TIBCO BusinessWorks™ Container Edition Getting Started

Software Release 2.4.4
May 2019
Important Information

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

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

ANY SOFTWARE ITEM IDENTIFIED AS THIRD PARTY LIBRARY IS AVAILABLE UNDER SEPARATE SOFTWARE LICENSE TERMS AND IS NOT PART OF A TIBCO PRODUCT. AS SUCH, THESE SOFTWARE ITEMS ARE NOT COVERED BY THE TERMS OF YOUR AGREEMENT WITH TIBCO, INCLUDING ANY TERMS CONCERNING SUPPORT, MAINTENANCE, WARRANTIES, AND INDEMNITIES. DOWNLOAD AND USE OF THESE ITEMS IS SOLELY AT YOUR OWN DISCRETION AND SUBJECT TO THE LICENSE TERMS APPLICABLE TO THEM. BY PROCEEDING TO DOWNLOAD, INSTALL OR USE ANY OF THESE ITEMS, YOU ACKNOWLEDGE THE FOREGOING DISTINCTIONS BETWEEN THESE ITEMS AND TIBCO PRODUCTS.

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

TIBCO, TIBCO ActiveMatrix BusinessWorks, TIBCO Rendezvous, TIBCO Enterprise Message Service, TIBCO Business Studio, TIBCO Enterprise Administrator, TIBCO ActiveSpaces, TIBