess that handles the order automatically. You can create the following expression that resolves to the proper direct subprocess name for each type of order:

```
if($IncomingOrer/orderAmount > 10000)
then 'MyProject.manualApproval' else
'MyProject.processOrder'
```

Use the full path and name of the direct subprocess file as stored in the project directory.

When you use the **Process Name Dynamic Override** field, you must also specify a direct subprocess to call in the **Process** field. The input, output, and error definitions of the specified direct subprocess must be the same as any direct subprocess that the expression in the **Process Name Dynamic Override** field can evaluate to.

> **ⓘ** **Note:** It may be helpful to create process definitions that act as programmatic interfaces when using the **Process Name Dynamic Override** field. In the example above, you may create a direct subprocess named orderProcessOrApprove. The only purpose of this direct subprocess is for specifying the input, output, and error schemas. Place this direct subprocess in the **Process Name** field of the **General** tab of the **Call Process** activity. Then, create the manualApproval and processOrder process definitions as copies of the orderProcessOrApprove direct subprocess. The **Call Process** activity then has the correct input, output, and error schemas for all direct subprocesses that can be called.

To dynamically call a direct subprocess, that has been created inside of a Shared Module, from an Application Module, use the following syntax:

```
'SM_Name/Package_Name.SubProcess_Name
```

> **ⓘ** **Note:** When the **Call Process** activity is in a local JDBC or JMS transaction, and the **Spawn** checkbox is not selected, the direct subprocesses called by the Call Process in the Local Transaction are also a part of the same local transaction.

Confirm

Confirm is a synchronous activity that confirms any confirmable message received by the process instance.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
ConfirmEvent	The message you want to confirm.
	This is a drop-down list of any process starter or activity awaiting an incoming event in the current process.
	You can confirm messages only from the reliable sources, for example, JMS messages. E-mail is also confirmable when received by the Receive Mail process starter (the e-mail message is deleted from the server when it is confirmed).
	Select only the event that sent the confirmable message.

Description

Provide a short description of the Confirm activity.

External Command

External Command is a synchronous activity that executes the command on the operating system. This activity waits for the command to complete before transition to the next activity. Optionally, the command output and errors can be included in this activity's output schema, written to a file, or both.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as a label for the activity in a process.
Command To Execute	Yes	The command line to execute. You can also specify any input and command-line arguments to the command in this field. You cannot specify more than one command by piping output from one command to another (for example, `ls | more` is not allowed).
Provide Command Output	None	Selecting this checkbox specifies that the output sent to the standard output and standard error by command is available in the output schema of this activity. **Caution:** Do not select the checkbox for the commands that produce a large amount of output. Instead, write the output to a file. This saves memory and you can use other activities, such as **Read File** and **Parse Data**, to handle the output file more efficiently.
Remove Parameter Quotes	Yes	If this checkbox is selected, the quotes surrounding the parameters are removed. The quotes are removed even if the parameter contains space(s). By default, this checkbox is clear.
Output Filename	Yes	The name and location of the file for storing any output or errors produced by the command. If this

Field	Literal Value/Process Property/Module Property	Description
		field is blank, no output file is created.
Output Line Splitting	None	Controls whether the command output and errors are split into multiple strings.
		Select any one from the following available options:
		• **None**: This activity produces a single string containing the command's output or errors. This string includes line end characters.
		• **At Operating System Line End**: This activity produces a repeating string element with one string element per output line. The lines are split by default on the carriage return/line feed character, depending on the operating system. The line end characters are not included in the output.
		• **At Specified Token**: The lines are split wherever the specified token occurs. The token is specified as an activity input element, and the token is not included in the output.

Description

Provide a short description of the activity.

Input

The following is input for this activity.

Input Item	Datatype	Description
command	string	The command to execute, including any input or

Input Item	Datatype	Description
		command-line arguments.
		Specifying this input item overrides the command specified on the **General** tab.
input	string	An input string to supply to the command's standard input.
		This schema item attains the same result as entering the command from a command prompt and then typing the input without specifying a carriage return.
outputFile	string	The file containing the output produced by the command for sending it to **stdout**.
environment	string	The environment variable settings used to run the command. This element is specified as a comma-separated list of *<name>=<value>* pairs where:
		• *<name>* is the name of the environment variable
		• *<value>* is the value of the environment variable
		If no value is specified for this element, the command is executed with the environment variable settings specified in the login scripts, for the user account used to execute the process engine.
		If a value is specified, the value of this element replaces any environment settings for the user account used to run the process engine.
		That is, the user environment is ignored and this element's value is used instead.
workingDirectory	string	The working directory for the command process.
		If not specified, the command process inherits its working directory from the process engine.

Output

The following is the output for the activity.

Output Item	Datatype	Description
returnCode	integer	The numeric return code returned by the command. Typically, a value of 0 (zero) in this element signifies a successful command execution. A non-zero value usually indicates termination of the command due to an error.
output	string	Any output produced by the command and sent to stdout. This output item is available only when you select the **Provide Command Output** checkbox on the **General** tab. This is a repeating element, if the line split configuration setting causes the command's output to be split into several lines.
error	string	Any errors produced by the command and sent to stderr. This element is only available if you select the **Provide Command Output** checkbox on the **General** tab. This is a repeating element, if the line split configuration setting causes the command's output to be split into several lines.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
CommandExecutionError	The command could not be executed.

Fault	Generated When..
FileIOError	There was an error when attempting to write the output to a file.
	Ensure that the user running the TIBCO BusinessWorks Container Edition engine, has the write permission to the output file.
InvalidInputException	An error in the activity's configuration or input mapping prevents the command from being executed.

Get Shared Variable

Get Shared Variable is a synchronous activity that retrieves the current value of a module shared variable or job shared variable. To use this activity to retrieve the value of a shared variable, use a **Critical Section** group to ensure that no other process instances alter the value of the shared variable at the same time.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Shared Variable Type	Specify the type of shared variable (**Module Shared Variable** or **Job Shared Variable**).
Shared Variable Name	The module shared variable or job shared variable whose value you want to retrieve.

Description

Provide a short description of the Get Shared Variable activity.

Output

The following is the output for the activity.

Output Item	Description
schema	The value of the shared variable specified in the **Shared Variable** field of the **General** tab is the output for this activity. The schema for the output is determined by the schema specified for the shared variable.

Inspector

Inspector activity is used to write the output of any or all activities and module properties to a file or stdout. This is particularly useful when debugging processes and you want to see the entire schema instead of mapping specific elements to the **Write File** activity.

You can use the **Inspector** activity to write the output of any activity or module properties in the current process. Activities and module properties in a subprocess are not available to the **Inspector** activity (but the output of a **Call Process** activity can be written using the Inspector activity). If you want to obtain the output from one or more activities or module properties in a subprocess, place the **Inspector** activity in the process of the subprocess.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Inspect	The output of the processes you want. Select any one from available items on the drop-down list to output. • All

Field	Description
	• ModuleProperties
	• _processContext
	Only activities that have output and are executed prior to the **Inspector** activity are visible in the drop-down list.
Stdout	Specifies whether the output of this activity should be sent to stout (standard output). Select this checkbox to send the output to **stdout**.
	If this checkbox is not selected, the output is not sent to **stdout**.
Append to File	Select this checkbox to specify that the output is appended to the file specified in this activity's input.
	Not selecting this checkbox overwrites any file that exists with the specified name.
Suppress Header	Select this checkbox to suppress the output of header information, such as the process instance ID and the timestamp.
	If this checkbox is not selected, the header information for the current process instance is written before the output of the activity or module property.

Description

Provide a short description of the Inspector activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
fileName	string	The absolute path to the file you want to write. This file contains the output of the specified activity or module property.

Output

The following is the output of this activity.

Output Item	Datatype	Description
output	string	The output of the module property or activity specified on the **General** tab in string form. You can use this output in subsequent activities, in addition to writing the output to a file or stdout. For example, you can use this output as the body of a TIBCO Rendezvous message.

Log

Log is a synchronous activity that writes a message to the log file. For each AppNode and application, there is a log file.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Logger Name	The name of a logger used in the logback.xml file. The value for the **Logger Name** is case sensitive and can be a single name or a hierarchical name.
Log Level	Select one of the following log levels recognized by the process engine. • **Error**: logs error conditions and messages. • **Warning**: is the warning message of an unexpected error in the process. • **Info**: logs informational messages highlighting the application/engine progress.

Field	Description
	• **Debug**: can be used for debug-level messages. • **Fatal**: can be used for fatal-level messages.
Suppress Job Info	If selected, no additional information is added to the log entry. If not selected, each log message appends JobId, ProcessInstanceId, Activity Name, Deployment Unit Name, and Version name to the message text.

Description

Provide a short description of the Log activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
msgCode	string	The error code of the message. This is used as an identifier for the message so that applications can scan the log for the message.
loggerName	string	The name of a logger used in the logback.xml file. The value for **loggerName** is case sensitive and can be a single name or a hierarchical name. If you specify a value in the input schema, that value overrides the value specified for the **Logger Name** field on the **General** tab.
logLevel	string	Select one of the following log levels recognized by the process engine. • **Error**: logs error conditions and messages. • **Warning**: is the warning message of an unexpected error in the process.

Input Item	Datatype	Description
		• **Info**: logs informational messages highlighting the application/engine progress. • **Debug**: can be used for debug-level messages. If you specify a value in the input schema, that value overrides the value specified for the **Log Level** field on the **General** tab.
message	string	The message to be displayed in the log. **Note:** If the message contains non-ASCII data, the default encoding of the Java Virtual Machine (JVM) used by the process engine, is used for text encoding, when writing to the log file.

Mapper

Mapper is a synchronous activity that adds a new process variable to the process. This variable can be an inline schema, primitive element, or a complex element. You can map the data values from the current list of process variables to the elements of the variable with the **Mapper** activity.

General

Specify the name of the activity here.

Description

A short description of the Mapper activity.

Input Editor

The **Input Editor** tab defines the structure of the process variable to add to the process. You can use a simple datatype, or you can define a group of data elements on this tab.

You can also reference an XML schema stored in the project. The data specified here becomes the input and output schema of the **Mapper** activity. This data is then available to other activities in the process.

Input

The input for the activity is defined by the specified data elements on the **Input Editor** tab.

Output

The output for the activity is defined by the specified data elements on the **Output Editor** tab.

Notify

The **Notify** activity allows a process instance to send data to a corresponding process instance containing a **Wait for Notification** activity or the **Receive Notification** process starter. The **Notify Configuration** resource and the key specified on the **General** tab, create the relationship between the **Notify** activity and the corresponding **Wait for Notification** activity or **Receive Notification** activity.

General

The General tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Shared Configuration	The Shared Configuration resource that defines a data schema for this activity. The Shared Configuration resource is required, but the schema can be empty if you do not want to pass data between processes.
Key	The key that coordinates a **Notify** activity with the corresponding **Wait for Notification** or **Receive Notification** activity.

Field	Description
Notification Timeout (msec)	The timeout (number of milliseconds) to keep the information for this **Notify** activity.

Description

Provide a short description of the Notify activity.

Input

This activity contains an input only if there is a schema defined on the **Notify Configuration** shared resource.

Output Item	Datatype	Description
schema	complex	The schema specified by the **Notify Configuration** shared resource on the **General** tab.
		This schema is used to pass data from the process instance containing the **Notify** activity into the process instance containing the related **Wait for Notification** activity or **Receive Notification** activity.

Output

This activity produces no output.

Set Shared Variable

Set Shared Variable is a synchronous activity that you can use to change the value of a shared variable.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Shared Variable Type	Specify the type of the shared variable (**Module Shared Variable** or **Job Shared Variable**).
Shared Variable Name	The module shared variable or job shared variable whose value you want to retrieve.

Description

Provide a short description of the Set Shared Variable activity.

Input

The following is the input for the activity.

Input Item	Description
schema	The value of the shared variable specified in the **Shared Variable** field of the **General** tab is the input for this activity. The schema for the input is determined by the schema specified for the shared variable.

Output

The following is the output for the activity.

Output Item	Description
schema or no output	The value of the shared variable specified in the **Shared Variable** field of the **General** tab is the output for this activity. The schema for the output is determined by the schema specified for the shared variable.

Sleep

Sleep is an asynchronous activity that suspends the process on the current transition for the specified time. For a process with multiple control flows, only the current execution branch of the process is suspended.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.

Description

Provide a short description of the Sleep activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
IntervalInMillisec	integer	The amount of time interval, in milliseconds, to suspend the current branch of the process. The default time interval is 3 minutes.

Timer

Timer is a process starter activity that starts the process at a specific time. You can also specify the processes to be started periodically.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Start Time	Yes	The day and time to start the process. To run the process periodically, the start time indicates the first time to run the process. The local machine's time zone is used to determine the process start. If the project is running on machines in different time zones, the time zone of the machine where the project was saved, is used. Use a process property in this field to provide start times relative to the time zone where the project is deployed. If you provide a process property for this field, the format of the specified time must be: `yyyy MMM dd HH:mm:ss z`
Run Once	None	Indicates this process should be run only once on the day and time indicated by the **Start Time** field. If you do not select this checkbox, the **Time Interval** and **Interval Unit** fields are displayed that you can use to specify the frequency of the process.
Cron Job	No	On selecting this checkbox, a job is scheduled as per the expression mentioned in the **Cron Expresion** field.
Use local time	None	On selecting this checkbox, the local time of the container is considered.
Cron Expression	Yes	This field is visible only when the **Cron Job** checkbox is selected. A cron expression is a string comprised of 6 or 7 fields separated by a white space. Fields can contain any of the allowed values, along with various combinations of the allowed special characters for that field. For more information, see https://cron-

Field	Literal Value/Module Property/Process Property	Description
		job.org/en/. For example, a cron expression can be as follows: `* * * * ? *` or `0/5 14,18,3-39,52 * ? JAN, MAR, SEP MON-FRI 2002-2010`
End Time for Cron Job	Yes	This field is visible only when the **Cron Job** checkbox is selected. The end time for the cron job.
Time Interval	Yes	The integer indicating the number of units specified in the **Interval Unit** field. For example, if a value of **1** is specified and the **Interval Unit** field is specified as **Hour**, a new process is started every hour beginning at the time specified in the **Start Time** field. • If a value of **2** is specified and the **Interval Unit** field is specified as **Month**, a process is started every other month, beginning at the time specified in the **Start Time** field. • If the value specified is less than one second (1000 milliseconds), the timer ignores the specified interval and is triggered once every second. **Note:** Using **Literal Value**, **Module Property**, and **Process Property** in this field, specifies the time interval in milliseconds and the **Interval Unit** field is not displayed.
Interval Unit	None	The unit of time to use with the **Time Interval** field to determine how often to start a new process. The units can be: **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, and **Year**.

Field	Literal Value/Module Property/Process Property	Description
End After	None	Specify the number of occurrences or the end time here. Select from the following available options: • **Never**: Click the **Never** radio button if you want to continuously execute the process instance. For example, if the Time Interval value specified is 5 and the Interval Unit specified is Second, the process instance executes every 5 seconds. • **Occurrences**: Clicking the **Occurrences** radio button displays the **Occurrences** field. Specify the number of occurrences of the process instance execution. This field can be configured with **Literal Value**, **Module Property**, and **Process Property**. • **End Time**: Clicking the **End Time** radio button displays the **End Time** field. Specify the end time and date when you want the process to end. This field can be configured with **Literal Value**, **Module Property**, and **Process Property**.

Description

Provide a short description of the Timer activity.

Advanced

The **Advanced** tab has the following fields.

Field	Description
Sequencing Key	This field contains an XPath expression that specifies which processes should run in sequence. Process instances with sequencing keys evaluating to the

Field	Description
	same value, are executed sequentially in the sequence the process instance was created.
Custom Job Id	This field contains an XPath expression that specifies a custom job ID for every job in the process instance.

Output

The following is the output of the activity.

Output Item	Description
TimerOutputSchema	The time the process instance started.
	The time is represented by several items in the output schema indicating:
	• Now
	• Hour
	• Minute
	• Second
	• Week
	• Month
	• Year
	• Date
	• Time
	• DayOfMonth
	Each of these items uses the appropriate datatype.

Wait for Notification

The **Wait for Notification** activity suspends execution of the process instance and waits for a **Notify** activity with a matching key to be executed in another process instance. The key specified in the **General** tab and the **Notify Configuration** resource, creates a relationship between the **Wait for Notification** activity and the corresponding **Notify** activity.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as a label for the activity in the process.
Shared Configuration	The shared configuration resource that defines a data schema for this activity.
	The shared configuration resource is required, but the schema can be empty if you do not want to pass data between processes.
Key	The key to coordinate a **Wait for Notification** activity with the corresponding **Notify** activity.
Activity Timeout (msec)	The timeout (number of milliseconds) for this **Wait for Notification** activity.

Description

Provide a short description of the Wait for Notification activity.

Input

This activity has no input.

Output

This activity contains an output only if there is a schema defined on the **Notify Configuration** shared resource.

The following is the output for the activity.

Output Item	Datatype	Description
schema	complex	The schema specified by the **Notify Configuration** shared resource on the General tab. This schema is used to pass data from the process instance containing the related **Notify** activity into the process instance containing the related **Wait for Notification** activity.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
ActivityTimedOutException	A timeout has been reached.

On Notification Timeout

The **On Notification Timeout** process starter specifies a process to execute, when a timeout is reached for storing notification data for a **Notify** activity.

You can specify an **On Notification Timeout** process for a specific shared configuration resource. The process with the **On Notification Timeout** process starter executes, when a **Notify** activity with a matching shared configuration undergoes a timeout.

It is recommended that you do not create more than one process for the same shared configuration resource. If a **Notify** activity experiences a timeout of it's stored notification information, that timeout can only apply to one **On Notification Timeout** process.

When the notification expires on timeout, the **On Notification Timeout** process starter is not triggered immediately. A background thread runs at a regular interval, and checks for expired messages. This thread runs at the default time interval of 30 minutes. To change this default time interval, which is specified in minutes, change the value in the optional

property bw.engine.activity.signalin.eventTimeout.purge.interval=30

> **ⓘ** **Note:** If the engine is running in memory mode, set the interval time to be between one to five minutes. If the engine is running in datastore mode, set the interval time to be anywhere from five to thirty minutes.

To change the default value, uncomment the property in the AppSpace `config.ini` file or pass it as a VM argument in TIBCO Business Studio for BusinessWorks.

You can also specify the **On Notification Timeout** process to apply to any **Notify** activity experiencing a timeout. If there is no **On Notification Timeout** specified for the shared configuration resource of the **Notify** activity, the process engine calls the **On Notification Timeout** process with the **Any Notify Config** field selected.

General

The General tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the resource.
Shared Configuration	This field is enabled only when the **Any Notify Config** checkbox is not selected.
	This field allows you to specify the shared configuration that this process applies to. Only timeouts for **Notify** activities with a matching shared configuration starts a process instance.
	Only one **On Notification Timeout** process can correspond to any shared configuration. You cannot create multiple processes using the same shared configuration.
Any Notify Config	Selecting this checkbox disables the **Shared Configuration** field.
	When the checkbox is selected, notifications from any **Notify** activity, associated with any shared configuration trigger the **On Notification Timeout** process starter.
	When this checkbox is not selected, notifications from any **Notify** activity

Field	Description
	associated with the same shared configuration as specified on the activity triggers the **On Notification Timeout** process starter. Do not select this checkbox, if you want to specify a process for a specific shared configuration resource.

Description

Provide a short description of the On Notification Timeout activity.

Output

The following is the output for the activity.

Output Item	Datatype	Description
notificationOutput	object	This element contains the schema specified in the shared configuration of the **Notify** activity that has timed out.

Receive Notification

This is a process starter activity. The **Receive Notification** activity starts a process when another process executes a **Notify** activity with a matching key and a shared configuration resource.

The key specified in the **Key** field of the **General** tab creates a relationship between the **Receive Notification** process starter and the corresponding **Notify** activity. The same shared configuration (**Notify Configuration**) resource must be specified by the corresponding **Receive Notification** and **Notify** activities to pass the data from the process containing the **Notify** activity to the process started with **Receive Notification**. The schema in the shared configuration resource however can be empty, if you do not want to pass data between processes.

General

The General tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Shared Configuration	The shared configuration resource that defines a data schema for this activity.
	The shared configuration resource is required, but the schema can be empty if you do not want to pass data between processes.
Key	The key that coordinates a **Receive Notification** process starter with the corresponding **Notify** activity.
	The key must be a fixed string.

Description

Provide a short description of the Receive Notification activity.

Advanced

The Advanced tab has the following fields.

Field	Description
Sequencing Key	This field contains an XPath expression that specifies which processes should run in order. Process instances with sequencing keys evaluating to the same value, are executed sequentially in the order the process instance was created.
Custom Job Id	This field contains an XPath expression that specifies a custom job ID for every job in the process instance.

Output

The following is the output for the activity.

Output Item	Datatype	Description
schema	complex	The schema specified in the **Shared Configuration** field on the **General** tab. This schema is used to pass data from the process instance containing the **Notify** activity into this process instance.

HTTP Palette

Using the HTTP Palette you can send and receive HTTP requests.

HTTP Receiver

HTTP Receiver is a process starter activity that starts the process based on the receipt of an HTTP request.

> ℹ️ **Note:** When configuring an **HTTP** activity, configuring an **HTTP Connector** shared resource as the value for a module property is not supported. Configure a literal value instead.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process property	Description
Name	None	The name to be displayed as the label for the activity in the process.
HTTP Connection	Yes	The **HTTP Connector** resource describes the characteristics of the connection used to receive incoming HTTP requests. For more information about specifying shared resources, see HTTP Connector.
Context Path	Yes	This is the prefix of a URL path that is used to select

Field	Literal Value/Module Property/Process property	Description
		the contexts to which an incoming request is passed. You can either specify both **Context Path** and **Path Spec**, any, or none of these elements. For example, the path displays as the `http://hostname.com/contextPath/pathSpec` format. If the **Context Path** is not specified, the **Context Path** is referred to as the root context. In a Send HTTP POST request where a **Path Spec** is not provided, add a forward slash (/) after the **Context Path** (for example: `http://<host>:<port>/<contextPath>/`). If you do not add the forward slash, the POST request is redirected as a GET request for the resource at `/<context_path>`. In a Send HTTP POST request where a **Path Spec** is provided no forward slash (/) is required (for example: `http://<host>:<port>/<contextPath>/<pathSpec>`).
Path Spec	Yes	If specified, it is added as a prefix of a URL of the form `http://hostname.com/contextPath/pathSpec`.
Output Style	None	The type of output. It can be either **String**, **Binary**, or **File**. **Note:** - If you are running POST/PUT requests using HTTP 2.0 (a binary protocol), you must set this field to **Binary**. - The File type output is used for HTTP Streaming.
Parse Post	None	Specifies to parse the message body of the HTTP

Field	Literal Value/Module Property/Process property	Description
Method Data		request into a schema for the output of the activity.
		When this checkbox is selected, the parameters specified in the **Parameters** field are used to validate the incoming request and parse it into the output schema. This field is present only when the **Output Style** is **String**.
		Note: For an incoming GET request, there is no message body, so this field has no effect on the incoming GET requests.
Parameters	None	The parameters of the incoming HTTP request.
		If you specify parameters in this table, the incoming request is parsed and represented as a schema in this output of the activity. For each parameter, you must provide a **name**, **datatype**, and whether the field is **Optional**, **Required**, or **Repeating**.
		Use the ⊞ and ✖ buttons to the right of the parameter table to add and remove parameters.
		Note: Specifying parameters in this field parses the query string of the request into the output schema, regardless of whether the **Parse Post Method Data** checkbox is selected. Selecting the **Parse Post Method Data** checkbox and specifying parameters here is required to parse the message body along with the query string of the request.
Expose Security Context	None	Places the information from the user's security context (either authentication or SSL certificate information) into the Context or SecurityContext output element.

Field	Literal Value/Module Property/Process property	Description
		This is a fairly expensive operation and SSL certificates can consume memory resources. Hence, select this option only if you require information from the user's security context for later use in your process.
Default Encoding	Yes	Specifies the encoding to use if no charset is specified in the **Content-Type** header of the message. This encoding is used for the message body and URL. However, when **URIEncoding** is specified in the HTTP Connector shared resource, the **Default Encoding** parameter specified here applies to the contentType alone and not to the **URIEncoding** property.

Description

Provide a short description for the HTTP Receiver activity.

Advanced

The **Advanced** tab has the following fields.

Field	Process Property/Module Property	Description
Fill Standard Headers	None	When this checkbox is selected, additional headers such as **Authorization**, **User-agent**, and **Host** are populated in the **Dynamic Headers** element in the server response received from the client.
Apply Policy to Subpaths	None	Select this checkbox to apply policy to sub paths.

Field	Process Property/Module Property	Description
Buffer Size	None	The default buffer size is 1024 bytes.
Write to File	None	Select this checkbox to write incoming requests that exceed the specified threshold size to a file instead of storing the request in memory. You can accept large incoming requests without consuming a great deal of memory. Selecting this checkbox displays the **Directory**, **Creating Non-Existing Directories**, and **Threshold Data Size(bytes)** fields. **Note:** This option is not intended to be used with the **Parse Post Method Data** option on the **General** tab. When you select **Write to File**, the **PostData** output element becomes a choice element containing either the output **fileName** or the **PostData**. This depends upon whether the data exceeds the size specified in the **Threshold Data Size** field. It is recommended to use either the **Write to File** option or the **Parse Post Method Data** option, but not both at the same time. Not selecting this field keeps the incoming requests in memory. **Note:** After writing, the files created using this option are not deleted automatically. You must manage the storage used by these files and delete them when they are no longer used.
Directory	Yes	The directory to write messages that are above the specified threshold. The process engine does not attempt to create the directory if the specified directory does not exist. Therefore, create the directory before starting the process engine.
Create Non-	None	When this checkbox is selected, all directories in the

Field	Process Property/Module Property	Description
Existing Directories		path specified in the **Directory** field are created, if they do not already exist.
		If this checkbox is not selected and there are one or more directories in the specified path in the **Directory** field that do not exist, an exception is raised.
Threshold Data Size (bytes)	Yes	The maximum size (in bytes) of an incoming request that can be kept in the memory. Requests larger than the specified size are written to a file in the specified directory. The file's name is output so that subsequent activities in the process can access the file and read its contents.
		Specify zero (0) in this field for all incoming requests to be saved to a file.
		Note: If configured with Module Property, the data type of the value should be **Long**.
Sequence Key	None	This field can contain an XPath expression that specifies which processes should run in sequence. Process instances with sequencing keys that evaluate to the same value are executed sequentially in the sequence the process instance was created.
Custom Job Id	None	This field can contain an XPath expression that specifies a custom ID for the process instance.

Special Characters in HTTP Requests

Depending upon the content type of the data for the request, the request can contain URL-encoded data and the server is expected to decode the data. For example, an incoming request may have the following message body:

```
name=John%20Smith&address=500%201%2F2%20Main%20Street
```

If the message body is not parsed (**Parse Post Method Data** checkbox is not selected on the **General** tab), the body of the message remains URL-encoded in the **PostData** output element. If the message body is parsed, the data is decoded before being placed in the **PostData** output element, and that element would display as follows:

```
name=John Smith&address=500 1/2 Main Street
```

> **Note:** The sequence of the parameters may not be the same in the **PostData** output element as the sequence in the original HTTP request sent by the client when the message body is parsed.

The **PostData** output element can contain different data and be a different length depending upon whether the message body is parsed. Hence, ensure that you obtain data from the parameters output element when the message body is parsed.

Output Editor

The **Output Editor** tab describes the data structure for the headers of the HTTP request. You can use the default structure, or you can alter the structure, if the incoming request has a specific data structure for the header of the request.

The header structure is defined by the HTTP protocol. For more information about the fields and content of the header of an HTTP request, see the HTTP specification at w3.org.

Output Header Element	Datatype	Description
Accept	string	This field specifies media types that are acceptable for response messages for the incoming request. For example, `text/*,text/html`. Media types are described in the HTTP specification at w3.org. If no **Accept-Header** field is present, it is assumed that the client accepts all media types.
Accept-Charset	string	This field specifies the character sets that are acceptable for response messages for the incoming request. For example, iso-8859-5, unicode-1-1. Character sets are described in the HTTP specification at w3.org.

Output Header Element	Datatype	Description
		If no **Accept-Charset** header is present, it is assumed that the client accepts any character set.
Accept-Encoding	string	This field specifies the content-coding values that are acceptable for response messages. For example, `compress`, `gzip`. For more information about this header field, see the HTTP specification at w3.org.
Content-Type	string	This field indicates the media type of the entity body sent to the receiver. For example, `text/html; charset=ISO-8850-4`. Media types are described in the HTTP specification at w3.org. **Note:** The option to filter out ContentType for HTTP Receiver activity is provided by using one of the following system type properties: Set the following system property or application property for Content Type validation in HTTP: Global Level - `bw.http.content.type.validation.list` Application Level - `-Dbw.http.content.type.validation.list(Application_name).application="text/plain"(Content_type)`. If both the properties are provided, then application level property takes precedence over the global level property.
Content-Length	string	This field indicates the size of the entity body (in decimal number of OCTETs) sent to the receiver. This field takes into account the encoding of the message body. For more information about when the message body is URL encoded, see Special Characters in HTTP Requests.

Output Header Element	Datatype	Description
Connection	string	Using this field the requestor can specify options required for this connection. For example, the option close specifies that the requestor would like the connection to be closed when the request is complete.
Cookie	string	If you want to receive more than one cookie, set the cardinality for this field to repeating (*). **Note:** For correct parsing and processing of cookies, input must be in the format "key=value". For more information about this field, see the HTTP specification at w3.org.
Pragma	string	This field is used to include implementation-specific directives that might apply to the receiver. For more information about this field, see the HTTP specification at w3.org.

Output

The following is the output for the activity.

Output Item	Datatype	Description
Method	string	The method specified in the request. For example, GET or POST.
RequestURI	string	The address portion of the request. At runtime it will contain the Query String after the question mark (?), if it is sent in the HTTP Request.
HTTPVersion	string	The version field of the HTTP request.

Output Item	Datatype	Description
PostData or BinaryContent or FileName or FilePath	string	The message body of the HTTP request. The content of this element depends upon whether the message body is parsed. Parsing behavior is controlled by the **Parse Post Method Data** field on the **General** tab. The content of this element is PostData if the **Output Style** field selected is of type String, and the element is BinaryContent if the **Output Style** field selected is of type Binary. For more information, see Special Characters in HTTP Requests.
QueryString	string	The query string portion of the request. This is the portion after the question mark (?).
Header	string	The header of the HTTP request.
Protocol	string	This can be HTTP or HTTPS depending upon the protocol used by the request.
Port	string	The port number on which the request was received. This is the port number configured in the HTTP Connector resource.
Headers	complex	The fields of the header specified on the **Output Editor** tab. For more information about these items, see the description of the Output Editor tab.
Parameters	complex	The parameters for the incoming request. These parameters are configured in the **Parameters** field on the **General** tab. When the **Parse Post Method Data** checkbox on the **General** tab is selected, the parameters include the parameters in the query string and

Output Item	Datatype	Description
		message body of the incoming HTTP request.
		When the **Parse Post Method Data** checkbox is not selected, the parameters include only the query string of the incoming request.
DynamicHeaders	complex	The dynamic header is an additional header parameter to receive header parameters from incoming HTTP requests. This element is specified in the **Output Editor** tab. The **DynamicHeaders** consists of the following information: • **Name**: the name of the header (case insensitive) • **Value**: the value of the header The following overriding conditions can be considered: • Overrides the value of HeaderName with the value found in **DynamicHeaders**, if it is a non-repeating header. If more than one occurrence of this header is found under **DynamicHeaders**, it generates the "`The header [ headerName ] is defined as non-Repeating Header in Input Headers. This header cannot have multiple occurences in DynamicHeaders.`" exception. • If it is a repeating element, add the respective name value pairs under dynamic headers, which is added to the existing list maintained for this element. • For a repeating element, if the new header name is not found under the Headers section declared through TIBCO Business Studio for BusinessWorks, the following is added into the HTTP Headers.

Output Item	Datatype	Description
		◦ Shows only one value, if found once in **DynamicHeaders**. ◦ Shows array of values, if found repeating in **DynamicHeaders**.
mimeEnvelopeElement	complex	This element contains the message attachments. This element contains a repeating element named **mimePart** that contains each mime attachment.

mimeHeaders	complex	This element contains the mime header for each **mimePart**. Mime headers can contain the following information:
		
		  • content-disposition: To suggest a filename for an attachment, use "*;filename=<filename>" in this element.
		
		    Note: HTTP servers may alter or ignore the suggested name.
		
		  • content-type
		
		  • content-transfer-encoding
		
		  • content-id
		
		  • any element
		
		For more information about MIME headers and their syntax, see http://www.faqs.org/rfcs/rfc2045.html.

Output Item	Datatype	Description
		Note: When the content type is specified as "text/*" (for example, "text/xml"), the attachment content is expected to be in either the **textContent** input element or the file name storing the attachment is expected to be in the **fileName** input element. When the content type is anything other than "text/*", the attachment content is expected to be in either the **binaryContent** input element or the file name storing the attachment is expected to be in the **fileName** input element.
binaryContent | textContent | fileName | mimeEnvelopeElement	choice	This element contains the mime attachment. The element can be one of the following: • **binaryContent**: content of the attachment content of the attachment when the attachment is binary data. • **textContent**: content of the attachment when the attachment is text data. • **fileName**: the file name of the attachment written to the disk. • **mimeEnvelopeElement**: If a server sends a response with nested attachments, it will reflect in this part at runtime. This element contains the message attachments. This element contains a repeating element named **mimePart** that contains each mime attachment. **Note:** To see this element in existing applications, we have to modify the activity to enable the **Save** button.
Context	complex	Contains information about the client's environment.

Output Item	Datatype	Description
		This element holds the optional **RemoteAddress** element.
RemoteAddress	string	The IP address of the client that submitted the HTTP request.

Threading Policy Details

The following are the details about the **HTTP Receiver** activity threading policy.

Activity Type	Multi-Threaded?	Default Thread Count	Thread Count Configuration
ProcessStarter	Yes	• Minimum QTP Threads = 10 • Maximum QTP Threads = 75	It can be configured on the **HTTP Connector** shared resource using the following fields: • Minimum QTP threads • Maximum QTP threads

Send HTTP Request

Send HTTP Request is an asynchronous activity that sends an HTTP request and waits for a response from the web server. This activity sends a request to a server that is compliant with either the HTTP 1.0, 1.1, or 2.0 specification. HTTP 2.0 is supported only when the HTTP version at the client shared resource is set to HTTP 2. TIBCO BusinessWorks Container Edition automatically sends the request using the correct version based on the version supported by the HTTP server. Therefore, do not specify the HTTP version of the server you are sending the request to.

> **Note:** When configuring an **HTTP** activity, configuring an HTTP Connector shared resource as the value for a module property is not supported. Configure a literal value instead.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
HTTP Client	Specifies the **HTTP Client** shared resource. For more information, see HTTP Client.
Parameters	The parameters of the HTTP request. For each parameter, you must provide: • **Parameter Name** • **Parameter Type** as string • **Parameter Cardinality** as **Optional**, **Required**, or **Repeating** These parameters are specified in the parameters element on the **Input** tab.
Post Data Type	The body of the HTTP message. You can either select **String**, **Binary**, or **File** format.

Description

Provide a short description for the Send HTTP Request activity.

Advanced

The **Advanced** tab has the following fields.

Field	Literal Value/ Module Property	Description
Provide Client Resource	None	Select this checkbox to dynamically select the needed HTTP client resource. When you select this checkbox, **Host** and **Port** fields disappear, and the **ClientResource** field appears in the **Input** tab.
Buffer Size	Yes	Specify the size of the buffer for streaming of chunk length payload. The streaming value must be in bytes, for example, 1024 or 2048 bytes. **Note:** By default, the chunk length for Apache HTTP Component is 4096 bytes, while for Jetty component it is customizable.
Write to File	None	Select this checkbox to write the incoming requests that exceed the specified threshold size to a file instead of storing the request in memory. You can accept large incoming requests without consuming a great deal of memory. Selecting this checkbox displays the **Directory** and **Threshold Data Size** fields. **Note:** • To keep the incoming requests in memory, do not select this checkbox. • The files created using this option after writing are not deleted automatically. You must manage the storage used by these files. Delete these files when they are no longer in use.
Directory	Yes	The directory to write messages that are above the specified threshold. The process engine does not attempt to create the directory if the specified directory does not exist. Therefore, create the directory before starting the process engine.
Creating	None	Selecting this checkbox creates all directories in the path

Field	Literal Value/ Module Property	Description
Non-Existing Directories		specified in the **Directory** field, if they do not already exist.
		Not selecting this checkbox with one or more non-existing directories in the specified path in the **Directory** field, raises an exception.
Threshold Data Size (bytes)	Yes	The maximum size (in bytes) of an incoming request to be kept in memory. Requests larger than the specified size are written to a file in the specified directory. The file's name is the output so that subsequent activities in the process can access the file and read its contents.
		Specifying zero (0) in this field causes all incoming requests to be saved to a file.
		Note: If configured with Module Property, the data type of the values should be **Long**.
Write Non MIME Content	None	Select this checkbox to save the non-MIME attachments or data downloaded from the server to a disk.

Input Editor

 Input Editor comprises the data structure of the headers of the HTTP request and HTTP reply message. You can either use the default structure or modify the structure, if the outgoing request or the reply to the request has a specific data structure for the header.

> **Note:** If you add a new primitive element, and that element is not a defined HTTP standard header, the server lists the element in the DynamicHeaders section.

Input Header	Datatype	Description
Accept	string	This field specifies media types that are acceptable for response messages for the request. For example, `text/*`, `text/html`. Media types are described in the HTTP specification at w3.org. If no **Accept-Header** field is specified, all media types are acceptable on the server.
Accept-Charset	string	This field specifies the character sets that are acceptable for response messages for the request. For example, `iso-8859-5`, `unicode-1-1`. The character sets are described in the HTTP specification at w3.org. If no **Accept-Charset** header is specified, any character set is acceptable on the server.
Accept-Encoding	string	This field specifies the content-coding values that are acceptable for response messages. For example, `compress` and `gzip`. For more information about this header field, see HTTP specification at w3.org.
Content-Type	string	This field indicates the media type of the entity body for the outgoing message and the incoming response. Media types are described in the HTTP specification at w3.org. An example of the media type is `text/html; charset=ISO-8850-4`.
Cookie	string	A name and value pair (also known as a cookie) containing information that the HTTP server may be expecting. For correct parsing and processing of cookies, input must be in the format "key=value". You can set the cardinality for this element to Repeating (*) to specify more than one cookie. You can also specify multiple name or value pairs in a single non-repeating element by separating each pair with a comma (for example, `"name1=value1, name2=value2"`).
Pragma	string	This field is used to include implementation-specific directives

Input Header	Datatype	Description
		that might apply to the receiver.
		For more information about this field, see the HTTP specification at w3.org.

Output Editor

Output Editor comprises the data structure of the headers of the HTTP request and HTTP reply message. You can either use the default structure or modify the structure, if the outgoing request or the reply to the request has a specific data structure for the header.

The HTTP protocol defines the header structure. For more information about the fields and content of the header of an HTTP request, see the HTTP specification at w3.org.

Output Header	Datatype	Description
Allow	string	This field lists the set of methods supported by the resource identified by RequestURI. The cardinality for this field is set to Repeating (*). **Note:** To see this element in the existing applications, modify and save the activity to get the changes reflected.
Content-Type	string	This field indicates the media type of the entity body for the outgoing message and the incoming response. Media types are described in the HTTP specification at w3.org. An example of the media type is `text/html; charset=ISO-8850-4`.
Content-Length	string	This field indicates the size of the entity body (in decimal number of OCTETs) of the response message.
Content-	string	This field is used as a modifier to the content-type. When

Output Header	Datatype	Description
Encoding		present, its value indicates what additional content encoding has been applied to the entity-body. Also, what decoding mechanisms must be applied to obtain the media-type referenced by the **Content-Type** header field. **Content-Encoding** is primarily used to allow a document to be compressed without losing the identity of its underlying media type. For more information about this field, see the HTTP specification at w3.org.
Date	string	The date and time when the response message was sent.
Location	string	This field is used to redirect the receiver to a location other than RequestURI for completion of the request or for identification of a new resource.
Set-Cookie	string	The cardinality for this field is set to Repeating (*). **Note:** For correct parsing and processing of cookies, input must be in the format "key=value". For more information about this field, see the HTTP specification at w3.org.
Pragma	string	This field is used to include implementation-specific directives that might apply to the receiver. For more information about this field, see the HTTP specification at w3.org.

Input

The following is the input for the activity.

Input Item	Datatype	Description
ClientResource	string	This field appears when you select the **Provide Client Resource** check box on the **Advanced** tab. Provide the HTTP client shared resource name to which you want to connect dynamically. The client resource name to be provided in the input tab should be of the form `<Package Name>.<Resource Name>`. **Note:** Do not provide `.httpClientResource` extension in the client resource name.
Host	string	The HTTP host you want to connect to. Specifying a value for this input item overrides any value specified on the **General** tab.
Port	string	The port number that the HTTP server uses for incoming requests. Specifying a value for this input item overrides any value specified on the **General** tab.
Method	string	The HTTP method to use for the request. All HTTP 1.1 methods are supported, but the connect method is unnecessary because TIBCO® Cloud Integration automatically uses the CONNECT method when connecting through a proxy server. If no method is specified in this element, the GET method is used by default.
RequestURI	string	The address portion of the request. This is the portion before the question mark (?)
PostData	string	The message body of the HTTP request. Do not specify this element when the method of the request is GET. For more information about this input element, see Sending Data in the HTTP Request

Input Item	Datatype	Description
QueryString	string	The query string portion of the request. This is the part after the question mark (?). For more information about this input element, see Sending Data in the HTTP Request
Timeout	integer	The amount of time (in milliseconds) to wait for a response from the HTTP server.
Headers	complex	The header fields to send for the request. When this element is specified on the **Input Editor** tab, you can use this input item to provide values for the header when sending the request.
DynamicHeaders	complex	The dynamic header is an additional header parameter to add runtime headers to the outgoing HTTP messages. This element is specified in the **Input** tab. **Note:** You can use only US-ASCII characters in **DynamicHeaders**. The **DynamicHeaders** consists of the following information: • **Name**: the name of the header • **Value**: the value of the header You can consider the following overriding conditions. • Overrides the value of HeaderName with the value found in **DynamicHeaders** if it is a non-repeating header. If more than one occurrence of this header is found under **DynamicHeaders**, it generates the following exception. The header [headerName] is defined as non-Repeating Header in Input Headers. This header cannot have multiple occurrences in DynamicHeaders.

Input Item	Datatype	Description

		• If it is a repeating element, add the respective name value pairs under dynamic headers, which is then added to the existing list maintained for this element.
		
		• For a repeating element, if the new header name is not found under the "Headers" section declared through TIBCO Business Studio for BusinessWorks, the following is added into HTTP headers.
		
		    ○ Shows only one value, if found once in **DynamicHeaders**.
		
		    ○ Shows an array of values, if found repeating in **DynamicHeaders**.
		
		If you add any Standard HTTP Header as a name value pair in the Dynamic Header section in the client end, on the server side this is listed under the Headers section.
		
		The Dynamic Headers can contain the following information:
		
		content-disposition: to suggest a filename for a streaming file, use the `"attachment;filename=<filename>"` in this element.
		
		Note: The HTTP servers may alter or ignore the suggested name.
		
		To Send the Compress Request with GZIP or Deflate add the content-encoding property through dynamic headers with appropriate compression type. Select the **Compression** checkbox on the HTTP Connector Shared Resource for server side compression.

Advanced	complex	The **Advanced** field is an additional field to add the Date Time pattern format of the `Expires` attribute inside cookie for parsing. This element is specified in

Input Item	Datatype	Description
		the **Input** tab. The **Advanced** field consists of the following information:
		`CookieExpiresDatePattern`: Date Time pattern format of the `Expires` attribute.
		Note: Whenever the `Expires` attribute value of cookie consists of a comma (,), enable the **Use Single Cookie Header** check box in the **HTTP Client** Shared Resource.
parameters	complex	This element contains parameters added in the **Parameters** field in the **General** tab.
mimeEnvelopeElement	complex	This element contains the message attachments. This element contains a repeating element named **mimePart** that contains each mime attachment.
mimeHeaders	complex	This element contains the mime header for each **mimePart**. mimeHeaders can contain the following information: • content-disposition: To suggest a filename for an attachment, use "*;filename=<filename>" in this element. **Note:** The HTTP servers may alter or ignore the suggested name. • content-type • content-transfer-encoding • content-id For more information about MIME headers and their syntax, see http://www.faqs.org/rfcs/rfc2045.html.

Input Item	Datatype	Description
		Note: When the content type is specified as "text/*" (for example, "text/xml"), the attachment content is in either the **textContent** input element or the file name storing the attachment is in the **fileName** input element. When the content type is anything other than "text/*", the attachment content is in either the **binaryContent** input element or the file name storing the attachment is in the **fileName** input element.
binaryContent | textContent | fileName | mimeEnvelopeElement	choice	This element contains the mime attachment. The element can be one of the following: • **binaryContent**: content of the attachment when the attachment is binary data. • **textContent**: content of the attachment when the attachment is text data. • **fileName**: the file name of the attachment written to the disk. • **mimeEnvelopeElement**: If a user wants to send a request with nested attachments, they can provide it in this part. This element contains the message attachments. This element contains a repeating element named **mimePart** that contains each mime attachment. **Note:** To see this element in existing applications, we have to modify the activity to enable the **Save** button.
FilePath	string	This element contains the streaming File of the HTTP request. Do not specify this element when the method of the request is GET.

Input Item	Datatype	Description
		Note: The file is used for HTTP Streaming. If you select Jetty as a client and file size is greater than 200 MB, then use `bw.jetty.max.streaming.response.length` system property like – `Dbw.jetty.max.streaming.response.length=1100145600`, where the value is in bytes.

Output

The following is the output of the activity.

Output Item	Datatype	Description
Header	complex	The fields of the header specified on the **Output Editor** tab. For more information about these items, see the description of the Output Editor tab.
statusLine	complex	This field is the first line of the response message. This consists of the protocol version, a numeric status code, and the text phrase explaining the status code. For more information about status codes in the HTTP responses, see the HTTP specification at w3.org.
httpVersion	string	The HTTP method to use for the request. All HTTP 1.1 methods are supported, but the connect method is unnecessary because TIBCO BusinessWorks Container Edition automatically uses the CONNECT method when connecting through a proxy server. If no method is specified in this element, by

Output Item	Datatype	Description
		default the GET method is used.
statusCode	string	HTTP status codes in output represents the status of the service. The codes also help to identify the problems caused. For more information on the standard HTTP status codes, see Status Code and Reason Phrase at w3.org.
reasonPhrase	string	The message body of the HTTP request. Do not specify this element when the method of the request is GET. For more information about this input element, see Sending Data in the HTTP Request
binaryContent	string	The binary content of the response to the request from the HTTP server.
asciiContent	integer	The ASCII content of the response to the request from the HTTP server.
filePath	string	The location of the file on the disk at which the non-MIME attachment is written.
Headers	complex	The header fields of the reply. The structure of this output item is specified on the **Output Headers** tab.
DynamicHeaders	complex	The dynamic header is an additional header parameter to add runtime headers to the outgoing HTTP messages. This element is specified in the **Input** tab. **Note:** You can use only US-ASCII characters in **DynamicHeaders**. The dynamic headers consist of the following information:

Output Item	Datatype	Description
		• **Name**: the name of the header
		• **Value**: the value of the header
		You can consider the following overriding conditions.
		• Overrides the value of **HeaderName** with the value found in **DynamicHeaders** if it is a non-repeating header. If more than one occurrence of this header is found under **DynamicHeaders**, it generates the following exception. `The header [ headerName ] is defined as non-Repeating Header in Input Headers. This header cannot have multiple occurences in DynamicHeaders.`
		• If it is a repeating element, add the respective name or value pairs under **DynamicHeaders**, which is added to the existing list maintained for this element.
		• For a repeating element, if the new header name is not found under the **Headers** section declared through TIBCO Business Studio for BusinessWorks, the following is added into HTTP headers. ○ Shows only one value, if found once in **DynamicHeaders**. ○ Shows an array of values, if found repeating in **DynamicHeaders**.
mimeEnvelopeElement	complex	This element contains the message attachments. This element contains a repeating element named **mimePart** that contains each mime attachment.
mimeHeaders	complex	This element contains the mime header for each

Output Item	Datatype	Description
		mimePart. mimeHeaders can contain the following information: • content-disposition: to suggest a filename for an attachment, use "*;filename=<filename>" in this element. > **Note:** The HTTP servers may alter or ignore the suggested name. • content-type • content-transfer-encoding • content-id For more information about MIME headers and their syntax, see http://www.faqs.org/rfcs/rfc2045.html. > **Note:** When the content type is specified as "text/*" (for example, "text/xml"), the attachment content is expected to be in either the **textContent** input element or the file name storing the attachment is expected to be in the **fileName** input element. When the content type is anything other than "text/*", the attachment content is in either the **binaryContent** input element or the file name storing the attachment is in the **fileName** input element.
binaryContent | textContent | fileName | mimeEnvelopeElement	choice	This element contains the mime attachment. The element can be one of the following: • **binaryContent**: content of the attachment when the attachment is binary data. • **textContent**: content of the attachment when the attachment is text data.

Output Item	Datatype	Description
		• **fileName**: the file name of the attachment written to the disk.
		• **mimeEnvelopeElement**: If a server sends a response with nested attachments, it will reflect in this part at runtime. This element contains the message attachments. This element contains a repeating element named **mimePart** that contains each mime attachment.
		Note: To see this element in existing applications, we have to modify the activity to enable the **Save** button.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
ActivityTimedOutException	A timeout has been reached.
HttpClientException	The HTTP server responds with a message containing the 4XX status code.
HttpServerException	The HTTP server responds with a message containing the 5XX status code.
HttpCommunicationException	An HTTP exception occurred while executing the specified method, or when trying to read the response.

Threading Policy Details

The following are the details about **Send HTTP Request** activity threading policy.

Activity Type	Multi-Threaded?	Default Thread Count	Thread Count Configuration
Asynchronous	Yes	Dynamic	This field can be configured by using the **Thread Pool** shared resource.

Policy

Activities that support policies display the **Policy** tab. To associate a new or existing policy with the **Send HTTP Request** activity, click the **Add Policy to Activity** icon. To edit policy details, click **Go to selected Policy** icon. The **Policy** tab has the following fields.

Field	Description
Name	The name of the policy.
Type	The type of policy associated with the activity. The **Send HTTP Request** activity can support the Basic Credential Mapping policy.
Description	A description of the policy.

Sending Data in the HTTP Request

You can use several HTTP methods in an HTTP request. Each method sends data in the request in a different manner. For example, the GET method uses the query string of the RequestURI to pass parameter and value pairs. Other methods use the HTTP message body to send data in the request.

The **Send HTTP Request** activity has the following three input elements for sending data in a request:

- **Post Data**: corresponds to the body of the HTTP message. All methods except the

GET method accept data in this element.

- **Query String**: corresponds to the query string of the RequestURI. You can use this input element to dynamically construct the query string using an XPath expression when you do not know the names or number of the input parameters for the request until the activity executes.

- **Parameters**: corresponds to the parameters defined in the **Parameters** field on the **General** tab. This is useful if you have a fixed set of parameters to send with the request. For requests using the GET method, these parameters are passed as the query string of the RequestURI. For requests using the POST method, these parameters are usually sent as the body of the HTTP message. They can also be included in the query string.

These input elements are mutually exclusive for some methods. For example, for POST requests, you can either specify parameters on the **General** tab and in the parameters input element or you can specify a **PostData** input element. However, do not specify both input elements. In the case of a POST request, the **PostData** input element is ignored when you specify parameters on the **General** tab.

For GET requests, you can either specify the parameters on the **General** tab and in the parameters input element or you can specify a **QueryString** input element. If you know the list of parameters for the request, configure the parameters on the **General** tab. If the list of parameters is not known until the activity executes, use the **QueryString** element.

However, when you specify all parameters on the **General** tab as **Optional**, you can use the **QueryString** input element instead of the parameters input element. If any element in the parameters element contains an expression, the **QueryString** element is ignored.

Special Characters in HTTP Requests

Depending upon the content type of the data for the request, the request can contain URL-encoded data and the server decodes the data. If this is the case and you want to send special characters such as +, /, or = in your HTTP request, your data string must be URL-encoded if you send the data using the **PostData** or **QueryString** input elements. If you send the data using the parameters specified on the **General** tab, encoding is done automatically.

For example, if you want to specify the following **PostData**:

name=John Smith&address=500 1/2 Main Street,

the **PostData** input element should result in the following string:

name=John%20Smith&address=500%201%2F2%20Main%20Street

For more information about the URL specification, see http://www.rfc-editor.org/rfc/rfc1738.txt.

Send HTTP Response

Send HTTP Response is a synchronous activity that sends a response to a previously received HTTP request. This activity is used in conjunction with the **HTTP Receiver** process starter activity and the **Wait for HTTP Request** activity. The default status line returned by this activity is `200 OK`. For HTTP 2 communication, the default status line returned by this activity is `200 null`.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Reply For	The **Wait for HTTP Request** activity or **HTTP Receiver** process starter activity that received the request. This is a list of available activities that can receive HTTP requests.
Flush Response	You can specify whether the response is to be flushed after each **Send HTTP Response** activity. By default, this property is not selected. This check box is useful when there are a large number of **Send HTTP Response** activities in a job. In such cases, the **Close Connection** check box is selected in the last activity to indicate the response is complete. However, selecting the **Flush Response** check box overrides this behavior and causes the response to be flushed after each **Send HTTP Response** activity.
Close Connection	Specifies that this activity contains the last part of an HTTP response. This

Field	Description
	field is used when you have more than one **Send HTTP Response** activities in a process and each activity sends a part of the response to the client. All **Send HTTP Response** activities in a process should have this check box clear, except the last **Send HTTP Response** activity. Select this check box for the last **Send HTTP Response** activity to indicate the response is complete.
	Select this check box, if the entire response is sent by only one **Send HTTP Response** activity in the process.
Buffer Size	The default buffer size is 1024 bytes.

Description

Provide a short description for the Send HTTP Response activity.

Input Editor

Input Editor tab describes the data structure for the headers of the HTTP response. You can use the default structure, or you can alter the structure, if the outgoing response has a specific data structure for the header of the request.

The header structure is defined by the HTTP protocol. For more information about the fields and content of the header of a HTTP request, see the HTTP specification at w3.org.

The following are the default header fields.

Header	Datatype	Description
StatusLine	string	This field is the first line of a response message. This consists of the protocol version, a numeric status code, and the text phrase explaining the status code.
		For more information about status codes in HTTP responses, see the HTTP specification at w3.org.
Content-Type	string	This field indicates the media type of the entity body sent to the receiver. The media types are described in the HTTP specification at w3.org. An example of the media type is:

199 | HTTP Palette

Header	Datatype	Description
		`text/html; charset=ISO-8850-4.` By default this item is set to `text` or `html`. If you are using this activity and the Retrieve Resources activity to retrieve a WSDL file, you should set the type to `text` or `xml` when specifying values on the **Input** tab.
Set-Cookie	string	For more information about this field, see the HTTP specification at w3.org. **Note:** For correct parsing and processing of cookies, input must be in the format "key=value".
Pragma	string	This field is used to include implementation-specific directives that might apply to the receiver. For more information about this field, see the HTTP specification at w3.org.
Location	string	This field is used to redirect the receiver to a location other than the RequestURI for the completion of the request or for identifying a new resource.

Input

The following is the input for the activity.

Input Item	Datatype	Description
binaryContent	binary	The binary content of the response to the request.
asciiContent	string	The ASCII content of the response to the request.
Headers	complex	The fields of the header specified on the **Input Editor** tab. See the description of the **Input Editor** tab for more information about these items.

TIBCO BusinessWorks™ Container Edition Binding and Palette Reference

Header	Datatype	Description
		`text/html; charset=ISO-8850-4.` By default this item is set to `text` or `html`. If you are using this activity and the Retrieve Resources activity to retrieve a WSDL file, you should set the type to `text` or `xml` when specifying values on the **Input** tab.
Set-Cookie	string	For more information about this field, see the HTTP specification at w3.org. **Note:** For correct parsing and processing of cookies, input must be in the format "key=value".
Pragma	string	This field is used to include implementation-specific directives that might apply to the receiver. For more information about this field, see the HTTP specification at w3.org.
Location	string	This field is used to redirect the receiver to a location other than the RequestURI for the completion of the request or for identifying a new resource.

Input

The following is the input for the activity.

Input Item	Datatype	Description
binaryContent	binary	The binary content of the response to the request.
asciiContent	string	The ASCII content of the response to the request.
Headers	complex	The fields of the header specified on the **Input Editor** tab. See the description of the **Input Editor** tab for more information about these items.

TIBCO BusinessWorks™ Container Edition Binding and Palette Reference

Header	Datatype	Description
		`text/html; charset=ISO-8850-4.` By default this item is set to `text` or `html`. If you are using this activity and the Retrieve Resources activity to retrieve a WSDL file, you should set the type to `text` or `xml` when specifying values on the **Input** tab.
Set-Cookie	string	For more information about this field, see the HTTP specification at w3.org. **Note:** For correct parsing and processing of cookies, input must be in the format "key=value".
Pragma	string	This field is used to include implementation-specific directives that might apply to the receiver. For more information about this field, see the HTTP specification at w3.org.
Location	string	This field is used to redirect the receiver to a location other than the RequestURI for the completion of the request or for identifying a new resource.

Input

The following is the input for the activity.

Input Item	Datatype	Description
binaryContent	binary	The binary content of the response to the request.
asciiContent	string	The ASCII content of the response to the request.
Headers	complex	The fields of the header specified on the **Input Editor** tab. See the description of the **Input Editor** tab for more information about these items.

Input Item	Datatype	Description
		Note: Only the first **Send HTTP Response** activity in the process can set the **Headers** element. If your process contains multiple **Send HTTP Response** activities, the headers are set by the first activity and this element is ignored for all subsequent activities.
DynamicHeaders	complex	The dynamic header is an additional header parameter to add runtime headers to the outgoing HTTP messages. This element is specified on the **Input** tab. **DynamicHeaders** consists of the following information: • Name: the name of the header • Value: the value of the header You can consider the following overriding conditions: • Overrides the value of the **HeaderName** with the value found in the **DynamicHeaders**, if it is a non-repeating header. If more than one occurrence of this header is found under **DynamicHeaders**, it throws the following exception: The header [headerName] is defined as non-Repeating Header in Input Editor. This header cannot have multiple occurences in DynamicHeaders. • If it is a repeating element, add the respective name value pairs under **DynamicHeaders**. This is added to the existing list maintained for this element. • For a repeating element, if the new header name is not found under the **Headers** section in TIBCO Business Studio™ for BusinessWorks™, the following is added into

Input Item	Datatype	Description
		the HTTP headers.
		○ Shows only one value, if found once in the **DynamicHeaders**.
		○ Shows an array of values, if found repeating in **DynamicHeaders**.
		Dynamic Headers can contain the following information:
		content-disposition- to suggest a filename for a streaming file, use the `"attachment;filename=<filename>"` in this element.
		Note: The HTTP servers may alter or ignore the suggested name.
mimeEnvelopeElement	complex	This element contains the message attachments. This element contains a repeating element named **mimePart** that contains each mime attachment.
mimeHeaders	complex	This element contains the mime header for each **mimePart**. mimeHeaders can contain the following information:
		• content-disposition: to suggest a filename for an attachment, use `"*;filename=<filename>"` in this element.
		Note: The HTTP servers may alter or ignore the suggested name.
		• content-type
		• content-transfer-encoding
		• content-id
		• any element

Input Item	Datatype	Description
		For more information about MIME headers and their syntax, see http://www.faqs.org/rfcs/rfc2045.html.
		Note: When the content type is specified as "text/*" (for example, "text/xml"), the attachment content is either the **textContent** input element or the file name storing the attachment is in the **fileName** input element. When the content type is anything other than "text/*", the attachment content is in either the **binaryContent** input element or the file name storing the attachment is in the **fileName** input element.
binaryContent | textContent | fileName | mimeEnvelopeElement	choice	This element is the content of the attachment. when the attachment is binary data. This element contains the mime attachment. The element can be one of the following: • **binaryContent**: content of the attachment when the attachment is binary data. • **textContent**: content of the attachment when the attachment is text data. • **fileName**: the file name of the attachment written to the disk. • **mimeEnvelopeElement**: If a server wants to send a response with nested attachments, it can be provided in this part. This element contains the message attachments. This element contains a repeating element named **mimePart** that contains each mime attachment. **Note:** To see this element in existing applications, we have to modify the activity to enable the **Save** button.

Input Item	Datatype	Description
FilePath	string	This element contains the streaming file of the HTTP response.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes* guide.

Fault	Generated When..
HttpCommunicationException	An HTTP exception occurred when sending the response.

JAVA Palette

The Java palette has activities for executing Java code as well as converting between Java objects and XML documents.

Configuring JAVA in an Application Module

To access JAVA code in TIBCO BusinessWorks™ Container Edition, you need to configure the application module with JAVA nature.

New Project

To configure a new project with JAVA nature:

Procedure

1. On the menu bar, click ![icon] icon.

   This opens the New BusinessWorks Application Module wizard.

2. Specify **Project name** and select the **Use Java configuration** checkbox and click **Finish**.
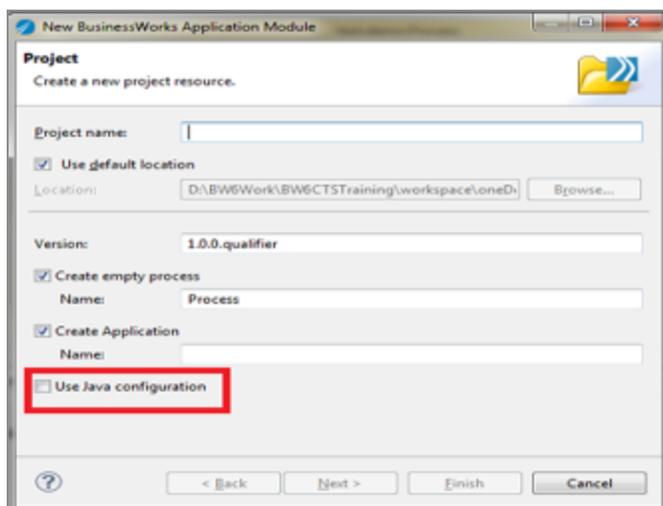
In **Project Explorer** you see a **JRE System Library** *<version>* folder created in your project.

Existing Project

To configure an existing project with JAVA nature:

1. In the **Project Explorer** pane, right-click on the project and select **Configure > Convert to Java Project**.
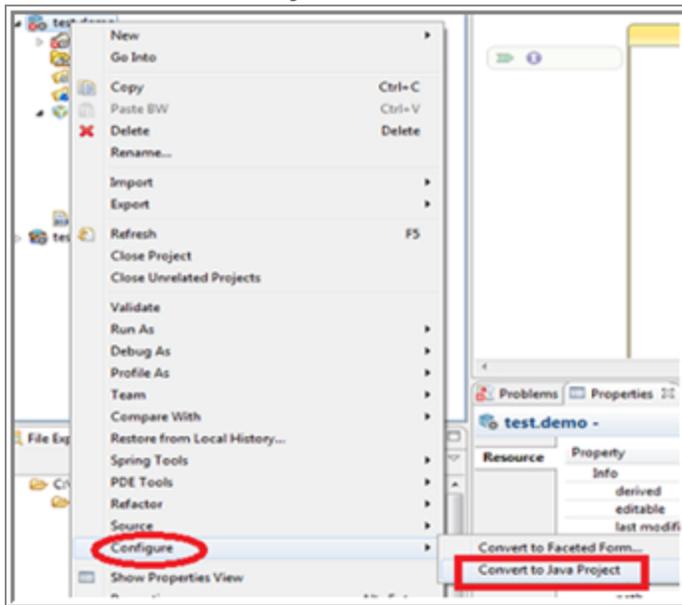
2. Import and run the existing JAVA source (which resides in the JAR file) inside the same application module.

3. In the TIBCO BusinessWorks Container Edition application module (which has java nature), select the **lib** folder.

4. Right-click and select **Import > Import > General > File System**.
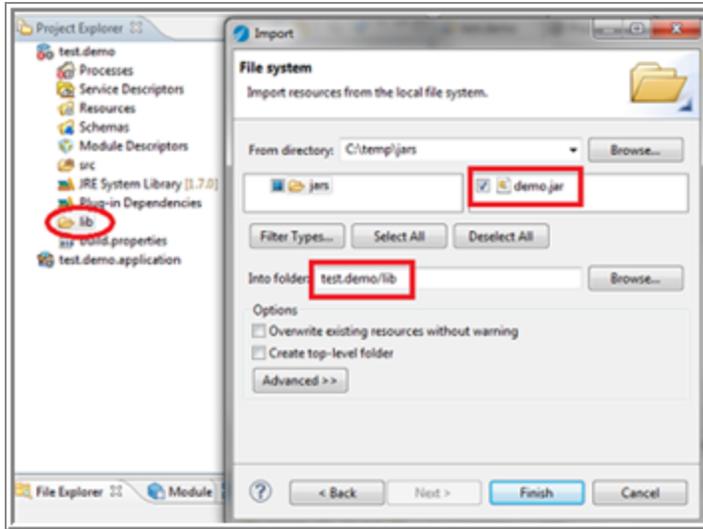
5. Click the **Browse** button of the **From directory** field and provide the location of jar file.

6. Now click the **Browse** button of the **Info folder** field and select the jar file you want to import to the **lib** folder, and click **Finish**.

   This copy and places the jar file in the application module's **lib** directory and adds it to the classpath.

Converting JAVA Class to XML Schema

This topic lists the rules that need to be followed when converting Java class to XML schema.

The Java class is converted to an XML schema using the following rules:

- The Java class public member should not have the public getter and setter methods for it.

- The Java class member public variable name is mapped to an XML element with the same name. For example, a Java class member variable declared as public int ZipCode is mapped to an XML element named *ZipCode*.

- The Java Bean accessors and modifiers are mapped to appropriate XML element names. For example, a Java class method public int getBalance() or public void setBalance(int Balance) are mapped to an XML element named *Balance*.

- Only one XML element is created regardless of how many members of the Java class

share name. For example, there may be an attribute named `MySalary` and accessors named `getMySalary()` and `setMySalary()`. This translates to one element named *MySalary* in the resulting XML document.

- All Java primitive types are supported. Datatypes that extend `java.util.Collections`, are also supported (for example, List, ArrayList, and Vectors). Arrays (for example, `int[ ]` and `string[ ]`) are also supported.

- Use Java generics and provide the type information while declaring and creating the collection for example,

  Instead of using `List list = new ArrayList();`

  use generics: `List<String> list = new ArrayList<String>();`

- The datatype `java.util.Map` or any types that extend `java.util.Map` are not supported. For example, *HashMap* is not supported.

- The variable name defined in the class should be exactly same as the getter and setter methods defined for the same variable. Otherwise, there can be a schema generation issue by **JavaToXML** and **XMLToJava** activities. For example, if a variable name is `airfileRouteStartTime`, names of the getter and setter methods must be `getAirfileRouteStartTime()`, and `setAirfileRouteStartTime()` respectively.

- Class should implement `java.io.Serializable` interface.

- The boolean variable names defined should not start with the keyword `is`. For example, "isemployee". It conflicts with the getter and setter methods `isemployee()` and `setemployee()` and results in the schema generation issue.

Creating Custom Xpath Functions

This topic provides the detailed procedure of creating a custom Xpath function group. You can use the Custom Xpath Function wizard to create your custom Xpath function group. It uses the `com.tibco.xml.cxf.customXPathFunction` extension.

Procedure

1. Open TIBCO Business Studio for BusinessWorks and select **File > New > Project**.

2. In the New Project window, select **Plug-in Project** and click **Next**.
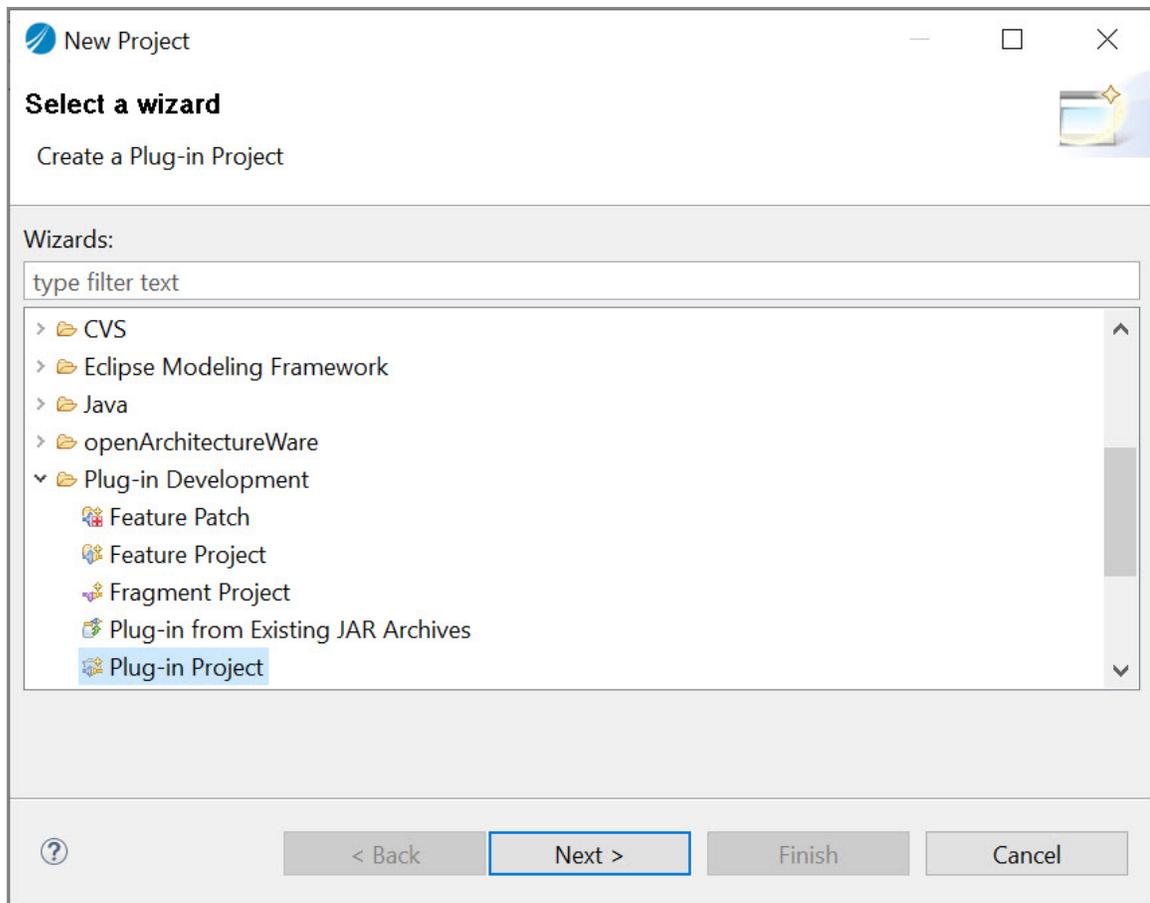
3. Specify a name for the project that reflects the Xpath functions, for example,
MyCustomXPathFunctions. Retain all other default selections and click **Next**.

4. On the Plug-in Content page, locate the **Options** group and select **This plug-in will make contributions to the UI**.

5. Accept all the defaults and click **Next**.

6. Select **Custom Xpath Function Wizard** on the Templates page, and click **Next**.
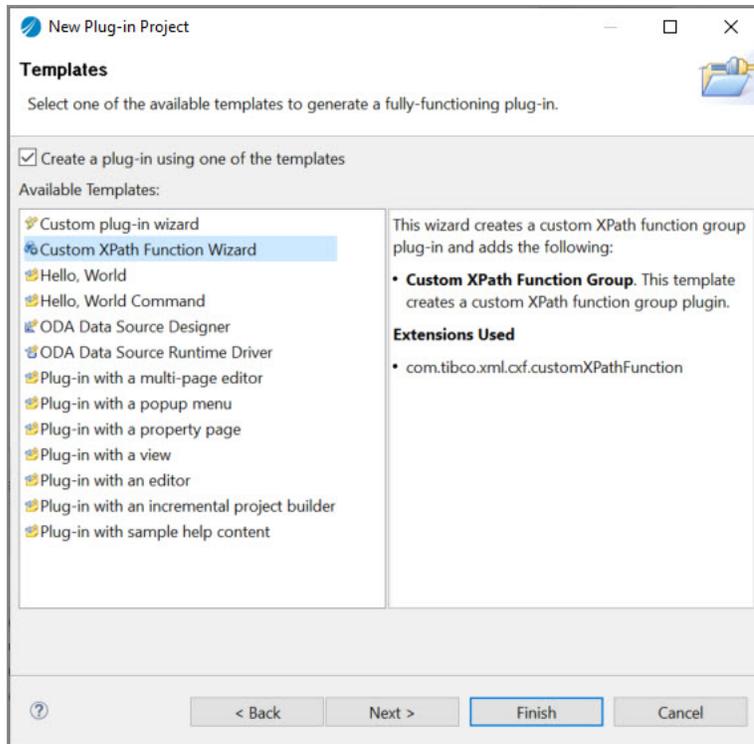
7. In the New Custom Xpath function group plug-in project window, provide values for the following fields and click **Next** to continue.
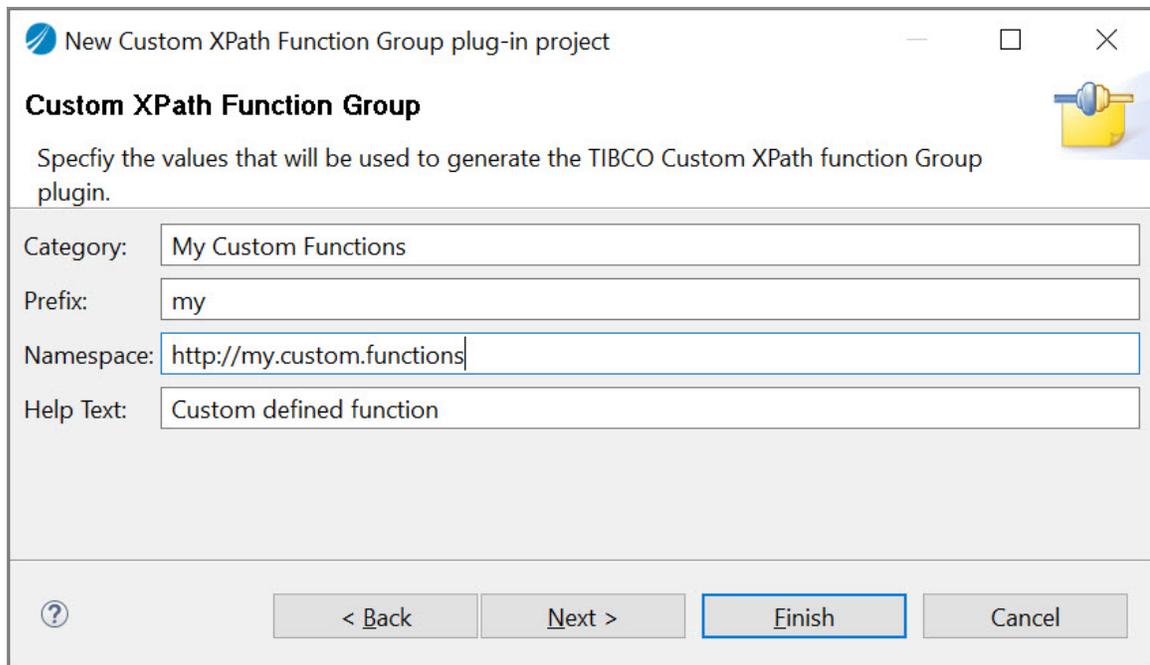
- **Category**: The name of the category that includes the custom Xpath functions

- **Prefix**: The prefix for the functions

- **Namespace**: The namespace for the functions

- **Help Text**: The description of the functions

8. Specify the **XPath Function** and **XPath Function Parameters** in the **XPath Function Group Creation Section** dialog.

9. Click the **Add** button on the right side of the **XPath Functions** table and provide values for:

- **Name**: the name of the function

- **Return Type**: the return type of the function

- **Description**: the description of the function

- **BW XML Types**: Select one of the XML types, such as item, node, item-set, node-

set, and atom.

10. Click the **Add** button on the right side of the **XPath Function Parameters** table. Specify values for the following parameters and click **Finish**:

- **Name**: the name of the parameter

- **Type**: the data type of the parameter

- **Optional**: select the checkbox if the parameter is optional.

- **BW XML Types**: Select one of the XML types as item, node, item-set, node-set, atom, Focus, DynamicContext, and TypedContext.

- **Is Hidden**: The checkbox is auto-selected for Focus, DynamicContext, and TypedContext types, all of which are BW XML Types. If unchecked (not hidden) for these types, the installed CXF does not show under the **Functions** tab.

11. Select the **Use Logger** checkbox to create the default logger object when creating the custom Xpath functions.

12. The Open Associated Perspective dialog displays. Click **Yes**.

TIBCO Business Studio for BusinessWorksopens in the Plug-in and the Plug-in Development perspective.

13. Open the Java class in the newly created Custom Xpath function plug-in and implement the Custom Xpath function.

Now your Custom Xpath function is ready for use at design time and run time.

> **Note:** The Custom Xpath functions work only when the plug-in is marked as singleton.

Creating Custom XPath Functions with Multiple Java Classes

You can also create custom XPath functions with multiple Java classes within the same plug-in.

Before you begin

Create the custom XPath function in the plug-in. For more information, see Creating
Custom Xpath Functions.

Procedure

1. Create different classes in the same plug-in.

2. Update the category, prefix, and namespace for the newly created Java class as per
   requirement. For example:

   ```
   @XPathFunctionGroup(category = "Another Custom Function",
   prefix = "mytns2",namespace = "http://my.xpath2", helpText =
   "Another Custom defined function")
   ```


3. Navigate to `plugin.xml` and add custom XPath extensions on the **Extensions** tab.

4. To update the class name and id, on the **Extension Element Details** pane click
   Browse.
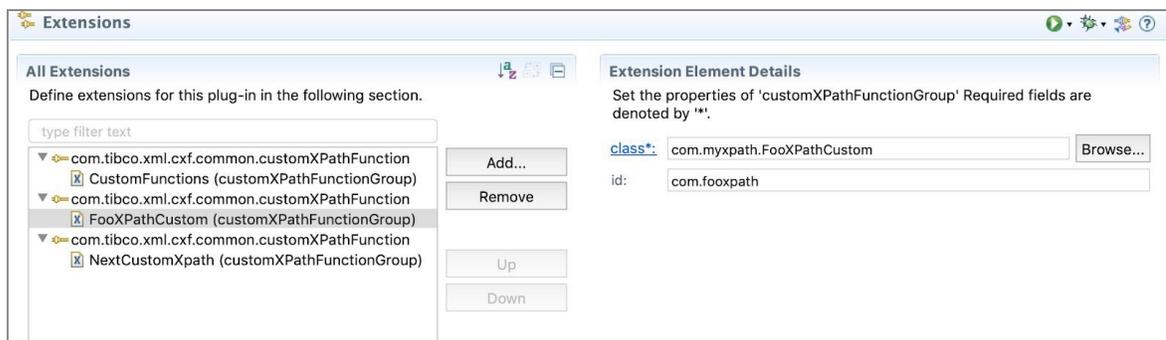
5. Create OSGi component for the newly created class as follows.

a. Create a copy of the existing `cxf.xml`.

b. For the newly created XML file, on the **Overview** tab of the **Component** pane, update **Name**, **Factory ID**, and **Class** fields.
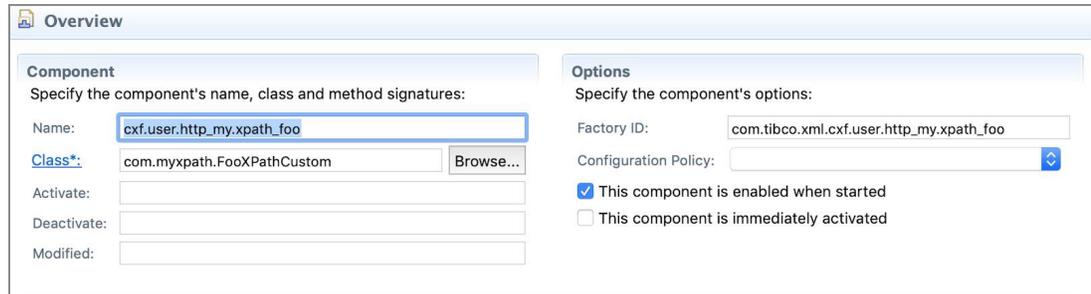
6. Repeat steps 3, 4, and 5 for other classes in the same plug-in.

7. Update the manifest file to include the `OSGI-INF/cxf.xml`, `OSGI-INF/foo_cxf.xml`, and `OSGI-INF/next_cxf.xml` service components.

8. Export the plug-ins and install into the host repository.

9. Restart TIBCO Business Studio for BusinessWorks.

Result
The new XPath functions are available for use.

What to do next
Run the application and custom XPath functions run from the same plug-in.

Using Custom XPath Function at Design Time and Run Time

You can use the newly created Custom XPath functions at design time and runtime.

Making Your Custom XPath Function Available at Design Time

Procedure

1. Right-click the **Custom XPath Function Plug-in** project and select **Export** from the menu. On the Export window, select **Plug-in Development > Deployable plug-ins and fragments** and click **Next**.
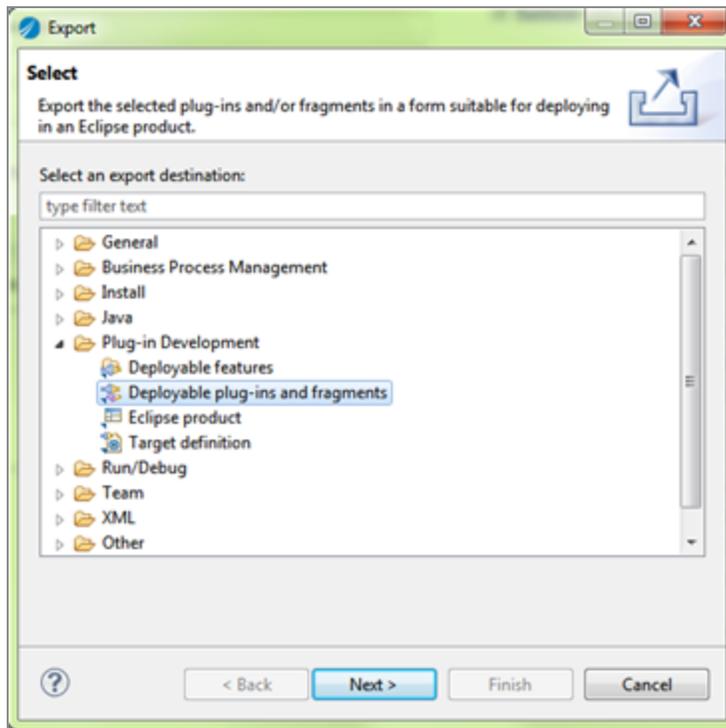
2. Select the **Install into host Repository** radio button on the **Deployable plug-ins and fragments** screen and click **Finish**.
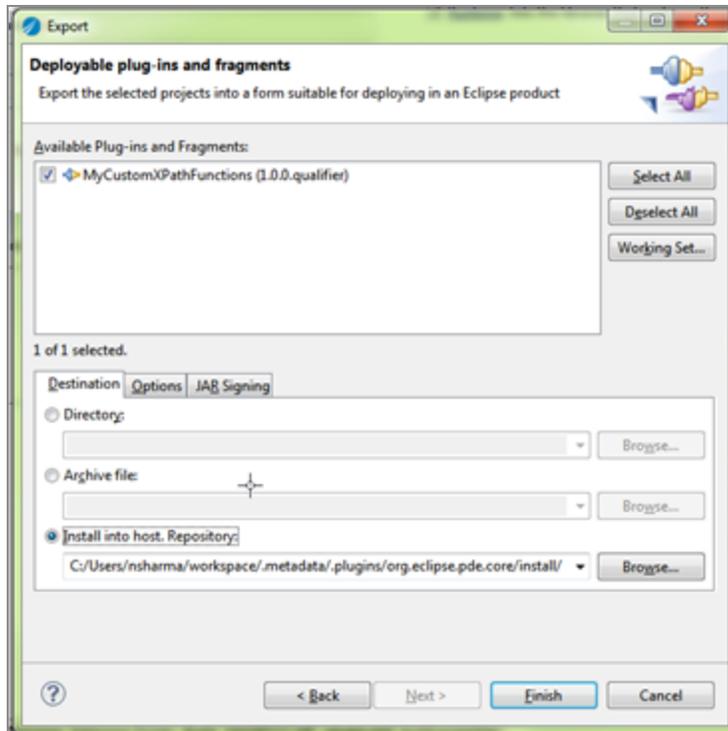
The Custom XPath function plug-in is exported into the TIBCO Business Studio for BusinessWorks instance to use at design time.

3. You are prompted with the following security warning message. Click **Install anyway**
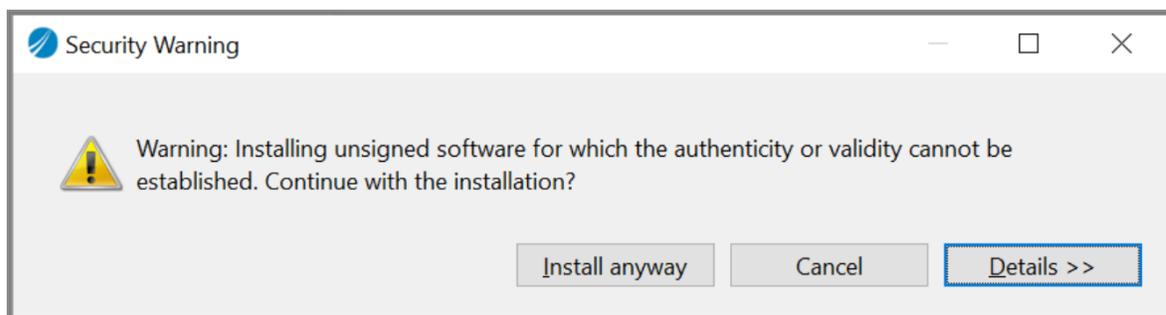
4. Click **Restart Now** when the following window is displayed.
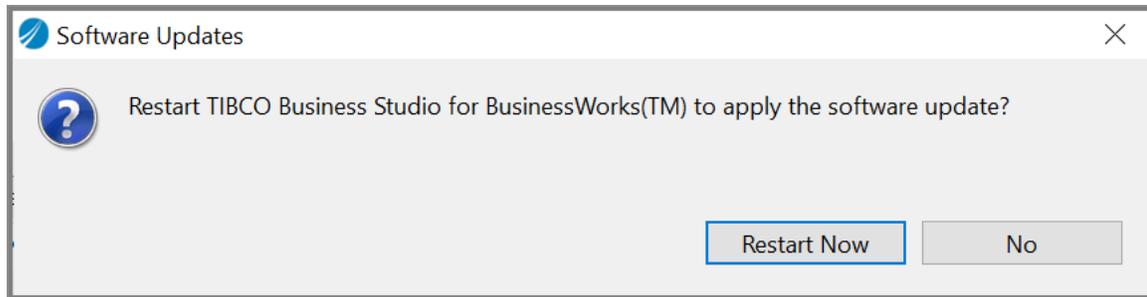
5. In TIBCO Business Studio for BusinessWorks, the Custom XPath function is available for use in any activity mapper in a process.
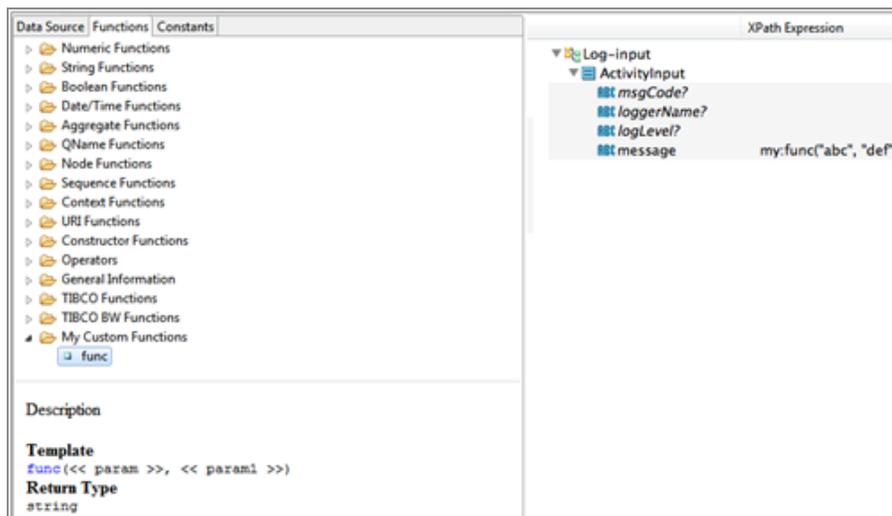
> **ⓘ Note:** Alternatively, you can install a custom XPath function using the following methods:
>
>   - Right-click the CXF project in the workspace and select **Install CXF Project**.
>
>   - Use the following command in BWDesign utility:
>
>     ```
>     system:export -cxf <CXF_Project_Name>
>     ```
>
>   For more information, see "Using the BWDesign Utility" in *TIBCO ActiveMatrix BusinessWorks™ Application Development*.

Making Custom XPath Function Available at Runtime

- After using the Custom XPath function in an activity in a process at design time, include the Custom XPath function plug-in the application project.
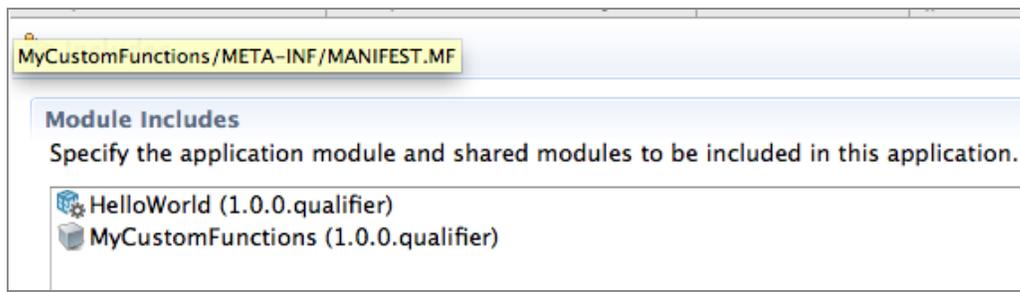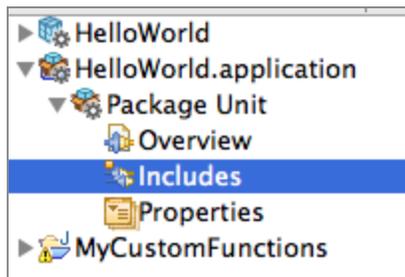
- After including the **Custom XPath Function Plug in** in the application project, test it in the **Debugger** and then, export it as an EAR, and deploy at runtime.

Deleting Custom XPath Functions

This topic lists the steps to uninstall a Custom XPath function so that it is not visible in the mapper.

Procedure

1. On the menu bar, click **Help > About TIBCO Business Studio for BusinessWorks™**. On Mac, this is located under the "BusinessWorks" menu.

2. Click **Installation Details**.

3. Select the Custom XPath function you want to uninstall, and click the **Uninstall** button.

4. If you uninstall a custom XPath project from the *{Tibco_Home}* location, the `CustomXpath.jar` is not removed from the `plugins` folder. To remove the jar file:

   a. In the file explorer, navigate to *{tibco_home}*`\studio\4.0\eclipse` folder.

   b. Open the command prompt here and type the following command:

   ```
   TIBCOBusinessStudio.exe -application
   org.eclipse.equinox.p2.garbagecollector.application
   ```

   The `CustomXpath.jar` is removed from the `plugins` folder.

5. Restart the workspace.

Java Invoke

Java Invoke is a synchronous activity that can be used to invoke a Java class method. You can construct an instance of the specified Java class, if you invoke the constructor for the class.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Java Global Instance	Yes	Specify the resource in this field to execute methods on shared Java objects created by Java Global Instance resources. For more information about the shared Java objects created by Java Global Instance resources, see Java Global Instance. This disables the options on the **Advanced** tab. In the

Field	Literal Value/Module Property	Description
		drop-down list in the **Method** field, you can select the method to execute.
		Note: If the configuration of the Java Global Instance resource changes, select and view any **Java Invoke** activities in your process that reference the Java Global Instance to automatically propagate the changes. An exception is raised if you attempt to test your process before refreshing the reference to the Java Global Instance. You may need to manually update the mappings of any input or output, if you select a new method with a different signature in the Java Global Instance resource.
MultiOutput	None	Select this check box to specify multiple parameters in the Java invoke method.
Class Name	None	The declared Java class whose method you want to execute. Use the **Browse A Class** button to select the class, or use the **Java Global Instance** field to specify a shared Java object.

Note: To log steps in appnode log, from the custom java code, add the jar file `rg.slf4j(1.8.0)` to Import Packages under Module dependencies and in the java code add the below code

```
import org.slf4j.*;
.
.
.
Logger logger = LoggerFactory.getLogger
("mylogger");
logger.info("my logger");
```

Under appnode, add the logger part in the `logback.xml`

Field	Literal Value/Module Property	Description
		file ``` <logger name="mylogger"> <level value="INFO"/> </logger> ```
Method	None	The method in the selected Java class you want to execute. You can use the drop-down list to select a different method in the selected Java class, if required.
Custom Icon	None	Provide the location of icon image file. Click the **Choose a custom icon** button to select the icon image file from your current workspace. Click the **Import an icon from the file system** button to select an icon image from your file system. Select the **Clear Value** button to remove the field value. **Important:** TIBCO recommends to use image file of size 48 by 48 pixels and of the form .png, .jpg, or .jpeg.
Parameters	None	The parameters for the **Java Invoke** activity. This element is available when the **MultiOutput** check box is selected. This element contains a list of input and output parameters that are required for the Java invoke. This element is not available when the selected method has no parameters.

Field	Literal Value/Module Property	Description
		Each Input and Output parameter has the following fields. • **Field Type**: It can be either `In` or `Out`. • **Field Name**: The name of the parameter. When creating a new parameter, provide the name of the parameter. The setter/getter methods get automatically generated for the same, or select the options from the drop-down list of the names of class member variables that have setter/getter methods. • **Type**: The datatype of the parameter. The default is `string`. Specify one of the supplied Java primitive types, or specify Object Reference when accepting a Java object from another Java activity as input. • **Occurrence**: Specify whether the parameter is **Required**, **Optional**, or **Repeating**. • Click to add the input parameter. • Click to add the output parameter. • Click to delete the parameter.

Description

Provide a short description for the Java Invoke activity.

Advanced

The **Advanced** tab has the following fields.

Field	Literal Value/Module Property	Description
Instantiate the referenced class	None	This field controls one of the following: • The referenced class must be constructed when the activity is executed, • An object reference must be supplied in the activity's input schema. Select this check box, if you want to construct the referenced class using the default constructor when the activity executes. In this case, the referenced class is not included in the activity's input schema. Do not select this check box, if you want to provide an object reference for the class name specified on the **General** tab. The object must be created and contained in the output schema of a previously executed activity. In this case, the referenced class is included in the activity's input schema. This check box is not available when **MultiOutput** on the **General** tab is selected.
Cache the object	None	This option is available only when the **Instantiate the referenced class** check box is selected. Select this check box to specify that all instances of this **Java Invoke** activity share the instance of the instantiated referenced class. A cached referenced object is kept in the memory until the process engine terminates. Not selecting this check box leads to each instance of this **Java Invoke** activity creating its own copy of the referenced class.
Invoke Cleanup Method	None	This check box is available only when the **Cache the object** check box is selected.

Field	Literal Value/Module Property	Description
		When selected, you can invoke a method on the cached object before the object is released from memory, when the process engine terminates. You can perform cleanup activities, such as closing connections, updating database tables, and so on. Select this check box to display the **Cleanup Method** field.
Cleanup Method	None	This field is available only when the **Invoke Cleanup Method** check box is selected. Using this field you can specify the name of the method to invoke, before the cached class in memory is released. The selected method cannot contain any parameters, therefore only methods with no parameters are listed in this field. Any values returned by the selected method are ignored, but exceptions raised by the selected method are sent to the log.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Parameters	varies	This element contains a list of input parameters that are required for the Java invoke **Method** selected on the **General** tab. This element is not available when the selected method has no parameters.

Output

The following is the output of the activity.

Output Item	Datatype	Description
MethodReturnValue	varies	The **Output** tab has the return value specified for the Java invoke **Method** selected on the **General** tab. This element is not available for methods that return void.
Parameters	varies	The **Output** tab lists all output parameters specified on the **General** tab. Selecting **MultiOutput** on the **General** tab makes this output item available.

Fault

The **Fault** tab lists the exceptions generated by this activity. Selecting a method to invoke lists the exceptions generated by the method.

In the **Java Invoke** activity:

- If **MultiOutput** is selected, the **Fault** tab generates the `InvocationException` exception with error schema element as `msg, msgCode, methodName, exceptionClassName, exceptionMessage`.

- If **MultiOutput** is not selected, the **Fault** tab displays the exceptions generated by the method selected in it with error schema elements as: `1:Java exception Object 2:exceptionClassName 3:exceptionMessage`

- If **MultiOutput** is not selected and method does not generate any exception, again the **Fault** tab shows `InvocationException` with error schema element as `msg, msgCode, methodName, exceptionClassName`, and `exceptionMessage`.

Error Schema Element	Datatype	Description
msg	string	The error messageFor more information, see *Error Codes*.
msgCode	string	The error codeFor more information, see *Error Codes*.

Error Schema Element	Datatype	Description
methodName	string	The name of the Java method that raised the exception.
exceptionClassName	string	The exception class name. The value of this element is obtained by calling the `<exception>.getClass().getName()` method.
exceptionMessage	string	The exception message. The value of this element is obtained by calling the `<exception>.getMessage()` method.

Java Process Starter

Java Process Starter is a process starter activity. You can use this activity to create a custom starter written in Java.

Java Process Starter Abstract Class

The JavaProcessStarter abstract class defines the interface of your Java Process Starter with the TIBCO ActiveMatrix BusinessWorks™ application. Define an implementation for the following methods:

- `init()`: this method is called when the process engine starts up. This method should initialize any resource connections. You could also specify a Java global instance on the **Advanced** tab that initializes resource connections. Java global instances are also loaded and initialized during application start-up. You can call `this.getJavaGlobalInstance()` to obtain the Java global instance resource in your process starter code.

- `onStart()`: this method is called by the process engine to activate the process starter. This method should activate any event notifier or resource observer code. The notifier or observer code can then call the `onEvent()` method to start a process instance.

- `onStop()`: this method is called by the application to deactivate the process starter. This method should deactivate any event notifier or resource observer code.

- `onShutdown()`: this method is called by the application when the AppNode shuts down. This method should release any resources and resource connections and perform any required clean up operations.

The following methods are already implemented and can be used in your code:

- `onEvent(Object object)`: this method is called when a listener or resource observer catches a new event. The input to this method is a Java object containing the event data.

- `getGlobalInstance()`: this method returns an object reference to the Java global resource specified on the **Advanced** tab of the process starter. This is useful if you want to place initialization code or other shared information in a Java global resource instead of in the `init()` method of this class.

- `onError()`: this method generates the exception specified in the input parameter. Use this method to propagate an error to the ActiveMatrix BusinessWorks™ process instance when a listener or resource observer fails to generate an event.

Passing Java Objects Between Java Activities

You can create instances of Java objects in **Java Invoke** activity or by using the **Java Invoke** or **XML To Java** activities. These Java objects can be passed using an output parameter to another activity later in the process definition. The **Java Invoke** activity receiving a Java object accepts the object into an input parameter. Map the output Java object to the input object of the receiving Java Invoke object.

> **Note:** Any Java objects passed by input and output parameters between activities should have the capability to serialize.

General

The **General** tab has the following fields.

Field	Process Property	Description
Name	No	The name to be displayed as the label for the activity in the process.

Field	Process Property	Description
Class Name	No	The name of the compiled Java class that is implemented by extending the JavaProcessStarter abstract class. Use the **Browse a Class** 🔍 button to select the class from the specified class library. This shows the classes that extend from the JavaProcessStarter abstract class.
Initialize with Parameters	No	Selecting this checkbox displays the **Init Method** field and **Init Parameters** table.
Init Method	No	Selecting the **Initialize with Parameters** checkbox displays this field. You can use the list, if required, to select a different method in the selected Java class.
Init Parameters	No	Specify default values in this table for the input parameters of the `init()` method. Each input parameter has three fields: • Parameter Name: the name of the parameter. • Parameter Type: the datatype of the parameter. • Parameter Value: specify a default value for the parameter. It can be a string or a module property.

Description

Provide a short description for the activity.

Advanced

The **Advanced** tab has the following fields.

Field	Literal Value/Module Property	Description
Sequence key	None	This field can contain an XPath expression that specifies which processes should run in sequence. Process instances with sequencing keys that evaluate to the same value are ran in the sequence the process instance was created.
Custom Job Id	None	This field can contain an XPath expression that specifies a custom job ID for the process instance.
Java Global Instance	Yes	A Java global instance resource. Specify a value in this field to reference a Java global instance resource in your JavaProcessStarter implementation class. For more information, see Java Global Instance. You can use the `getGlobalInstance()` method to obtain a reference to the Java global instance resource.

Output

The following is the output of the activity.

Output Item	Datatype	Description
javaObject	varies	The Java object that was passed to the `onEvent()` method. By default, this element is a Java object reference.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
JavaEventSourceException	Problem occurs on some event.

Java To XML

Java To XML is a synchronous activity. You can use this activity to convert Java object's data members into an XML document.

> **Note:** If the class does not have a public data member and only has a Java bean modifier that sets the data, the input schema contains an element for the modifier, but the resulting XML document has no value set for the corresponding element. For example, the Java object has a method declared as public int `setID()`, but there is no method for getting the ID and the data member ID is not public. In this case, an element named ID in this activity's output schema exists, but that element has no value because there is no public mechanism for getting the data.

General

The **General** tab has the following fields.

Field	Module Property	Description
Name	No	The name to be displayed as the label for the resource.
Class Name	No	The class name in the Java schema to use. Use the **Browse** button to bring up the dialog to select the Java class. Click the **Reload** button to regenerate the schema.

Description

Provide a short description for the Java to XML activity.

Advanced

The **Advanced** tab has the following fields.

Field	Description
Instantiate the referenced class	This field controls one of the following: • The referenced class must be constructed when the activity is executed. • An object reference must be supplied in the activity's input schema. Select this check box to construct the referenced class using the default constructor when the activity executes. In this case, the referenced class is not included in the activity's input schema. Do not select this check box if you want to provide an object reference for the class name specified on the **General** tab. The object must be created and contained in the output schema of a previously executed activity. In this case, the referenced class is included in the activity's input schema.
Cache the object	This option is available only when you select the **Instantiate the referenced class** check box. Select this check box to specify that all instances of this **Java Invoke** activity share the instance of the instantiated referenced class. A cached referenced object is kept in the memory until the process engine terminates. Not selecting this check box leads to each instance of this **Java To XML** activity creating its own copy of the referenced class.

Input

The following is the input for the activity.

Input item	Datatype	Description
referenced class	Java Object	An object reference to the **Class Name** specified on the **General** tab. This element is only available when the **Instantiate the referenced class** check box is clear on the **Advanced** tab.

Output

The following is the output of the activity.

Output Item	Datatype	Description
XML Schema	varies	The output is the XML schema from the converted Java object. The elements in the schema display in alphabetical order.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
`JavaToXMLConversionException`	An error occurred when converting the Java object to XML.

XML To Java

XML to Java is a synchronous activity. Use this activity to create an instance of a Java object based on data from an XML document. The XML schema for providing the input to the Java object or Java schema specified on the **General** tab of this activity.

The specified Java class must meet the following requirements:

- The Java class must have a public default constructor (that is, a constructor with no arguments).

- The Java class must be eligible to be serialized (that is, the class must implement or be a subclass of a class that implements `java.io.Serializable`).

If the class does not have a public data member and only has a Java bean accessor that retrieves the data, the input schema contains an element for the accessor. However, the resulting Java object has no value set for the member. For example, the object has a method declared as public int `getID()`, but there is no method for setting the ID and the data member ID is not public. In this case, an element named ID in this activity's input schema, but mapping a value to the element does not result in setting the ID member of the output Java object. This is because there is no public mechanism for setting the data.

General

The **General** tab has the following fields.

Field	Module Property	Description
Name	No	The name to be displayed as the label for the resource.
Class Name	No	The class name in the Java schema to use. Use the **Browse** button to bring up the dialog to select the Java class.

Description

Provide a short description for the XML to Java activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
XML Schema	varies	The XML schema you want to convert to a Java object. The content of this schema depends upon the Java class or Java schema defined in the **Class Name** field on the **General** tab. The elements in the schema display in an alphabetical order.

Output

The following is the output of the activity.

Output Item	Datatype	Description
referenced class	Java Object	A Java object that is created out of the XML schema.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the possible corrective actions, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
XMLToJavaConversionException	An error occurred when converting the XML schema to a Java object.

JDBC Palette

The JDBC palette contains activities for querying, updating, or calling stored procedures in the database.

The CPU usage can be lowered by passing the following properties as a VM argument for eclipse or the `config.ini` file for AppSpace or AppNode at runtime.

- `-Dbw.application.jdbc.validateConnection=false`

- `bw.application.jdbc.validateConnection.frequency=<int>`

where, `<int>` represents a number which must be smaller than the Connection Idle Timeout property in the JDBC Shared Resource

> **Note:** To avoid errors for the JDBC MS SQL Server Database, pass the `trustServerCertificate=true` property in the JDBC MS SQL Server URL.

Migrating Existing JDBC Applications Configured with Progress DataDirect Drivers

Starting from TIBCO Business Studio for BusinessWorks 2.9.0 release, the JDBC Progress DataDirect Drivers are not supported directly.

> **ⓘ Note:**
> - Starting from TIBCO Business Studio for BusinessWorks 2.9.0, the Oracle (ojdbc8.jar) jar files are going to be shipped as out of box. However, you can also install and use the jars having different versions using the existing `bwinstall <db_name>-driver` command if you don't want to use the shipped jar version. Also, at runtime ensure to select the bundled jar provided or installed so applications can run with the installed jars.
>
> - TIBCO will no longer continue to ship the following jar files present at **TIBCO -> *<BW_Home>* -> config -> design -> thirdparty** location:
>   - TIdb2
>   - TIinformix
>   - TImysql
>   - TIoracle
>   - TIpostgresql
>   - TIsqlserver

Migrating from Progress DataDirect Driver configured applications to Native Driver

In TIBCO Business Studio for BusinessWorks:

In TIBCO BusinessWorks Container Edition pass the **Database URL** in the VM argument section in the following format:

`-Dbw.application.jdbc.db_name.url=<Database Connection URL>`

For example, for PostgreSQL database, use the following format: `-Dbw.application.jdbc.postgres.url=jdbc:postgresql://localhost:5432/postgres`

> **ⓘ Note:** If the **Database Driver** and **Database URL** fields are hard-coded with the Progess DataDirect driver details, then after importing the project to TIBCO Business Studio for BusinessWorks 2.9.0, the **Database Driver** field is blank and the **Database URL** field is populated as-is with the older URL details.

Migrating from Progress DataDirect Driver configured applications to Custom Drivers (Bring Your Own License)

After importing the project to TIBCO Business Studio for BusinessWorks 2.9.0, if you want to continue using the Progress DataDirect Drivers then you need to create a custom driver class.

1. Create the runtime bundle just like creating a regular custom JDBC bundle. While creating this bundle provide the valid license key and other details as applicable and, keep the unzipped bundle at *TIBCO/BWCE_HOME/system/shared* location.

   > **Note:** Once the bundle is created you can reuse this bundle with other applications for the same database.

2.]

3. To run the application, select the runtime bundle and pass both the **Database Driver** and **Database URL** fields in the VM arguments section in the following format:
   ```
   bw.application.dd.jdbc.db_name.driver=db_drivername
   bw.application.dd.jdbc.db_name.url="db_url"
   ```

   For example,
   ```
   bw.application.dd.jdbc.postgres.driver=
   tibcosoftwareinc.jdbc.postgresql.PostgreSQLDriver
   bw.application.dd.jdbc.postgres.url=
   "jdbc:tibcosoftwareinc:postgresql://localhost:5432;DatabaseName=postgres"
   ```

   > **Note:**
   > - In TIBCO Business Studio for BusinessWorks, when the Database URL is passed in the VM arguments section, the URL must be in double quotes ("")
   >
   > - Ensure the same URL which is used in older project or existing EAR is passed, otherwise the project does not run.

Passing Multiple Database URLs

While migrating from Progress DataDirect to Native Driver if there are multiple instances of the same database with different versions, then pass the Database URL in the VM argument in the following format separated by a hash ("#")

```
-Dbw.application.jdbc.db_name.url=<Database Connection URL1>#<Database
Connection URL2>
```

While migrating from Progress DataDirect to Progress DataDirect using the custom jar approach (Bring Your Own License), if there are multiple instances of the same database with different versions, then pass both the Database URL and the Driver name in the VM argument in the following format separated by a hash ("#")

```
-Dbw.application.dd.jdbc.db_name.driver=tibcosoftwareinc.jdbc.db_name.db_
Driver>
```

> **Note:** Multiple URL separation option works only in case of the same Database having different versions. In case of multiple databases, the URL's will be different or separate.

JDBC Call Procedure

JDBC Call Procedure is an asynchronous activity that calls a database procedure or function using the specified JDBC connection. If this activity is not part of a transaction group, the SQL statement is committed after the activity completes.

If this activity is part of a transaction group, the SQL statement is committed or rolled back with the other JDBC activities in the group at the end of the transaction.

To override the default behavior of transaction groups for certain JDBC activities in a transaction group, select the **Override Transaction Behavior** checkbox on the **Advanced** tab. This specifies that the activity is outside of the transaction and the SQL statement is committed when the activity completes, even if the activity is in a transaction group.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
JDBC Shared Resource	Yes	The path to the shared resource containing the JDBC connection information. For more information about JDBC resources, see JDBC Connection.
Timeout (sec)	Yes	The time (in seconds) to wait for the procedure call to complete. If the call does not complete in the specified time limit, it returns an error.
Maximum rows	Yes	The maximum number of rows to retrieve. To retrieve all rows, specify 0.

Description

Provide a short description for the JDBC Call Procedure activity.

Procedure

The **Procedure** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Schema	Yes	The schema in which the procedure resides. This name is used to resolve naming conflicts, if more than one schema in the database has the selected procedure with the same name.

Field	Literal Value/Process Property/Module Property	Description
		Configure the **Schema** field to use a **Process Property** or a **Module Property** if a schema is defined for the project, and you know the schema name. To search for a schema, configure the field to use a **Literal Value**, and click the **Browse** button to select a schema from the database that is using the specified JDBC Connection.
Catalog or Package	Yes	The catalog or package in which the procedure resides. This name is used to resolve naming conflicts if more than one catalog or package in the database has selected procedures with the same name. Configure the **Catalog or Package** field to use a **Process Property** or a **Module Property** if a catalog or package is defined for the project, and you know their names. To search for a catalog or package, configure the field to use a **Literal Value**, click the **Browse** button, and select a catalog or package from the database using the specified **JDBC Connection**. If a schema is specified in the **Schema** field, only catalogs or packages in the specified schema are listed. This field is only applicable to databases with more than one catalog and package. For more information about catalogs and packages, see your database documentation.
Procedure / Function	Yes	The name of the database procedure or user-defined function to call. Configure the **Procedure / Function** field to use a **Process Property** or a **Module Property** if a procedure or function is defined for the project, and

Field	Literal Value/Process Property/Module Property	Description
		you know the procedure or function name. To search for a procedure or function, configure the field to use a **Literal Value**, and click the **Browse** button to select a procedure or function from the database that is using the specified JDBC Connection. Click the **Browse** button to select the database for available procedures or functions after specifying the **JDBC Connection** field. Click the **Refresh** button to retrieve the changes from the database, if you change the parameters or signatures while editing your project. The list of available procedures and functions is limited by the values supplied in the **Schema and Catalog** or **Package** fields.
Parameters	No	This field displays the parameters for the procedure and you can override the data type (IN, OUT, IN/OUT) for each parameter in the **Direction** column. The parameters listed in the table show the **Parameter Name**, **Data Type**, and the **Direction** (IN, OUT, IN/OUT) of each parameter. All IN and IN/OUT parameters listed in the **Parameters** table are shown on the **Input** tab of the activity for the user to provide values.

Advanced

The **Advanced** tab has the following fields.

Field	Description
ThreadPool SharedResource	Specifies a queue of threads available to run a queue of tasks. Thread pools are used to improve performance when running large numbers of asynchronous tasks by reducing per task invocation overhead, provide a means of bounding, and managing the resources consumed when running a collection of tasks.
Override Transaction Behavior	Overrides the default behavior of a transaction group. If this activity is in a transaction group, the activity is normally committed or rolled back with the other transactional activities.
	Select this checkbox to specify that this activity is not part of the transaction group and is committed when it completes. Also, select this checkbox to use a separate database connection to perform the activity and commit the SQL statement.
Override JDBC Connection	Selecting this option overrides the default datasource connection, and allows for SQL queries to be run against the optional datasource.
	After selecting this option, the **sharedResourceName** field is enabled under the **Input > Data Source** tab. From the **sharedResourceName** field, type the complete path with the name of the optional datasource, which can be found under the **Resources** folder. For example, if a JDBC shared resource is located at **Resources > Package_Name > JDBC_SR_ Name**, then the value you enter should be Package_Name.JDBC_SR_ Name. The value you enter in the **XPath Expression** field is case sensitive and must be a string.
	To avoid mapping issues, ensure that the optional datasource maintains a structure that is identical to the default database, and that both datasources are using the same datatypes.
	Note: Transactions between multiple datasources are not supported.
Interpret Empty String As Null	Specifies how empty strings in the activity's input elements should be handled. Selecting this checkbox sends the nulls to the database where empty strings are supplied.
	Empty strings are treated as zero-length strings, if this checkbox is not selected. Whether you use this option or not, you can still use XPath to

Field	Description
	set input elements explicitly to null.
ResultSets Use Schema	Specifies that the design-time must try to fetch the schema when a result set is expected from a stored procedure call.
	When selected, TIBCO Business Studio for BusinessWorks tries to invoke the stored procedure with default values and get the metadata for the resultant.
	If the checkbox is not selected, or if TIBCO Business Studio for BusinessWorks does not get any metadata, the results are displayed as 'unknown result sets'.
Interpret Invalid XML Chars	Specifies how invalid characters should be handled. If the checkbox is clear and invalid characters are sent from the database to TIBCO BusinessWorks Container Edition, the call procedure execution fails.
	On the other hand, if invalid characters are sent, and the checkbox is selected, TIBCO BusinessWorks Container Edition runs it successfully.
Override S/A DataType Schema	Select the checkbox to override the schema appended to the object type parameter. The schema in the **Schema** field on the **Procedure** tab overrides the schema in the **Parameters** field at run time.
Use Nil	Specifies whether NULLs are represented as optional schema elements or whether each item that can contain a NULL has subitems.

Input

The input for this activity is dependent on the input parameters of the database procedure.

Input Item	Datatype	Description
inputs	complex	The input parameters of the database procedure. The datatype contained in this element varies depending on the parameters specified for the database procedure.
		The IN and IN/OUT parameters displayed in the

Input Item	Datatype	Description
		Parameters table on the **Procedure** tab are displayed here for the user to set values.
		You can see **inputSet** once you select **Procedure** in the **General** tab.
ServerTimeZone	string	Specifies the time zone for the database server.
timeout	number	The time (in seconds) to wait for the procedure call to complete. The call if not completed in the specified time limit, returns an error.
maxRows	number	The maximum number of rows to retrieve. To retrieve all rows, specify 0.

Output

The output of the activity is depends on the output parameters of the database procedure.

Output Item	Datatype	Description
outputSet	complex	The output of the database procedure. The datatype contained in this element varies depending on what the database procedure returns.
UnresolvedResultSets	string	The output of the database procedure may return other result sets. These result sets are returned in this output item as an XML string. Create a schema resource to describe these result sets, and then use the **Parse XML** activity to parse the XML based on the schema. You can then use the parsed output in subsequent activities.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the possible corrective action to take, see the *TIBCO BusinessWorks*

Container Edition Error Codes.

Fault	Generated When.
JDBCConnectionNotFoundException	An error occurred when attempting to connect to the specified database.
InvalidTimeZoneException	An invalid time zone was specified.
JDBCSQLException	An SQL error occurred.
LoginTimedOutException	A timeout occurred while attempting to connect to the database.
ActivityTimedOutException	A timeout has been reached.

JDBC Query

JDBC Query is an asynchronous activity that performs the specified SQL SELECT statement. If this activity is not part of a transaction group, the SQL statement is committed after the activity completes.

If this activity is part of a transaction group, the SQL statement is committed or rolled back with the other JDBC activities in the group at the end of the transaction.

If you want to override the default behavior of transaction groups for certain JDBC activities in a transaction group, you can select the **Override Transaction Behavior** checkbox on the **Advanced** tab. This specifies that the activity is outside of the transaction and the SQL statement is committed when the activity completes, even if the activity is in a transaction group.

> **Note:** Do not use the **Override JDBC Connection** option in a transaction. Selecting this option in a transaction results in failure of the transaction.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
JDBC Shared Resource	Yes	The path to the shared configuration resource containing the JDBC connection information. For more information, see JDBC Connection.
Timeout (sec)	Yes	The time (in seconds) to wait for the query to complete. If the query does not complete in the given time limit, an error is returned.
Maximum rows	Yes	The maximum number of rows to retrieve. To retrieve all rows, specify 0.

Description

Provide a short description for the JDBC Query activity.

Statement

The **Statement** tab has the following fields.

Field	Description
Statement	This is the SQL SELECT statement to perform in the database. You can use a SQL Builder wizard to build the query for the desired database.
Prepared Statement	The **Prepared Statement** contains the **Parameter Name** and **Data Type** fields. All the parameters defined in the **Prepared Statement** table are shown in the **Input** tab of the activity for the user to provide values. Each prepared statement corresponds to the question mark in the same position in the SQL statement. That is, the first prepared statement in the list corresponds to the first question mark, the second prepared statement in the list corresponds to the second question mark, and so on. The warning is for

Field	Description
	the purpose of information only. Ensure that the parameters in this field correctly correspond to the statement.

You can optionally specify names for each prepared parameter. By default, the prepared statements are named Parameter1, Parameter2, and so on. Supply the datatype of each parameter to the SQL statement, and this data type is used in the input schema for the statement.

Run the Query Button.

Runs the JDBC Select statement entered in the **Statement** field and shows the results in the **Results View**. These results are persisted in the workspace.

SQL Button

Open Eclipse SQL Builder. For more information about using the SQL Query Builder to visually build queries, see Eclipse Documentation at eclipse.org.

Fetch Button.

Using the **Fetch** button on this activity, you can synchronize the activity with the contents of the database. When you first configure a JDBC activity, click the **Fetch** button to retrieve the schema for the output result set before applying your changes by saving your project.

After you have configured your activity, the **Fetch** button is useful when you make a change to the database while editing a process containing this activity in TIBCO BusinessWorks™ Container Edition. The **Fetch** button synchronizes with the database and changes the output schema, if necessary.

Advanced

The **Advanced** tab has the following fields.

Field	Description
ThreadPool SharedResource	Specifies a queue of threads available to run a queue of tasks. Thread pools are used to improve performance when running large numbers of asynchronous tasks by reducing per task invocation overhead, provide a means of bounding, and managing the resources consumed when

Field	Description
	running a collection of tasks.
Loop Subset in Calling Processes	**Note:** This field can be activated on selecting the **Process in Subsets** checkbox.
	Use this field to configure the loop with lastSubset in another process. This another process calls the subprocess in which the **JDBC Query** activity is used.
	Provide the names of the processes as a comma-separated list.
	For example, mypackage1.myprocess1, mypackage2.myprocess2.
	If no input is provided for this field and the **Loop Subset in Current Process** is also not selected it means that the loop is in a direct call process of the subprocess.
	Either **Loop Subset in Calling Processes** or **Loop Subset in Current Process** can be selected at a time.
Loop Subset in Current Process	**Note:** This field can be activated on selecting the **Process in Subsets** checkbox.
	Select this checkbox to configure the loop with lastSubset inside the current process for activity.
	Either **Loop Subset in Calling Processes** or **Loop Subset in Current Process** can be selected at a time.
Override Transaction Behavior	Overrides the default behavior of a transaction group. If this activity is in a transaction group, the activity is normally committed or rolled back with the other transactional activities.
	This activity is not part of the transaction group and is committed when it completes, when this checkbox is selected. Selecting this option uses a separate database connection to perform the activity and commit the SQL statement.
Override JDBC Connection	Selecting this option overrides the default datasource connection, and allows for SQL queries to be run against the optional datasource.

Field	Description
	Note: Do not use the **Override JDBC Connection** option in a transaction. Selecting this option in a transaction results in failure of the transaction.
	After selecting this option, the **sharedResourceName** field is enabled under the **Input > Data Source** tab. From the **sharedResourceName** field, type the complete path with the name of the optional datasource, which can be found under the **Resources** folder. For example, if a JDBC shared resource is located at **Resources > Package_Name > JDBC_SR_ Name**, then the value you enter should be `Package_Name.JDBC_SR_Name`. The value you enter in the **XPath Expression** field is case sensitive and must be a string.
	To avoid mapping issues, ensure that the optional datasource maintains a structure that is identical to the default database, and that both datasources are using the same datatypes.
	Note: Transactions between multiple datasources are not supported.
Use Nil	Specifies whether NULLs are represented as optional schema elements or whether each item that can contain a NULL has subitems.
Interpret Empty String as Null	Specifies how empty strings in the activity's input elements should be handled. Selecting this checkbox sends the nulls to the database where empty strings are supplied. Empty strings are treated as zero-length strings, when this checkbox is not selected. Whether you use this field or not, you can still use XPath to set input elements explicitly to null. **Note:** Many databases treat empty strings and nulls as the same, so this field does not affect how the database interprets empty strings.
Interpret Invalid XML Chars	Specifies how invalid characters should be handled. If the checkbox is clear and invalid characters are sent from the database, the JDBC Query execution fails. When the checkbox is selected and invalid characters are sent, TIBCO BusinessWorks Container Edition runs it successfully.

Field	Description
Process In Subsets	Selecting this checkbox activates the **Loop Subset in Calling Processes** and **Loop Subset in Current Process** fields.
	This field provides the option to process the result set in smaller batches rather than processing the entire result set at once. When this checkbox is selected, the subsetSize input element is displayed for you to specify the size of each batch of records you want to process. Also, the lastSubset output element is displayed and is set to `true` when the last batch of records is being processed.
	Not selecting this checkbox returns the whole result set.
Override SQL Statement	Selecting this checkbox overrides the SQL SELECT statement specified on the **Statement** tab, and enables you to add a new SQL statement on the **Input** tab.
	Tip: To connect to another database that has a schema with the same table structure, select the **Override JDBC Connection** checkbox. After doing this, go to the **Input** tab, and specify a Shared Resource at **sharedResourceName**.

Input

The following is the input for the activity.

Input Item	Datatype	Description
ServerTimeZone	string	Specifies the time zone for the database server.
timeout	number	The time (in seconds) to wait for the query to complete. If the query does not complete in the given time limit, an error is returned.
maxRows	number	The maximum number of rows to retrieve. To retrieve all rows, specify 0.
SqlStatement	string	Displays after the **Override SQL Statement** checkbox on

Input Item	Datatype	Description
		the **Advanced** tab has been selected. This is the SQL SELECT statement to perform in the database. You can use a SQL Builder wizard to build the query for the desired database. You can also use a module property here.
		Note: If you create a SQL statement, ensure that the new statement does not differ from the output schema specified on the **Output** tab. If the new SQL statement does not match the output schema, an error is generated at run time. The output schema is determined from the original SQL statement on the **Statement** tab.

Output

The following is the output of the activity.

Output Item	Datatype	Description
resultSet	complex	The root class for the output of the **JDBC Query** activity. This class contains all output items for the activity.
Record	complex	The result of the database query. The contents of the **Record** element depends on the columns returned by the query.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
JDBCConnectionNotFoundException	An error occurred when attempting to connect to the specified database.
InvalidTimeZoneException	An invalid time zone was specified.
JDBCSQLException	An SQL error occurred.
LoginTimedOutException	A timeout has occurred while attempting to connect to the database.
InvalidSQLTypeException	A parameter's datatype does not match the datatype of the table column.
DuplicatedFieldNameException	A duplicate field name was specified.
ActivityTimedOutException	A timeout has been reached.

JDBC Update

JDBC Update is an asynchronous activity that performs the specified SQL INSERT, UPDATE, or DELETE statement.

If this activity is not part of a transaction group, the SQL statement is committed after the activity completes. If this activity is part of a transaction group, the SQL statement is committed or rolled back with the other JDBC activities in the group at the end of the transaction.

If you want to override the default behavior of transaction groups for certain JDBC activities in a transaction group, you can select the **Override Transaction Behavior** checkbox on the **Advanced** tab.

> **Note:** Do not select the **Override Transaction Behavior** checkbox for a JDBC resource in a transaction group.

This specifies that the activity is outside of the transaction and the SQL statement is committed when the activity completes, even if the activity is in a transaction group.

The SQL statement is committed after the activity completes.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
JDBC Shared Resource	Yes	The path to the shared resource containing the JDBC connection information. For more information, see JDBC Connection.
Timeout (sec)	Yes	The time (in seconds) to wait for the query to complete. If the query does not complete in the given time limit, an error is returned.

Description

Provide a short description for the JDBC Update activity.

Statement

The **Statement** tab has the following fields.

Field	Description
Statement	This is an SQL INSERT UPDATE OR DELETE statement to perform in the database. You can use the SQL Builder wizard to build the query for the desired database.
Prepared Statement	The **Prepared Statement** contains the **Parameter Name** and **Datatype** fields. All the parameters defined in the **Prepared Statement** table are shown in the

Field	Description
	Input tab of the activity for the user to provide values.
	Each prepared statement corresponds to the question mark in the same position in the SQL statement. That is, the first prepared statement in the list corresponds to the first question mark, the second prepared statement in the list corresponds to the second question mark, and so on.
	The warning is for informational purposes only. You must make sure the parameters in this field correctly correspond to the statement.

Advanced

The **Advanced** tab has the following fields.

Field	Description
ThreadPool SharedResource	Specifies a queue of threads available to run a queue of tasks. Thread pools are used to improve performance when running large numbers of asynchronous tasks by reducing per task invocation overhead, provide a means of bounding, and managing the resources consumed when running a collection of tasks.
Insert if record does not exist	Select this checkbox to insert the record if it does not exist in the database.
Override Transaction Behavior	Overrides the default behavior of a transaction group. If this activity is in a transaction group, the activity is normally committed or rolled back with the other transactional activities.
	If you select this checkbox, this activity is not part of the transaction group and is committed when it completes. Selecting this option uses a separate database connection to perform the activity and commit the SQL statement.
Override JDBC Connection	Selecting this option overrides the default datasource connection, and allows for SQL queries to be run against the optional datasource.
	After selecting this option, the **sharedResourceName** field is enabled

Field	Description
	under the **Input > Data Source** tab. From the **sharedResourceName** field, type the complete path with the name of the optional datasource, which can be found under the **Resources** folder. For example, if a JDBC shared resource is located at **Resources > Package_Name > JDBC_SR_Name**, then the value you enter should be `Package_Name.JDBC_SR_Name`. The value you enter in the **XPath Expression** field is case sensitive and must be a string.
	To avoid mapping issues, ensure that the optional datasource maintains a structure that is identical to the default database, and that both datasources are using the same datatypes.
	Note: Transactions between multiple datasources are not supported.
Interpret Empty String As Null	Specifies how empty strings in the activity's input elements should be handled.
	Selecting this checkbox sends the nulls to the database where empty strings are supplied.
	When the checkbox is not selected, empty strings are treated as zero-length strings. Whether you use this checkbox option or not, you can still use XPath to set input elements explicitly to null.
	Note: Many databases treat empty strings and nulls as the same, so this checkbox does not affect how the database interprets empty strings.
Batch Update	Selecting this checkbox signifies that you want to perform multiple statements by supplying an array of records as input to the activity where each record matches the prepared parameters for the statement. The statements are performed in one batch at the end of the activity's execution.
	If you do not select this checkbox, the expected input is the list of prepared parameters for the statement (no array of records). Some database drivers may not support batch updates. A `JDBCSQLException` is generated, if the database driver does not support batch updates.

Field	Description
Override SQL Update Statement	Selecting this checkbox overrides the SQL UPDATE statement specified on the Statement tab, and enables you to add a new SQL statement on the Input tab. **Note:** To connect to another database that has a schema with the same table structure, select the Override JDBC Connection checkbox. After doing this, go to the Input tab, and specify a Shared Resource at sharedResourceName.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Record	complex	This field appears when the **Batch Update** checkbox is selected on the **Advanced** tab.
InsertStatement	string	This field appears when the **Insert if record doesn't exist** checkbox is selected on the **Advanced** tab.
ServerTimeZone	string	Specifies the time zone for the database server.
timeout	number	The time (in seconds) to wait for the query to complete. If the query does not complete in the given time limit, an error is returned.
sharedResourceName	string	This field appears when the **Override JDBC Connection** checkbox is selected on the **Advanced** tab.
SqlUpdateStatement	string	Displays after the Override SQL Update Statement checkbox on the Advanced tab has been selected. This is the SQL UPDATE statement to perform in the database. You can use a SQL Builder wizard to build

Input Item	Datatype	Description
		the query for the desired database. You can also use a module property here.
		Note: If you create a SQL Update statement, ensure that the new statement does not differ from the output schema specified on the Output tab. If the new SQL statement does not match the output schema, an error is generated at run time. The output schema is determined from the original SQL Update statement on the Statement tab.

Output

The following is the output for the activity.

Output Item	Datatype	Description
noOfUpdates	integer	The number of updates performed by the statement specified on the **General** tab.
		This element can also return −2 that indicates, in the event of a batch update, that the updates were successful, but it is unknown how many updates were performed.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When.
JDBCConnectionNotFoundException	An error occurred when attempting to connect to the specified database.

Fault	Generated When.
InvalidTimeZoneException	An invalid time zone was specified.
JDBCSQLException	An SQL error occurred.
LoginTimedOutException	A timeout has occurred while attempting to connect to the database.
InvalidSQLTypeException	A parameter's datatype does not match the datatype of the table column.
ActivityTimedOutException	A timeout has been reached.

SQL Direct

SQL Direct is an asynchronous activity that runs an SQL statement that you supply. With this activity you can build an SQL statement dynamically using other activities, then pass the SQL statement into this activity's input. You can use this activity to run DDL SQL statements like CREATE and DROP. You can also run SQL statements that are not supported by other activities in the JDBC palette.

If this activity is not part of a transaction group, the SQL statement is committed after the activity completes. If this activity is part of a transaction group, the SQL statement is committed or rolled back with the other JDBC activities in the group at the end of the transaction.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the

Field	Literal Value/Module Property	Description
		process.
JDBC Shared Resource	Yes	The path to the shared resource containing the JDBC connection information. For more information, see JDBC Connection.
Timeout (sec)	Yes	The time (in seconds) to wait for the query to complete. If the query does not complete in the given time limit, an error is returned.
Maximum rows	Yes	The maximum number of rows to retrieve. To retrieve all rows, specify 0.

Description

Provide a short description for the SQL Direct activity.

Advanced

The **Advanced** tab has the following field.

Field	Description
ThreadPool Shared resource	Specifies a queue of threads available to run a queue of tasks. Thread pools are used to improve performance when running large numbers of asynchronous tasks by reducing per task invocation overhead, provide a means of bounding, and managing the resources consumed when running a collection of tasks.
Override Transaction Behavior	Overrides the default behavior of a transaction group. If this activity is in a transaction group, the activity is normally committed or rolled back with the other transactional activities. If you select this checkbox, this activity is not part of the transaction group

Field	Description
	and is committed when it completes. Selecting this option uses a separate database connection to perform the activity and commit the SQL statement.
Override JDBC Connection	Selecting this option overrides the default datasource connection, and allows for SQL queries to be run against the optional datasource. After selecting this option, the **sharedResourceName** field is enabled under the **Input > Data Source** tab. From the **sharedResourceName** field, type the complete path with the name of the optional datasource, which can be found under the **Resources** folder. For example, if a JDBC shared resource is located at **Resources > Package_Name > JDBC_SR_Name**, then the value you enter should be `<Package_Name>.<JDBC_SR_Name>`. The value you enter in the **XPath Expression** field is case sensitive and must be a string. To avoid mapping issues, ensure that the optional datasource maintains a structure that is identical to the default database, and that both datasources are using the same datatypes. **Note:** Transactions between multiple datasources are not supported.
Use Nil	Specifies whether NULLs are represented as optional schema elements or whether each item that can contain a NULL has subitems.

Input

The following is the input for the activity.

Input Item	Datatype	Description
statement	string	The SQL statement you want to run. You can use any process variables to build the statement dynamically, if required.
ServerTimeZone	string	Specifies the time zone for the database server.
timeout	number	The time (in seconds) to wait for the query to complete. If the query does not complete in the specified time limit,

Input Item	Datatype	Description
		an error is returned.
maxRows	number	The maximum number of rows to retrieve. To retrieve all rows, specify 0.

Output

The following is the output of the activity.

Output Item	Datatype	Description
noOfUpdates	string	The number of updates performed by the statement specified on the **General** tab.
		This element can also return −2 that indicates, in the event of a batch update, that the updates were successful, but it is unknown how many updates were performed.
unknownResultSet	string	The result of the database statement. The result element depends upon what is returned by the statement.
		The result set is a list of zero or more rows with zero or more columns. Each column contains a name and a value element. You can use XPath expressions to retrieve the desired row and column from the result set. There can be multiple results sets returned by the statements run by this activity.
		Because of this, this output item is a repeatable list of result sets. The first item in the list of result sets corresponds to the result set returned by the first statement run by this activity. The second item in the list of result sets corresponds to the result set returned by the second statement run by this activity, and so on.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When.
JDBCConnectionNotFoundException	An error occurred when attempting to connect to the specified database.
InvalidTimeZoneException	An invalid time zone was specified.
JDBCSQLException	An SQL error occurred.
LoginTimedOutException	A timeout has occurred while attempting to connect to the database.
ActivityTimedOutException	A timeout has been reached.

JMS Palette

Java Message Service (JMS) is a specification about sending and receiving messages between two or more applications in a Java environment. The **JMS** palette is used to send and receive JMS messages in a process.

The models supported are:

- Point-to-Point (queues): Message delivered to one recipient

- Publish/Subscribe (topics): Message delivered to multiple recipients

JMS supports these two models by using distinct interfaces within the API for each messaging model.

For more information about JMS and its message models, see the documentation of the JMS provider or JMS specifications.

Common JMS Properties and Headers

JMS properties comprise message headers and message properties. MessageHeader properties are set by the JMS client sending the message. You can view these after the message is received. You can also set MessageProperties on the outgoing messages on the **Input** tab of the activity that sends messages.

Message Headers

The following table describes the messages headers.

MessageHeaders Property	Datatype	Description
JMSDestination	string	The destination where the message is sent.
JMSReplyTo	string	The JMS destination where the reply message should be sent.

MessageHeaders Property	Datatype	Description
JMSDeliveryMode	string	The delivery mode of the message. Can be one of the following: • PERSISTENT: signifies the messages are stored and forwarded. • NON_PERSISTENT: messages are not stored and may be lost due to failures in transmission.
JMSMessageID	string	The unique ID of the message.
JMSTimestamp	long	The time a message was handed off to a JMS provider to be sent. The time is expressed as the amount of time, in milliseconds, since midnight, January 1, 1970, UTC.
JMSExpiration	long	The expiration time of the message. The time is expressed as the amount of time, in milliseconds since midnight, January 1, 1970, UTC. If set to 0, the message does not expire.
JMSRedelivered	boolean	Typically this item is set when the JMS provider has delivered the message at least once before.
JMSPriority	integer	The priority of the message. Priority is a value from 0-8. Higher numbers signify a higher priority (that is, 8 is a higher priority than 7).
JMSCorrelationID	string	This ID is used to link a response message with its related request message. This is usually the message ID of a request message when this field is found in a reply message.
JMSType	string	The JMS provider-supplied string to describe the type of the message. Some JMS providers use this property to define messages in the provider's repository. For more information about the use of this field, see the JMS provider documentation.

Message Properties

The following table describes the messages properties.

MessageProperties	Datatype	Description
JMSXUserID	string	The ID of the user sending the message. This property is optionally set on incoming messages by the JMS application. This property is not set on outgoing messages.
JMSXAppID	string	The ID of the application sending the message. This property is optionally set on incoming messages by the JMS application. This property is not set on outgoing messages.
JMSXProducerTIXID	string	The transaction identifier of the transaction where this message was produced. This property is optionally set on incoming messages by the JMS application. This property is not set on outgoing messages.
JMSXConsumerTXID	string	The transaction identifier of the transaction where this message was consumed. This property is optionally set on incoming messages by the JMS application. This property is not set on outgoing messages.
JMSXRcvTimestamp	integer	The time the JMS server delivered the message to the consumer. The time is expressed as the amount of time, in milliseconds since midnight, January 1, 1970, UTC. This property is optionally set on incoming messages by the JMS server. This property is not set on outgoing messages.

MessageProperties	Datatype	Description
JMSXDeliveryCount	integer	The number of delivery attempts for this message.
JMSXGroupID	string	The identity of the message group this message is a part of. You can set this property on outgoing messages to group messages into a numbered sequence.
JMSXGroupSeq	integer	The sequence number of this message in its group. You can set this property on outgoing messages to group messages into a numbered sequence.

Get JMS Queue Message

Get JMS Queue Message is an asynchronous activity that retrieves a message from the specified queue. You can use this activity to perform a receive operation on the queue as opposed to waiting for a queue message to be delivered to the **Wait for JMS Queue Message** activity.

You can use the **Message Selector** field on the **Advanced** tab to retrieve a specific queue message from the queue.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.

Field	Literal Value/Process Property/Module Property	Description
JMS Connection	Literal Value Module Property	The JMS connection. For more information, see JMS Connection.
Destination	Yes	The name of the queue from which to retrieve the message. The syntax of the destination name is specific to the JMS provider that you are using. For more information about queue names, see the JMS provider documentation.
Message Type	None	The type of the message. This can be one of the following: • **Text**: the message is a `java.lang.String`. • **Byte**: a stream of bytes. • **Map**: a set of name or value pairs. The names are strings, and the values are simple datatypes (JAVA primitives), an array of bytes (use the Binary datatype when mapping this data), or a string. Each item can be accessed sequentially or by its name. • **Object**: a Java object that can be serialized. • **Object Ref**: an object reference to a Java object. • **Simple**: a message with no body part. • **Stream**: a stream of Java primitives, strings, or arrays of bytes. Each value must be read sequentially. • **XML Test**: the message is XML text.

Field	Literal Value/Process Property/Module Property	Description
Acknowledge Mode	Yes	The acknowledge mode for incoming messages. It can be one of the following: • **Auto**: the message is automatically acknowledged, when it is received. • **Client**: the message is acknowledged at a later point by using the Confirm activity. If the message is not confirmed before the process instance ends, the message is redelivered and a new process instance is created to handle the new incoming message. Ensure that your process confirms the message when using the acknowledge mode. • **Dups Ok**: the message is acknowledged automatically when it is received. JMS provides this mode for lazy acknowledgment, but TIBCO Cloud Integration acknowledges messages on receipt. • **TIBCO EMS Explicit Client**: (Only available for TIBCO Enterprise Message Service) a message that is not acknowledged using the **Confirm** activity before the process instance ends, is redelivered instead of all messages in the session. The session is not blocked and one session handles all incoming messages for each process instance. • **TIBCO EMS Explicit Client Dups OK**: (Only available for TIBCO Enterprise Message Service) a message that is not acknowledged using the **Confirm** activity before the process instance ends, is redelivered instead of all messages in the session. The session is not blocked and one session handles all the

Field	Literal Value/Process Property/Module Property	Description
		incoming messages for each process instance. The messages however, are lazily acknowledged. • **TIBCO EMS No Acknowledge**: messages delivered using this mode do not require acknowledgment. Therefore, messages in this mode are not redelivered regardless of whether the delivery was successful.
Default Map Type Value	Yes	When the message is sent with the message type as **Map**, the optional XML schema elements having data type as **Int**, **Integer**, **Float**, **Decimal**, or **Double** with null values, and the **Default Map Type Value** checkbox is selected, then the default value is added while creating the node at the receiver side for respective element, otherwise the node is not created.

Description

Provide a short description for the Get JMS Queue Message activity.

Advanced

The **Advanced** tab has the following fields.

Field	Literal Value/Process Property	Description
Message Selector	Yes	A string to determine whether a message should be received. The syntax of the message selector is determined by the JMS provider (where message

Field	Literal Value/Process Property	Description
		properties are used instead of table column names).
		For more information and syntax for a message selector string, see the JMS provider documentation.
Application Properties Type	None	The type of application-specific message properties that is part of the message.
JMS Fetch Timeout	Yes	This field specifies the amount of time (in milliseconds) that the JMS Consumer waits to fetch message from the queue before it times out. If set to zero (0), the JMS Consumer does not time out. **Note:** JMS Fetch Timeout must be less than Activity Timeout in the **Input** tab.

Output Editor

The **Output Editor** tab defines the schema to use for messages of type Map, Stream, or XML Text. Map messages are name or value pairs, and using this schema you can define the structure of the retrieved destination queue message. The schema defined on the **Output Editor** tab becomes the body of the message on the **Output** tab. For XML Text message type, select an XSD element. For Map and Stream message types, select an XSD type in the **Output Editor** tab.

Input

The following is the input for the activity.

Input Item	Datatype	Description
destinationQueue	string	The queue to which to send the request. This input item overrides the **Destination** field on the **General** tab.

Input Item	Datatype	Description
timeout	integer	This field specifies the amount of time (in milliseconds) that the activity waits before it times out. If set to zero (0), the activity times out after default time, which is three minutes.
selector	string	A string to determine whether a message should be received. The value of this element overrides any value specified in the **Message Selector** field on the **Advanced** tab. The syntax of the message selector is determined by the JMS provider, but it is usually a subset of SQL92 (where message properties are used instead of table column names). For more information and syntax for a message selector string, see the JMS provider documentation.

Output

The following is the output of the activity.

Output Item	Datatype	Description
JMSHeaders	complex	The message header fields for the message. For more information about message header fields, see Common JMS Properties and Headers. Only properties applicable to this type of message are displayed.
JMSProperties	complex	The message properties for the message. For more information about message properties, see Common JMS Properties and Headers. Only properties applicable to this type of message are displayed.
Body	as per message	The body of the message.

Output Item	Datatype	Description
	type	
DynamicProperties	complex	Dynamic properties are an additional parameter to add runtime property to the outgoing JMS messages that are specified in the Input tab. This is an optional element with only one instance. Dynamic property can have multiple property elements.

Each property element denotes a single dynamic property and can contain the following elements:

- **Name** - Required. Name of the property with one instance.

- **Value** - Required. Value of the property with one instance.

- **Type** - Optional. Type of the property. If not provided, it is considered as string.

The following data types are supported:

- string

- boolean

- short

- integer

- long

- float

- double

- byte

Note: The DynamicProperty overwrites the value of a property (with the same name) added using the **JMS Application Property**. The DynamicProperties are also added to the outgoing message.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see *TIBCO BusinessWorks Container Edition Error Codes.*

Fault	Generated When..
JMSInvalidInputException	The input to the activity is not valid.
JMSSessionCreateException	The JMS session could not be created.
JMSReceiveException	The JMS receives operation failed.
ActivityTimedOutException	A timeout has been reached.

JMS Receive Message

JMS Receive Message is a process starter activity that starts the process based on the receipt of a message for the specified JMS destination.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Messaging Style	Yes	Select from one of the following available messaging style options: • **Generic**: This option corresponds to an

Field	Literal Value/Process Property/Module Property	Description
		abstract JMS Destination. If you are not sure whether your destination is a **Queue** or **Topic**, select **Generic**. • **Queue** • **Topic**
JMS Connection	None	The JMS connection. For more information, see JMS Connection.
Destination	Yes	The JMS destination from where the process starter receives the message.
Message Type	None	The type of the message. This can be one of the following: • **Text**: the message is a `java.lang.String`. • **Byte**: a stream of bytes. • **Map**: a set of name or value pairs. The names are strings, and the values are simple datatypes (JAVA primitives), an array of bytes (use the Binary datatype when mapping this data), or a string. Each item can be accessed sequentially or by its name. • **Object**: a Java object that can be serialized. • **Object Ref**: an object reference to a Java object. • **Simple**: a message with no body part. • **Stream**: a stream of Java primitives, strings, or arrays of bytes. Each value must be read sequentially. • **XML Test**: the message is XML text.

Field	Literal Value/Process Property/Module Property	Description
Durable Subscriber	Yes	Specifies a durable subscription. Specify a name in the **Subscription Name** field that is registered with the JMS application as the durable subscriber name. This field is only available if the **Messaging Style** is **Topic**.
Subscription Name	Yes	The subscription name registered with the JMS application for durable subscriptions. This field is only available when you select the **Durable Subscriber** checkbox. After creating a durable subscription, you can only remove this subscription by accessing the administration tool of the JMS provider. For more information about managing durable subscriptions, see the documentation of the JMS provider.
Suppress Local Messages	Yes	Specifies not to receive messages on the specified topic name when the message origin is the JMS application on the same connection as the process engine. If the process publishes and subscribes to messages with the same topic name, this option is useful if you want to specify whether to receive messages sent by the same JMS application that published the message. Selecting this checkbox prevents the process from receiving messages sent by the same connection. Keep this checkbox clear to specify that the messages sent by the same connection should be received.

Field	Literal Value/Process Property/Module Property	Description
Acknowledge Mode	Yes	The acknowledge mode for incoming messages. It can be one of the following: • **Auto**: the message is automatically acknowledged, when it is received. • **Client**: the message is acknowledged at a later point by using the **Confirm** activity. If the message is not confirmed before the process instance ends, the message is redelivered and a new process instance is created to handle the new incoming message. Ensure that your process confirms the message when using the acknowledge mode. • **Dups Ok**: the message is acknowledged automatically when it is received. JMS provides this mode for lazy acknowledgment, but TIBCO BusinessWorks Container Edition acknowledges messages on receipt. • **TIBCO EMS Explicit Client**: (Only available for TIBCO Enterprise Message Service) a message that is not acknowledged using the **Confirm** activity before the process instance ends, is redelivered instead of all messages in the session. The session is not blocked and one session handles all incoming messages for each process instance. • **TIBCO EMS Explicit Client Dups OK**: (only available for TIBCO Enterprise Message Service) a message that is not acknowledged using the **Confirm** activity before the process instance ends, is redelivered instead of all messages in the session. The session is not blocked and one session handles all the

Field	Literal Value/Process Property/Module Property	Description
		incoming messages for each process instance. The messages however, are lazily acknowledged.
		• **TIBCO EMS No Acknowledge**: messages delivered using this mode do not require acknowledgment. Therefore, messages in this mode are not redelivered regardless of whether the delivery was successful.
Max Sessions	None	Specifies the maximum number of client sessions that can connect with the messaging server. This property is enabled only when the **Acknowledge mode** is selected as **Client**. The default value is 1. **Note:** Do not use the `flowlimit` property when the **Max Session** value is greater than 1.
Shared Subscription	None	This field is enabled only when the **Messaging Style** field has **Topic** option selected. Select this checkbox to support shared subscription feature of JMS 2.0 specification. By default, the checkbox is clear.
Default Map Type Value	Yes	When the message is sent with the message type as **Map**, the optional XML schema elements having data type as **Int**, **Integer**, **Float**, **Decimal**, or **Double** with null values, and the **Default Map Type Value** checkbox is selected, then the default value is added while creating the node at the receiver side for respective element, otherwise the node is not created.

Description

Provide a short description for the JMS Receive Message activity.

Load-Balancing for Incoming Messages

One common application of a JMS message is to distribute messages across multiple receivers, thus balancing the processing of the messages. To achieve this goal, both the JMS server and TIBCO BusinessWorks Container Edition must be properly configured. The JMS server must allow the messages to be distributed across multiple receivers. For example, in TIBCO Enterprise Message Service, the exclusive property on the queue controls whether messages can be delivered across receivers or not. In TIBCO BusinessWorks Container Edition, the process definition containing the JMS Receiver must be deployed across multiple process engines. This creates multiple queue receivers for the same queue.

> **Note:**
> If the Sequencing Key field is set to preserve the order of incoming messages, to confirm the messages sequentially you must either set the Acknowledge mode to TIBCO EMS Explicit Client Acknowledge mode or set the Acknowledge mode to Client and Max Sessions to 1. Setting Max Sessions to 1 can limit the system's throughput, so using TIBCO Enterprise Message Service and TIBCO EMS Explicit Client Acknowledge is a better choice.

Advanced

The **Advanced** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Message Selector	Yes	A string to determine whether a message should be received. The syntax of the message selector is determined by the JMS provider.
		For more information and syntax for a message

Field	Literal Value/Process Property/Module Property	Description
		selector string, see the JMS provider documentation.
Application Properties Type	None	Any application-specific message properties that are part of the message. This is specified by the JMS application properties shared configuration object.
Polling Interval (sec)	None	Specifies the polling interval to check for new messages. If no value is specified for the property, the default polling interval is 2 seconds.
		Set a value in this field to specify the polling interval (in seconds) for this activity. The value in this field overrides the default polling interval.
Sequence Key	None	This field can contain an XPath expression that specifies which processes should run in sequence. Process instances with sequencing keys that evaluate to the same value are run in the sequence the process instances were created.
Custom Job Id	None	This field can contain an XPath expression that specifies a custom ID for the process instance.

Output Editor

The **Output Editor** tab defines the schema to use for incoming messages whose **Message Type** is **Map**, **Stream**, or **XML Text**. Map messages are name or value pairs, and you can use the schema to define the structure of the incoming message. The schema on the **Output Editor** tab becomes the structure used for the body of the message displayed on the **Output** tab. For XML Text message type, select an XSD element. For Map and Stream message types, select an XSD type in the **Output Editor** tab.

Output

The following is the output of this activity.

Output Item	Datatype	Description
JMSHeaders	complex	The message header fields for the message. For more information about message header fields, see Common JMS Properties and Headers. Only properties applicable to this type of message are displayed.
JMSProperties	complex	The message properties for the message. For more information about message properties, see Common JMS Properties and Headers. Only properties applicable to this type of message are displayed.
Body	as per message type	The body of the message.
DynamicProperties	complex	Dynamic properties are an additional parameter to add runtime property to the outgoing JMS messages that are specified in the **Input** tab. This is an optional element with only one instance. Dynamic property can have multiple property elements. Each property element denotes a single dynamic property and can contain the following elements: • Name - Required. Name of the property with one instance. • Value - Required. Value of the property with one instance. • Type - Optional. Type of the property. If not provided, it is considered as string. The following data types are supported: • string • boolean

Output Item	Datatype	Description
		• short
		• integer
		• long
		• float
		• double
		• byte
		Note: The DynamicProperty overwrites the value of a property (with the same name) added using the **JMS Application Property**.
		The DynamicProperties are also added to the outgoing message.

JMS Request Reply

JMS Request Reply is an asynchronous activity that is used to send a request to a JMS destination and wait for a response from the JMS client.

> **Note:** This activity uses temporary destinations to ensure that reply messages are received only by the process that sent the request. However, the user has the ability to use preexisting JMS destinations.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Messaging Style	Yes	Select from one of the following available options: • Generic • Queue • Topic
JMS Connection	None	The JMS connection. For more information, see JMS Connection.
Destination	Yes	The name of the destination of the outgoing message.
Message Type	None	The type of the message. This can be one of the following: • **Text**: the message is a `java.lang.String`. • **Byte**: a stream of bytes. • **Map**: a set of name or value pairs. The names are strings, and the values are simple datatypes (JAVA primitives), an array of bytes (use the Binary datatype when mapping this data), or a string. You can access each item sequentially or by its name. • **Object**: a Java object that can be serialized. • **Object Ref**: an object reference to a Java object. • **Simple**: a message with no body part. • **Stream**: a stream of Java primitives, strings, or arrays of bytes. Each value must be read sequentially.

Field	Literal Value/Process Property/Module Property	Description
		• **XML Text**: the message is XML text.

Description

Provide a short description for the JMS Request Reply activity.

Advanced

The **Advanced** tab has the following fields.

Field	Description
Reply To Destination	The destination to use for replies for this activity. **Note:** If more than one job has the same **Reply To Destination**, each job may not receive the correct reply. Ensure to specify an expression in this field that assigns a different **Reply To Destination** to each process instance.
Deliver Mode	The delivery mode of the message. Can be one of the following: • **Persistent**: signifies that the messages are stored and forwarded. • **Non_Persistent**: signifies that the messages are not stored and may be lost due to transmission failure. • **TIBCO_EMS_Reliable_Delivery**: this mode is only available when using TIBCO Enterprise Message Service. See the TIBCO Enterprise Message Service documentation for more information.
JMS Expiration (msec)	Corresponds to the `JMSExpiration` property that specifies how long the message can remain active in milliseconds. If set to 0, the message does not expire.

Field	Description
Priority	The priority of the message. You may set the priority to a value from 0-8. The default value is 4.
Type	The value to supply to the JMSType header property.
Application Properties Type	Any application-specific message properties that are part of the message. This is specified by the JMS application properties shared configuration object.

Input Editor and Output Editor

The **Input Editor** and **Output Editor** tabs define the schema to use for messages of type **Map**, **Stream**, or **XML Text**. Map messages are name or value pairs, and you can use the schema to define the structure of the outgoing request and the incoming reply. The schema defined on the **Input Editor** tab becomes the body of the message on the **Input** tab. The schema defined on the **Output Editor** tab becomes the body of the message on the **Output** tab. For XML Text message type, select an XSD element. For Map and Stream message types, select an XSD type in the **Input Editor** or **Output Editor**.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Destination	string	The destination to which to send the request. This input item overrides the **Destination** field on the **General** tab.
replyTo	string	The destination to use for replies for this activity. **Note:** If more than one job has the same **Reply To Destination**, then each job may not receive the correct reply. Ensure to specify an expression in this field that assigns a different **Reply To Destination** to each process instance.

Input Item	Datatype	Description
JMSExpiration	integer	Specifies how long the message can remain active in milliseconds. If set to zero (0), the message does not expire.
JMSPriority	string	The priority of the message. This item overrides the priority set on the **Advanced** tab.
JMSDeliveryMode	string	The delivery mode of the message. Can be one of the following: **Persistent**: signifies that the messages are stored and forwarded. **Non_Persistent**: signifies that the messages are not stored and may be lost due to transmission failure. **TIBCO_EMS_Reliable_Delivery**: this mode is only available when using TIBCO Enterprise Message Service. For more information, see the TIBCO Enterprise Message Service documentation.
JMSCorrelationID	string	This ID is used to link a response message with its related request message. This is usually the message ID of a request message when this field exists in a reply message.

Input Item	Datatype	Description
		Note: If the correlation ID is not specified the JMS Message ID is passed as a selector. The **Reply To JMS Message** activity replies for that corresponding ID only. If you use the **JMS Send Message** activity instead of the **Reply To JMS Message** activity for sending the reply to the **JMS Request Reply** activity, then you must explicitly map the JMS Correlation ID in the JMS Send Message activity with the Message ID in the **JMS Request Reply** activity to maintain the correct order of the request and reply.
JMSType	string	The type of the message. This item overrides the value specified on the **Advanced** tab.
JMSProperties	complex	The message properties. For more information, see Common JMS Properties and Headers.
Body	depends on the message type	The body of the message.
DynamicProperties	complex	Dynamic properties are an additional parameter to add runtime property to the outgoing JMS messages that are specified in the Input tab. This is an optional element with only one instance. Dynamic property can have multiple property elements. Each property element denotes a single dynamic property and can contain the following elements: **Name** - Required. Name of the property with one instance. **Value** - Required. Value of the property with one instance. **Type** - Optional. Type of the property. If not

Input Item	Datatype	Description
		provided, it is considered as string.
		The following data types are supported:
		• string
		• boolean
		• short
		• integer
		• long
		• float
		• double
		• byte
		Note: The DynamicProperty overwrites the value of a property (with the same name) added using the **JMS Application Property**.
		The DynamicProperties are also added to the outgoing message.
requestTimeout	integer	This field specifies the amount of time (in milliseconds) that the activity waits before it times out.

Output

The following is the output of this activity.

Output Item	Datatype	Description
JMSHeaders	complex	The message header fields for the message. For more information about message header fields, see Common JMS Properties and Headers.
		Only properties applicable to this type of message are displayed.

Output Item	Datatype	Description
JMSProperties	complex	The message properties for the message. For more information about message properties, see Common JMS Properties and Headers. Only properties applicable to this type of message are displayed.
Body	as per message type	The body of the message.
DynamicProperties	complex	Dynamic properties is an additional parameter to add runtime property to the outgoing JMS messages that are specified in the Input tab. This is an optional element with only one instance. Dynamic property can have multiple property elements. Each property element denotes a single dynamic property and can contain the following elements: Name - Required. Name of the property with one instance. Value - Required. Value of the property with one instance. Type - Optional. Type of the property. If not provided, it is considered as string. The following data types are supported: • string • boolean • short • integer • long • float • double

Output Item	Datatype	Description
		• byte
		Note: The DynamicProperty overwrites the value of a property (with the same name) added using the **JMS Application Property**.
		The DynamicProperties are also added to the outgoing message.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
JMSInvalidInputException	The input to the activity is not valid.
JMSMessageCreateException	The JMS message could not be created.
JMSSessionCreateException	The JMS session could not be created.
JMSSendException	The JMS sends operation failed.
JMSReceiveException	The JMS receives operation failed.
ActivityTimedOutException	A timeout has been reached.

JMS Send Message

JMS Send Message is a synchronous activity that sends a message to the specified JMS destination.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Messaging Style	None	Select from one of the following available options: • Generic • Queue • Topic
JMS Connection	None	The JMS connection. For more information, see JMS Connection.
Destination	Yes	The name of the destination of the outgoing message.
Message Type	None	The type of the message. This can be one of the following: • **Text**: the message is a `java.lang.String`. • **Byte**: a stream of bytes. • **Map**: a set of name or value pairs. The names are strings, and the values are simple datatypes (JAVA primitives), an array of bytes (use the Binary datatype when mapping this data), or a string. Each item can be accessed sequentially or by its name. • **Object**: a Java object that can be serialized. • **Object Ref**: an object reference to a Java object.

Field	Literal Value/Process Property/Module Property	Description
		• **Simple**: a message with no body part.
		• **Stream**: a stream of Java primitives, strings, or arrays of bytes. Each value must be read sequentially.
		• **XML Text**: the message is XML text.

Description

Provide a short description for the JMS Send Message activity.

Advanced

The **Advanced** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Reply To Destination	Yes	The destination to use for replies for this activity. **Note:** If more than one job has the same **Reply To Destination**, each job may not receive the correct reply. Ensure to specify an expression in this field that assigns a different **Reply To Destination** to each process instance.
Delivery Mode	None	The delivery mode of the message. Can be one of the following: • **Persistent**: signifies that the messages are stored and forwarded.

Field	Literal Value/Process Property/Module Property	Description
		• **Non_Persistent**: signifies that the messages are not stored and may be lost due to transmission failure.
		• **TIBCO_EMS_Reliable_Delivery**: this mode is only available when using TIBCO Enterprise Message Service. For more information, see the TIBCO Enterprise Message Service documentation.
JMS Expiration (msec)	None	Corresponds to the `JMSExpiration` property that specifies how long the message can remain active in milliseconds. If set to 0, the message does not expire.
Delivery Delay (msec)	Yes	Delivery Delay feature, which is supported in EMS, is now supported in TIBCO BusinessWorks Container Edition 2.3.2. Delivery Delay is the minimum length of time in milliseconds that must elapse after a message is sent before the JMS provider may deliver the message to a consumer.
Priority	None	The priority of the message. You may set the priority to a value from 0-8. The default value is 4.
Type	None	The value to supply to the `JMSType` header property.
Application Properties Type	None	Any application-specific message property that is part of the message. This is specified by the JMS application properties shared configuration object.
Override	None	Overrides the default behavior of a transaction group.

Field	Literal Value/Process Property/Module Property	Description
Transaction Behavior		When the activity is in a transaction group, the JMS message is normally committed or rolled back with the other participants in the transaction.
		When the checkbox is selected, the JMS message does not participate in the transaction.

Input Editor

The **Input Editor** tab defines the schema to use for outgoing messages whose message type is **Map**, **Stream**, or **XML Text**. Map messages are name or value pairs, and you can use the schema to define the structure of the outgoing message. After defining the schema on the **Input Editor** tab, it becomes the structure used for the body of the message displayed on the **Input** tab. For the XML Text message type, select an XSD element. For Map and Stream message types, select an XSD type in the **Input Editor** tab.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Destination	string	The destination to which to send the message. This input item overrides the **Destination** field on the **General** tab.
replyTo	string	The destination to use for replies for this activity.
		Note: If more than one job has the same **Reply To Destination**, each job may not receive the correct reply. Ensure to specify an expression in this field that assigns a different **Reply To Destination** to each process instance.

Input Item	Datatype	Description
JMSExpiration	integer	Specifies how long the message can remain active in milliseconds. If set to 0, the message does not expire.
JMSPriority	string	The priority of the message. This item overrides the priority set on the **Advanced** tab.
JMSDeliveryMode	string	The delivery mode of the message. Can be one of the following: • **Persistent**: signifies that the messages are stored and forwarded. • **Non_Persistent**: signifies that the messages are not stored and may be lost due to transmission failure. • **TIBCO_EMS_Reliable_Delivery**: this mode is only available when using TIBCO Enterprise Message Service. For more information, see the TIBCO Enterprise Message Service documentation.
JMSCorrelationID	string	This ID is used to link a response message with its related request message. This is usually the message ID of a request message when this field exists in a reply message.
JMSType	string	The type of the message. This item overrides the value specified on the **Advanced** tab.
JMSProperties	complex	The message properties. For more information, see Common JMS Properties and Headers.
Body	depends on the message type	The body of the message.

Input Item	Datatype	Description
DynamicProperties	complex	Dynamic properties are an additional parameter to add runtime property to the outgoing JMS messages that are specified in the Input tab. This is an optional element with only one instance. Dynamic property can have multiple property elements. Each property element denotes a single dynamic property and can contain the following elements: • **Name** - Required. Name of the property with one instance. • **Value** - Required. Value of the property with one instance. • **Type** - Optional. Type of the property. If not provided, it is considered as string. The following data types are supported: • string • boolean • short • integer • long • float • double • byte **Note:** The DynamicProperty overwrites the value of a property (with the same name) added using the **JMS Application Property**. The DynamicProperties are also added to the outgoing message.

Output

The following is the output of the activity.

Output Item	Datatype	Description
MessageID	string	The unique identifier of the message.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
JMSInvalidInputException	The input to the activity is not valid.
JMSMessageCreateException	The JMS message could not be created.
JMSSessionCreateException	The JMS session could not be created.
JMSSendException	The JMS sends operation failed.

Reply to JMS Message

Reply To JMS Message is a synchronous activity that sends a reply to a previously received JMS queue or topic message. The **Reply For Event** field in the **General** tab lists the activities that can receive the JMS message. The activity you select determines the response of the reply message.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label of the activity.
Reply For Event	A drop-down list of activities that can receive JMS queue or topic messages. The activity you select determines the message this activity replies to. The reply-to queue or topic name from the message in the selected activity is used to send the reply message. The listed activity is **WaitforJMSRequest**.
Message Type	The type of the message. This can be one of the following: • **Text**: the message is a `java.lang.String`. • **Byte**: a stream of bytes. • **Map**: a set of name or value pairs. The names are strings, and the values are simple datatypes (JAVA primitives), an array of bytes (use the Binary datatype when mapping this data), or a string. Each item can be accessed sequentially or by its name. • **Object**: a Java object that can be serialized. • **Object Ref**: an object reference to a Java object. • **Simple**: a message with no body part. • **Stream**: a stream of Java primitives, strings, or arrays of bytes. Each value must be read sequentially. • **XML Text**: the message is XML text.

Description

Provide a short description for the Reply to JMS Message activity.

Advanced

The **Advanced** tab has the following fields.

Field	Description
Delivery Mode	The delivery mode of the message. This can be one of the following:

Field	Description
	• **Persistent**: signifies that the messages are stored and forwarded.
	• **Non_Persistent**: signifies that the messages are not stored and may be lost due to transmission failure.
	• **TIBCO_EMS_Reliable_Delivery**: this mode is only available when using TIBCO Enterprise Message Service.
	For more information, see the TIBCO Enterprise Message Service documentation.
JMS Expiration (msec)	Corresponds to the `JMSExpiration` property that specifies how long the message can remain active in milliseconds. If set to zero (0), the message does not expire.
Priority	The priority of the message. You may set the priority to a value from 0-8. The default value is 4.
JMS Type	The value to supply to the `JMSType` header property.
Application Properties Type	Any application-specific message property that is part of the message. This is specified by the JMS application properties shared configuration object.

Input Editor

The **Input Editor** tab defines the schema to use for outgoing messages whose message type is **Map**, **Stream**, or **XML Text**. Map messages are name or value pairs, and you can use the schema to define the structure of the outgoing message. After defining the schema on the **Input Editor** tab, it becomes the structure used for the body of the message displayed on the **Input** tab. For XML Text message type, select an XSD element. For Map and Stream message type, select an XSD Type in the **Input Editor** tab.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Destination	string	The destination to which to send the request. This input item overrides the **Destination** field on the **General** tab.
replyTo	string	The destination to use for replies for this activity. **Note:** If more than one job has the same **Reply To Destination**, each job may not receive the correct reply. Ensure to specify an expression in this field that assigns a different **Reply To Destination** to each process instance.
JMSExpiration	integer	Specifies how long the message can remain active in milliseconds. If set to 0, the message does not expire.
JMSPriority	string	The priority of the message. This item overrides the priority set on the **Advanced** tab.
JMSDeliveryMode	string	The delivery mode of the message. Can be one of the following: • **Persistent**: signifies that the messages are stored and forwarded. • **Non_Persistent**: signifies that the messages are not stored and may be lost due to transmission failure. • **TIBCO_EMS_Reliable_Delivery**: this mode is only available when using TIBCO Enterprise Message Service. For more information, see the TIBCO Enterprise Message Service documentation.
JMSCorrelationID	string	This ID is used to link a response message with its related request message. This is usually the message ID of a request message when this field exists in a

Input Item	Datatype	Description
		reply message.
JMSType	string	The type of the message. This item overrides the value specified on the **Advanced** tab.
JMSProperties	complex	The message properties. For more information, see Common JMS Properties and Headers.
Body	depends on the message type	The body of the message.
DynamicProperties	complex	Dynamic properties are an additional parameter to add runtime property to the outgoing JMS messages that are specified in the Input tab. This is an optional element with only one instance. Dynamic property can have multiple property elements.

Each property element denotes a single dynamic property and can contain the following elements:

- **Name** - Required. Name of the property with one instance.
- **Value** - Required. Value of the property with one instance.
- **Type** - Optional. Type of the property. If not provided, it is considered as string.

The following data types are supported:

- string
- boolean
- short
- integer
- long

Input Item	Datatype	Description
		• float
		• double
		• byte
		Note: The DynamicProperty overwrites the value of a property (with the same name) added using the **JMS Application Property**.
		The DynamicProperties are also added to the outgoing message.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
JMSInvalidInputException	The input to the activity is not valid.
JMSMessageCreateException	The JMS message could not be created.
JMSSessionCreateException	The JMS session could not be created.
JMSSendException	The JMS sends operation failed.

Mail Palette

The Mail Palette is used to receive incoming emails or send outgoing emails.

Receive Mail

Receive Mail is a process starter activity that polls a mail server for the new mail. After detecting and retrieving a new mail, the **Receive Mail** activity starts the process.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Host	Yes	The host name or IP address for the mail server.
Port	Yes	The default (110) port used to connect to the POP3 server.
Protocol	No	The protocol used by receive mail activity to connect to the mail server. The following are the types of protocols: • POP3 • GraphAPI • IMAP

Field	Literal Value/Module Property/Process Property	Description
Authentication Type	No	Select the authentication type. There are two types of authentication: • Basic • OAuth 2.0
Username	Yes	The username for logging into the server.
Password	Yes	The password for logging into the POP3 server. When the **Authentication Type** is selected as **OAuth 2.0**, the **Password** field is disabled. When the **Protocol** is selected as **GraphAPI**, the **Password** field is disabled.
OAuth Configuration	Yes	The name of the OAuth Configuration resource. In the OauthConfiguration Resource Template wizard, create a resource to connect to OAuth Configuration. This field is enabled only when **IMAP** and **GraphAPI** is selected in the **Protocol** field. For more information, see OAuth Configuration Resource.
Polling Interval (Sec)	Yes	The polling interval (in seconds) to check for new mail. The default is 5, if no polling interval is specified.
Delete Mail	None	Select this checkbox to delete the mail from the server after the process starter has retrieved it.

Field	Literal Value/Module Property/Process Property	Description
		Note: To delete mail, place a **Checkpoint** activity immediately after the **Receive Mail** process starter. This ensures that the message is not lost in the event of a machine failure. Alternatively, you can leave this field clear.
Enable Confidentiality	None	This checkbox specifies whether a Secure Sockets Layer (SSL) Client should be used to specify the SSL configuration. When this checkbox is selected, the **SSL Client** field displays.
SSL Client	Yes	The name of the resource. In the SslClientResource Resource Template wizard, create a resource to connect to the SSL client.

Test Connection Button

The **Test Connection** button tests the connection to the specified mail server. Use this button to ensure that your **Receive Mail** activity is properly configured to receive mail from the specified mail server. This button is enabled only when the POP3 protocol is selected, and the **Host**, **User Name**, and **Password** fields on the **General** tab are populated.

Description

Provide a short description for the receive mail activity.

Advanced

The **Advanced** tab contains the following fields:

Field	Description
Sequence Key	Contains an Xpath expression that specifies which process runs in a sequence. Process instances with sequencing keys evaluating to the same value, are run in the sequence the process instance was created.
Custom Job Id	This field can contain an Xpath expression that specifies a custom ID for the process instance.
Provide Raw Message	When selected, this field specifies the whole message to be provided in the binary form. The output schema changes to include an element named rawMessage that contains the message instead of bodyText and mimeEnvelopeElements. The **Receive Mail** process starter uses the `javax.mail` API. This API assumes that the content type of the message is `multipart/mixed`. When the `multipart` subtype is not `mixed` (such as **multipart/signed** or `multipart/alternative`) or when another content type is used (for example, `application/*`), the `javax.mail` API may not correctly parse the message. Select this checkbox to send the raw message to a **Java Invoke** activity that uses the Java activation framework to implement the appropriate handlers to process the message. For more information about how the Java activation framework can be used for this purpose, see the *API documentation* for `javax.activation` and `javax.mail`. You can also feed the binary message to the `javax.mail.MimeMessage` constructor in the form of a `java.io.InputStream` to construct a `MimeMessage` object. The default `javax.mail` API handler is used in such a case.
Write to File	Select this checkbox to specify that the incoming messages whose body and attachments (or raw message size, if **Provide Raw Message** is selected) exceed the specified threshold size. It must be written to a file instead of being stored in memory. You can accept large incoming messages without consuming more memory. The incoming message is written to the file with mail headers when the body and attachment exceed the specified size. Selecting this checkbox displays the **Directory**, **Creating Non-Existing Directories**, and **Threshold Data Size(bytes)** fields.

Field	Description
	Do not select to keep the incoming messages in memory. **Note:** The files created by using this option are not deleted automatically. Manage the storage used by these files and delete them when they are no longer used.
Directory	The directory to write messages that are above the specified threshold. The process engine does not attempt to create the directory if the specified directory does not exist. Therefore, create the directory before starting the process engine.
Create Non-Existing Directories	The directory to write messages that are above the specified threshold. If the specified directory does not exist, the process engine does not attempt to create the directory. Therefore, select the checkbox to create the directory before starting the process engine. Not selecting this checkbox with one or more non-existing directories in the specified path in the **Directory** field, raises an exception.
Threshold Data Size (bytes)	The maximum size (in bytes) of an incoming message that can be kept in memory. Messages larger than the specified size are written to a file in the specified directory. The file name is an output so that the subsequent activities in the process properties can access the file and read its contents. Specifying zero (0) in this field determines all incoming messages to be saved to a file.

Output Editor

Incoming messages may contain custom headers. From the **Output Editor** tab, define a custom schema for the headers of the incoming mail messages. You can also specify any standard header supported by the `javax.mail` package. For example, X-Mailer or X-Priority. Values of headers in the incoming message populate the corresponding defined output headers of the same name.

Output

The following is the output of the activity.

Output Item	Datatype	Description
From	String	The email address of the sender of the email.
to	String	The recipient list of the email.
cc	String	The cc (carbon copy) list of the email.
replyTo	String	The replyTo list of the email.
subject	String	The subject of the email.
sentdate	String	The date the email was sent.
Headers	Complex	This element contains the schema you defined on the **Output Editor** tab. You can specify custom headers that can be included in the incoming mail messages.
bodyElement	String	Contains a choice element. You can either provide the bodyText element or fileName element.
mimeEnvelopeElement	Complex	This field is available when the **Provide Raw Message** checkbox on the **Advanced** tab is not selected. Contains the message attachments. This element contains a repeating element named mimePart that comprises each mime attachment.
mimeHeaders	Complex	This element contains the mime header for each mimePart. mimeHeaders contain the following information: • Content-disposition - To suggest a filename for an attachment use `"*;filename=<filename>"` in this element.

Output Item	Datatype	Description
		Note: E-mail servers may alter or ignore the suggested name. • Content-type • Content-transfer-encoding • Content-id • Any element For more information about MIME headers and their syntax, see http://www.faqs.org/rfcs/rfc2045.html. **Note:** When the contentType is specified as "`text/*`" (for example, "`text/xml`"), specify the attachment content in either the textContent input element or the file name storing the attachment in the fileName input element. When the contentType is anything other than "`text/*`": • The attachment content must be in the binaryContent input element. • The file name storing the attachment must be in the fileName input element.
binaryContent | fileName | textContent	choice	This element contains the mime attachment. It can be any of the following: • binaryContent: the content of the attachment when the attachment is binary data. • fileName: the file name of the attachment written on the disk. • textContent: the content of the attachment when the attachment is text data.

Send Mail

Send Mail is a synchronous activity that sends an email by way of an SMTP server.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Protocol	None	The protocol used by send mail activity to connect to the mail server. The following are the types of protocols: • GraphAPI • SMTP
SMTP Property	Yes	The property name of the SMTP server. If your server is configured for using a different port, specify it in this field.
OAuth Configuration	Yes	The name of the OAuth Configuration shared resource. In the OauthConfiguration Resource Template wizard, create or select a resource to connect to OAuth Configuration. This field is enabled only when **GraphAPI** is selected in the **Protocol** field. For more information, see OAuth Configuration Resource.
SSL Client	Yes	The name of the SSL Client shared resource. In the SslClientResource Resource Template wizard, create or select a resource to connect to the SSL client.

Description

Provide a short description for the **Send Mail** activity.

Advanced

The **Advanced** tab has the following fields.

Field	Description
Allow Non-Standard Email IDs	By default, email IDs specified in the to, from, cc, bcc, or replyTo input elements must conform to the RFC 822 standard for email addresses.
	Select this checkbox to specify email IDs that do not conform to the RFC 822 standard. This is useful for sending mail to a server that has extensions for sending email to devices such as a fax machine.
	For more information about email address syntax, see http://www.faqs.org/rfcs/rfc822.html.
	Note: When this field is not selected, not all the RFC 822 syntax rules are enforced. For example, addresses composed of simple names (with no "@domain" part) are allowed.
	For more information about enforcing RFC 822, see the description of the strict flag of the parse method of `javax.mail.internet.InternetAddress` at https://docs.oracle.com/.

Input Editor

You may want to add custom headers to outgoing mail messages. From the **Input Editor** tab, you can define a custom schema for the headers of the outgoing mail message. You can specify any standard header supported by the `javax.mail` package. For example, X-Mailer or X-Priority.

Input

The following is the input for the activity.

Input Item	Datatype	Description
username	String	The username to use when authenticating to the mail server.
password	String	The password to use when authenticating to the mail server.
from	String	The email address of the sender.
to	String	The recipient list for the email. Use this repeatable element to send mail to more than one recipient. Provide a list of "to" recipients in a single string by using either a comma or a semicolon (but not both in the same string) as delimiters.
cc	String	The "cc" (carbon copy) list for the email. Use this repeatable element to place more than one recipient on the "cc" list, if required. Provide a list of "cc" recipients in a single string by using either a comma or a semicolon (but not both in the same string) as delimiters.
bcc	String	The "bcc" (blind carbon copy) list for the email. Use this repeatable element to place more than one recipient on the "bcc" list. Provide a list of "bcc" recipients in a single string by using either a comma or a semicolon (but not both in the same string) as delimiters.
replyTo	String	The "replyTo" list for the email. Use this repeatable element to place more than one recipient on the "replyTo" list. Provide a reply to list a single string by using either a comma or a semicolon (but not both in the same string) as delimiters to separate the addresses in the list.

Input Item	Datatype	Description
subject	String	The subject of the email.
sentdate	String	The date stamp for the email. **Note:** The email server provides the actual date stamp for the email, so this input item is ignored. Do not attempt to place a value in this input item.
bodyElement	Complex	Contains a choice element. You can either provide the bodyText element or fileName element.
bodyText	String	The text of the email message.
fileName	String	The file name of the attachment written to the disk.
Headers	Complex	Contains the schema you defined on the **Input Editor** tab. Specify custom headers for the outgoing mail messages. Specify any standard header supported by the `javax.mail` package. For example, X-Mailer or X-Priority. To use attachments, specify the Content-Type header as `multipart/*` (where * is a valid subtype of the multipart).
contentType	String	The mime content type for the message. You can specify a character encoding in this element for the encoding of the body of the message.
mimeEnvelopeElement	Complex	Contains the message attachments.
mimePart	Complex	This repeating element comprises each mime

Input Item	Datatype	Description
		attachment.
mimeHeaders	Complex	This element contains the mime header for each mimePart.
		mimeHeaders contain the following information:
		• Content-disposition - To imply a filename for an attachment, use `"*;filename=<filename>"` in this element.
		Note: E-mail servers may alter or ignore the suggested name.
		• Content-type
		• Content-transfer-encoding
		• Content-id
		• Any element
		For more information about MIME headers and their syntax, see http://www.faqs.org/rfcs/rfc2045.html.
		Note: When the contentType is specified as `"text/*"` (for example, `"text/xml"`), specify the attachment content in either the textContent input element or the file name storing the attachment in the fileName input element. When the contentType is anything other than `"text/*"`:
		• The attachment content must be in the binaryContent input element.
		• The file name storing the attachment must be in the fileName input element.
binaryContent | fileName | textContent	Choice	This element contains the mime attachment. It can be any of the following:

Input Item	Datatype	Description
		• binaryContent: the content of the attachment when the attachment is binary data.
		• fileName: the file name of the attachment written on the disk.
		• textContent: the content of the attachment when the attachment is text data.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
InvalidInputException	The input is not valid.
UnknownHostException	The mail server host name is not resolved. The exception contains the name of the mail server host.
SendFailedException	The send operation failed for some reason.

Parse Palette

The **Parse** palette provides shared configuration resources for parsing and rendering the formatted text. This is useful if you want to transform the formatted lines of text into a data schema or transform a data schema into a formatted text string.

The text lines can be formatted either by delimiters separating each field or offsets can be specified to determine where each field begins and ends.

Parsing a Large Number of Records

The input for this activity is placed in a process variable and takes up memory as it is being processed. When reading a large number of records from a file, the process may consume significant machine resources. To avoid too much memory, you may want to read the input in parts, parsing and processing a small set of records before moving on to the next set of records.

This procedure is a general guideline for creating a loop group for parsing a large set of input records in parts. You may want to modify the procedure to include additional processing of the records, or you may want to change the XPath expressions to suit your business process. If processing a large number of records, do the following.

Procedure

1. Select and drop the **Parse Data** activity on the process editor.

2. On the **General** tab, specify the fields and select the **Manually Specify Start Record** checkbox.

3. Select the **Parse Data** activity and click the group icon on the tool bar to create a group containing the **Parse Data** activity.

4. Specify **Repeat Until True Loop** as the Group action, and specify an index name (for example, "i").

   The loop must exit when the EOF output item for the **Parse Data** activity is set to true. For example, the condition for the loop can be set to the following: `string ($ParseData/Output/done) = string(true())`

5. Set the **noOfRecords** input item for the **Parse Data** activity to the number of records you want to process for each execution of the loop.

    If you do not select the **Manually Specify Start Record** checkbox on the **General** tab of the **Parse Data** activity, the loop processes the specified **noOfRecords** with each iteration, until there are no more input records to parse.

    You can optionally select the **Manually Specify Start Record** checkbox to specify the **startRecord** on the **Input** tab. If you do this, you must create an XPath expression to properly specify the starting record to read with each iteration of the loop. For example, the count of records in the input starts at zero, so the **startRecord** input item could be set to the current value of the loop index minus one. For example, $i - 1.

MimeParser

This activity is responsible for parsing MTOM messages into a SOAP message that contains binary attachments as an inline data of the SOAP message. The output can be either in a Text or in a Binary format.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Input Style	The input message format. (Currently only MTOM is available.)
Output Style	The output message format. You can select either **Text** or **Binary** format for the output messages.

Description

Provide a short description for the MimeParser activity.

Advanced

The **Advanced** tab has the following.

Field	Description
Generate Strict Parsing Errors	Select this checkbox to generate strict parsing validation errors.

Input

The following is the input for the activity.

Input Item	Datatype	Description
MimeBinaryData	Binary	This is the incoming binary data.

Output

The following is the output of the activity.

Output Item	Datatype	Description
soapMessage	Binary or String	Depends on the type selected in the **Output Style** format.
StrictValidationErrors	Complex	This displays only when the **Generate Strict Parsing Errors** checkbox is selected.
ErrorString	String	Shows strict parsing validation error logs.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When...
XOPParseException	The input has XOP-related exceptions.
MimeParseException	The input has MIME related exceptions.
InvalidXMLException	The input has invalid xml related exception.

Parse Data

The **Parse Data** activity takes a text string or input from a file and processes it by converting it into a schema tree based on the specified **Data Format** shared resource.

You can use any mechanism to obtain or create a text string for processing. For example, you can use the **Read File** activity to obtain text from a file. You can also use this activity to specify a text file to read.

You can use this activity in various scenarios. For example, a user has a file comprising multiple lines with comma-separated values (as in data obtained from a spreadsheet) and this data has to be inserted into a database table. In such a scenario, read and parse the file into a data schema with the **Parse Data** activity. Then, use the **JDBC Update** activity to insert the data schema into a database table.

General

The **General** tab has the following fields.

Field	Module Property	Description
Name	No	The name to be displayed as the label for the activity in the process.
Data Format	No	The Data Format shared resource to use when parsing the text input.
Input Type	No	Specify the type of input for this activity.

Field	Module Property	Description
		Input can either be is **String** or **File**. If the input is a text string, provide the string to the `text` input item. If the input is a file, provide the file name and location to the `fileName` input item.
Encoding	Yes	The encoding of the input file.
		To enable this field, select the **File** option in the **Input Type** field. Any valid Java encoding name can be used.
Skip Blank Spaces	No	Select this checkbox to skip any empty records when parsing the text input.
		When this checkbox is not selected, parsing stops at the first blank line encountered in the input.
Manually Specify Start Record	No	You can specify the record in the input where you want to start parsing.
		This is useful if you have a large number of records and you want to read the input in parts (to minimize memory usage).
		Selecting this checkbox displays the startRecord input item. For more information about how to read the input stream in parts, see Parsing a Large Number of Records.
Strict Validation	No	Validates every input line for the specified number of fields for the fixed-format text.
		For example, if the format states that there are three fields per line and this checkbox is selected, all lines in the input must contain three fields.
Continue On Error	No	Continues parsing the next record in the input after encountering an error, if any.
		If an error occurs, the error information is separated from the output of the successfully parsed records and is provided in the output schema of the activity.

Field	Module Property	Description
		When this checkbox is not selected, the **Parse Data** activity quits parsing if an error is encountered while parsing the records in the input. Irrespective of whether this checkbox is selected or not, the **Parse Data** activity quits when any data validation errors occur.

Description

Provide a short description for the Parse Data activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
text	string	The text string to parse. This input item is available only when **String** is specified in the **Input Type** field of the **General** tab.
fileName	string	The location and name of the file to read. The file's content is used as the input text string for this activity. This input item is available only when **File** is specified in the **Input Type** field of the **General** tab.
startRecord	number	The line number of the input stream to begin parsing. All lines before the specified line are ignored. This input item is available only if the **Manual Specify Start Record** checkbox on the **General** tab is selected.

Input Item	Datatype	Description
		The input stream begins with the line number 1 (one). This is useful for reading the input stream in parts to minimize memory usage. For more information, see Parsing a Large Number of Records
noOfRecords	number	The number of records to read from the input stream. Specify -1 if you want to read all records in the input stream. This is useful for reading the input stream in parts to minimize memory usage. For more information, see Parsing a Large Number of Records.
SkipHeaderCharacters	integer	The number of characters to skip when parsing. You can skip over any file headers or other unwanted information.

Output

The following is the output of the activity.

Output item	Datatype	Description
Rows	complex	This output item contains the list of parsed lines from the input. This is useful to determine the number of records parsed by this activity. The schema specified by the **Data Format** resource is contained in this output item.
schema	complex	The schema containing the data from the parsed input text. This output item contains zero or more parsed records.
ErrorRows		This output item is available when you select **Continue on**

Output item	Datatype	Description
		Error, and errors while parsing the records in the input.
		Raw input data is put in the error string.
		This field contains the list of error lines for the records from the input that failed parsing.
done	boolean	true if no more records are available for parsing. false if there are more records available.
		This output item is useful to check whether there are no more records in the input stream when reading the input in parts to preserve memory.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
FileNotFoundException	The specified input file cannot be located.
BadDataFormatException	The input format is not valid.

Parsing a Large Number of Records

The input for this activity is placed in a process variable and takes up memory as it is being processed. When reading a large number of records from a file, the process may consume significant machine resources. To avoid too much memory, you may want to read the input in parts, parsing and processing a small set of records before moving on to the next set of records.

This procedure is a general guideline for creating a loop group for parsing a large set of input records in parts. You may want to modify the procedure to include additional processing of the records, or you may want to change the XPath expressions to suit your business process. If processing a large number of records, do the following:

1. Select and drop the **Parse Data** activity on the process editor.

2. On the **General** tab, specify the fields and select the **Manually Specify Start Record** checkbox.

3. Select the **Parse Data** activity and click the group icon on the tool bar to create a group containing the **Parse Data** activity.

4. Specify **Repeat Until True Loop** as the Group action, and specify an index name (for example, "i").

   The loop must exit when the EOF output item for the **Parse Data** activity is set to true. For example, the condition for the loop can be set to the following:

   ```
   string($ParseData/Output/done) = string(true())
   ```

5. Set the **noOfRecords** input item for the **Parse Data** activity to the number of records you want to process for each execution of the loop.

   If you do not select the **Manually Specify Start Record** checkbox on the **General** tab of the **Parse Data** activity, the loop processes the specified **noOfRecords** with each iteration, until there are no more input records to parse.

   You can optionally select the **Manually Specify Start Record** checkbox to specify the **startRecord** on the **Input** tab. If you do this, you must create an XPath expression to properly specify the starting record to read with each iteration of the loop. For example, the count of records in the input starts at zero (0), so the **startRecord** input item could be set to the current value of the loop index minus one. For example, $i - 1.

Render Data

The **Render Data** activity takes an input of a data schema and renders it as a text string. The schema processed is based on a specified **Data Format** shared resource.

You can use this activity in various scenarios, for example, retrieving a result set from a database table. You may want to:

- Format this result set as a formatted text string (with line breaks between each row in the result set), and then write that text string out to a file.

- Use the **Render Data** activity to render the data schema as a formatted text string.

- Use the **Write File** activity to write the string to a file.

General

The **General** tab has the following fields.

Field	Module Property	Description
Name	No	The name to be displayed as the label for the activity in the process.
Data Format	No	The **Data Format** shared resource to use when rendering the text output.

Description

Provide a short description for the Render Data activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
Rows	complex	The element containing the list of items to render.
root	complex	The complex element containing data schema (specified by the Data Format shared resource) to render as a text string. This is a repeating element which renders more than one output record.

Output

The following is the output of the activity.

Output Item	Datatype	Description
Text	string	The output text string as a result of rendering the specified data schema. Line breaks separate records of the data schema.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective actions to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When.
BadDataFormatException	The output format is not valid.

Rendezvous Palette

Using the Rendezvous palette you can send and receive TIBCO Rendezvous® messages.

> **(!) Important:** To run Rendezvous® applications on UNIX, set the LD_LIBRARY_PATH variable and RV_HOME specifying the RV lib path.

> **(!) Important:** To run applications in TIBCO Business Studio for BusinessWorks you must install Rendezvous® to use the Rendezvous palette features. Refer to the product readme, `readme.txt`, for the supported versions.

Datatype Conversion

When sending or receiving TIBCO Rendezvous® messages, TIBCO BusinessWorks Container Edition converts the fields of the message to the correct datatypes. Data in the incoming Rendezvous® messages is converted to TIBCO BusinessWorks Container Edition datatype (and represented in the activity's output). Data in an outgoing message is converted to Rendezvous® datatypes.

The following tables describe the corresponding datatypes between TIBCO BusinessWorks Container Edition and Rendezvous, depending on whether TIBCO BusinessWorks Container Edition is sending or receiving the message.

Rendezvous to TIBCO BusinessWorks Container Edition Datatype Conversion

Rendezvous Datatype	XSD Datatype
TIBRVMSG_BOOL	boolean
TIBRVMSG_I8	byte

Rendezvous Datatype	XSD Datatype
TIBRVMSG_I16	short
TIBRVMSG_I32	int
TIBRVMSG_I64	long
TIBRVMSG_U8	unsignedByte
TIBRVMSG_U16	unsignedShort
TIBRVMSG_U32	unsignedInt
TIBRVMSG_U64	unsignedLong
TIBRVMSG_F32	float
TIBRVMSG_F64	double
TIBRVMSG_OPAQUE	hexBinary or base64Binary, depending on what is defined in the schema.
TIBRVMSG_DATETIME	date
TIBRVMSG_STRING	string

TIBCO BusinessWorks Container Edition to Rendezvous Datatype Conversion

XSD Datatype	Rendezvous Datatype
boolean	TIBRVMSG_BOOL
byte	TIBRVMSG_I8
short	TIBRVMSG_I16

XSD Datatype	Rendezvous Datatype
int	TIBRVMSG_I32
long	TIBRVMSG_I64
integer	TIBRVMSG_I64
unsignedByte	TIBRVMSG_U8
unsignedShort	TIBRVMSG_U16
unsignedInt	TIBRVMSG_U32
unsignedLong	TIBRVMSG_U64
float	TIBRVMSG_F32
double	TIBRVMSG_F64
hexBinary or base64Binary, depending on what is defined in the schema.	TIBRVMSG_OPAQUE
date	TIBRVMSG_DATETIME
dateTime	TIBRVMSG_DATETIME
time	TIBRVMSG_STRING
string	TIBRVMSG_STRING

Field Names in Messages

A TIBCO Rendezvous® message can contain field names that are not compliant with XML naming rules. For example, XML element names can neither begin with a number, nor can they contain special characters, such as # or ^. When activities in the Rendezvous palette send or receive messages, element names in the input schemas or output schemas for the activity must comply with XML naming rules. Any element in referenced schemas in the

input schemas or output schemas is automatically altered to comply with XML naming rules. From the **XML-Compliant Field Names** field, you can handle the incoming or outgoing messages when the field names are not valid XML names.

When the **XML-Compliant Field Names** field is clear, any field name in the incoming or outgoing messages not compliant with XML naming rules, is altered to comply with the rules. When this field is selected, field names in the message are left unaltered.

When publishing a message with **XML-Compliant Field Names** field not selected, the field names in the message sent over the transport to the receiving application are altered to comply with XML naming rules. By selecting this field, you can send the message with the original field names intact; no alterations are made to the field names. This is useful in the case where you reference a schema in the **Input Editor** tab that contains invalid XML names. The receiving application receives the message with the original field names, if the **XML-Compliant Field Names** field is not selected.

When receiving a message, not selecting the **XML-Compliant Field Names** field ensures that the field names in the incoming message are altered to comply with XML naming rules. If the message contains fields that do not comply with XML naming rules, the output schema of the subscriber activity must specify the correct altered name to match the altered name of the field in the incoming message. Keeping the field clear leaves the field names of the message intact.

> **Note:** Not selecting the **XML-Compliant Field Names** field involves extra processing of incoming and outgoing messages. If you are certain that messages contain only field names that comply with XML naming rules, you can clear this field to improve performance.

Globalization Support

TIBCO Rendezvous® activities use `bw.palette.messageEncoding` property as the wire format encoding.

If this property is not set, the default wire format encoding value that supports all the character set, is UTF-8. To change the wire format encoding other than UTF-8, set the `bw.palette.messageEncoding` property to the required encoding. In design time, you can specify the property in VM arguments of Debug or Run Configurations like –`Dbw.palette.messageEncoding=ISO8859-1` in the Debug Configurations dialog. See the following screenshot for reference.
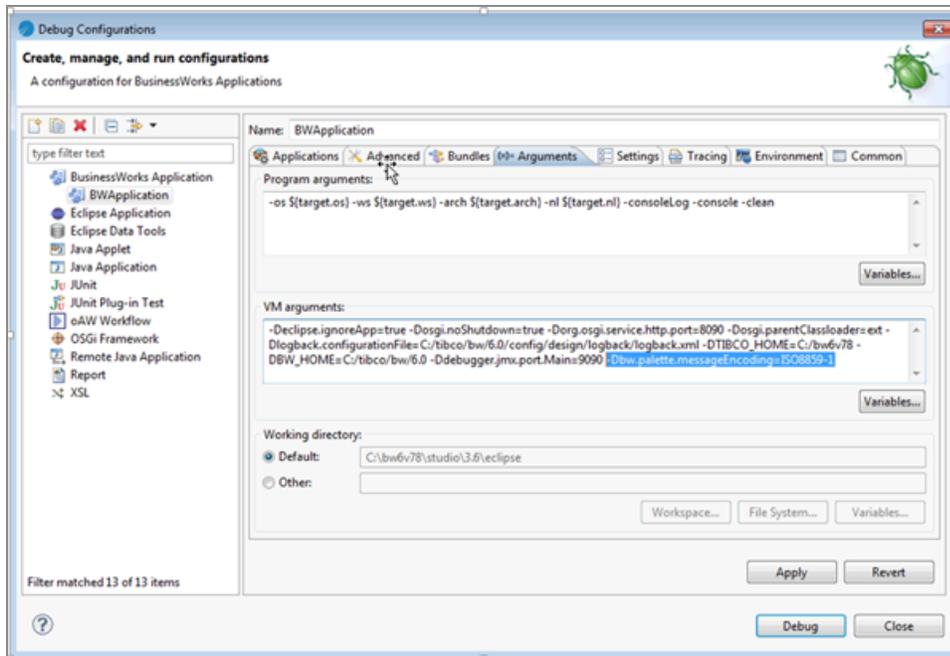
Note: ISO8859-1 supports English and other western European languages that belong to ISO Latin-1 character set. UTF-8 can send and receive characters that are not in the ASCII and Latin-1 character set.

In run time, you can specify the property in the `config.ini` file of the AppNode.

Rendezvous Publisher

Rendezvous Publisher is a synchronous activity that publishes a TIBCO Rendezvous® message on the specific subject with the specified message content.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Subject	Yes	The subject on which to publish the Rendezvous® message. Specify a subject on the **Input** tab to override this value.
Transport	Literal Value/Module Property	The Rendezvous transport parameters. These are specified as **Rendezvous Transport** shared configuration resource.
Pre Register Listener	Yes	When using TIBCO Rendezvous Certified Messaging (RVCM), this field specifies the cm names of the RVCM subscribers. The publisher preregisters with any expected listeners. If multiple cm names are specified, separate each cm name with a comma. For more information about RVCM, see the *TIBCO Rendezvous®* documentation.
XML Format	None	If selected, the body of the Rendezvous® message is placed into a single field named XML. This field is a byte array containing the body of the message. The datatype of the field is TIBRVMSG_XML.
XML Compliant Field Names	None	Specifies whether the field names of the outgoing message should be altered so that they comply with XML naming rules. Clear this checkbox to alter only the field names that do not comply with XML naming rules. When selected, the field names are left unaltered.

Description

Add a short description for the activity.

Input Editor

Define a schema element for the body of the Rendezvous message.

Input

The following is the input of this activity.

Field	Data Source	Description
subject	string	The subject of the Rendezvous message.
replySubject	string	The subject to send replies for this Rendezvous message. This is useful if you want to use the **Wait for Rendezvous Message** activity later in the process to receive replies to the message.
preRegisterListner	string	When using TIBCO Rendezvous Certified Messaging (RVCM), this field specifies the cm names of the RVCM subscribers. This allows the publisher to pre-register with any expected listeners. If multiple cm names are specified, separate each cm name with a comma. Any listeners preregistered on the **Transport** tab are also preregistered with the listeners specified in this element.
body	complex	The body of the Rendezvous message. This field displays only when any schema is provided in the **Input Editor** tab.

Fault

The **Fault** tab lists the possible exceptions generated by this activity.

Fault	Generated When..
TransportException	The transport cannot be created.

Fault	Generated When..
SerializationException	The message cannot be serialized.
RVPluginException	An error occurred when sending the message.

Rendezvous Transport

Rendezvous Transport resource describes a TIBCO Rendezvous ® transport. This resource is used when specifying activities from the **Rendezvous** palette. For more information about specifying these fields, see the TIBCO Rendezvous documentation.

Configuration

The **Configuration** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	No	Resource label.
Description	No	Short description of the shared resource.
Daemon	Yes	If the Rendezvous daemon is running on the same machine as the process engine: Daemon is not specified. If the Rendezvous daemon is running on a different machine, then the process engine: Host name followed by the socket number. For example: `ssl:acct:5785`

Field	Literal Value/Process Property/Module Property	Description
Network	Yes	Host name, IP address, network name, or interface name. For example: `;224.34.103.4`
Service	Yes	TIBCO Rendezvous service name or port number: For example: *<service name>* *<port number>*
SSL	No	Use a Secure Sockets Layer (SSL) when communicating with the TIBCO Rendezvous daemon. Select this field to enable the Configure SSL button. For more information, see Configure SSL Button.

Configure SSL Button

The Configure SSL button allows you to configure the SSL parameters for communicating with the TIBCO Rendezvous daemon. For more information about how SSL is configured for TIBCO Rendezvous daemons and clients, see the TIBCO Rendezvous documentation.

The SSL Configuration for TIBCO RV dialog tab has the following fields:

Field	Description
Daemon Certificate	File containing one or more certificates from trusted daemon certificate authorities. Select this field to connect to a trusted (non-rogue) daemon and retrieve a daemon certificate. Once a certificate is retrieved, select a project folder in the TIBCO Rendezvous

Field	Description
	administration interface and choose `Tools>Trusted Certificates>Import Into PEM Format` to import the certificate.
	Alternatively, instead of importing or embedding them, configure certificates at deployment time by declaring a global variable and setting its absolute path to a certificate in PEM format.
	To learn more about retrieving, importing, and configuring daemon certificates using the administration interface, see the TIBCO Rendezvous documentation.
Identity	Identity resource used to authenticate the TIBCO Rendezvous daemon.
	Click **Browse** to select an identity resource from a list of resources whose Type field is set to Identity File or Username/Password.

Advanced

The **Advanced** tab has the following fields:

Field	Global Variable?	Description
RV Type	No	Type of Rendezvous connection (reliable (standard RV transport), certified (RVCM), or Distributed Queue (RVCMQ)). **Note:** Use **Confirm** activity with RV Certified messaging. The fields on the **Advanced** tab correspond to the value selected for this field.
Certified Transport		
CM Name	Yes	Name of the delivery-tracking session, in the same format as the TIBCO Rendezvous subject names.
Ledger File	Yes	Name and location of the persistent ledger file that tracks certified messages. If not specified, the certified message ledger is kept in process memory only.

Field	Global Variable?	Description
Sync Ledger File	Yes	Specifies whether to keep the ledger file synchronous with the current messages.
Relay Agent	Yes	Name of the relay agent to use. Relay agents are useful when clients are disconnected from the network from time to time. The relay agents store inbound certified messages and labeled messages (and other messages related to certified delivery features) on behalf of their disconnected client programs. When a client is connected, the relay agent receives inbound messages immediately.
Require Old Message	Yes	Select this checkbox if you want to require the retention of messages for which delivery has not been confirmed. These messages are been resent.
Message Timeout (sec)	Yes	Time limit for certified message delivery.
Distributed Queue Transport		
CMQ Name	Yes	Name of the distributed queue, in the same format as TIBCO Rendezvous subject names.
Worker Weight	Yes	Weight of the worker (this pertains to the worker processing queue requests, not to BusinessWorks process engines). Relative worker weights assist the scheduler in assigning tasks. When the scheduler receives a task, it assigns the task to the available worker with the greatest worker weight.
Worker Tasks	Yes	Sets the task capacity for the worker (this pertains to the worker processing queue requests, not to BusinessWorks process engines). Task capacity is the maximum number of tasks that a worker can accept. When the number of accepted tasks reaches this maximum, the worker cannot accept additional tasks until it completes one or more of them.

Field	Global Variable?	Description
Worker Complete Time	Yes	The amount of time the scheduler waits for a worker process to complete. If the worker process does not complete the process in the specified time, the scheduler reassigns the message to another worker.
Scheduler Weight	Yes	Weight represents the ability of this member to fulfill the role of scheduler, relative to other members with the same name. Cooperating distributed queue transports use relative scheduler weight values to elect one transport as the scheduler. Members with higher scheduler weight take precedence. Acceptable values range from 1 to 65535.
Scheduler Heartbeat (sec)	Yes	Interval in which scheduler sends heartbeat messages. All members with the same name must specify the same value for this parameter. The value must be strictly positive.
Scheduler Activation (sec)	Yes	Maximum amount of time that the heartbeat signal from the scheduler is silent before the member with the greatest scheduler weight becomes the new scheduler. All members with the same name must specify the same value for this parameter. The value must be positive.

Rendezvous Reply To Request

Rendezvous Reply to Request is a synchronous activity that is used to send a reply to a received TIBCO Rendezvous® message.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Reply to	None	This field sends reply to the Rendezvous® activity or process starter that received the request. This is a selection list of the following available activities that can receive Rendezvous messages. • RendezvousSubscriber • WaitForRendezvousMessage
Reply Subject	Yes	The reply subject of the received Rendezvous message. You can override this value by specifying a subject on the **Input** tab.
XML Compliant Field Names	None	Select this checkbox to specify whether the field names of the outgoing message should be altered so that they comply with the XML naming rules. Clear this checkbox to alter only the field names that do not comply with XML naming rules. When selected, the field names are left unaltered.

Description

Provide a short description for the activity.

Input Editor

Use the **Input Editor** tab to define a custom schema for the body of the Rendezvous message.

Conversations

You can initiate the conversation here. Click the **Add New Conversation** button to initiate multiple conversations.
For more information about conversations, see the *TIBCO BusinessWorks™ Container Edition Application Development* guide.

Input

The following is the input for the activity.

Input Item	Datatype	Description
replySubject	string	The reply subject of the Rendezvous® message.

Fault

The **Fault** tab lists the possible exceptions generated by this activity.

Fault	Generated When..
RVPluginException	An error occurred when sending the reply message.

Rendezvous Subscriber

Rendezvous Subscriber is a process starter activity that creates a process when a TIBCO Rendezvous® message on the specified subject is received.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Subject	Yes	The subject on which to listen for the Rendezvous® message.
Transport	Literal Value/Module Property	The Rendezvous transport parameters. These are specified as a **Rendezvous Transport** shared resource.
XML Format	None	If you select this check box, the body of the incoming Rendezvous message is a single field named "xml". This field is a byte array containing the body of the message. TIBCO BusinessWorks Container Edition automatically deserializes the xml field and represents the message as a schema tree in this activity's output. **Note:** When selected, the **Need Input Filteration** check box gets disabled.
Need Output Filtration	None	Specifies that only fields that match the name and datatype of the schema elements specified on the **Output Editor** tab should be processed. Any elements that do not match the specified schema are ignored and not included in this activity's output. This is useful if you need only a subset of the fields of an incoming message. Any unnecessary fields are not included in the activity output. This saves memory, and consequently improves the performance.

Field	Literal Value/Process Property/Module Property	Description
		Note: This option is valid, only when the **XML Format** check box is not selected. Do not select this check box when the **XML Format** checkbox is also selected.
XML Compliant Field Names	None	Specifies whether the field names of the incoming message should be altered so that they comply with XML and ActiveEnterprise naming rules. Only field names that do not comply with XML and ActiveEnterprise naming rules are altered when this check box is not selected. When selected, field names are left unaltered.
Raw-RV-Object Mode	None	Selecting this checkbox disables **XML Format**, **Needs Output Filtration**, and **XML-Compliant Field Names** check boxes.
		Select this check box to specify that the body of the RV message is the output as a Java object reference. A subsequent **Java Invoke** activity in the process can accept the reference as an input parameter and then construct the RV message from the object. To use the Java object, the **Java Invoke** activity must cast the object reference as a byte array. For example:
		`TibrvMsg tibrvMsg = new`
		`TibrvMsg ((byte[])in_var_1);`
		Where `in_var_1` is the input parameter for the Java Invoke activity that is mapped to the object reference of the object containing the Rendezvous message.

Description

Provide a short description for the activity.

Advanced

The **Advanced** tab has the following fields.

Field	Description
Sequence Key	This field can contain an XPath expression that specifies which processes should run in a sequence. Process instances with sequence keys that evaluate to the same value, are run sequentially in the sequence the process instance was created.
Custom Job Id	This field can contain an XPath expression that specifies a custom job ID for the process instance.

Output Editor

Use the **Output Editor** tab to define a custom schema for the body of the Rendezvous® message.

Conversations

You can initiate the conversation here. Click the **Add New Conversation** button to initiate multiple conversations.
 For more information about conversations, see *Application Development*.

Output

The following is the output of the activity.

Output Item	Datatype	Description
sendSubject	string	The subject of the received message.
replySubject	string	The reply subject of the Rendezvous® message.

Rendezvous Request Reply

Rendezvous Request Reply is an asynchronous activity that publishes a TIBCO Rendezvous® message on the specified subject with the specified message content. A reply to the message on the specified reply subject and with the specified reply message content is received by this activity. This activity waits for a reply on the reply subject and outputs the content of the reply. This activity uses the INBOX mechanism to ensure that reply messages are received only by the process that sent the request.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Subject	Yes	The subject on which to publish the Rendezvous® message. You can override this value by specifying a subject on the **Input** tab.
Transport	Yes	The Rendezvous transport parameters. These are specified as a Rendezvous Transport shared configuration resource.
Request Timeout (msec)	Yes	The length of time to wait (in milliseconds) for a reply to the Rendezvous message. If no reply is received in the specified time limit, it returns an error.
Pre-register Listener	Yes	When using TIBCO Rendezvous Certified Messaging (RVCM), this field specifies the cm names of the RVCM subscribers. The sender can preregister with any expected listeners. If multiple cm names are specified, separate each cm name with a comma.

Field	Literal Value/Process Property/Module Property	Description
		For more information about RVCM, see the *TIBCO Rendezvous*® documentation .
XML Format	None	If this check box selected, the body of the Rendezvous® request and reply messages is sent as a single field named "xml". This field is a byte array containing the body of the message.
Need Output Filtration	None	Specifies that only fields in the reply message that match the name and datatype of the schema elements specified on the **Output Editor** tab should be processed. Any elements that do not match the specified schema are ignored and not included in this activity's output. This is useful if you need only a subset of the fields of a reply message. Any unnecessary fields are not included in the activity output. This saves memory, and consequently improves performance. **Note:** This option is only valid when the **XML Format** check box is not selected.
XML Compliant Field Names for Input	None	Specifies whether the field names of the input message should be altered so that they comply with XML naming rules. Only field names that do not comply with XML naming rules are altered when this check box is not selected. Select this check box if you want the field names to be unaltered.
XML Compliant Field Names	None	Specifies whether the field names of the outgoing message should be altered so that they comply with XML naming rules. Only field names that do not comply

Field	Literal Value/Process Property/Module Property	Description
for Output		with XML naming rules are altered when this check box is not selected.
		Select this check box if you want the field names to be unaltered.

Description

Provide a short description for the activity in this field.

Input Editor

In the **Input Editor** tab you can define custom schemas for the body of the outgoing Rendezvous request and body of the incoming Rendezvous response.

Output Editor

In the **Output Editor** tab you can define custom schemas for the body of the outgoing Rendezvous request and body of the incoming Rendezvous response.

Input

The following is the input for this activity.

Input Item	Datatype	Description
subject	string	The subject of the Rendezvous message.
replySubject	string	The reply subject for this activity.
timeout	number	The amount of time to wait (in milliseconds) for a reply to the Rendezvous message. If no reply is received in the given time limit, an error is returned.

Output

The following is the output of this activity.

Output Item	Datatype	Description
replySubject	string	A reply message can optionally have a reply subject, presumably so that the two applications can communicate with further messages. This element contains the reply subject set on the reply message, if one is set. If no reply subject is set on the reply message, this element is not populated.

Fault

The **Fault** tab lists the possible exceptions generated by this activity.

Fault	Generated When..
TransportException	The transport cannot be created.
SerializationException	The message cannot be serialized.
RVPluginException	An error occurred when sending the message.
ActivityTimedOutException	A timeout has been reached.

REST and JSON

The activities of REST and JSON Palette can be used to invoke RESTful web services and expose processes as RESTful web services. The activities in this group can also be used to convert data between JSON format and XML format.

BW JSON Utils

BW JSON utils provide an easier way to create XML schema (XSD files) from JSON payload or SQL statements.

Create XSD Schema from JSON payload

To create a schema from an available JSON payload:

1. From the **Project Explorer** pane, right-click on the **Schemas** folder and click **New > XML Schema File from JSON Payload**.

2. In the JSON to Schema Wizard, enter a schema file name in the **Schema File Name** field.

3. Enter the JSON payload, to be used during the conversion, in the **JSON Sample** field, and click **Next**.

   > **ⓘ** **Note:** A validation error displays in TIBCO Business Studio for BusinessWorks if any values from the JSON payload are not valid.

If the conversion is successful, the message JSON payload parsed successfully...Created schema <Name>.xsd displays in the JSON to Schema Wizard window, and you can close the wizard.

Create Schema from an SQL String

To create an SQL schema:

1. In the Process package, click **Resources > JDBC Connection Resource** and verify whether the JDBC connection is able to connect to the database successfully.

2. Right-click the fully configured **JDBC Connection Resource > BW JSON Utils > Create Schema from SQL**.

3. In the SQL Query Builder, provide the query statement to fetch the required information and click **OK**.

4. A "Schema has been created successfully under the Schemas folder with <Name>.xsd." message is displayed.

Invoke REST API

Invoke REST API asynchronous activity invokes RESTful web services and receives responses from the service provider.

> **Note:** For the **Invoke REST API** activity to work, a **HTTP Client** shared resource is required. In the **HTTP Client** shared resource, make sure **Apache HttpComponents (Supported by HTTP and REST)** is selected as the **Implementation Library** in the **HTTP Client** section.

> **Note:** To invoke the **DELETE** method of a REST service implemented in Opaque mode, it is not possible to send the request body as an asciiContent for the **Invoke REST API** activity. To send the request body, we have to use the **Send HTTP Request** activity with the **Jetty HTTP Client** implementation library in the **HTTP Client** shared resource.

> **Note:** MIME or multipart data format attachments are not supported as response or request in **Invoke REST API** activity.

General

The **General** tab has the following fields.

Field	Module Property	Description
Name	No	The name to be displayed as the label for the activity in the process.
HTTP Client	Yes	The HTTP client resource.
Resource Path	No	The resource of the REST service. **Note:** • Spaces and special characters are not supported in the path and query parameter name. • To use special characters in a query parameter name, percent-encode the question mark (?) as `%3F`. This field is mandatory.
HTTP Method	No	The REST methods used for the requests. Select from one of the following available methods: • POST • GET • PUT • DELETE • PATCH
Request Type	No	The value of the Content-Type header set while invoking the REST operation. • JSON: the service provider returns the data in the JSON format. • XML: the service provider returns the data in the XML format. • Custom: to override the **Content-Type** value in the **Input** tab, select **CUSTOM** and provide the value in the **Input** tab.

Field	Module Property	Description
Response Type	No	The value of the **Accept** header set while invoking the REST Operation. • JSON: the service provider returns the data in the JSON format. • XML: the service provider returns the data in the XML format. • Custom: to override the **Accept header** value in the **Input** tab, select **CUSTOM** and provide the value in the **Input** tab.

Description

Provide a short description for the Invoke Rest API activity.

Advanced

The **Advanced** tab has the following fields.

Field	Description
Output HTTP Response Headers	Selecting this checkbox displays the HTTP headers received in the response.
Request Entity Processing	This field has two values: • BUFFERED: the request entity is buffered in memory to determine the content length that is sent as a Content-Length header in the request. • CHUNKED: the entity is sent as chunked encoded (no Content-Length is specified, the entity is streamed). The Transfer-Encoding header is set to Chunked. The default value is Chunked.

Field	Description
Activity Timeout (seconds)	Time out to invoke a RESTful web service and receive a response from the service provider after an Invoke REST API call is sent. The Default Timeout is three (3) minutes.

Input

The following is the input for the activity.

Input Item	Datatype	Description
ResourcePath	String	This value overrides the value provided in the **Resource Path** field on the **General** tab.
Message Body	String	The body of the HTTP request message.
asciiContent	String	The ASCII content (ASCII content of the request like POST or PUT to the server) of the request to the HTTP server.
HttpHeaders	String	The header of the HTTP request. The header structure is defined by the HTTP protocol. For more information about the fields and content of the header of an HTTP request, see the W3C HTTP specification at w3.org. • **Accept**: specifies the media types that are acceptable for response messages for the incoming request. For example, `text/*`, `text/html`. For more information about media types, see the W3C HTTP specification at w3.org **Note:** To get status messages for exceptions, set the **Accept** field to `*/*` or `text/html`.

Input Item	Datatype	Description
		• **Accept-Charset**: specifies the character sets that are acceptable for response messages for the incoming request. For example, iso-8859-5, unicode-1-1. For more information about character sets, see the W3C HTTP specification at w3.org. • **Accept-Encoding**: specifies the content-coding values that are acceptable for response messages. For example, compress, gzip. • **Content-Type**: the media type of the entity body sent to the receiver. For more information about media types, see the W3C HTTP specification at w3.org. • **Content-Length**: indicates the size of the entity body sent to the receiver. • **Connection**: the requestor can specify options desired for this connection. For example, the option close specifies that the requestor would like the connection to be closed when the request is complete. • **Cookie**: For more information about this field, see the W3C HTTP specification at w3.org. **Note:** For correct parsing and processing of cookies, input must be in the format "key=value". • **Pragma**: is used to include implementation-specific directives that might apply to the receiver. For more information about this field, see the W3C HTTP specification at w3.org.

Input Item	Datatype	Description
DynamicHeaders	Complex	The dynamic header is an additional header parameter to add runtime headers to the outgoing HTTP messages. The DynamicHeaders consists of the following information: • Name: the name of the header • Value: the value of the header
DynamicConfigurations	Complex	Specifies the values to override. **activityTimeout**: The **Invoke REST API** activity timeout in seconds. At runtime, it takes precedence over the Activity Timeout value in the **Advanced** tab. **Note:** To see this element in existing applications, we have to modify the activity to enable the **Save** button.

Output

The **Output** tab has the following fields.

| Output Item	Description
statusCode	The HTTP Response Code.
reasonPhrase	ReasonPhrase is intended for giving a short textual description of statusCode.
MessageBody	The body of the response message. This is asciiContent.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO ActiveMatrix BusinessWorks™ Error Codes*.

| Fault	Generated When..
HttpCommunicationException	An HTTP exception occurred when trying to run the specified method, or when trying to read the response.
HttpClientException	The HTTP server replied with a message that has the 4XX status code.
HttpServerException	The HTTP server replied with a message that has the 5XX status code.
Resilience4jCircuitBreakerOpenException	This exception is generated when the circuit breaker is open.

Support for Path and Query Parameters

Invoke REST API activity supports both Path and Query parameters. For **Path Parameters**, enclose each path parameter in **{ }** brackets. For example, if the resource path is set as `/books/{isbn}`, a PathParameters element is added to the activity input.
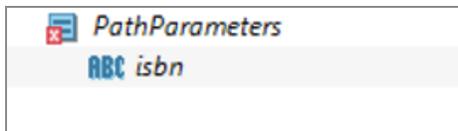
Multiple path parameters can be defined in a similar way. The activity also supports static URL content mixed with path parameters, for example, `/books/{isbn}/events/{author}`.

Query parameters are defined using the pattern `/staticUrl?Param1&Param2`.

A QueryParameters element is added to the **Input** tab of the activity with each of the parameter listed. For example, if the resource path is set as `/books?isbn&author`, a **QueryParamters** element is added to the activity input.
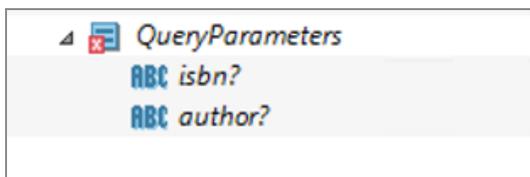
Policy

Activities that support policies display the **Policy** tab. To associate a new or existing policy with the **Invoke REST API** activity, click the **Add Policy to Activity** icon. To edit policy details, click the **Go to selected Policy** icon. The **Policy** tab has the following fields.

Field	Description
Name	The name of the policy.
Type	The type of policy associated with the activity. The **Invoke REST API** activity can support the Basic Credential Mapping policy.
Description	A description of the policy.

Restrictions on XML Schema

This topic lists the restrictions on XML Schema.

General Restrictions

- No wildcards or attribute wildcards. For example, any element and any attribute is not supported.

- Complex types might not contain both an attribute and a child element with the same local name.

- Complex types might not contain mixed content.

- Attributes that are not part of the default (empty) namespace, cannot be used for Complex Elements.

- The 'choice' and 'sequence' compositors might not have maxOccurs > 1 (same as the restriction on 'all' in the schema specification).

- Substitution groups are not supported.

- Element of simple type with an attribute is not supported.

- The `elementFormDefault` can only be qualified for schemas used by REST binding and JSON activities.

- Schemas should not contain cyclic dependencies within same schema, or on the other schemas.

- Schemas should not have a type that has two child members with the same local name, but different namespaces.

- For float and double values, XML schema always shows exponential values of type `1.0E0`

Parse JSON

Parse JSON activity takes the JSON data, processes it, and converts it into XML data.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to display as the label for the activity in the process.
Schema Type	None	Specifies the mode for the output data. The following schema types are supported: • Generic: converts a JSON string to an XML string • XSD: converts a JSON string to an XML document defined using a schema specified in the **Output Editor**. The default value in this field is the XSD schema type.

Field	Literal Value/Process Property/Module Property	Description
		Note: Ensure the schema resource that you select does not contain cyclic dependencies on other schemas, or a type that has two child members with the same local name, but different namespaces.
Badgerfish	None	Specifies the following conversion rules: • Select this checkbox for converting JSON data to XML data with the mapping of XML namespace, XML attribute, CDATA, and so on. • Do not select this checkbox if you want normal conversion rules for mapping data from JSON to XML. XML attributes are converted into elements when the checkbox is not selected
Input JSON Style	None	Specifies the method to parse the data. • **None** : Select this option to exclude the root element in the JSON string. • **Json with Root** : Select this option to include the root element in the input JSON string. • **Anonymous Array** : Select this option to accept a JSON array without the parent element, where the root element has exactly one child of the type Array.
Use Null For Empty Values	None	Select the checkbox to use null in the place of empty values in the JSON. This option is applicable only for Objects and Arrays. This has no effect on primitive types. By default, the checkbox is not selected.

Field	Literal Value/Process Property/Module Property	Description
Ignore Additional JSON Fields	None	This checkbox is available when the XSD option is selected in the **Schema Type** field. Select the checkbox to ignore the additional fields received due to the changes made in external payload while processing the schema. This checkbox is not selected by default.

Description

Provide a short description for the Parse JSON activity.

Output Editor

From the **Output Editor** you can define or reference an XML schema for the activity output. You can use the BW JSON Utils tool to generate an XML schema file with a specified JSON file.

Input

The **Input** tab has the following fields.

Field	Description
jsonString	The input JSON data for translation.

Output

The output for the **Parse JSON** activity varies depending on the data schema you specified in the **Schema Type** field on the **General** tab.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
JSONParserException	An exception occurred when parsing the JSON data.

Render JSON

Render JSON activity takes XML data and renders it as a JSON string.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Schema Type	None	Specifies a schema type for the output data. The following are the supported schema types: • Generic: converts an XML string to a JSON string. • XSD: converts an XML string to a JSON document defined using a schema specified in the **Output Editor**. The default value in this field is the XSD schema type.

Field	Literal Value/Process Property/Module Property	Description
Badgerfish	None	Specifies the following conversion rules: • Select this checkbox for converting JSON data to XML data with the mapping of XML namespace, XML attribute, CDATA, and so on. • Do not select this checkbox if you want normal conversion rules for mapping data from JSON to XML. XML attributes are converted into elements when the checkbox is not selected
Output JSON Style	None	Specifies the method to render the data. • **None** : Select this option to exclude the root element in the JSON string. • **Json with Root** : Select this option to include the root element in the input JSON string. • **Anonymous Array** : Select this option to return a JSON array without the parent element, where the root element has exactly one child of the type Array.
Use Empty Values For Null	None	Select the checkbox to use empty values in place of null in JSON. This option is applicable only for Objects and Arrays. This has no effect on primitive types. By default, the checkbox is not selected.

> **Note:**
> - The "`com.tibco.plugin.restjson.xml2json.emptytonull`" property is newly added for the **Rendor JSON** activity. This property when set to true converts the "" (refers to empty string) and "{}" (refers to empty element or object) to the "null" value in the output generated by the **Rendor JSON** activity.
>
> - The "`com.tibco.bw.rest.renderjson.generic.parseDataTypes`" property is newly added for the **Rendor JSON** activity with a generic schema type. On setting this property to true, numeric values are rendered without quotes else the default behavior of enclosing all values within quotes is followed.

Description

Provide a short description for the Render JSON activity.

Input Editor

From the **Input Editor** tab, you can define or reference an XML schema for the activity input. You can also use the BW JSON Utils tool to generate an XML schema file with a specified JSON file.

Input

The input data for the **Render JSON** activity is in the XML format. The schema type of the XML data varies depending on the schema type you specified in the **Schema Type** field on the **General** tab.

Output

The **Output** tab contains the following field.

Field	Description
jsonString	The translated data in JSON string format.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
JSONRenderException	An exception occurred when rendering the data.

Transform JSON

Transform JSON activity takes input data in JSON format and transforms it into JSON format according to the Jolt specifications.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property	Description
Name	No	The name to be displayed as the label for the activity in the process.
Spec File	Yes	Specify the location of Jolt specification file. For more information, see Jolt Specification at Github.com. **Note:** The input specification file takes priority followed by the Spec File in the General tab. If none of the specification files are provided, then the specification file generated from mappings is used.
Input Json	Yes	The input json payload file location. The file provided

Field	Literal Value/Module Property	Description
Schema File		must be a valid json file with a `.json` file extension. The .json file must contain either a sample json payload or a valid json schema string. The first element of the payload must have all the elements that need to be mapped. If the input provided is a json schema file then output must also be json schema.
Output Json Schema File	Yes	The output json payload file location. The file provided must be a valid json file with a `.json` file extension. This `.json` file must contain the output json payload. **Note:** The first element of the payload must have all the elements that need to be mapped. This is valid only in case of the json payload file.
Build Spec File	No	Select the **Map Input Output Json** button to open the Jolt Json Mapper dialog. For more on the Jolt Json Mapper dialog, see Jolt Json Mapper

Description

Provide a short description for the Transform JSON activity.

Input

The **Input** tab has the following fields.

Field	Description
JsonInput	Mandatory. Input content in the JSON format.
SpecContent	Provide Jolt specification file content in the JSON format. You can either provide a path to the file in the **General** tab or provide file content in the **SpecContent** field.

Output

The **Output** tab has the following fields.

| Field	Description
JsonOutput	Output in the JSON format converted as per the Jolt specification used.

Fault

The **Fault** tab has the following fields.

Fault	Generated When.
JSONTransformException	There is invalid Jolt specification file or JSON input.

> 🛈 **Note:** If you are using an external file for JOLT specification in **SpecContent** field, make sure that the file path is accessible to the application container.

Jolt Json Mapper Dialog

The Jolt Json Mapper creates a Jolt Specification file as mappings added in the **Jolt Json Mapper** dialog.

The Json Transform Mapper dialog is displayed when the correct and valid input and output json files are added in the **Input Json Schema File** and **Output Json Schema File** fields and the **Map Input Output Json** button in the **General** tab.
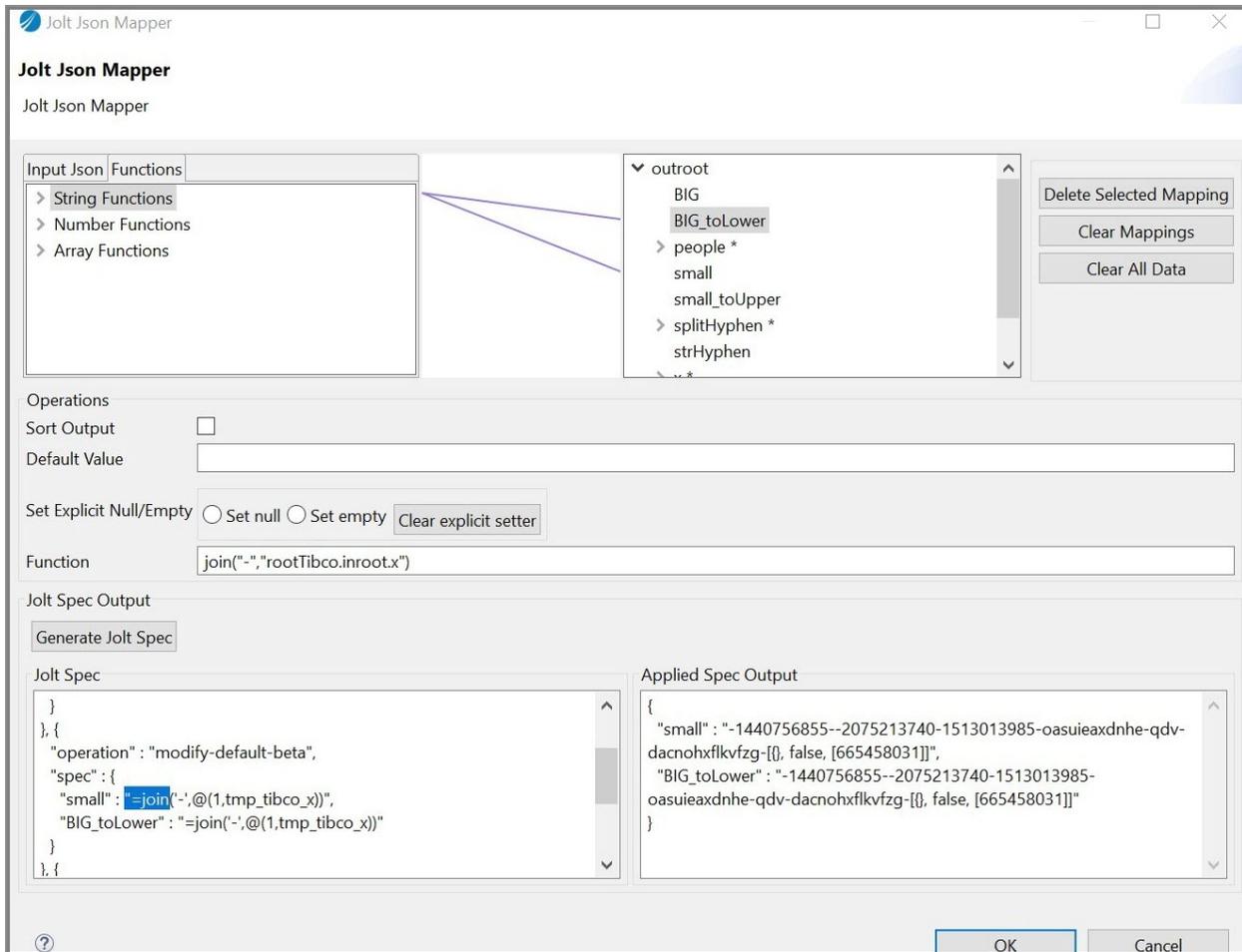
Input and Output Tree Panels

- The input and output tree can be expanded by clicking on the arrows beside the elements in the tree.

- Elements that have the same json type can only be mapped from the input to the output tree. For example, a json array cannot be mapped to a json object or a json string element and vice versa.

- The parent array should be mapped before mapping its child elements. If not mapped then on selecting the **Generate Jolt Spec** button an error occurs.

- The Functions tab applies the String, Number or Array to the output tree.

Sort Output

- To sort the json output activity select the **Sort Output** check box. If this check box is not selected, the output will not be in a sorted order.

> **ⓘ** **Note:** For sorting, only ascending order is supported and sorting is done alphabetically on the element names in the output json.

Default Value

- To add a default value for an output element, click on the element and then add the default value in the **Default Value** text box.

- If a mapping is done on an element for which a default value is provided, the mapping takes precedence. The default value is cleared when the input tree element is dragged and dropped on the output tree element.

Set Explicitly Null/Empty

- The **Set null** or **Set empty** radio buttons sets the values of any elements in the output json with a default value as Null or Empty.

> **ⓘ** **Note:** The **Set empty** button is only to String data type. For all other data types apply the **Set null** button. The **Set null** can also be used for String data type.

Function

- The **Function** text box displays the functions signature as added in the **Functions** tab.

> **ⓘ** **Note:** If we are using special characters like "+, ()", add a escape character "\" before each of the special characters.
> For example, concat("rootTibco.inroot.BIG",\)\+,"rootTibco.inroot.small",+)

- The String, Number, and Array Functions are supported.

- To apply a function to an output field, drag and drop a function from the left function tree to the required element in the output tree. After dropping the function, the output element displays the function signature in the Function text box in the Input/Output trees.

Jolt Spec Output

- After the elements are mapped correctly, click the **Generate Jolt Spec** button. This

generates the jolt specifications based on the mappings, functions applied the **Sort Output** check box and the if any default values provided for the output element.

- The generated jolt specification string is displayed in the **Jolt Spec** text box. The output json from applying the jolt specification generated to the input json is displayed in the **Applied Spec Output** text box.

Delete Selected Mapping

- The **Delete Select Mapping** button deletes a mapping from the input to output tree. It also deletes a mapping from the underlying data model. Select the mapping line to delete a particular mapping. Regenerate the jolt specification after deleting a mapping.

Clear Mappings

- The **Clear Mappings** buttons clears all the mappings from the input to the output tree.

Clear All Data

- The **Clear All Data** button resets the mapper and all the data mappings stored in it. This button clears all the mappings from the input to the output tree, clears the Sort Button (if selected), and removes any default values set on the output tree.

> **i** **Note:** It is recommended to use this button if new input and output json or json schema files are provided in the **General** tab.

Policy Resources

Policies are resources that you can define and configure in TIBCO Business Studio for BusinessWorks. You can define a policy and then reference it from multiple activities or binding in the same or different process. For example, you can define a **Basic Authentication** policy, and refer it either from an **HTTP Receiver** activity, or from a REST binding.

Basic Authentication Policy

Applying this policy ensures that users cannot access your application unless their credentials are verified.

When you create a **Basic Authentication** policy, the following policy resources are created:

- BasicAuthentication
- BasicAuthentication_AuthenticationProvider
- XML Schema related to Authentication

Basic Authentication

The **Basic Authentication** policy resource provides options that you can configure when you set up a basic authentication security policy.

General

The **General** section has the following fields.

Field	Description
Package	The name to be displayed as the label of the policy resource package.
Name	The name of the policy resource.
Description	A short description of the policy resource.

Service Provider Details

The service provider that you select authenticates username and password credentials. The **Service Provider Details** section has the following fields.

Authentication Tab

Field	Description
Select Authentication Type	Select a way to authenticate user credentials: • **LDAP Authentication** • **XML File Authentication**
LDAP Resource	Create a **LDAP Authentication** resource, or specify an existing **LDAP Authentication** resource. The Basic Authentication policy references the **LDAP Authentication** resource that you select.
XML File Resource	Create a **XML Authentication** resource, or specify an existing **XML Authentication** resource. The Basic Authentication policy references the **XML Authentication** resource that you select. For creating an XML file manually, refer to Create XML File for Credentials.

Authorization Tab

Field	Description
Authorization by Role	Select the checkbox to enable the following fields: • Radio buttons: **Permit** and **Deny** • Roles section. For more information, see XML Authentication.
Permit	Select the radio button to authorize the specific roles specified in the Roles table.
Deny	Select the radio button to deny the specific roles specified in the Roles table.

Field	Description
Roles	Add or remove roles to which authorization access is allowed or denied. For XML Authentication, groups are defined in an `XMLUsers.xml` file under group-mapping tag in the **XML Authentication** shared resource. For LDAP Authentication, groups are defined under the **Users and Groups** tab of the **LDAP Authentication** shared resource. Roles are case-sensitive. To sort roles in alphabetical order click the Roles header. To add roles, click the **Add** button. Add roles by clicking a last empty row. To remove roles, click **Delete** button. To move roles, click **Move** button. **Note:** When you select the **Permit** or **Deny** options, it is mandatory to add roles.

Basic Credential Mapping

Configure the **Basic Credential Mapping** policy resource to set up a basic credential-mapping security policy.

General

The **General** section has the following fields.

Field	Description
Package	The name to be displayed as the label of the policy resource package.
Name	The name of the policy resource.
Description	A short description of the policy resource.

Credential Mapping

You can choose to enforce either **Fixed** credential mapping or **Conditional** credential mapping.

Fixed

Select this option to ensure that a fixed set of credentials is applied to all users regardless of their user type or role. The **Fixed** section has the following fields:

Field	Description
Identity Provider	Select an **Identity Provider** to authenticate user credentials.

Conditional

Select **Conditional** credential mapping to enforce credential mapping for authenticated users and anonymous users. Credentials are mapped for authenticated users if the request comes from an authenticated service. Credentials are mapped for anonymous users if the request comes from an unauthenticated service.

The **Conditional** section has the following fields:

Field	Description
Role-based Identity Providers	Type roles for authenticated users and associate an **Identity Provider** with each role. You can reuse the same **Identity Provider** for multiple roles.
Authenticated Identity Provider	Select an **Identity Provider** shared resource for authenticated users.
Anonymous Identity Provider	To allow anonymous users access to your secure application, specify an **Identity Provider**. not To allow anonymous users access, do not specify an **Identity Provider**.

Field	Description
	Note: Application logic can also affect how credentials are mapped for anonymous users. For example, application logic might dictate that anonymous requests are redirected to specific entry points. If an anonymous request is directed to an enforced entry point, the request is rejected.

WSS Consumer

Configure the WSS Consumer policy to enforce confidentiality, integrity, and timestamping, and credential mapping.

General

The **General** section has the following fields.

Field	Description
Package	The name to be displayed as the label of the policy resource package.
Name	The name of the policy resource.
Description	A description of the policy resource.

Shared Resource for WSS Processing

The **Shared Resource for WSS Processing** section has the following fields.

Field	Description
WSS Verification	The **WSS Verification** shared resource that the WSS Consumer policy references.

Service Provider Details

The Service Provider Details section comprises of the **Confidentiality** tab, the **Integrity** tab, the **Timestamp** tab, and the **Credential Mapping** tab.

Confidentiality

To maintain confidentiality, the policy can be configured for an outbound request to be encrypted and an inbound response to be decrypted at its endpoint. The **Confidentiality** tab has the following fields:

Field	Description

Encrypt Request	Specify the following fields:
	
	1. **Trust Provider**: Select a **Trust Provider** shared resource.
	
	2. **Key Alias**: Specify a **Key Alias**.
	
	3. **Algorithm Suite**: Specifies the algorithm suite required for performing cryptographic operations with symmetric or asymmetric key based security tokens. An algorithm suite specifies actual algorithms and allowed key lengths. The default selection is Basic128. You can select a different algorithm suite from the drop-down menu.
	
	4. **Encrypt**: Specify to **Encrypt Parts** or to **Encrypt Elements** of the message.
	
	   a. **Encrypt Parts**: Select this option to encrypt the **Body**, **Header**, or both parts of the message.
	
	   b. **Encrypt Elements**: Select this option to encrypt elements in the request message. When specifying the **Element**, ensure you also specify the **Namespace** of the element, and **Prefix** of the element if it has one.

Decrypt Response	To Decrypt response, provide the **Subject Provider** or the **Subject Provider (with Trust Credential)** value in the **WSS Authentication** policy resource, and select the **Enable Decryption** checkbox on the **Basic Configuration** section of the **WSS Authentication** policy resource.

Integrity

To maintain integrity, the outbound request can be signed and the signature verified in the inbound response. The **Integrity** tab has the following fields:

Field	Description

Sign Request	Specify the following fields:
	
	1. **Subject Provider**: Select a **Subject Provider** shared resource.
	
	2. **Algorithm Suite**: Specifies the algorithm suite required for performing cryptographic operations with symmetric or asymmetric key based security tokens. An algorithm suite specifies actual algorithms and allowed key lengths. Default type is **Basic128**. You can select a different algorithm suite from the drop-down menu.
	
	3. **Digest Algorithm for Signature**: The algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input. Default type is **SHA-256**. You can select a different type from the drop-down menu.
	
	4. **Sign**: Specify to **Sign Parts** or to **Sign Elements**.
	
	    a. **Sign Parts**: Select this option to sign the **Body**, **Header**, or both parts of the message.
	
	    b. **Sign Elements**: Select this option to sign elements in the request message. When specifying the **Element**, ensure you also specify the **Namespace** of the element, and **Prefix** of the element if it has one.

Verify Signature on Response	Select the checkbox to enable the **Verify parts that are Signed** field. Select from the following options from the drop-down menu : 1. **Entire message** 2. **Message header** 3. **Message body**

Timestamp

Under the **Timestamp** tab, configure the following fields to insert a timestamp in an outbound request and verify a timestamp in the inbound response.

Field	Description
Set Timestamp on Request	Specify time-to-live in seconds.
Verify Timestamp on Response	No additional configuration required.

Credential Mapping

Under the **Credential Mapping** tab, select either **Username Token** credential mapping or **SAML Token** credential mapping to map credentials to the outbound request.

Field	Description
No Credentials	Select this option to ensure credential mapping is not enforced.
Username Token based Credential Mapping	Select **Fixed** or **Conditional**: • If you select **Fixed**, specify an **Identity Provider** resource in the **Identity Provider** field. • If you select **Conditional**, specify the types of users your application maps credentials for. You can choose to map credentials for authenticated users with roles, authenticated users, and anonymous users. For configuration details, see Basic Credential Mapping.
SAML Token based Credential Mapping	Configure the following fields: 1. **SAML Token Profile**: Select a token type. Specify either **SAML 1.1 Token 1.1** or **SAML 2.0 Token 1.1**. 2. **Sign SAML Assertion**: If you select this option, the following fields are enabled:

Field	Description

- **Subject Provider:** Specify a **Subject Provider** shared resource.

- **Digest Algorithm for Signature:** Select one of the following options from the drop-down menu:

  - SHA1

  - SHA256

  - SHA384

  - SHA512

- **Algorithm Suite:** Select one of the following options from the drop-down menu:

  - Basic128

  - TripleDes

  - Basic256Rsa15

  - Basic192Rsa15

  - Basic128Rsa15

  - TripleDesRsa15

  - Basic256Sha256

  - Basic192Sha256

  - Basic128Sha256

  - TripleDesSha256

  - Basic256Sha256Rsa15

  - Basic192Sha256Rsa15

  - Basic128Sha256Rsa15

  - TripleDesSha256Rsa15

3. **SAML Issuer Name**: Type a SAML issuer name.

4. **SAML Assertion Validity**: Select **SAML Assertion Validity (forever)** to ensure that the SAML assertion is valid indefinitely. Optionally, you can select **Specify Validity Period (sec)** to specify the number of seconds the SAML assertion is valid.

WSS Provider

The WSS Provider policy acts on the server side to ensure that the confidentiality, integrity, and timestamp of a request remains secure.

General

The **General** section has the following fields.

Field	Description
Package	The name to be displayed as the label of the policy resource package.
Name	The name of the policy resource.
Description	A short description of the policy resource.

Shared Resource for WSS Processing

The **Shared Resource for WSS Processing** section has the following fields.

Field	Description
WSS Verification	The **WSS Verification** shared resource that the WSS Provider policy references.

Service Provider Details

The Service Provider Details section comprises of the **Authentication** tab, **Confidentiality** tab, the **Integrity** tab, and the **Timestamp** tab.

Authentication

The **Authentication** tab has the following fields that you can enable to enforce authentication on a request message.

Field	Description
No Verification	Select this option to ensure that credentials are not authenticated through username token or SAML token.
Verify username token	Select this option to authenticate user credentials through username token. If you select this option, ensure you have configured the **User Authentication** tab on the **WSS Authentication** Shared Resource.
Verify SAML token	Select this option to authenticate user credentials through SAML token. Select one of the following confirmation methods: 1. **Bearer** 2. **Holder of Key** 3. **Sender Vouches** Select one of the following security token types: 1. **SAML 1.1 Token 1.1** 2. **SAML 2.0 Token 1.1** Specify Issuer Name

Confidentiality

An inbound request can be decrypted and an outbound response can be encrypted to maintain confidentiality. The **Confidentiality** tab has the following fields:

Field	Description
Decrypt Request	To Decrypt request, provide the **Subject Provider** or the **Subject Provider (with Trust Credential)** value in the **WSS Authentication** policy resource, and select the **Enable Decryption** checkbox on the **Basic Configuration** section of the **WSS Authentication** policy resource.
Encrypt Response	Specify the following fields: 1. **Trust Provider**: Select a **Trust Provider** shared resource.

Field	Description

2. **Key Alias**: Specify a **Key Alias**.

3. **Algorithm Suite**: Specifies the algorithm suite required for performing cryptographic operations with symmetric or asymmetric key based security tokens. An algorithm suite specifies actual algorithms and allowed key lengths. The default selection is **Basic128**. You can select one of the following algorithms suite from the drop-down menu:

   - Basic128
   - TripleDes
   - Basic256Rsa15
   - Basic192Rsa15
   - Basic128Rsa15
   - TripleDesRsa15
   - Basic256Sha256
   - Basic192Sha256
   - Basic128Sha256
   - TripleDesSha256
   - Basic256Sha256Rsa15
   - Basic192Sha256Rsa15
   - Basic128Sha256Rsa15
   - TripleDesSha256Rsa15

4. **Encryption Algorithm**: Select one of the following encryption algorithms from the drop-down menu:

   - AES_128
   - AES_192
   - AES_256
   - AES_128_GCM
   - AES_192_GCM

Field	Description

- AES_256_GCM

- TRIPLE_DES

5. **Encrypt**: Specify to **Encrypt Parts** or to **Encrypt Elements** of the message.

   a. **Encrypt Parts**: Select this option to encrypt the **Body**, **Header**, or both parts of the message.

   b. **Encrypt Elements**: Select this option to encrypt elements in the response message. When specifying the **Element**, ensure you also specify the **Namespace** of the element, and **Prefix** of the element if it has one.

Integrity

Maintain integrity by verifying the signature on an inbound request and signing an outbound response. The **Integrity** tab has the following fields:

Field	Description
Verify Signature on Request	Select from the following options from the drop-down menu **Verify parts that are signed** : 1. **Entire message** 2. **Message header** 3. **Message body**
Sign Response	Specify the following fields: 1. **Subject Provider**: Select a **Subject Provider** shared resource. 2. **Algorithm Suite**: Specifies the algorithm suite required for performing cryptographic operations with symmetric or asymmetric key based security tokens. An algorithm suite specifies actual algorithms and allowed key lengths. Default type is **Basic128**. Select a one of the following algorithm suites from the drop-down menu:

Field	Description
	• Basic128
	• TripleDes
	• Basic256Rsa15
	• Basic192Rsa15
	• Basic128Rsa15
	• TripleDesRsa15
	• Basic256Sha256
	• Basic192Sha256
	• Basic128Sha256
	• TripleDesSha256
	• Basic256Sha256Rsa15
	• Basic192Sha256Rsa15
	• Basic128Sha256Rsa15
	• TripleDesSha256Rsa15

3. **Digest Algorithm for Signature**: The algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input. Default type is **SHA-256**. Select one of the following options from the drop-down menu.

   - SHA1
   - SHA256
   - SHA384
   - SHA512

4. **Sign**: Specify to **Sign Parts** or to **Sign Elements** of the message.

   a. **Sign Parts**: Select this option to sign the **Body**, **Header**, or both parts of the message.

   b. **Sign Elements**: Select this option to sign elements in the response message. When specifying the **Element**, ensure you also specify the **Namespace** of the element, and **Prefix** of the element if it has one.

Timestamp

To track the time of the request, a timestamp is inserted in the request. The **Timestamp** tab has the following fields:

Field	Description
Verify Timestamp on Request	No additional configuration required.
Set Timestamp on Response	Specify the time-to-live in seconds.

Shared Resources

Shared resources are used to define a resource that contains configuration data that can be referenced from multiple processes.

You can define a shared resource and then reference it from multiple activities in the same or different process. For example, you can define a **JDBC Connection** resource and then use it in any of the JDBC activities in your process to connect to the database.

Shared resources (such as **JDBC Connection, HTTP Connection, JMS Connection**) are available at design time. At runtime, the referencing activities and event sources have full access to their instances and configuration. Shared resources can be grouped in packages, similar to the way process packages and Java packages are presented in the file system.

When defined in an application module, shared resources are not visible outside the application module. However, when defined in a shared module, they are visible outside the shared module.

For more information about basic concepts like application module, shared module, and shared resource, see General Concepts in the *TIBCO BusinessWorks Container Edition Concepts*.

Independent Start of a Component

A TIBCO BusinessWorks Container Edition application contains one or more components in it. During application startup, if the shared resource encounters any errors or issues, then the application goes in an impaired or startFailed state.

The Independent start of a component feature allows an application to start even if there is component failure (due to shared resource failure) so that the user can use other working components in the application without any issue. This feature mainly targets areas where shared resource creation fails, that results in component initialization failure, which also results in the application to be in an impaired or startFailed state.

When the `bw.independent.component.startup=true` property is enabled, it bypasses the shared resource creation failure error and the component initialization failure and starts the application so that you can use the other working components in the application.

Data Format

Data Format shared resource contains the specification for parsing or rendering a text string using the **Parse Data** and **Render Data** activities. This resource specifies the type of formatting for the text (delimited columns or fixed-width columns), the column separator for delimited columns, the line separator, the fill character, and field offsets for fixed-width columns.

Also specify the data schema to use for parsing or rendering the text. When parsing text, each column of an input line is transformed into the corresponding item in the specified data schema. The first column of the text line is turned into the first item in the data schema, the second column is transformed into the second item, and so on. Each line is treated as a record, and multiple lines result in a repeating data schema containing the lines of the input text string. The following figure illustrates how an input text string is parsed into a specified data schema.

Parsing a Text String into a Data Schema
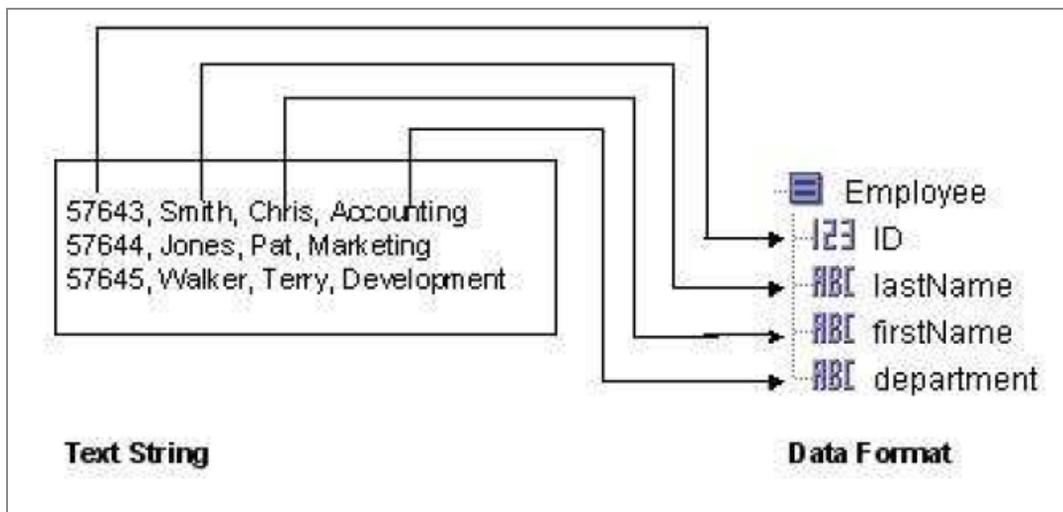
When rendering text, each record in the input data schema is transformed into a line of output text. The first item of the data schema is transformed into the first column of the text line in the output text string. Rendering a data schema into a text string is exactly the opposite process of parsing a text string into a data schema.

Data Format Editor

This editor contains **General**, **Data Format Configuration**, **Data Format Editor**, and **Field Offsets** sections.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

Data Format Configuration

You can define the structure of the data file by using the Data Format Configuration panel.

Field	Description
Format Type	The Type of formatting for the text. The text can be either Delimiter separated or Fixed format.
	When you select the Delimiter separated option, the text in each column is separated by a delimiter character, specified in the **Col Separator** field. Each line is separated by the character specified in the **Line Separator** field.
	When you select the `Fixed format` option, the text in each column occupies a fixed position on the line. For fixed format text, you must specify the fill character, the line length, and the column offsets.
Col Separator	This field specifies one or more separator characters between columns when the `Delimiter separated` option is specified in the **Format Type** field.
	When rendering text, each element in the input data schema is separated by the column separator in the output text string. If more than one character is specified in this field, the **Render Data** activity places the entire string specified in this field between each column.
	For example, if "`:;`" is specified in this field, then "`:;`" displays between each column in the rendered string.
	When parsing text, each column becomes an element in the output data

Field	Description
	schema. If more than one character is specified in this field, the **Parse Data** activity uses the rules specified in the **Col Separator Parse Rule** field to determine how to parse the data. This field supports the Module property. **Note:** Space is not supported using the module property as Column separator in the **Data Format** shared resource for **Render Data** activity.
Col Separator Parse Rule	Specifies the rules to use for multiple column separator characters when parsing data. The choices are the following: • Treat all characters entered as a single column separator string. The characters entered into the **Col Separator** field are treated as a single string that acts as a separator. For example, if the specified **Col Separator** is ":;" then `Apple:;Orange::Pear` is treated as three columns. • Treat each character specified as a potential column separator Any of the characters acts as a column separator. For example, if the specified Col Separator is ":;" `Apple;Orange:Pear` is treated as three columns.
Line Separator	Specifies one or more characters that determines the end of each line. Available Line Separator characters are: • New Line • Carriage Return • Carriage Return/Line Feed(windows) When parsing text, each line is treated as a new record in the output data schema. When rendering text, each data record is separated by the line separator character in the output text string. The last line in your input file must be ended by the specified line separator.
Fill	When processing fixed-format columns, this is the type of character that is

Field	Description
Character	used to fill the empty space in a column and between columns. This field is only available when you select the Fixed format option in the **Format Type** field. The fill character is used only by the Render Data activity. You can select any one from the following available options: • Space: fills with space • Dash: fills with dash • Others: specify your own custom fill character in the **Fill With** field For example, you have a column that holds an integer and the specified width is 10. One row has the value "588" for that column. Because the width of 588 is three and the column width is 10, the remaining 7 characters are filled with the specified fill character.

Data Format Editor

You can define a custom schema for the text in the Data Format Editor panel. You can define your own datatype here. After defining the data type, the data specified here is used to parse a text string into the specified schema or render the specified schema as a text string. The header element contains the following:

- @attribute
- (any element)
- primitive
- (any element)

The line length is the total length of input lines, including the line separator characters. Include the appropriate number of characters for the selected line separator in the **Data Format Configuration** to the total length of each line.

Field Offsets

When processing fixed-format text, you must specify the line length and the column offsets. This enables the Parse Data or Render Data activity to determine where columns and lines begin and end. You can specify the format of fixed-width text using the **Field Offsets**.

The column offset is the starting and ending character position on each line for the column. Each line starts at zero (0). For each column of the line, you must specify the name of the data item associated with this column (this is the same name you specified for the corresponding elements in the data schema), the starting offset for the column, and the ending offset for the column.

Each column offset can begin where the last column offset ended. Many fixed-format data files are used by databases or are generated by automated processes. These files have rigid file record formats and may not have additional padding space between the columns. When you define each column offset to begin where the last column-offset ends, you can read the data more quickly. This is because you can read sequentially the bytes of the input records.

Consider the following text file. The first two lines of the file indicate offset numbers (each 0 indicates another 10 characters), and the fill character between the columns is spaces:

0 12 30 45 012345678901234567890123456789012345678901234567890012345678 57643 Smith Chris Account 57644 Jones Pat Marketing 57645 Walker Terry Development

Delimiter Separated Fields

When processing delimiter-separated text, each field in the input line is separated by the delimiter specified by the **Column Separator** field. Leading and trailing spaces are stripped from each field and the specified line separator determines when a new record starts.

In some situations, you may not be able to choose a column separator character that does not display in any column data. For example, if you choose a comma as the column separator, there may be commas in some of the column values. To process data that contains column separator characters in a column, you can surround the column with double quotes (" "). You can also use double quote to include leading and trailing spaces as well as line breaks in a field. If you want to display a double quote in a field, escape the double quote by using two consecutive double quotes. That is, use "" to represent a double quote in a field.

The following data illustrates input lines with each field separated by commas. some fields, however, contain commas, leading or trailing spaces, double quotes, and line breaks.

```
57643, Smith, "Chris", Accounting, "State: Be prepared"!
```

```
57644, Jones, "Pat ", Marketing, "Statement: To paraphrase JFK, "Ask not what
your company can do for you, ask what you can do for your company."
```

```
57645, Walker, "Terry", Development, "Statement: May goal is to be CEO
someday".
```

Notice that Pat Jones' statement spans two lines and contains double quotes as well as a comma. The entire field is surrounded by double quotes, so it is still treated as part of the same record.

FTP Connection

FTP Connection shared resource describes a connection to an FTP server. Use the FTP connections when configuring activities in the FTP palette.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

FTP Connection

This section has the following fields.

Field	Literal Value/Module Property	Description
Host	Yes	The host name or IP address of the FTP server.
Port	Yes	The port number of the FTP server. For connections inside of a firewall, when no value is specified in this field, the default value is 21.
Username	Yes	The username to use when logging into the FTP server.

Field	Literal Value/Module Property	Description
Password	Yes	The password to use when logging into the FTP server. This field is mandatory.
Mode	Yes	Specify whether you want to use Active or Passive mode to connect to the FTP server.
Connection Timeout (msec)	Yes	The timeout specified in this field (in milliseconds) applies to the following scenarios: • Making the control socket connection to the FTP Server • Making the data socket connection to the FTP server (passive mode) • Waiting for the data socket connection from the FTP server (active mode) • Reading a reply from control socket after sending an FTP control command • Reading data from the data sockets The default value is 60000 milliseconds. If the specified time is reached when attempting any of these operations, the activity using this FTP connection ends without error and takes any matching success transitions to the next activity in the process.
Remote Filename Encoding	None	A drop-down list of values for specifying the character encoding of file names supplied to FTP commands using this connection. The encoding specified in this field is used when sending path and file names supplied to FTP commands that use this **FTP Connection** resource. Set this field when the FTP server has a different system encoding than the machine where the engine is running.

Field	Literal Value/Module Property	Description
Firewall	Yes	Specifies whether the FTP host resides outside of a firewall. Selecting this checkbox enables the **Proxy Host**, **Proxy Port**, and **Proxy User Name** fields. If this checkbox is selected, you must supply the **Proxy Host** and **Proxy Port** fields with the name and the port of the proxy server used to access sites outside of a firewall. You can optionally specify the username and password for the proxy server, if required.
Proxy Host	Yes	Specifies the host name or IP address of the firewall server. This is used when the remote FTP server is outside of a firewall.
Proxy Port	Yes	Specifies the port number of the firewall server. This is used when the remote FTP server is outside of a firewall.
Proxy username	Yes	Specifies the username to use to connect to the proxy server, if necessary. This is used when the remote FTP server is outside of a firewall.
Proxy Password	Yes	Specifies the password to use for the specified username when connecting to the proxy server, if necessary. This is used when the remote FTP server is outside of a firewall.

Test Connection

When you click the **Test Connection** button, the activity attempts to connect to the specified remote FTP server on the specified port using the specified username and password. You can use this button to determine whether the specified connection configuration is correct.

Available Commands

The **Available Commands** button displays a dialog with details about which FTP commands are available on the specified remote server. FTP servers on different operating

systems and from different vendors support slightly different commands. Use this button to determine the available commands on the remote FTP server before attempting to run them with the **FTP Quote** activity.

Security

Select the **Confidentiality** checkbox to display the **SSL Client** field.

Field	Description
SSL Client	The name of the resource. In the SslClientResource Resource Template wizard, create a resource to connect to the SSL client.

HTTP Client

The **HTTP Client** resource template represents an outgoing HTTP connection.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

HTTP Client

This section has the following fields.

| Field	Literal Value/Module Property	Description
Default Host	Yes	The name of the host that accepts the incoming requests.
		For machines that have only one network card, the default value `localhost` specifies the current machine. For machines with more than one network card, this field specifies the host name of the card used to accept the incoming HTTP requests.
		The default value is localhost.
Default Port	Yes	The port number on which to invoke outgoing HTTP requests.
		The default value is 80.
HTTP Version	No	The HTTP version to be used for communication. By default, it is HTTP 1.1. If required, you can select HTTP 2.0 as the version for the HTTP 2 communication. It is mandatory for the server to support HTTP 2 as well if HTTP 2 is selected as the client protocol. When HTTP 2 is selected as the version, the option of the client in the **Implementation Library** field, changes to **Jetty HTTP Client**, and other options do not appear, as only the Jetty client supports HTTP 2. When HTTP 1.1 is selected as the version, the previous options reappear along with the new option **Jetty HTTP Client**, which also supports HTTP 1 communication.
		When HTTP 2.0 is selected as the HTTP version in the HTTP client shared resource, enabling SSL through the Confidentiality or Default Confidentiality option is compulsory.
		• If Confidentiality is enabled at the HTTP client and HTTP 2 is the version used, it is necessary

Field	Literal Value/Module Property	Description
		to select **Enable Mutual Authentication** at the corresponding SSL client resource and configure the identity store provider, key alias name, and key password.
		Note: HTTP 2 is currently supported only for the HTTP palette along with HTTP 1.1. The SOAP and REST bindings continue to use only HTTP 1.1 for communication.
Implementation Library	No	Specifies the implementation library to be used: • **Apache HttpComponents(Supported by HTTP 1, SOAP and REST)**: must be used for the SOAP and REST Reference Binding, and Invoke REST API activity. When HTTP 2 is selected as the protocol version in the HTTP Connector resource, with Follow HTTP1 Compliance selected, and HTTP Components client is used as the implementation library in the HTTP Client resource, the application fails due to lack of common cipher suites. Hence this scenario is presently not supported for HTTP 2 in TIBCO BusinessWorks™ Container Edition . • **Jetty HTTP Client:** can be used for HTTP activities (both HTTP 1.1 and HTTP 2). This client library is currently not supported for REST and SOAP. • **Apache Commons(Deprecated)**: Apache Commons library is now no longer used, because it is deprecated. Older imported projects using Apache Commons, continue to display Apache Commons as the client option,

Field	Literal Value/Module Property	Description
		but Apache Components is used as the client for executing the HTTP requests, because this library is obsolete.
		The default is Apache HttpComponents(Supported by HTTP 1, SOAP, and REST), when HTTP 1.1 is selected as the protocol version. When HTTP 2 is selected as the protocol version, the Jetty HTTP Client is the only Implementation option available, because currently only the Jetty Client supports HTTP 2.
		Note: - The Jetty HTTP Client implementation can only be chosen for HTTP applications, and is not compatible with SOAP/REST applications. - After an Axis2 upgrade, the Apache Commons client library is no longer used in HTTP. The projects migrated from ActiveMatrix BusinessWorks 5.x shall now show Apache Components as the client library instead of Apache Commons.
Disable Connection Pooling	Yes	Indicates whether to use the single or multi-threaded connection manager. If you select this checkbox, **Maximum Total Connections**, **Idle Connection Timeout(ms)**, **Maximum Total Connections Per Host/Route**, and **Stale Check** fields are disabled. The default value is False. This option is not enabled when the Jetty HTTP Client is selected as the implementation library, because connection pooling happens internally in the Jetty Client library.

Field	Literal Value/Module Property	Description
Maximum Total Connections	Yes	Specifies the maximum number of concurrent, active HTTP connections allowed by the resource instance to be opened with the target service. This property is enabled only if connection pooling is enabled (the Disable Connection Pooling parameter is deselected). The default value is 200. For applications that create many long-lived connections, increase the value of this parameter.
Idle Connection Timeout(ms)	Yes	Keep-alive time (in milliseconds) for idle HTTP connection in the Persistent Connection Pool.
Maximum Total Connections Per Host/Route	Yes	Specifies the maximum number of concurrent, active HTTP connections allowed by the resource instance to the same host or route. This property is enabled only if connection pooling is enabled (the Disable Connection Pooling parameter is unchecked). **Note:** This number cannot be greater than **Maximum Total Connections**. Every connection created here also counts into Maximum Total Connections. The default value is 20.
Stale Check	Yes	Selecting this checkbox determines whether the stale connection check is to be used. Disabling the stale connection check can result in slight performance improvement at the risk of getting an I/O error, when executing a request over a connection that has been closed at the server side.
Cookie Policy	No	Specifies the Cookie Policy to use:

Field	Literal Value/Module Property	Description
		Note: Ensure you have selected **Apache HttpComponents (Supported by HTTP and REST)** as the Implementation Library when selecting a Cookie Policy.
		• **BROWSER_COMPATIBILITY**: Choose this option to select a cookie policy that is compatible with the common cookie management practices, including those that are not fully standards compliant.
		• **NETSCAPE**: Select this option to use the Netscape cookie draft compliant policy. When you select this option, Cookie2 headers are not sent in the request message.
		• **RFC_2109**: Select this option to use the RFC2109 compliant policy. This policy is supported for HTTP palette activities.
		Note: If this policy is selected for the **Invoke API** activity or the REST Reference Binding, the **BEST_MATCH** policy is selected instead.
		• **IGNORE_COOKIES**: Select this option if cookies sent by the server need be disabled.
		• **BEST_MATCH**: This is the default selection. Select this option to choose the **BEST_MATCH** Cookie Policy.
Disable connection state tracking	Yes	This checkbox is enabled only when **Apache HttpComponents (Supported by HTTP and REST)** option is selected in the **Implementation Library** field. When this checkbox is selected, the HTTPS

Field	Literal Value/Module Property	Description
		connections are not reset by the Apache components client, and a single HTTPS connection can be reused. By default, the checkbox is clear. **Caution:** If you select this checkbox, there might be a security risk of leasing a persistent SSL connection to the wrong user. This option is not enabled when the Jetty HTTP Client is selected as the implementation library.
Thread Pool	No	Specifies a queue of threads available to run a queue of tasks. Thread pools are used to improve performance when executing large numbers of asynchronous tasks by reducing per task invocation overhead, provide a means of bounding, and managing the resources consumed when executing a collection of tasks. You can optionally create this client thread pool to route messages to the target service. The thread pool resource can be created by either selecting a thread pool resource template or creating a new one. For more information, see the Thread Pool topic.
Service Discovery Configuration		
Enable Service Discovery	No	Select this checkbox to enable service discovery.
Service Name	Yes	The service name to be discovered on Eureka or Consul. You can also browse the services from Consul or Eureka by using 🔍 icon.

Field	Literal Value/Module Property	Description
		• Consul: To browse the services from Consul, select **Consul** and the service to be set for the HTTP Client Service Registry Configuration. • Eureka: To browse the services from Consul, select **Eureka** and select the service to be set for the HTTP Client Service Registry Configuration. **Note:** Set the **Consul** and the **Eureka** URLs in the Service Registry preference.
Enable Caching	Yes	When enabled, services are discovered only from the local cache instead of accessing the Eureka or Consul servers.
Thread Pool	No	Specifies a queue of threads available to run a queue of tasks. Thread pools are used to improve performance when executing large numbers of asynchronous tasks by reducing per task invocation overhead, provide a means of bounding, and managing the resources consumed when executing a collection of tasks.
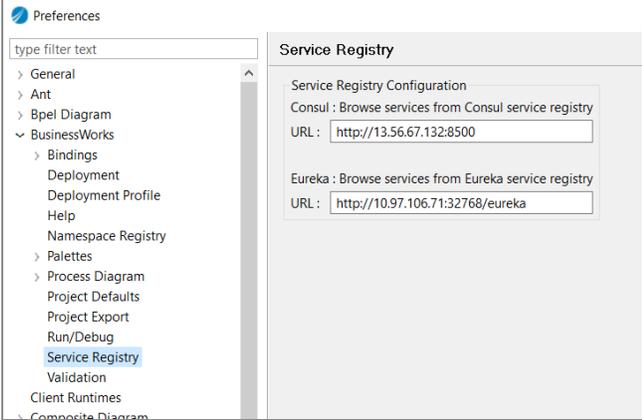
Field	Literal Value/Module Property	Description
		You can optionally create this client thread pool to route messages to the target service. The thread pool resource can be created by either selecting a thread pool resource template or creating a new one. For more information, see the Thread Pool topic.
Enable Loadbalancing	Yes	If enabled, service instances are invoked in a round-robin fashion. If not enabled, the first cached service instance is always invoked irrespective of the number of service instances.

Advanced

This section has the following fields.

Field	Literal Value/Module Property	Description
Socket Timeout (ms)	Yes	Specifies the number of milliseconds to wait for data before the connection is severed. The value zero (0) is interpreted as an infinite timeout and is used when no socket timeout is set in the HTTP method parameters.
Connection Timeout (ms)	Yes	Specifies the number of milliseconds to wait while a connection is being established. The value zero (0) is interpreted as no timeout.
Accept Redirect	Yes	Indicates whether the HTTP method should automatically follow HTTP redirects.

Field	Literal Value/Module Property	Description
		This option is used when client connection receives the redirect responses from server like Moved Permanently, Moved Temporarily, Temporary Redirect and so on. The default value is False.
Retry Count	Yes	Specifies the maximum number of retry attempts for connecting to the server if an exception occurs. When the client library implementation is set to **Apache Components**, the retry count at HTTP client shared resource is accessed internally by the Apache Components Client to create a DefaultHttpRequestRetryHandler (https://hc.apache.org/httpcomponents-client-ga/httpclient/apidocs/org/apache/http/impl/client/DefaultHttpRequestRetryHandler.html), which retries to send the failed requests number of times specified in the Retry Count field. Interrupted IO Exception, Unknown Host Exception, Connect Exception, and SSL Exception are the types of exceptions for which the requests cannot be retried. The requests are retried for all other exceptions, only if the request has not been successfully sent. The default value of retry count is 3.
Use Single Cookie Header	Yes	Select this checkbox if multiple name/value pairs must be sent in a single non-repeating Cookie header element for outgoing HTTP requests. The default value is False.
Use Non-Preemptive Authentication	Yes	Select this checkbox if you want to use non-preemptive authentication.
Log	Yes	Select this checkbox to log the time required for sending and

Field	Literal Value/Module Property	Description
Request/Response Time		receiving the HTTP response in the **Send HTTP Request** activity.
HTTP Proxy	No	Specifies the HTTP proxy server to be used to gain access outside of a firewall. The **Proxy Configuration** shared configuration resource specifies the configuration of the proxy server. For more information, see the Proxy Configuration.

Circuit Breaker Configuration

Field	Literal Value/Module Property	Description
Enable Circuit Breaker	Yes	Select **Enable Circuit Breaker** to configure properties for Resilience 4j configuration. The default value is False.
Circuit Breaker Name	Yes	The specific Resilience4j command. The circuit breaker name should be unique for each **HTTP Connector** shared resource.
Circuit Breaker Configuration		
Sliding Window Type	No	This property configures the type of sliding window which is used to record the outcome of calls when the CircuitBreaker is closed. Sliding window can either be count-based or time-based. If the sliding window is *COUNT_BASED*, the last

Field	Literal Value/Module Property	Description
		slidingWindowSize calls are recorded and aggregated.
		If the sliding window is *TIME_BASED*, the calls of the last slidingWindowSize seconds recorded and aggregated.
		The default value is *COUNT_BASED*.
Failure Rate Threshold	Yes	This property configures the rate of failure threshold in percentage.
		When the rate of failure is equal or greater than the threshold, the CircuitBreaker transitions to open and starts short-circuiting calls.
		The default value is 50.
Minimum Number of Calls	Yes	This property configures the minimum number of calls which are required (per sliding window period) before the CircuitBreaker can calculate the error rate or slow call rate.
		For example, if minimumNumberOfCalls is 10, then at least 10 calls must be recorded, before the failure rate can be calculated.
		If only 9 calls have been recorded the CircuitBreaker will not transition to open even if all 9 calls have failed.
		The default value is 100.
Permitted Number of Calls in HalfOpen State.	Yes	This property configures the number of calls permitted when the CircuitBreaker is half open.
		The default value is 10.
Wait Duration in Open State	Yes	The time required for the CircuitBreaker to wait before transitioning from open to half-open.
		The default value is 60000 (ms).

Field	Literal Value/Module Property	Description
(ms)		
Slow Call Rate Threshold	Yes	This property configures threshold in percentage. The CircuitBreaker considers a call as slow, when the call duration is greater than slowCallDurationThreshold When the percentage of slow calls is equal or greater than the threshold, the CircuitBreaker transitions to open and starts short-circuiting calls. The default value is 100.
Slow Call Duration Threshold	Yes	This property configures the duration of threshold above which calls are considered as slow and increases the rate of slow calls. The default value is 60000 (ms)
Automatic Transition From Open to HalfOpen Enabled	Yes	If this is checkbox is enabled, the CircuitBreaker will automatically transition from open to half-open state and no call is needed to trigger the transition. A thread is created to monitor all the instances of CircuitBreakers to transition them to HALF_OPEN once waitDurationInOpenState passes. If the checkbox is disabled, the transition to HALF_OPEN only happens if a call is made, even after waitDurationInOpenState is passed. The advantage here is no thread monitors the state of all CircuitBreakers. The default value is false.
Sliding Window Size	Yes	This property configures the size of the sliding window which is used to record the outcome of calls when the CircuitBreaker is closed. The default value is 100.

Field	Literal Value/Module Property	Description
Max Wait Duration in HalfOpen State (ms)	Yes	This property configures a maximum wait duration which controls the longest amount of time a CircuitBreaker could stay in Half Open state, before it switches to open. Value 0 means Circuit Breaker would wait infinitely in HalfOpen State until all permitted calls have been completed. The default value is 6000 (ms)
Exceptions to record	No	This table lists exceptions that record as a failure and thus increase the failure rate. Any exception matching or inheriting from the list is not counted as a failure or if a list of the exceptions is specified, all other exceptions count as a success unless explicitly ignored by `ignoreExceptions`. **Note:** The complete qualified names of the exceptions should be added in the Exceptions to record table. For example, adding `ConnectException` will not work. You need to add the complete qualified name, `java.net.ConnectException`
Exceptions to ignore	No	This table lists exceptions that are ignored and neither count as a failure or success. Any exception matching or inheriting from the lists is not counted as a failure or success, even if the exceptions are a part of the `recordExceptions`. **Note:** The complete qualified names of the exceptions should be added in the Exceptions to ignore table.For example, adding `ConnectException` will not work. You need to add the complete qualified name, `java.net.ConnectException`
Record Failure Predicate	No	Using this field you can browse and select a java class that implements the predicate interface.

Field	Literal Value/Module Property	Description
		Note: The application module must be converted to JAVA projects and the predicate class should be created under the src folder.

For example,

```
public class CustomPredicate implements
Predicate<Throwable> {
 @Override
 public boolean test(Throwable throwable) {
 return !(throwable instanceof
java.net.NoRouteToHostException);
 }
}
```

The above predicate ignores the `NoRouteToHostException` exception and records other exceptions.

Specific error codes can be recorded or ignored by using the below exception classes:

`BWCircuitBreakerClientExceptions` and `BWCircuitBreakerServerExceptions`

Examples of conditions that can be used in predicate class to handle error codes:

`BWCircuitBreakerClientExceptions)throwable).getCode () == 4xx`

`BWCircuitBreakerServerExceptions)throwable).getCode () == 5xx`

Security

Security can include any combination of the following characteristics:

Field	Literal Value/Module Property	Description
Authentication	None	To be included when the messages must be authenticated. Authentication can be performed with user names and passwords.
		Identity Provider : Provides user name and password credentials encapsulated in an identity provider resource. Selecting this checkbox activates the **Identity Provider** field.
		Note: You can enforce authentication on requests by selecting the **Authentication** checkbox, or by associating the Basic Credential Mapping policy with REST or HTTP activities. If you choose to use the policy, ensure the **Authentication** checkbox on the **HTTP Client** resource is not selected.
Oauth Configuration	None	Selecting this checkbox enables the OAuth configuration.
		OAuth configuration requires additional configurations. For more information about OAuth configuration, see OAuth Configuration Resource.
Kerberos Authentication	None	Selecting this checkbox enables Kerberos Authentication.
		Kerberos Authentication requires a few additional configurations. For more information about configurations, see the section "Kerberos Authentication Configuration".
		To enforce Kerberos authentication on REST binding,

Field	Literal Value/Module Property	Description
		you must select **Apache HttpComponents (Supported by HTTP and REST)** library. **Note:** Kerberos authentication is not supported on **Invoke REST API**. Use REST Reference Binding or HTTP to invoke a kerberized service.
Username	None	Enter the Active Directory user name to be used for the Kerberos authentication. This field is enabled only when the **Kerberos Authentication** field is selected.
Password	None	Enter the password for the specified user name. This field is enabled only when the **Kerberos Authentication** field is selected. This field is mandatory.
Default Confidentiality	Yes	Select the checkbox to encrypt or decrypt messages. Select the checkbox to enable the SSL with default configuration. For more information about SSL Client configuration, see the SSL Client Configuration. **Note:** If you select **Default Confidentiality**, there is no need to create a shared resource. SSL is enabled with **Default Configuration**. If custom configurations for SSL are required, select **Confidentiality**. When using **Default Confidentiality** ensure that the correct root certificates are present at the `<TIBCO_HOME>\tibcojre64\1.8.0\lib\security\cacerts` trust store.

Field	Literal Value/Module Property	Description
Confidentiality	Yes	Select the checkbox when encrypting or decrypting messages.
		When you select a module property for this field and set it to true, or select the checkbox, the **SSL Client Configuration** field is visible.
		For more information about SSL Client configuration, see the SSL Client Configuration.
JWT Authentication	No	Select the checkbox to use the JSON Web Token (JWT) authentication when exposing a REST Reference.
		Select the checkbox to enable the JWT Token field.
		For more information about JWT, see JWT.io
		By default, the checkbox is clear.
JWT Token	Yes	Provide a JWT value.

Kerberos Authentication Configuration

The `krb5.conf` and `login.conf` files are present under the config folder at the location, *BW Home*/config. We recommend that the configurations in the `login.conf` file should not be changed. However, values in the `krb5.conf` file can be changed according to the Kerberos environment you are working in.

You can configure the following values in the `krb5.conf` file:

- **libdefaults**
- **realms**
- **domain_realm**

HTTP Connector

HTTP Connector resource describes the characteristics of the connection used to receive incoming HTTP requests. This resource is used when the process requires an HTTP request on a specific port where the HTTP server is running. For example, the process starter **HTTP Receiver** and the signal-in **Wait for HTTP Request** activity receiving HTTP requests.

There can be only one process with an **HTTP Receiver** or **Wait for HTTP Request** activity that uses the same **HTTP Connection** resource. With this restriction the HTTP server listening for, the incoming requests dispatches the request to the correct process.

General

The **General** section has the following fields.

Field	Module Property	Description
Package	No	The name of the package in which you want to create a shared resource.
Name	No	The name of the shared resource.
Description	No	A short description of the shared resource.

Basic Configuration

The **Basic Configuration** section has the following fields.

Field	Module Property	Description
Host	Yes	**Note:** Do not change the value for this property. By default, it is set to the BW.HOST.NAME module property.
		Specifies the name of the host that accepts the incoming requests. For machines that have only one network card, the

Field	Module Property	Description
		default value localhost specifies the current machine. For machines that have more than one network card, this field specifies the host name of the card that is used to accept incoming HTTP requests. If there is more than one network card on the machine, and you specify localhost in this field, all network cards on the machine listen for incoming HTTP requests on the specified port. When using a local host for the system property, use the full name of your machine as the host in the client. **Note:** You can start more than one HTTP server on a single port by using **Context** and **Path** fields on **HTTP Receiver** and **Wait for HTTP** activities. Ensure that you use a different port, if different **Context** or **Path** are not specified. To allow or restrict IP addresses in the HTTP Connector shared resource, add the following VM arguments in TIBCO Business Studio for BusinessWorks: • `-Dbw.plugin.http.httpConnector.allowIPs=<IP address>` • `-Dbw.plugin.http.httpConnector.restrictIPs=<IP address>` You can provide a list of IP addresses by using a semicolon (;) as a separator. For example, -Dbw.plugin.http.httpConnector.allowIPs=10.98.179.102;10.98.179.110 Use hyphen to provide a range of IP addresses. For example, -Dbw.plugin.http.httpConnector.restrictIPs=10.98.179.102-10.98.179.110

Field	Module Property	Description
		You can also provide a range of IP addresses along with other IP addresses not in the address range by semicolons (;). For example, - Dbw.plugin.http.httpConnector.restrictIPs=10.98.179.102-10.98.179.110;10.98.179.120
Port	Yes	Specifies the port number on which to listen for incoming HTTP requests. The default value for **PORT** field is set to `BW.CLOUD.PORT`. The `BW.CLOUD.PORT` is a module property whose default port value is 8080. **Note:** Do not use port 7777. It is reserved for an internal use.
HTTP Version	No	The HTTP version to be used for communication. By default, it is HTTP 1.1. If required, you can select HTTP 2.0 as the version for the HTTP 2 communication. It is mandatory for the communicating client to support HTTP 2 as well if HTTP 2 is selected as the server protocol. If multiple clients are used, which may or may not support HTTP 2, it is recommended to select the **Follow HTTP1 Compliance** checkbox, subject to the condition that the clients have the ability to negotiate the HTTP version with the server. Currently, among the TIBCO Cloud Integration HTTP clients, only the Jetty client supports HTTP 2 communication. When HTTP 2.0 is selected as the HTTP version in the HTTP connector shared resource, enabling SSL through the Confidentiality or Default Confidentiality option is compulsory. • If Confidentiality is enabled at the HTTP connector and HTTP 2 is the version used, it is necessary to select **Enable Mutual Authentication** at the corresponding SSL server resource and configure the keystore provider.

Field	Module Property	Description
		Note: Both HTTP 2 and HTTP 1.1 are currently supported only for the HTTP palette. The SOAP and REST bindings continue to use only HTTP 1 for communication.
Follow HTTP1 Compliance	Yes	This option is enabled only when the HTTP 2.0 is selected as the HTTP version. This option, if selected enables negotiation between the server and client to decide the HTTP version to be used. HTTP 1.1 and HTTP 2.0 are the supported versions. If the client does not have the ability to negotiate the protocol, selecting this checkbox causes the server to fall back to the default HTTP 1 implementation. Hence this checkbox should be selected only if the HTTP 2 client can negotiate the protocol with the server. Currently, no client used in ActiveMatrix BusinessWorks 6.x can negotiate the protocol version. Hence it is recommended not to enable this option, if HTTP 2 communication is required. Currently, HTTP communication with this option works only when the Jetty HTTP Client is selected as the client library, for TIBCO BusinessWorks Container Edition applications.
Accept Queue Size	Yes	Specifies the number of connection requests to be queued before the operating system starts sending rejections. The default values are: -1 and 0. These default values signify that the queue size is 50 or OS-specific.
Acceptor Threads	Yes	These are the Jetty server threads. Acceptor threads are HTTP socket threads for an **HTTP Connector** resource that accept the incoming HTTP requests. The value should be compulsorily set between 1 and (the number of processors in the runtime + 3)/4. This shall ensure optimal performance during request processing using Non-Blocking IO. The Jetty 9 library used for the HTTP Connector, uses Non-Blocking IO as the only means

Field	Module Property	Description
		of IO.
		The default value is 1.
		For more information about Acceptor Threads, see Jetty Documentation.
Hide Server Implementation	Yes	Select the checkbox to hide additional details of a server implemented in the HTTP response, and log files in case of errors.
		By default, the checkbox is clear.

Advanced Configuration

The **Advanced Configuration** section has the following fields.

Field	Module Property	Description
Header Buffer Size (B)	Yes	Specify to set the size of the buffer for request headers. An idle connection at most has one buffer of this size allocated. The default value is 4K. **Note:** Increase the default value while sending HTTP requests with large header values and using Jetty HTTP Client library, set the new property "com.tibco.bw.http.jettyclient.requestBufferSize" with the proper request size.
Request Buffer Size (B)	Yes	Specify to set the size of the content buffer for receiving requests. These buffers are used only for active connections that have requests with bodies that do not fit within the header buffer. The default value is 8K.
Use Direct Buffers	Yes	Specify to determine whether direct byte buffers are used or not. The default value is True.

Field	Module Property	Description
Response Buffer Size (B)	Yes	Specify to set the size of the content buffer for sending responses. These buffers are used only for active connections that are sending responses with bodies that do not fit within the header buffer. The default value is 24K.
URI Encoding	Yes	Specifies the character encoding used to decode the URI bytes. If you do not set any value for uriEncoding, the default UTF-8 is used.
Max Idle Time (ms)	Yes	Specify to set the maximum idle time for a connection. The **Max Idle Time** is applied in the following cases: • When waiting for a new request to be received on a connection; when reading the headers and content of a request • When writing the headers and content of a response Jetty interprets this value as the maximum time between the progress made on the connection. The timeout is reset if implemented by jetty if a single byte is read or written.
Enable DNS Lookups	Yes	Selecting this checkbox enables a domain name system (DNS) lookup for HTTP clients so that the IP address is resolved to a DNS name. This can adversely affect the throughput. This checkbox is not selected as default. Select this checkbox only when required. The default value is `False`.
Low Resource Max Idle Time (ms)	Yes	Specifies to set the period in `ms` for the connection to be idle. This results in the server rapidly closing idle connections to gracefully handle high load situations.
Compression	Yes	Specifies if the output of the HTTP Connection is

Field	Module Property	Description
		compressed using the HTTP/1.1 GZIP compression. The acceptable values for this field are: `True` or `False` The default value is `False`. **Note:** The GZip compression type is only supported for server side compression.
Linger Time (ms)	Yes	Specifies to set the *LINGER_TIME* on the connection socket. This setting affects only the close connection socket.
Compressible Mime Types	Yes	Specifies the list of MIME types for which HTTP compression may be used. The default value for this comma-separated list is `text/html, text/xml, text/plain`.
Session Timeout (s)	Yes	The timeout (in seconds) for the HTTP session to expire after inactivity.
Max Post Size	Yes	Specifies the maximum size in bytes of the POST that the container FORM URL parameter parsing can handle. By default, the value of this field is set to 2MB or 2097152 bytes. Setting the value less than or equal to zero disables the limit for maxPostSize.
Reverse Proxy Host	Yes	Specifies the name of the reverse proxy host that accepts the incoming requests.
Max Save Post Size	Yes	Specifies the maximum size in bytes of the POST that the container can `save/buffer` during FORM or CLIENT-CERT authentication. • FORM authentication: the request message is saved while the user is redirected to the login page. It is retained until the user authentication succeeds or the session associated with the authentication request expires.

Field	Module Property	Description
		• CLIENT-CERT authentication: the request message is buffered for the entire duration of the SSL handshake. After processing the request, the buffer is emptied. However, in both the cases the buffering happens before the user authentication. By default, the value of this field is set to 4Kb or 4096 bytes. Setting the value to -1 disables the limit for this field.
Reverse Proxy Port	Yes	The reverse proxy port number on which to listen for the incoming HTTP requests.
Minimum QTP Threads	Yes	The Queued Thread Pool (QTP) uses the default job queue configuration to The QTP threads accept the requests from the acceptor threads. This field specifies the minimum number of QTP threads available for the incoming HTTP requests. The HTTP server creates the number of threads specified by this parameter when it starts up. The default value is 10.
Maximum QTP Threads	Yes	This field specifies the maximum number of threads available for the incoming HTTP requests. The HTTP server does not create more than the number of threads specified by this parameter. The default value is 75. This limit is useful for determining number of incoming requests that can be processed at a time. Setting a high number creates that many threads and drastically reduce the performance.
Enable Access Logs	Yes	Select the **Enable Access Logs** checkbox to print the HTTP access logs.

Field	Module Property	Description
Custom Access Logs Format	Yes	This field provides the custom Jetty access logger format. The value provided in this field takes effect at runtime only when **Enable Access Logs** is enabled. The logger format can be a combination of format strings available here: https://www.eclipse.org/jetty/javadoc/jetty-9//org/eclipse/jetty/server/CustomRequestLog.html
Disable HTTP Methods	Yes	Disable specific HTTP methods by adding the required methods to the field list. To add a method to the list, click the **Add** icon . To remove a method from the list, select the required method and click the delete icon . **Note:** The first time you click the **Add** icon , by default the TRACE method is selected and added to the list of methods to be disabled.
Share Across Applications	Yes	Select the **Share Across Applications** checkbox to share **HTTP Connector** shared resource across several application modules. **Note:** This checkbox is applicable if the HTTP Connector shared resource is present in the shared module, and the checkbox is not be applicable if the HTTP Connector shared resource is present in the application module.

Security

Security can include any combination of the following characteristics:

Field	Module Property	Description
Confidentiality	Yes	To be included when encrypting or decrypting the messages. Selecting this checkbox displays the **SSL Server Configuration** field. Older security protocols TLS v1.0, v1.1 and SSL v3 are not supported, in order to prevent security vulnerabilities.
JWT Authentication	No	Enable this to use JSON Web Token (JWT) authentication when exposing a BusinessWorks REST Service. JWT is an open standard that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. Selecting this checkbox enables the following fields: • **JWT Algorithm**: Use the drop-down list to specify an HMAC algorithm (**HS256**, **HS384** or **HS512**) or RSA algorithm (**RS256**, **RS384** or **RS512**) • **Secret Key**: When HMAC is used, a shared secret (password) is required to encrypt the JWT token. Enter the password in the **Secret Key** field. • **Public Key**: This field is visible when any of the RSA algorithms are selected. The public key is used for encryption and can be provided by uploading the respective file using the **File Picker** option. The file should be of `.pem` format. • **Private Key**: This field is visible when any of the

Field	Module Property	Description
		RSA algorithms are selected. The private key is used for decryption and can be provided by uploading the respective file using the **File Picker** option. The file should be of `.pem` format.

> **Note:** When Jetty debug logs are enabled, intermittent `ClosedChannelException` error messages occur in the logs. These messages are to be ignored, since this is an expected behavior due to internal implementation of the Jetty server. These messages are not an indication of any errors.

Service Registry Configuration

Field	Module Property	Description
Service Name	Yes	The name of the service to register on Eureka or Consul.
Eureka Configuration	No	Select this checkbox to modify the App Metadata configuration for Eureka Service Registry. By default, the checkbox is clear.
Service Port	Yes	The service port with which the service gets registered on the Eureka server. The default value is 8080.
Eviction Duration (S)	Yes	Specifies the length of the lease (in seconds) for the service. The default value is 30.
Renewal Interval (S)	Yes	Specifies the time (in seconds) after which the lease should be renewed. The default value is 30.

Field	Module Property	Description
Home Page URL	Yes	This is the default location of a useful endpoint for home page. To override this parameter, provide input in the following format: `<port>/<url>`
Status URL	Yes	This is the default location of a useful endpoint for status. To override this parameter, provide input in the following format: `<port>/<url>`
Health check URL	Yes	This is the default location of a useful endpoint for health check. To override this parameter, provide input in the following format: `<port>/<url>/`
Data Center Name	Yes	This sets the data center information for where the instance is running. Default value is MyOwn.

Identity Provider

Identity Provider shared resource provides username and password credentials.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.

Field	Description
Description	A short description of the shared resource.

Identity Provider

This section has the following fields.

Field	Description
Username	Enter a username.
Password	Enter a password.

JDBC Connection

JDBC Connection resource describes a JDBC connection. JDBC connections are used when specifying activities from the JDBC palette.

General

The **General** section has the following fields.

Field	Hot Update	Description
Package	No	The name of the package in which you want to create a shared resource.
Name	No	The name of the shared resource.
Description	No	A short description of the shared resource

JDBC Connection

This section has the following fields.

Field	Literal Value/Module Property	Hot Update	Description
Connection Type	None	No	Specifies the kind of JDBC connection that you want to create.
Maximum Connections	Yes	No	The maximum number of database connections to allocate. The default maximum is 10. The minimum value that can be specified is 1.
Login Credentials	Yes	Yes	The username used to connect to the database. The password used to connect to the database.

JDBC Driver

This section has the following fields.

Field	Literal Value/Module Property	Hot Update	Description
Database Driver	Yes	No	The name of the JDBC driver class. If you select a supported driver from a list of supported drivers, the **Database URL** field is populated with a template for the URL of the driver. The following drivers are packaged: • PostgreSQL • Microsoft SQL Server In addition, the JDBC palette also supports

Field	Literal Value/Module Property	Hot Update	Description
			the MySQL, Oracle, and DB2, and MariaDB database. **Note:** The TIBCO Business Studio for BusinessWorks product can connect with MariaDB ® or MySQL through use of MariaDB® Connector/J, which is subject to free and open-source software license terms and available at https://mariadb.com/downloads/mariadb-tx/connector. MariaDB Connector/J is not part of the product and therefore not within the scope of your license for the TIBCO Business Studio for BusinessWorks product. Accordingly, MariaDB Connector/J is not covered by the terms of your agreement with TIBCO, including any terms concerning support, maintenance, or warranties. Download and use of MariaDB Connector/J is solely at your own discretion and subject to the free and open-source license terms applicable to MariaDB Connector/J. When editing this field, you can also specify for the JDBC Palette to use a custom JDBC driver. For instructions on how to do this, see Using Custom Drivers. **Note:** Custom drivers must be compliant with the JDBC 4.0 API Specification.
Database URL	Yes	Yes	The URL to use to connect to the database. A template of the URL is supplied for the selected JDBC driver. Supply the portions of the URL that are in angle brackets (<>).

Field	Literal Value/Module Property	Hot Update	Description
Login Timeout (secs)	Yes	No	The length of time (in seconds) to wait for a successful database connection. Only JDBC drivers that support connection timeouts can use this field. If the JDBC driver does not support connection timeouts, the value of this field is ignored. Most JDBC drivers support connection timeouts.
Connection Idle Timeout	Yes	No	This option monitors the given time. When the time lapses, it cleanly closes the connection awaiting inside the pool of connections. You can provide time in any of the following units: **Seconds** **Minutes** **Hours** When you choose **Minutes** option and then select the **Hours** option, the value is rounded up to the nearest whole number. That is, 300 seconds are converted to 5 minutes or 1 hour. When you choose **Hours** and then select the **Minutes** option, the rounding takes place at the most convenient whole number. That is, 1 hour becomes 60 minutes or 3,600 seconds. The default value is 300 seconds. **Note:** If you enter a value less than or equal to Zero (0), the monitoring does not take place. The pool does not close any physical connection.

Test Connection

Click the **Test Connection** button to test whether the specified configuration fields result in a valid connection to a database.

Set MSI Authentication

This button is visible only when the `com.microsoft.sqlserver.jdbc.SQLServerDriver` driver is selected in the **Database Driver** field and the **Enable Azure AD Managed Identity Authentication** field is selected from **Windows > Preferences > BusinessWorks > Palettes > JDBC**.

To use the Set MSI Authentication in Azure Cloud, you are required to use the ServicePrincipal Authentication in TIBCO Business Studio for BusinessWorks. This is needed to access the metadata database while using the TIBCO Business Studio for BusinessWorks.

When the application is ready to be deployed, navigate to the shared resource and click the **Set MSI Authentication** button. Once this is done, the application is ready to be used on Azure Cloud.

Installing Supported Drivers for Run Time

If you want to use other databases, other than PostgreSQL or Microsoft SQL Server, you must add them separately. This is a one-time process for every installation of TIBCO BusinessWorks Container Edition

To add the drivers at run time for Oracle, MySQL, DB2, or MariaDB, run the `bwinstall` command from `TIBCO_HOME`/bwce/`version`/bin. Ensure that you run `bwinstall` with no arguments to access the help and instructions to add the driver to the run time.

Supported Drivers

The following database drivers are supported:

- `com.mysql.jdbc.Driver`

- `org.postgresql.Driver`

- `oracle.jdbc.OracleDriver`

- `com.microsoft.sqlserver.jdbc.SQLServerDriver`

- `com.ibm.db2.jcc.DB2Driver`

- `org.mariadb.jdbc.Driver`

> **ℹ Note:** To connect to MySQL Server, you can select the MariaDB native driver (`org.mariadb.jdbc.Driver`) from the **Database Driver** dropdown menu.

The following Data Direct Drivers are supported:

- `tibcosoftwareinc.jdbc.postgresql.PostgreSQLDriver`

- `tibcosoftwareinc.jdbc.mysql.MySQLDriver`

- `tibcosoftwareinc.jdbc.sqlserver.SQLServerDriver`

- `tibcosoftwareinc.jdbc.db2.DB2Driver`

- `tibcosoftwareinc.jdbc.oracle.OracleDriver`

Using Custom Drivers

Installing custom drivers comprises of three parts:

1. Enabling Custom Drivers
2. Configuring JDBC Connection to Use a Custom Driver
3. Setting up Custom Drivers for Run Time

Enabling Custom Drivers

In TIBCO Business Studio for BusinessWorks, you can enable the JDBC palette to use custom drivers. Before you do this, ensure that you place the drivers in *TIBCO_HOME*/bwce/*version*/config/design/thirdparty.

1. Click **Click Here to Set Preferences**. Optionally, go to **Windows > Preferences > BusinessWorks > Palettes > JDBC**.

2. Select the **Allow use of custom driver** checkbox and click **Apply**.
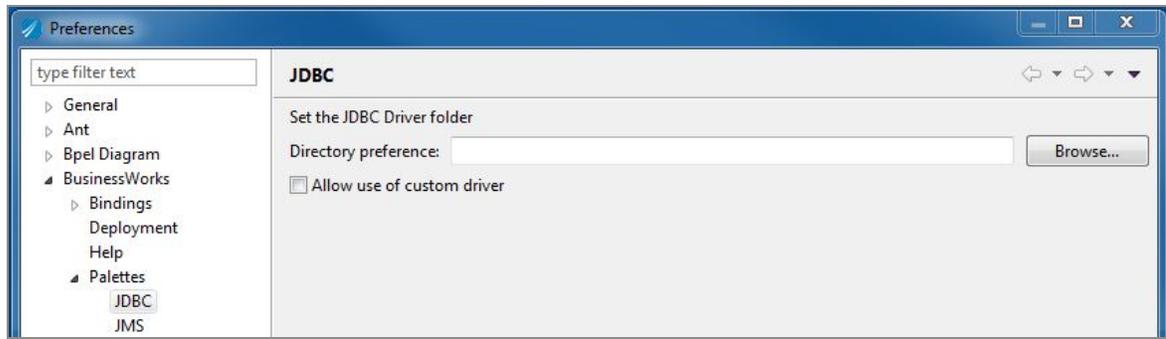
3. Click **OK** and restart TIBCO Business Studio for BusinessWorks for this configuration to take effect.

Configuring JDBC Connection to Use a Custom Driver

Follow these steps to configure the **JDBC Connection** Shared Resource to use a custom driver:

1. Open the **JDBC Connection** Shared Resource from the Project Explorer.

2. Click in the **Database Driver** field, and select the **custom.jdbc.driver** option from the dropdown menu.

3. Edit the following fields that display under the **Login Timeout (sec) field**:

   - **Database Driver**: Type the name of the driver class to use.

   - **Database Name**: Type the name of the database to connect to. If the database name is the name of a schema, type the schema to connect to.

   - **Jar File**: Type the name of the Jar archive file stored at `TIBCO_HOME/bwce/version/config/design/`. This file to create the driver. If you are adding multiple jar files, separate the file names with commas.

     > ⓘ **Note:** To ensure that TIBCO Business Studio for BusinessWorks creates a driver bundle for the application at run time, confirm that the Jar archive file is stored under the third-party folder at `<BWCE_Home>/config/design/thirdparty`.

4. Click **Test Connection**. If you have entered the correct database for connection details, and have already created a plug-in project to support the custom driver, TIBCO Business Studio for BusinessWorks confirms that you have successfully connected to the database.

> **ⓘ** **Note:** If you are unable to connect to the database you specified, check the configurations that you made to the **JDBC Connection** Shared Resource. The database connection field can also fail if you have not created a plug-in project to support the custom driver at run time. In this case, TIBCO Business Studio for BusinessWorks displays a warning message indicating that it does not recognize the custom driver you selected. Click **OK** and complete step 5.

5. Follow these steps:

   a. In the Create New Plug-in Project wizard that displays, enter a name in the **Project name** field, accept the default selections, and click **Next**.

   b. In the Content pane, accept the default values, and click **Next**.

   c. In the Custom Driver Properties pane, edit the following fields:

   > **✓** **Tip:** Refer to your JDBC driver documentation for additional details.

   - **Datasource Class**: Type the name of the class that TIBCO Cloud™ Integration should use to establish a connection data source between the **JDBC Connection** Shared Resource and the database.

   - **Pooled Datasource Class**: Type the name of the class that TIBCO Cloud Integration should use to create a pooled connection data source between the **JDBC Connection** Shared Resource and the database.

   d. Click **Finish**, and the newly created plug-in project displays on the **Project Explorer** pane. Click the custom plug-in project to expand it.

   e. Click the src folder to display the java file, and double-click the java file to edit it. Locate the following instructions in the Java file, and follow them to import packages that contain the datasource and pooldatacourse classes:

   ```
   /*******************************
   TODO*************************************
   * Import the packages from jar that contain the data source
   and drivers  *
   * by hovering over the data source class name with your mouse
   ```

```
and          *
* select the import option
          *


*************************************************************
***********/
```

f. Edit the methods and properties provided in the template file, or add your own.

> **ⓘ Note:** Methods with the @Override annotation can be modified. Whether you edit @Override methods, do not remove them from the template file.

Setting up Custom Drivers for Run Time

Follow these steps to set up custom drivers for run time:

1. Export the custom driver JAR file by right-clicking on the Plug-in-Project from the **Project Pane** and clicking **Export > Plug-in Development > Deployable plug-ins and fragments**.

2. In the Export wizard, complete the following steps:

   a. Select a plug-in project.

   b. On the **Destination** tab, and specify `<BWCE_HOME>/system/shared` as the target directory to export the runtime bundle.

3. Click **Finish** to export the plug-in project as a JAR file. The JAR file is exported to the plug-ins folder at `<BWCE_HOME>/system/shared/plugins`.

   For Docker, see Extending the Base Docker Image in the *TIBCO BusinessWorks Container Edition Application Development*.

   For Cloud Foundry, see The TIBCO BusinessWorks™ Container Edition Buildpack in the *TIBCO BusinessWorks Container Edition Application Development*.

> **ⓘ Note:** The `plug-ins` folder is automatically generated after you export the JAR file, but you can delete this folder after moving the JAR file to the shared folder at `<BWCE_HOME>/system/shared`.

Uninstalling Drivers for Run Time

To uninstall the drivers at run time for Oracle, MySQL, DB2, and MariaDB, run the `bwuninstall` command from `TIBCO_HOME`/bwce/`version`/bin/thirdparty. Ensure that you run `bwuninstall` with no arguments to access the help and instructions to uninstall the driver at run time.

Azure Authentication

Azure Authentication helps to connect the SQL Server database with the JDBC shared resource.

The following are the connection properties to connect the Azure Authentication with the SQL Server database.

- Azure Password Authentication.

- Azure Managed Identity Authentication.

Azure Password Authentication:

This connection property is used to connect to the SQL Server database with Azure Authentication username and password.

The Azure Password Authentication comprises of the following parts:

1. Enabling the JDBC palette to use Azure Password Authentication

2. Configuring the JDBC Connection Shared Resource to use Azure Password Authentication

3. Setting up Azure Authentication for Run Time

Enabling the JDBC palette to use Azure Password Authentication

In TIBCO Business Studio for BusinessWorks, you can enable the JDBC palette to use azure authentication for. Before you do this, create a folder **AzureLib** in `TIBCO_HOME`/bwce/`version`/config/design/thirdparty and ensure that you place all the required jars to connect to Azure in this folder.

1. Navigate to **Windows > Preferences > BusinessWorks > Palettes > JDBC**.

2. Select the **Enable Azure AD Password Authentication - (Required Jars must be provided)**] checkbox and click **Apply**.

3. Click **OK** and restart TIBCO Business Studio for BusinessWorks for this configuration to take effect.

Configuring the JDBC Connection Shared Resource to use Azure Password Authentication

Follow these steps to configure the **JDBC Connection** Shared Resource to use Azure authentication:

1. Open the **JDBC Connection** Shared Resource from the Project Explorer.

2. On the `JDBC Driver` tab, click the **Database Driver** field, and select the **com.microsoft.sqlserver.jdbc.SQLServerDriver** option from the dropdown menu.

3. On the **Database URL** field, add the URL to connect to the database.

4. Edit the fields that display under the **Login Timeout (sec) field**.

5. Click **Test Connection**. If you have entered the correct database connection details, TIBCO Business Studio for BusinessWorks confirms that you have successfully connected to the database.

> **i** **Note:** If you are unable to connect to the database you specified, check the configurations that you made to the **JDBC Connection** Shared Resource.

Setting up Azure Authentication for Run Time

Follow these steps to set up Azure authentication for run time:

1. Stop all the applications, AppNodes, and BWAgents.

2. Back up the `com.tibco.bw.tpcl.jdbc.datasourcefactory.sqlserver_4.0.XXX` folder from `<BWCE_HOME>\2.5\system\shared` to `tmp` directory.

3. Add the required jar in the `lib` folder in `com.tibco.bw.tpcl.jdbc.datasourcefactory.sqlserver_4.0.XXX`.

4. In the META-INF folder, locate the MANIFEST.MF file and modify the Bundle-ClassPath in this file, and append the jar names added in step 3.

5. Copy the modified `com.tibco.bw.tpcl.jdbc.datasourcefactory.sqlserver_ 4.0.XXX` folder in `<BWCE_HOME>\2.5\system\shared`.

Azure Managed Identity Authentication:

This connection property is used to connect to the SQL Server database from an Azure resource with support for **Identity** enabled.

The Azure-Managed Identity Authentication comprises of the following parts:

1. Enabling the JDBC palette to use Azure-Managed Identity Authentication

2. Configuring the JDBC Connection Shared Resource to use Azure-Managed Identity Authentication

Enabling the JDBC palette to use Azure-Managed Identity Authentication

In TIBCO Business Studio for BusinessWorks, you can enable the JDBC palette to use Azure authentication for Managed Identity. Before you do this, create a folder **AzureLib** in *TIBCO_ HOME*/bwce/*version*/config/design/thirdparty and ensure that you place all the required jars to connect to Azure in this folder.

1. Navigate to **Windows > Preferences > BusinessWorks > Palettes > JDBC**.

2. Select the **Enable Azure AD Managed Identity Authentication - (Required Jars must be provided)** checkbox. Add the required details in the AD Service-Prinicipal ID and AD Service-Principal Secret fields and click **Apply**.

3. Click **OK** and restart TIBCO Business Studio for BusinessWorks for this configuration to take effect.

Configuring the JDBC Connection Shared Resource to use Azure-Managed Identity Authentication

Follow these steps to configure the JDBC Connection Shared Resource to use Azure authentication:

1. Open the **JDBC Connection** Shared Resource from the Project Explorer.

2. On the `JDBC Driver` tab, click the **Database Driver** field, and select the **com.microsoft.sqlserver.jdbc.SQLServerDriver** option from the dropdown menu.

3. On the **Database URL** field, add the URL to connect to the database.

4. Edit the fields that display under the **Login Timeout (sec) field**.

5. Click **Test Connection**. If you have entered the correct database connection details, TIBCO Business Studio for BusinessWorks confirms that you have successfully connected to the database.

Enabling the module property in JDBC Call Procedure/Function for JDBC Call Procedure Activity

In TIBCO Business Studio for BusinessWorks, you can enable the module property in JDBC Call Procedure/Function by following the below steps:

1. In TIBCO Business Studio for BusinessWorks, navigate to **Windows > Preferences > BusinessWorks > Palettes > JDBC**.

2. Select the **Allow module property in JDBC Call Procedure/Function** checkbox and click **Apply and Close**.

3. Click **OK** and restart TIBCO Business Studio for BusinessWorks for this configuration to take effect.

JMS Connection

JMS Connection resource describes a JMS connection. This resource is used to specify activities on the JMS palette.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

Basic Configuration

This section has the following fields.

Field	Literal Value/Module Property	Description
Connection Factory Type	No	A connection factory is the object used by a client to create a connection to a provider. Select from one of the following available options: • **JNDI**: select to look up the connection factory object. Selecting this option displays the **Queue Connection Factory** and **Topic Connection Factory** fields, when the **Messaging Style** selected is **Queue/Topic**. • **Direct**: select if you want to connect to the JMS server using a URL. Selecting this option displays the **Provider URL** field. Ensure to select the **Queue/Topic** in the **Message Style**.
Messaging Style	None	The available messaging style options are: • Generic • Queue/Topic You can combine these styles in a single application or the application can use any one of these messaging styles.

Field	Literal Value/Module Property	Description
Queue Connection Factory	Yes	This field is available when you select **JNDI** in the **Connection Factory Type** field and **Queue/Topic** in the **Message Style** field. The QueueConnectionFactory object stored in JNDI is used to create a queue connection with a JMS application. For more information about creating and storing QueueConnectionFactory objects, see the JNDI provider documentation.
Topic Connection Factory	Yes	This field is available only when you select **JNDI** in the **Connection Factory Type** field and **Queue/Topic** in the **Message Style** field. The TopicConnectionFactory object stored in JNDI. This object is used to create a topic connection with a JMS application. For more information about creating and storing TopicConnectionFactory objects, see the JNDI provider documentation.
Connection Factory JNDI Name	Yes	This field is available only when you select the **JNDI** in the **Connection Factory Type** field and **Generic** in the **Message Style** field. The GenericConnectionFactory object stored in JNDI is to create connection with the JMS server.
JNDI Configuration	No	A JNDI Configuration shared configuration object that specifies the JNDI connection information. For more information, see JNDI Configuration.

Field	Literal Value/Module Property	Description
		Note: The **New** button is used to create a JNDI Configuration Shared Resource using the preferences set by the user.
Provider URL	Yes	This field is available only when you select **Direct** in the **Connection Factory Type** field.
		The URL to the JNDI service provider, for example,`tcp://<host>:<port>`.
Use UFO Connection Factory	Yes	Use this checkbox when using EMS Unshared State Failover factories.
		This field appears only when the option **Direct** is selected in the **Connection Factory** field.
		When using Unshared failover setup, if a connection loss is detected to server (A), TIBCO BusinessWorks™ Container Edition tries to connect to server (B) as defined in the connection factory configuration.
		After enabling this checkbox you can pass the provider URL in the form of UFO. For example: `tcp://serverA:7222+tcp://serverB:7222`, where serverA and serverB are two EMS servers configured for the Unshared State Failover.
		Unsupported Features and Activities - JMS Palette - JMS Queue Requestor - JMS Topic Requestor - Any other Requestor activity - SOAP Palette - All Activities

Field	Literal Value/Module Property	Description

		• Service Palette
		
		    ○ All activities
		
		• AE Adaptor
		
		    ○ All activities
		
		Looking up factories using JNDI is not supported
		
		• UFO factories cannot be retrieved through JNDI directly.
		
		For more information about Unshared State Failover, see *TIBCO Enterprise Message Service User's Guide*.

Test Connection

Click the **Test Connection** button to test the connection specified in the configuration of this resource.

Using Third-Party JMS Drivers

The JMS palette supports the EMS, IBM MQ, TIBCO Custom JMS, and JBoss JMS drivers.

Adding Drivers for Runtime

For using IBM MQ and JBoss JMS drivers, you must add them separately, for run time. This is a one time process for every installation of TIBCO Business Studio for BusinessWorks. To add the JMS drivers at run time for IBM MQ, or JBoss run the bwinstall command from `<BWCE_Home>/2.x/bin`. Ensure that you run `bwinstall` with no arguments to access the help and instructions to add the driver to the run time.

The script creates driver bundles and adds it to the specific TIBCO BusinessWorks Container Edition environment. Any App Nodes that are created and started in that TIBCO BusinessWorks Container Edition environment includes the drivers automatically. Since the drivers are a part of the system and not part of each application, all the applications on a particular AppNode use the same drivers.

However, this is not true of App Spaces. The drivers belong to an AppNode and not an AppSpace.

To use another location for the drivers, a preference is provided in TIBCO Business Studio for BusinessWorks which can be updated for the JMS palette to use drivers at design time.

To change the **Directory Preference** in TIBCO Business Studio for BusinessWorks, navigate to **Windows > Preferences > BusinessWorks > Palettes > JMS**

If you have changed the preferences and want to use other databases with the JMS palette at design time, place the drivers at `<BWCE_Home>/2.x/config/design/thirdparty` location.
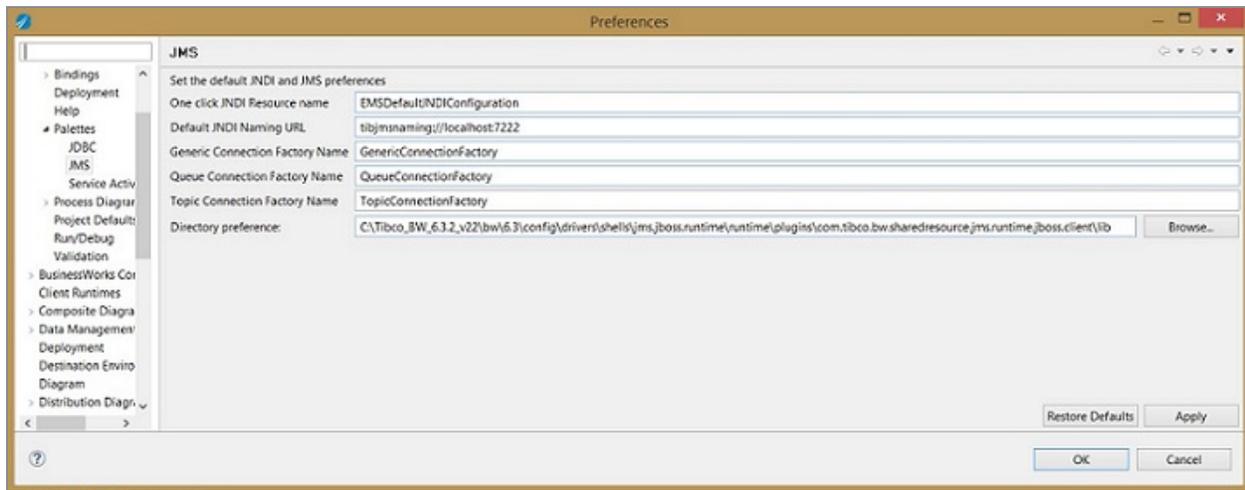
Field	Description
One click JNDI Resource name	The name of **JNDI Connection** shared resource that is auto created when you click the **New** button. This is used only if the shared resource uses JNDI and not Direct.
Default JNDI Naming URL	The default URL to be added in the **JNDI Connection** shared resource when a new resource is created.
Generic Connection Factory Name	The value to be added in the **Generic Connection Factory** field when a new JMS Connection Shared Resource is created.
Queue Connection Factory Name	The value to be added in the **Queue Connection Factory** field when a new JMS Connection shared resource is created.

Field	Description
Topic Connection Factory Name	The value to be added in the **Topic Connection Factory** field when a new JMS Connection shared resource is created.
Directory Preferences	Click the **Browse** button and navigate to the location where the driver libraries are stored.

> **Note:** While testing the JMS connection for JMS drivers other than EMS, ensure that only the jars related to the selected third-party JMS driver are present in the location specified in the directory preference.

Security

Security section has the following field.

Field	Literal Value/Module Property	Description
Login Credentials	Yes	The login credentials to log in to the JMS server. Click the **Username+Password** radio button, if you want to use this option for a secure login. This displays the **Username** and **Password** fields.

Advance Configuration

This section has the following fields.

Field	Literal Value/Module Property	Description
Auto-generate Client-ID	No	Select this checkbox if you want to automatically generate the client ID if no client ID is specified or if the specified ID is already in use.
		When this checkbox is selected, a value cannot be specified for the clientID.
		When an automatically generated client ID is required, the following format is used for the client ID:
		`BW-<projectName>-<topic or queue>-<engineName>-<timestamp>`
		Do not select this checkbox if you do not want to use the value specified in the **Client ID** field. If no value is specified in the **Client ID** field, no client ID is set. If the value specified in the **Client ID** field is already in use, it generates an exception.
		Note: Selecting the **Auto-generate Client-ID** checkbox generates a new Client ID every time a connection is made to the JMS Server. If you are using Durable Subscribers, manually set the client ID for the JMS Connection.
Client ID	Yes	The client ID for the connection. The JMS providers have a provider-specific format for client IDs.
		For more information about client IDs, see the JMS provider's documentation.
		Each connection must use a unique client ID. If you do not specify a value in this field, TIBCO BusinessWorks™ Container Edition ensures the uniqueness of the client ID for each connection.

SSL

Select the **Confidentiality** checkbox to configure the SSL connection parameters.

Field	Literal Value/Module Property	Description
SSL Client	None	Specify an **SSL Client** Shared Resource. For more information about how to configure the Shared Resource, see SSL Client.
Connection Factory SSL Password	Yes	This field is only available if you select the **Confidentiality** checkbox. The SSL configuration is specified in the ConnectionFactory object, except for the client SSL password. You can specify your client SSL password in this field, or you can leave this field empty. If your password is not specified, the private key password is used.

Uninstalling Drivers for Run Time

To uninstall the drivers at run time for IBM MQ or JBOSS, run the `bwuninstall` command from `<BW_Home>/1.0/bin`. Ensure that you run `bwuninstall` with no arguments to access the help and instructions to uninstall the driver at run time.

JNDI Configuration

JNDI Configuration shared resource provides a way to configure the JNDI configuration that can be shared with other activities. This resource can be specified in any resource that permits JNDI connections. For example, JDBC Connection and JMS Connection can use JNDI connections.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource

JNDI Connection

This section has the following fields.

Field	Literal Value/Module Property	Description
JNDI Provider	No	The name of the JNDI connection. Select any one of the JNDI providers from the following options: • JBoss • TIBCO EMS • TIBCO Custom JMS • IBM WebSphere MQ
Initial Context Factory	Yes	The initial context factory class for accessing JNDI. (`com.tibco.tibjms.naming.TibjmsInitialContextFactory`).
Naming Providing URL	Yes	The name of the JNDI provider URL. For example, `tibjmsnaming://localhost:7222`.

JNDI Property Configuration

This section contains optional properties to pass to the JNDI server. Use the + and X keys to add and delete properties in the list.

Field	Literal Value/Module Property	Description
Name	No	The name of the custom JNDI property.
Type	No	The datatype for the custom JNDI property.
Value	Yes	The value for the custom JNDI property.

> **Note:** See the documentation for your JNDI provider for more information about properties that can be passed to the JNDI server.

Security

This section has the following field.

Field	Description
Login Credentials	The login details to access the JNDI connection. Select any one of the following options. • **None**: Selecting this radio button does not display the Username+Password option. • **Username+Password**: Valid username and password to access the JNDI connection.

Using TIBCO Custom JMS JNDI Provider

Installing custom drivers comprises two parts:

1. Enabling the JMS palette to use Custom Drivers

2. Configuring the JNDI configuration shared resource

Enabling custom libraries

In TIBCO Business Studio for BusinessWorks, you can enable the **JMS** palette to use custom drivers.

Before you begin

Ensure that you place the drivers at the `<BW_Home>/config/design/thirdparty` location.

Procedure

1. Click the link **Click Here to Set Preferences**. Optionally, go to **Windows > Preferences > BusinessWorks > Palettes > JMS**.

2. Select required directory preference and click **Apply**.

3. Click **OK** and restart TIBCO Business Studio for BusinessWorks for the configuration to take effect.

Configuring and Setting up Custom Drivers for Runtime

Follow these steps to configure the **JNDI Configuration** shared resource to use a custom driver:

Procedure

1. Open the **JNDI Configuration** shared resource from the **Project Explorer**.

2. Select the **TIBCO Custom JMS** option from the drop-down list in **JNDI Provider** field.

3. Provide **Initial Context Factory** and **Naming Provider URL** values.

> ℹ **Note:** Multiple initial context factories are supported.

4. Click **Test Connection** to display Custom JMS Provider Runtime Support wizard.

5. Click **Yes** to create a Plug-in project. Specify desired project name and click **Next**.

6. In the **Content** pane, accept the default values, and click **Next**.

7. Click **Finish**

   The newly created plug-in project displays on the **Project Explorer** pane.

Adding Multiple Initial Context Factories

You can add multiple initial context factories in the newly created plug-in project.

Before you begin

Newly created plug-in project is displayed on the **Project Explorer** pane.

Procedure

1. Double-click the `Activator.java` file from the `src` folder of the plug-in project.

   `Activator.java` file is opened. By default, there are `getProvider()` and `getProviders()` methods are present in the `Activator.java` file.

2. Uncomment the `getProviders()` method.

3. Create another copy of the `CustomJMSProvider.java` file from the `src` folder in the same folder. You can rename the copy file.

4. Double-click the newly copied file.

   Newly created Java file is opened.

5. In the `getInitialContextFactory()` method, remove the existing context factory and add the name of an initial context factory that is to be supported.

6. Open the `Activator.java` file and create an object of a newly created class.

7. Add the object to the AbstractJMSProvider array.

8. Expand the `OSGI-INF` directory to see the `CustomJMSProvider.xml` file.

9. Create copy of the `CustomJMSProvider.xml` file with the same name as given to the newly created copy of the `CustomJMSProvider.java` file.

10. Double-click the newly created copy of the `CustomJMSProvider.xml` file.

    Newly created XML file is opened.

11. Replace an existing class name with the new class name in the `<implementation class>` attribute.

12. Expand the `MetaINF` folder to see the `MANIFEST.MF` file.

13. Add an entry of the newly created XML file in the `service-implement` attribute in the form `OSGI-INF/<class name>.xml`

14. Save all the open files.

> ℹ **Note:** If you want to add more context factories, repeat the procedure.

Java Global Instance

You can use the **Java Global Instance** shared configuration resource to specify a Java object to be shared across all process instances in a Java Virtual Machine (JVM).

When the engine is started, an instance of the specified Java class is constructed. When the process engine is shut down, if specified, a cleanup method is invoked on the object. The object is released before the engine shuts down. Any **Java Invoke** activity can be configured to access the shared Java global instance when the process engine runs. Any **Java Invoke** activity can access the shared java global instance by invoking the static methods of the configured Java class.

If multiple process instances access the shared Java global instance, you may want to ensure that only one process instance can access the object at a time. You can accomplish this by either declaring the methods of the configured class as synchronous or by using a critical section group.

General

The **General** field has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

Java Global Instance

The section has the following fields.

Field	Description
Class	Use the **Browse a Class** button to locate the class you want to use.
Method	Select a constructor from the drop-down list of constructors available for this class. This is used to construct the class when the engine is started. This field displays only the default constructor with no parameters or constructors that have only parameters of Java primitive types. You cannot create an object using a constructor that requires an object, array, or non-primitive type for a parameter.
Parameter Input	The input parameters for the constructor selected in the **Method** field. Each input parameter has the following three fields: • **Parameter**: the name of the parameter. • **Type**: the datatype of the parameter. • **Value**: specify a value for the parameter in this field. You can set the value by using a module property. **Note:** When a string is the datatype for the parameter, do not use quotes around the string when specifying the value.

Advanced Configuration

This section has the following fields.

Field	Description
Invoke Cleanup Method	Selecting this checkbox signifies that you want to invoke a method on the object before the object is released from memory when the engine ends. You can perform cleanup activities, such as closing connections, releasing resources. Selecting this checkbox enables the **Cleanup Method** field.

Field	Description
Cleanup Method	This field is only available when you select the **Invoke Cleanup Method** checkbox.
	You can specify the name of the method to invoke before the object in memory is released. The method selected cannot contain any parameters, therefore only methods without parameters are listed in this field. Any values returned by the selected method are ignored, but exceptions raised by the selected method are sent to the log.

Keystore Provider

Keystore Provider shared resource provide access to a keystore.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

Keystore

This section has the following fields.

Field	Literal Value/Module Property	Description
Provider	None	The name of the keystore provider. Select from the following available options: • SUN • IBMJCE When you select keystore Type as **PKCS#11**, this field becomes editable, and a drop-down list shows a list of PKCS#11 security providers. Default: Empty. The first matching provider supporting the format is selected.
URL	Yes	The location of the keystore. This field is disabled when you select keystore Type as **PKCS#11**.
Password	Yes	The password for the keystore.
Type	None	The type of the keystore. Select from the following types: • JCEKS: Java Cryptography Extension keystore • JKS: Java keystore • PKCS#11: Hardware keystore type • PKCS#12: Standard keystore type The default value is JKS. **Note:** To configure PKCS#11, enable FIPS configuration of a system. For more information about enabling FIPS mode, see JDK 8 PKCS#11 Reference Guide at Oracle.com.
Refresh Interval	Yes	Refresh interval, greater than 0. If the keystore provider is accessed after the refresh interval has expired:

Field	Literal Value/Module Property	Description
		• The refresh timer is reset to zero. • The operations on the keystore provider are performed on the refreshed copy. This field is disabled when you select keystore Type as **PKCS#11**. The default value is 3600000.

> **Note:** It is recommended not to use expired certificates.

LDAP Authentication

LDAP Authentication shared resource represents configuring the connection to an LDAP server. This connection is used by component implementations to look up names in an LDAP directory server.

LDAP Authentication is primarily used for HTTP basic authentication in TIBCO BusinessWorks Container Edition. The following fields are commonly used:

- **Server URL**: LDAP Server Location, such as, `ldap://10.107.170.145:389`

- **User Search Expression**: search expression for the user, such as `(&(cn={0})(objectclass=user))`

- **User DN Template**: user distinguished name template, such as `cn={0},cn=users,dc=na,DC=tibco,DC=com`

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

LDAP Authentication

The **LDAP Authentication** section has the following tabs:

Connection

The **Connection** tab has the following fields.

Field	Literal Value/Module Property	Description
Server URL	Yes	The URL with the host and port number on which the LDAP server is listening for connections. The default is `ldap://localhost:389`
User Search Expression	Yes	Search expression for the user, such as `(&(cn={0}) (objectclass=user))`.
Admin User Credential	None	The username and password in LDAP format such as `cn={0},CN=Users,DC=ce,DC=na,DC=tibco,DC=com`. Create a limited or restricted user that can only search your tree. Selecting this checkbox displays the **Credentials Provider** option.
Credential Provider	None	Select the identity provider having LDAP server admin credentials.

Field	Literal Value/Module Property	Description
User Search Base DN	Yes	The LDAP user search expression to be used for performing the user search in a specified hierarchy. Leave this field blank to start search from the top-level element in the directory.
User Search Scope	None	This specifies the number of levels in the directory server that is searched for a matching user profile. Select any from the following available options: • **Subtree**: indicates all the subentries (all lower levels) • **One level**: indicates all entries immediately (one level) The default value is Subtree.
User DN Template	Yes	This field appears when the **Admin User Credentials** checkbox is not selected. Write user distinguished name template, such as `cn={0},cn=users,dc=na,DC=tibco,DC=com`.
SSL Configuration	None	Select this checkbox to configure an SSL Client.
SSL Client	None	The name of the resource. In the SslClientResource Resource Template wizard, create a resource to connect to the LDAP server over SSL using SSL client shared resource.

Users and Groups

The **Users and Groups** tab has the following fields.

Field	Literal Value/Module Property	Description
User's name	Yes	The name of the user in the LDAP directory.
Additional user attributes to retrieve	Yes	The field specifies the LDAP attribute used to search and retrieve the user profile. For example, using a different attribute in your directory server such as the given name.
Group Indication	None	Specifies the following groups: • **No Group Info** • **User Attributes Indicates Group**: used for grouping the user that matches the specified attribute. When selected, displays the **User Attribute Group Name** field. • **User DN Indicates Group**: group of user DN. When selected, displays the **User Attribute Group Name** field. • Search Groups: Selecting the **Search Groups** option displays the following groups: • **Group Root DN** • **Group Search Expression** • **Group Attribute User Names** • **Group Attribute Group Name** • **Group Attribute Subgroup Name** **Note:** The Group Indication is useful for Authorization by roles in the **Basic Authentication** policy resource.

Search

The **Search** tab has the following fields.

| Field	Literal Value/Module Property	Description
User Search Scope	None	This specifies the number of levels in the directory server that is searched for a matching user profile. Select any from the following available options: • **Subtree**: indicates all the subentries (all lower levels) • **One level**: indicates all entries immediately (one level) The default is Subtree.
Group Root DN	Yes	The root distinguished name (DN) group name.
Group Search Scope	None	Provides the search scope option, **true** or **false**. • Select **true**, if you want to scan the entire LDAP tree beneath the DN (subtree scope). • Select **false**, if you want to scan only direct children of that DN (one-level scope).
Group Search Expression	Yes	The group search expression property

Advanced

The **Advanced** tab has the following fields.

Field	Literal Value/Module Property	Description
Follow Referrals	None	When you select this checkbox, the LDAP server does not return results. Instead it returns a reference (a referral) to another LDAP server that may contain additional

Field	Literal Value/Module Property	Description
		information such as the names and locations of other objects.
Connection Pool	Yes	Specify the number of the connections in this field. The default is 10. All activities that are part of the same transaction, use the same connection in the connection pool. The first activity in a transaction attempts to reestablish an invalid connection. If a connection becomes invalid during a transaction, the transaction is rolled back and must be retried, if necessary.
Search Timeout	Yes	Specifies the timeout in milliseconds for LDAP search. The default is 0.

Test Configuration

The **Test Configuration** tab has the following fields.

Field	Description
Test	Click the **Test** button to start testing the LDAP search operation.
Search Results	Shows the search results returned by the LDAP search operation.

OAuth Configuration Resource

The OAuth Configuration Resource template represents an OAuth connection.

General

The **General** section has the following fields:

Field	Description
Package	The name of the package in which you want to create a shared resource
Name	The name of the shared resource
Description	A short description of the shared resource

OAuth Configuration Resource

This section has the following fields:

Field	Literal Value/Module Property	Description
Client ID	Yes	Enter Client ID for OAuth Configuration.
Client Secret	Yes	Enter Client Secret for OAuth Configuration.
Access Token URL	Yes	Enter Access Token URL Authorization Server.
Scope	Yes	Enter scope for OAuth Configuration.
Grant Type	None	Select a grant type from the Grant Type drop down list. The following options are supported: • **Client credential** • **Resource password** By default **Client credential** option is selected.
Username	Yes	Enter a username for the **Resource password** grant type.
Password	Yes	Enter a password for the **Resource password** grant type.
Custom Parameters	No	Select this checkbox to add key and values from users.

Test Connection

Click the **Test Connection** button to test whether the specified configuration fields result in a valid connection to an Authorization Server.

Security

Select either the **Default Confidentiality** or **Confidentiality** checkbox to configure the SSL connection parameters.

Field	Literal Value/ Module Property	Description
Default Confidentiality	Yes	Select the checkbox to encrypt or decrypt messages. Select the checkbox to enable the SSL with default configuration. **Note:** If you select **Default Confidentiality**, there is no need to create a shared resource. SSL is enabled with **Default Configuration**. If custom configurations for SSL are required, select **Confidentiality**. When using **Default Confidentiality** ensure that the correct root certificates are present at the `<TIBCO_ HOME>\tibcojre64\1.8.0\lib\security\cacerts` trust store.
Confidentiality	Yes	Select the checkbox when encrypting or decrypting messages. When you select a module property for this field and set it to true, or select the checkbox, the **SSL Client Configuration** field is visible. For more information about SSL Client configuration, see SSL Client Configuration.

Note: OAuth 2.0 is supported with **HTTP Client** shared resource with REST reference binding, **Invoke REST API** or **Send HTTP Request** activities.

Note: The OAuth Refresh Token is a built-in functionality and it depends on the OAuth configuration.

Proxy Configuration

Proxy Configuration shared resource is used to specify a proxy HTTP server when HTTP requests are sent outside of a firewall.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

Proxy Connection Configuration

This section has the following fields.

Field	Literal Value/Module Property	Description
Host	Yes	The host name or IP address of the proxy server.
Port	Yes	The port number of the proxy host.

Security

This section has the following fields.

Field	Description
Authentication	Select the checkbox to specify the authentication to be used.
Identity Provider	Create a UserId resource in the UserIdResource Resource Template

Field	Description
	wizard. This provides access to the username and password credentials to access the proxy connection.

SMTP Resource

A **SMTP Resource** shared resource template represents a connection to an SMTP server. Using the **SMTP Resource**, you can transfer e-mail messages between servers. The SMTP communications are transported by TCP to ensure end-to-end transport.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

SMTP

This section has the following fields.

Field	Literal Value/Module Property	Description
Machine Name	Yes	The name of the host that accepts the incoming requests. The default value is localhost.
Port	Yes	The port number on which you can listen for SMTP

Field	Literal Value/Module Property	Description
		requests. The default value is 25.
Timeout (ms)	Yes	The waiting time for a response from the server. The timeout must be greater than zero (0). A timeout of zero is interpreted as an infinite timeout. The timeout value input in this field also applies to the socket I/O timeout.
Username	Yes	The valid username used to authenticate connections to the server.
Password	Yes	The valid password used to authenticate connections to the server.
Enable STARTTLS	Module Property	Select the **Enable STARTTLS** checkbox to use SMTP over SSL. This checkbox is selected by default.

Security

Select the **Confidentiality** checkbox to display the **SSL Client** field.

Field	Description
SSL Client	The name of the resource. In the SslClientResource Resource Template wizard, create a resource to connect to the SSL client.

SSL Server Configuration

You can specify the SSL parameters for the HTTP connection here.

The following are the fields in the SSL Server Configuration for the HTTP connector.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

Basic SSL Server Configuration

This contains the following fields.

Field	Module Property	Description
Identity Store Provider	No	Used to provide identity store.
Key Alias Name	Yes	The name of the alias used to access the identity.
Key Alias Password	Yes	The password for the alias.
Enable Mutual Authentication	No	Indicates whether the client in the SSL connection authenticates to the server. Selecting this checkbox displays the identity fields.
Client Auth Type	No	Select the type of client authentication from the drop-down list. The available types are: • **optional** • **required**

Field	Module Property	Description
		Note: Avoid using the **optional** option in the production environment as it is vulnerable to attacks.
Keystore Provider as Trust Store	No	The name of a keystore provider resource instance that maintains a keystore that confirms an identity.

Advanced SSL Server Configuration

Use the following fields for advanced SSL server configuration.

Field	Module Property	Description
SSL Security Provider	Yes	Optional. The SSL security provider. **Note:** This is the name for the JSSE's cryptographic provider implementing SSLContext. If you are using non-default providers, such as PDCS#11 ones, you might want to override it.
SSL Protocol	No	The SSL protocol to use in the SSL connection. Select from the following options: • TLSv1 • TLSv1.1 • TLSv1.2 • TLSv1.3 • SSLv3–Use of this protocol is discouraged. The default value is TLSv1.2. Selecting a protocol implies the support of higher versions as well.

Field	Module Property	Description
		Note: • TSLv1.3 as SSL Protocol is supported for HTTP, REST, SOAP, Mail, and TCP Palettes. • TLSv1.3 supports only RSA certificates.
SSL Cipher Class	No	The number of bits in the key used to encrypt data: • **No Exportable Ciphers** • **All Ciphers** • **At Least 128 Bit** • **More Than 128 Bit** • **At Least 256 Bit** • **FIPS Ciphers** • **Explicit Ciphers**
Explicit Cipher List	Yes	A list of ciphers. Enabled when **SSL Cipher Class** is set to **Explicit Ciphers**. Use the JSSE format for ciphers names.
Verify Remote Host Name	No	Indicates whether the name on the server's certificate must be verified against the server's host name. If the server's host name is different than the name on the certificate, the SSL connection fails. You can verify the name on the certificate against another name by specifying **Expected Remote Hostname**. Selecting this checkbox displays the **Expected Remote Hostname** field. Default: This checkbox is deselected.
Expected Remote Hostname	Yes	Optional. The expected name of the remote host. The default is None.

SSL Client Configuration

You can specify SSL parameters for the HTTP client shared resource here.

SSL Client Configuration

The **General** section in the SSL Client Configuration has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

Basic SSL Client Configuration

This section has the following fields.

Field	Module Property	Description
Keystore Provider As Trust Store	No	The name of a keystore provider resource instance that maintains a keystore that confirms an identity.
Enable Mutual Authentication	No	Indicates whether the client in the SSL connection authenticates to the server. Select this checkbox to enable the identity fields.
Identity Store Provider	No	The name of the keystore provider resource that maintains a keystore used to assert an identity.
Key Alias Name	Yes	The name of the alias used to access the identity.
Key Alias Password	Yes	The password for the alias.

Advanced Client SSL Configuration

This section contains the following fields.

Field	Module Property	Description
SSL Security Provider	Yes	Optional. The SSL security provider. **Note:** This is the name for the JSSE's cryptographic provider implementing SSLContext. If you are using non-default providers, such as PDCS#11, you might want to override it.
SSL Protocol	No	The SSL protocol to use in the SSL connection: - TLSv1 - TLSv1.1 - TLSv1.2 - TLSv1.3 - SSLv3 - Use of this protocol is discouraged. The default value is TLSv1.2. Selecting a protocol implies the support of higher versions as well. **Note:** - TSLv1.3 as SSL Protocol is supported for HTTP, REST, SOAP, Mail, and TCP Palettes. - TLSv1.3 supports only RSA certificates.
SSL Cipher Class	No	The number of bits in the key used to encrypt data: **No Exportable Ciphers** **All Ciphers** **At Least 128 Bit**

471 | Shared Resources

Field	Module Property	Description
		• **More Than 128 Bit** • **At Least 256 Bit** • **FIPS Ciphers** • **Explicit Ciphers** The greater the number of bits in the key (cipher strength), the more possible key combinations and the longer it takes to break the encryption. The default is At Least 128 Bit.
Explicit Cipher List	Yes	A list of ciphers. Enabled when **SSL Cipher Class** is set to **Explicit Ciphers**. Use the JSSE format for ciphers names.
Verify Remote Host name	No	Indicates whether the name on the server's certificate must be verified against the server's host name. If the server's host name is different than the name on the certificate, the SSL connection fails. The name on the certificate can be verified against another name by specifying **Expected Remote Hostname**. Selecting this checkbox displays the **Expected Remote Hostname** field. Default: This checkbox is deselected.
Expected Remote Hostname	Yes	Optional. The expected name of the remote host. The default is None.

Subject Provider

The Subject Provider resource provides support for authenticating user credentials through a keystore or Trust Store.

TIBCO BusinessWorks™ Container Edition Binding and Palette Reference

472 | Shared Resources

General

The **General** section has the following fields.

Field	Description
Package	The name to be displayed as the label of the policy resource package.
Name	The name of the policy resource.
Description	A short description of the policy resource.

Subject Provider

The **Subject Provider** section has the following fields.

Field	Literal Value/Module Property	Description
Credential Provider for Identity Store	None	Refers to a keystore Provider resource. The **Keystore Provider** maintains the keystore used to assert an identity.
Key Alias Name	Yes	The name of the alias used to access the identity.
Key Alias Password	Yes	The password for the alias.
Credential Provider for Trust Store	None	Optional. Select a **Trust Provider** resource.

Security

The **Security** section has the following fields.

TIBCO BusinessWorks™ Container Edition Binding and Palette Reference

Field	Literal Value/Module Property	Description
SOAP Actor	Yes	**SOAP Actor** field can be used to provide the SOAP actor attribute to address the SOAP header element to a specific endpoint.
Protect Token	None	**Protect Token** checkbox can enable or disable signing of Binary Security Token. By default, the checkbox is selected.

TCP Connection

The TCP Connection shared resource specifies the connection information for the TCP server. This resource is used when a process acts as a TCP client connecting to a remote server or when a process acts as a TCP server accepting incoming TCP.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a shared resource.
Name	The name of the shared resource.
Description	A short description of the shared resource.

TCPConnection

This section has the following fields.

Field	Literal Value/Module Property	Description
Host	Yes	When a process acts as a client, this field specifies the host name or host IP address of the TCP server to connect to. When a process acts as a server, this field specifies the hostname or IP address of the machine where the process engine is running. You can specify localhost, or if the machine has more than one network interface card, you can specify the IP address of the card you want to use to accept the TCP/IP connections.
Port	Yes	This field specifies the port number on which a TCP server is listening for requests.
Enable Connection Pool	Yes	Selecting this checkbox enables the **When Exhausted Connections**, **Maximum Connections**, **Maximum Wait Time**, and **Idle Timeout** fields.
When Exhausted Connections	Yes	When the connections are exhausted on the server, select any one option from the following available options. • **Block**: the pool is blocked when the pool is exhausted, that is, the maximum number of active objects has been reached, until a connection is available, or the maximum wait time has been reached. • **Fail**: when the pool is exhausted, that is, the maximum number of connections have been consumed, it fails. • **Grow**: when this policy is selected a new connection is created for every request of the client, therefore there is no maximum connection limit.

Field	Literal Value/Module Property	Description
Maximum Connections	Yes	Specifies the maximum number of simultaneous client sessions that can connect with the server. This parameter is enabled only if connection pooling is enabled, that is, the **Enable Connection Pool** checkbox is selected. The default value is 10.
Maximum Wait Time (msec)	Yes	Specifies the maximum wait time in milliseconds to connect to the TCP server. This parameter is enabled only if connection pooling is enabled. That is, the **Enable Connection Pool** checkbox is selected. The default value is 10000.
Idle Timeout (msec)	Yes	Specify the idle timeout for the connections in milliseconds. The default value is 1.

Security

Select the **Confidentiality** checkbox to encrypt or decrypt messages. When you select the checkbox, the SSL Client Configuration field is visible.

For more information, see SSL Client Configuration.

Trust Provider

The Trust Provider resource maintains the identity of a trusted resource.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a policy resource.
Name	The name of the policy resource.
Description	A short description of the policy resource.

Trust Configuration

The **Trust Configuration** section has the following fields.

Field	Description
Credential Store used for Trust	Refers to a keystore Provider. The keystore is a trust store.
Enable Trust Store Access	Select this option to enable access to the specified trust store.

WSS Authentication

A WS-Security ASP resource template enables a connection to Web Services Security authentication services.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a policy resource.
Name	The name of the policy resource.
Description	A short description of the policy resource.

Basic Configuration

The **Basic Configuration** section has the following fields.

Field	Description
Security Token	Security Token is an online security credential that adds an extra layer of identity protection. **X.509** token is the default selection. X.509 is an ITU-T standard for a public key infrastructure (PKI) and Privilege Management Infrastructure (PMI).
Enable Signature Verification	Select to indicate whether to verify the signatures. By default, this option is not selected. If selected, it activates the **Trust Provider** field.
Enable Decryption	Select to indicate whether to enable decryption. By default, this option is not selected. If selected, it activates the **Subject Provider** field. **Note:** Select both **Enable Signature Verification** and **Enable Decryption** to activate the **Subject Provider (with Trust Credential)** field.

User Authentication

The **User Authentication** section has the following fields.

Field	Description
Select the type of authentication	Choose to authenticate users through **XML File Authentication** or **LDAP Authentication**.
XML File Resource	The name of the **XML Authentication** shared resource used to authenticate user credentials.
LDAP Resource	The name of the **LDAP Authentication** shared resource used to authenticate user credentials.

XML Authentication

The **XML Authentication** resource provides support for XML file-based authentication.

General

The **General** section has the following fields.

Field	Description
Package	The name of the package in which you want to create a policy resource.
Name	The name of the policy resource.
Description	A short description of the policy resource.

XML Authentication Configuration

The **XML Authentication Configuration** section has the following fields.

Field	Literal Value/Module Property	Description
XML File	Yes	The XML File containing user credentials and group mappings. The group mappings are used in the Authorization by role in the **Basic Authentication** policy resource.

Notify Configuration

The **Notify Configuration** specifies a schema to use for passing data between running process instances. The corresponding **Receive Notification, Notify** and **Wait for Notification** activities use the same **Nofity Configuration** to define the data for inter-process communication. The schema can be empty, if you do not want to pass data between processes.

General

The **General** tab has the following fields.

Field	Description
Package	The name of the package in which you want to create a Configuration resource.
Name	The name of the configuration resource.
Description	A description of the configuration resource.

Notify Configuration Editor

The **Notify Configuration Editor** is used to define a custom schema. The schema can be empty, if you do not want to pass data between processes.

You can define your own datatype on this tab, and you can reference XML schema or ActiveEnterpise classes stored in the project. Once defined, the data specified on the tab appears on the **Input** or **Output** tab of the **Receive Notification, Wait for Notification** or **Notify** activity where this shared configuration resource is used.

Thread Pool

The thread pool is a queue of threads available to run a collection of tasks. Thread pools are used to improve performance when running large numbers of asynchronous tasks by reducing per task invocation overhead. This provides a means of bounding and managing the resources consumed when running a collection of tasks.

General

The **General** section has the following fields.

Field	Description
Name	The name to be displayed as the label for the resource.
Description	A short description of the resource.

Thread Pool

This section has the following fields.

Field	Literal Value/Module Property	Description
Core Pool Size	Yes	Must be greater than or equal to zero. The default is 5.
Max Pool Size	Yes	The maximum number of threads in the pool. Must be greater than zero and greater than or equal to core pool size. The default is 10.
Keep Alive Time (s)	Yes	The length of time an idle thread remains in the pool before being reclaimed, if the number of threads in pool is more than core pool size. The default is 30 seconds.
Autostart Core Threads	Yes	Indicates to create and start the core pool size threads when the thread pool is created. Normally core threads are created and started only when new tasks arrive.
Thread Pool Name Prefix	Yes	A string prefixed to the name of each thread.
Daemon	Yes	Select the checkbox to specify whether the threads can be started as a daemon or a user. The default is `false`.

Field	Literal Value/Module Property	Description
Priority	Yes	The default priority of the threads in the pool. The default is 5.
Rejection Policy	No	The policy applied when no thread is available to run a task: • BLOCKING: the task is blocked until a thread from the thread pool picks up this task. • CALLER runs: the task is run in the calling thread. • ABORT: the task is stopped and an exception is generated. The default is BLOCKING.

TCP Palette

The TCP palette provides activities and resources that can send and receive data using the TCP protocol. TCP is often the inherent communication layer for high-level protocols such as HTTP or FTP.

You could use the resources in the TCP palette to communicate with HTTP or FTP severs, but the activities in the HTTP palette or FTP palette are best suited for this purpose. Use this palette when communicating with TCP, when there is no corresponding TIBCO BusinessWorks Container Edition palette for the protocol. For example, using the TCP palette to handle the incoming and outgoing data from a custom application that communicates with other applications by way of TCP.

Using the TCP Palette Activities

You can use the TCP palette activities to create a general sequence of events in a process.

Procedure

1. Create a TCP Connection shared configuration resource that describes the connection information.

2. Depending on whether the process is a TCP client or server, one of the following occurs:

   a. A connection opens using the **TCP Open Connection** activity, when a process works as a TCP client communicating with a TCP server.

   b. The process starts either with a **TCP Receiver** process starter or has a **Wait for TCP Request** activity that waits for an incoming connection request, when a process works as a TCP server waiting for the incoming TCP requests.

3. Use the **Read TCP Data** and **Write TCP Data** activities to read and write data to the in process TCP connection.

4. After the process is complete, close the TCP connection using the **TCP Close Connection** activity.

Read TCP Data

Read TCP Data is an asynchronous activity that reads data from an open TCP connection. The connection must be opened by an activity or a process starter that ran previously in the process.

Activities that can open a TCP connection are: **TCP Receiver**, **TCP Open Connection**, and **Wait for TCP Request**. The activity that opens a TCP connection places a handle to the open connection in its connectionKey output element. The connectionKey is mapped to this activity's input.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Type	None	Type of data to be read. This can be either **Text** or **Binary**. Specifying **Text** in this field enables the **Separator** and **Encoding** fields. When **Binary** is specified, the **Bytes To Read** field is enabled.
Separator	Yes	The type of separator. This activity reads data from the TCP connection until the separator or the end of the data stream is encountered. Select **User-defined binary separator (specify comma-separated byte values)** from the drop-down list to enable the **User Defined Separator** field.
User-Defined	Yes	This field specifies the separator character to use.

Field	Literal Value/Module Property/Process Property	Description
Separator		When **Binary** is selected in the **Type** field, a user-defined separator can be: • A string representation of a single byte, or • A comma-separated byte values (for example, 0x0a, 0x0b, 0x0c). For valid string representations of byte values, refer to the Java documentation for the `decode(String n)` method of `Java.lang.Byte`.
Encoding	Yes	Specifying **Text** in the **Type** field, determines the character encoding of the text data in this field.
Bytes To Read	Yes	Specify the number of bytes to be read by this activity. This field is enabled when you select **Binary** in the **Type** field. Leaving this field blank makes the activity read until the end of the data stream is encountered.
Timeout (msec)	Yes	The time to wait (in milliseconds) for this activity to complete.

Description

Provide a short description for the Read TCP Data activity.

Input

The following is the input for the activity.

> ℹ **Note:** When **Text** is specified in the **Type** field, the **Input** tab displays **connection**, **encoding**, **separator**, and **timeout** elements. Specifying **Binary** in the **Type** field displays **connection**, **bytesToRead**, and **timeout** elements.

Input Item	Datatype	Description
connection	binary	The handle to the connection to read the data from. This connection is obtained from a previously ran activity in the process that opened the connection.
encoding	string	Specifies the character encoding of the text when reading the text data. This value overrides the value specified in the **Encoding** field in the **General** tab.
separator	string	An optional element. Specifies the boundary between the body elements after writing the body element to the TCP connection, while reading the text data. This element is available only when the User-Defined Separator option is selected on the **General** tab. This value overrides the value specified in the **General** tab.
bytesToRead	number	An optional element. Specifies the number of bytes to be read when reading the binary data. This value overrides the value specified in the **Bytes To Read** field in the **General** tab.
timeout	number	An optional element. The time to wait (in milliseconds) for this activity to complete.

Output

The following is the output of the activity.

Output Item	Datatype	Description
data	string	When the data is specified as text, this output element contains the text read from the TCP connection. The separator character if specified, is not included in the output.

Output Item	Datatype	Description
		When the data is specified as binary, this output element contains the binary data read from the TCP connection.
endOfStreamReached	boolean	This element is set to true, when the end of the data stream is reached.
		This is useful when the activity is in an iteration loop and the activity is configured to read a number of bytes with each iteration. Use this element in the condition of the loop, to determine when the last set of bytes was read.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
TCPActivityFault	There is some problem associated with the activity at runtime.

TCP Close Connection

TCP Close Connection is a synchronous activity that closes a TCP connection opened by a previously ran activity or a process starter.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.

Description

Provide a short description for the TCP Close Connection activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
connection	binary	The handle to the connection that you want to close. This value is obtained from a previously ran activity in the process that opened the connection.

Output

The activity has no output.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
TCPActivityFault	There is some problem associated with the activity at runtime.

TCP Open Connection

TCP Open Connection is a synchronous activity that opens a connection to a TCP server. After establishing the connection, the activity places a handle to the open connection in the connection output element. The subsequent activities in the process use this connection key for the TCP Connection to read data from, write data to, or close the connection.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
TCP Connection	Yes	The **TCP Connection** is a shared configuration resource that specifies the connection information for the TCP server. Specify the required value in the TCP connection property.
Local Host	None	Specifies the IP address on the local machine to be used as the TCP client.
Timeout	None	Specifies a timeout value for the **TCP Connection**. • Default value: 0. • Lower limit: 1. • Upper limit: default OS timeout value. **Note:** For the default value and timeout value higher than the default OS timeout value, the timeout occurs within the OS default timeout.

Description

Provide a short description for the TCP Open Connection activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
connection	complex	An optional element. This element contains the information about the TCP server that you want to connect to. It comprises **host**, **port**, and **localhost** elements.
host	string	An optional element. The name or IP address of the TCP server you want to connect to. This value overrides the value specified by the TCP connection property in the **General** tab.
port	number	An optional element. The port number on which the TCP server is listening for requests. This value overrides the value specified by the TCP connection property in the **General** tab.
localhost	string	An optional element. The name or the IP address of the TCP client.

Output

The following is the output of the activity.

Output Item	Datatype	Description
connection	binary	The handle to the open connection. The subsequent activities in the process use this connection key to read data from, write data to, or close the connection.

490 | TCP Palette

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
TCPActivityFault	There is some problem associated with the activity at runtime.

TCP Receiver

TCP Receiver is a process starter activity that starts the process when a client requests a TCP connection. The connection element in the activity output contains the handle to the TCP connection. The subsequent activities in the process can use this handle for the TCP connection to read data from, write data to, or close the connection.

General

The **General** has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.
Security	Yes	To be included when encrypting or decrypting the messages. The drop-down list displays the SSL Server Configuration field. Use this field to specify the SSL parameters. For more information, see SSL Server Configuration.
Host	Yes	The hostname or the host IP address of the TCP server to

TIBCO BusinessWorks™ Container Edition Binding and Palette Reference

Field	Literal Value/Module Property/Process Property	Description
		connect to.
Port	Yes	Specify the port number on which this TCP server is listening for requests.
Enable DNS Lookup	None	Select this checkbox to enable a domain name system (DNS) lookup for resolving the IP address to a DNS name. **Caution:** Select this element only when required, as this may adversely affect throughput.
Keep Alive	None	Select this checkbox to specify for the TCP connections to use the keep alive feature of sockets.

Description

Provide a short description for the TCP Receiver activity.

Advanced

The **Advanced** tab contains the following fields.

Field	Description
Sequence Key	This field can contain an XPath expression that specifies which processes should run in a specified order. Process instances with sequencing keys evaluating to the same value are ran sequentially in the order of the process instance creation.
Custom Job Id	This field can contain an XPath expression that specifies a custom ID for the process instance.

Output

The following is the output of the activity.

Output Item	Datatype	Description
connection	binary	The handle to the open connection. This connection can be used by subsequent activities in the process definition to specify the TCP connection to write data to. Read data from or close.
IP	string	The IP address of the client sending the TCP request.
Host	string	The host name of the client sending the TCP request. This field is available only when you select the **Enable DNS Lookup** field in the **General** tab.
Port	integer	The port number of the client sending the TCP request.

Write TCP Data

Write TCP Data is an asynchronous activity that sends data on the specified TCP connection. The connection must be opened by an activity or process starter that ran previously in the process.

General

The **General** tab has the following fields.

Field	Literal Value/Module Property/Process Property	Description
Name	None	The name to be displayed as the label for the activity in the process.

Field	Literal Value/Module Property/Process Property	Description
Type	None	The type of the data to write. This can be either **Text** or **Binary**.
		Specifying **Text** in this field displays the **Separator** and **Encoding** fields.
		Specifying **Binary** in this field displays the **User Defined Separator** field.
Separator	Yes	The type of separator to use. The separator is written to the TCP connection after the specified data is written.
		Select **User-defined separator** from the drop-down list to enable the **User Defined Separator** field.
User-Defined Separator	Yes	This field specifies the separator character to use.
		When **Binary** is selected in the **Type** field, a user-defined separator can be a string representation of a single-byte value or it can be comma-separated byte values (for example, 0x0a, 0x0b, 0x0c). For valid string representations of byte value, refer to the Java documentation for the decode(String n) method of java.lang.Byte.
Encoding	Yes	When text data is specified, this field determines the character encoding to use for the text data.

Description

Provide a short description for the Write TCP Data activity.

Input

The following is the input for activity.

Input Item	Datatype	Description
connection	binary	The handle to the connection that you want to write data to. This connection is obtained from a previously ran activity in the process that opened the connection.
data	string or binary	The data to write to the TCP connection. The data type of this field is either **String** or **Binary**, depending on the data selected in the **Type** field in the **General** tab.
encoding	string	An optional element. When writing text data, this specifies the character encoding of the text. This value overrides the value specified in the **Encoding** field in the **General** tab.
separator	binary	An optional element. This is available only when you select the **s** option on the **General** tab.
timeout	number	An optional element. The time to wait (in milliseconds) for this activity to complete.

Output

This activity has no output.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When.
TCPActivityFault	There is some problem associated with the activity at runtime.

XML Activities Palette

The XML Activities palette provides activities for parsing XML strings into schemas and rendering schemas into XML strings.

Parse XML

Parse XML is a synchronous activity that takes a binary XML file or an XML string and converts it into an XML schema tree based on the XSD specified.

Parsing Date and Datetime Strings

In XML documents parsed by the **Parse XML** activity, datetime values are read in accordance with the ISO-8601 standard, as described in the XML Schema specification. For example, the value:

2002-02-10T14:55:31.112-08:00

is 55 minutes, 31 seconds, and 112 milliseconds after 2.00 p.m. on February 10, 2002 in a time zone that is 8 hours, 0 minutes behind UTC.

If no time zone field is present, the value is interpreted in the time zone of the machine that is performing the parsing. This can lead to complications if you are processing XML from a different time zone, so you are encouraged to always use a time zone.

General

The **General** tab has the following fields.

Field	Description
Name	The name to be displayed as the label for the activity in the process.
Input Style	Can be **Binary**, **Text**, or **Dynamic**.

Field	Description
	In **Binary mode,** the binary content is read. The encoding used for parsing the content is either the value specified in the forceEncoding input item, the encoding specified in the XML header or the xmlBinary input item, or UTF-8 (the default encoding, if no encoding is specified).
	In **Text mode**, (the default and preferred choice) an XML string is passed as an input item.
	In **Dynamic mode**, a choice is offered for input. You can either supply **binary** or **text** input. You can use a choice statement and set substitution in the mapping to supply the correct type of input at run time.
Validate Output	When selected, specifies that the output of the activity should be validated against the schema specified in the **Output Editor** tab.

Description

Provide a short description for the Parse XML activity.

Output Editor

Use the **Output Editor** tab to define a schema for the XML output.

Input

The following is the input for the activity.

Input Item	Datatype	Description
xmlString	string	The XML string to parse.
		This input element is available when the **Input Style** field in the **General** tab is set to **Text**.
xmlBinary	complex	Available when you set the **Input Style** field in the **General** tab to **Binary**.
		This input item contains a required bytes item for

Input Item	Datatype	Description
		which you specify the input XML bytes.
		This input item also contains an optional forceEncoding item. This specifies the encoding to use when parsing the XML. If this element is not specified, the encoding specified in the XML header of the byte element is used to parse the XML. If the encoding value is not specified in the XML header, the default encoding (UTF-8) is used.
xmlBinary|xmlString	choice	Available when the **Input Style** in the **General** field is set to **Dynamic**. You can specify a choice statement and set substitution and supply the correct input type at run time.

Output

The following is the output of the activity.

Output Item	Datatype	Description
XML Schema	complex	The parsed XML schema is the output of this activity. The contents of the schema are determined by the schema defined in the **Output Editor** tab.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
XMLParseException	An exception occurred when parsing the data.
UnsupportedEncodingException	An unsupported encoding was specified in the forceEncoding input item.
MissingByteCountException	An exception occurred while parsing the data with an invalid or missing byte count.
ValidationException	The XML data supplied to the parser was invalid with respect to its governing XML Schema.

Render XML

Render XML is a synchronous activity that takes an instance of an XML schema element and renders it as a stream of bytes containing XML or an XML string.

The **Render XML** activity takes an instance of an XML schema element and renders it as a stream of bytes containing XML or an XML string. The schema is processed based on the XSD file specified.

Rendering Date and Datetime Strings

UTC time is used when datetime strings are generated. For example, the time 55 minutes, 31 seconds, and 112 milliseconds after 2.00 p.m.on February 10, 2002 would be represented by the **Render XML** activity as 2002-02-10T14:55:31.112Z.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	No	The name to be displayed as the label for the activity in the process.
Output Style	No	Specifies whether the output should be text or binary data.
Validate Input	No	When selected, specifies that the data input to the activity should be validated against the schema specified in the **Input Editor** tab. By default, the checkbox is clear.
Format Using Default Namespace Prefix	No	Prepends the namespace with the default namespace prefix.
Disable Pretty Print	No	Returns output data in the form of non-formatted XML. By default, the checkbox is clear.
Suppress XML Declaration	No	Removes XML declaration in the beginning of an XML document of the form `<?xml version = "1.0" encoding = "UTF-8" standalone = "no" ?>`. By default, the checkbox is clear.
Encoding	Yes	This field is visible only when the **Output Style** field is first set to **Binary**, then set to **Text**. This field specifies the encoding value to place into the XML header of the rendered XML data. You can specify any valid XML encoding value listed in IANA-Charsets.

Description

Provide a short description for the Render XML activity.

Advanced

Click the **Advanced** tab and specify which elements in the input schema can contain CDATA sections. Using CDATA sections you can enclose text within an element that should not be treated as XML.

Click the **Add** button to add elements of the input schema to the list. When you click the **Add** button, a dialog displays. Using this dialog locates and selects the input schema by location in the project or by namespace. Use the **By Location** tab if your input schema is located in the project directory. Use the **By Namespace** tab if you constructed the input schema on the **Input Editor** tab.

Use the **X** button to remove the elements from the list and the up and down arrows to move elements in the list.

Input Editor

From the **Input Editor** tab, you can define a schema for the incoming XML data you want to render.

Input

The following is the input for the activity.

Input Item	Type	Description
byteEncoding	string	This is only available when you set the **Output Style** field in the **General** tab to **binary**.
		This specified the encoding value to place into the XML header in the rendered XML output. This is also the encoding used to encode the binary data. If this item is not specified, the default encoding of the java virtual machine used by the process engine is used to encode the binary data. This value may not be a valid XML encoding.

Input Item	Type	Description
		You can specify any valid XML encoding value listed in IANA-Charset. For a complete list, refer to http://www.iana.org/assignments/character-sets. The encoding names are case-sensitive.
XML Schema	string	The XML schema to render. The contents of the schema are determined by the schema specified in the **Input Editor** tab.

Output

The following is the output of the activity.

Output Item	Type	Description
xmlString	string	Available when **Text** is selected in the **Output Style** field in the **General** tab. This item contains a stream of bytes representing the rendered XML.
xmlBytes	bytes	Available when **Binary** is selected in the **Output Style** field in the **General** tab. This item contains a stream of bytes representing the rendered XML.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When..
XMLRenderException	An exception occurred when rendering the data.
UnsupportedEncodingException	An unsupported encoding was specified in the byteEncoding input item.

Fault	Generated When..
MissingByteCountException	An exception occurred while rendering the data with an invalid or missing byte count.

Transform XML

Transform XML is a synchronous activity that invokes the built-in XSLT processor to apply an XSLT style sheet to an XML source document and produce a result document.

Normally, transformation of data occurs by mapping process variables to an activity's input and applying XPath expression to perform any transformation. If you have an XSLT file that you are using for transformation, or if an outside source supplies an XSLT file, this activity helps you to use the XSLT file instead of manually creating the mappings.

For more information about creating and editing XSLT files, see the XSLT specifications at http://www.w3.org/TR/xslt.

General

The **General** tab has the following fields.

Field	Literal Value/Process Property/Module Property	Description
Name	No	The name to be displayed as the label for the activity in the process.
Style sheet	Yes	The XSLT file shared configuration resource to use when transforming the XML. You can also specify an XSLT Style sheet in the Style sheet input element, if desired. Style sheets specified in the activity input override the Style sheet specified in this field.
Disable	No	Returns output data in the form of non-formatted XML.

Field	Literal Value/Process Property/Module Property	Description
Pretty Print		By default, the checkbox is clear.
Engine Type	No	In this dropdown, you can select either the TIBCO XSLT 2.0 and Saxon-B XSLT 2.0 for the transformation. The default option or engine used is TIBCO XSLT 2.0. • TIBCO XSLT 2.0 - This is the default XSLT engine provided by TIBCO. • Saxon-B XSLT 2.0 - XSLT 2.0 compliant engine provided by Saxon.
Input and Output Style	No	Specifies whether the input and output is **Text** or **Binary**.

Description

Provide a short description for the Transform XML activity.

Input

The following is the input for the activity.

Input Item	Datatype	Description
xmlString	text or binary	The XML to transform. If the **Input and Output Style** is set to binary, this element is named xmlBytesIn. If the **Input and Output Style** is set to text, this element is named xmlString.
styleSheet	string	This optional element contains the XSLT Style sheet to use to transform the XML input. This Style sheet overrides

Input Item	Datatype	Description
		the Style sheet specified in the **styleSheet** field in the **General** tab.
parameter	complex	This repeating element contains the input parameters for the Style sheet. Each input parameter is specified as name/value pairs. The name is a string that corresponds to the name of the parameter specified in the XSLT Style sheet parameter list.
name	string	The name of the Style sheet input parameters.
value	string	The value to supply for the specified Style sheet input parameters.
isXMLDocument	boolean	Specifies whether the parameter is an XML document.

Output

The following is the output for the activity.

Output Item	Datatype	Description
xmlString	text or binary	A binary or text value. The element is named **xmlString** and is of type **Text** when you select **Text** for the **Input and Output Style** on the **General** tab. The element is named **xmlOutput** and is of type binary when you select **Binary** for the **Input and Output Style** on the **General** tab.

Fault

The **Fault** tab lists the possible exceptions generated by this activity. For more information about error codes and the corrective action to take, see the *TIBCO BusinessWorks Container Edition Error Codes*.

Fault	Generated When.
XMLTransformException	An exception occurred when transforming the data.
UnsupportedEncodingException	Data was supplied that is encoded in an unsupported encoding.
MissingByteCountException	An exception occurred while transforming the data with an invalid or missing byte count.

Activity Icons Reference

Basic Activities Palette		
Critical Section	Constructor	For Each
Iterate	Repeat	Repeat on Error
Scope	While	Receive
Compensate	Empty	Exit
Get Context	Rethrow	Invoke
		Set EPR
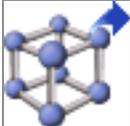
Basic Activities Palette

Reply   Set Context

Throw   Start   End

General Activities Palette

Assign

Confirm   Mapper

Get Shared Variable   Inspector   Log

Set Shared Variable   Sleep   Timer

Call Process

Notify   Wait for Notification

File Palette

Copy File	Create File	File Poller	List Files
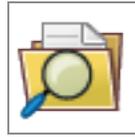
Read File	Remove File	Rename File	Write File
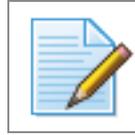
FTP Palette

FTP Change Default Directory	FTP Delete File	FTP Dir

FTP Get	FTP Get Default Directory	FTP Make Remote Directory

FTP Put	FTP Quote	FTP Remove Remote Directory

FTP Rename File	FTP SYS Type

HTTP Palette

HTTP Receiver   Send HTTP Request   Send HTTP Response

Java Palette

Java Invoke   Java Process Starter   Java To XML   XML To Java

JDBC Palette

JDBC Call Procedure   JDBC Query   JDBC Update   SQL Direct

JMS Palette

Get JMS Queue Message   JMS Receive Message   JMS Send Message

Reply to JMS Message   RequestReply

Mail Palette

Receive Mail   Send Mail

Parse Palette

MiimeParser

Parse Data   Render Data

REST and JSON Palette

Invoke REST API   Parse JSON   Render JSON
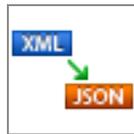
TCP Palette
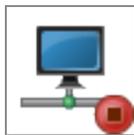
Read TCP Data   TCP Close Connection   TCP Open Connection   TCP Receiver

Write TCP Data

XML Activities Palette

Parse XML   Render XML   Transform XML
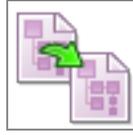
TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

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

The Product Documentation website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation for this product is available on the TIBCO BusinessWorks™ Container Edition page:

- *TIBCO BusinessWorks™ Container Edition Release Notes*
- *TIBCO BusinessWorks™ Container Edition Installation*
- *TIBCO BusinessWorks™ Container Edition Application Development*
- *TIBCO BusinessWorks™ Container Edition Application Monitoring and Troubleshooting*
- *TIBCO BusinessWorks™ Container Edition Bindings and Palettes Reference*
- *TIBCO BusinessWorks™ Container Edition Concepts*
- *TIBCO BusinessWorks™ Container Edition Error Codes*
- *TIBCO BusinessWorks™ Container Edition Getting Started*
- *TIBCO BusinessWorks™ Container Edition Maven Plug-in*
- *TIBCO BusinessWorks™ Container Edition Migration*
- *TIBCO BusinessWorks™ Container Edition Performance Benchmarking and Tuning*
- *TIBCO BusinessWorks™ Container Edition REST Implementation*
- *TIBCO BusinessWorks™ Container Edition Refactoring Best Practices*

- *TIBCO BusinessWorks™ Container Edition Samples*

How to Contact Support for TIBCO Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our product Support website.

- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the product Support website. If you do not have a username, 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 TIBCO Community.

Legal and Third-Party Notices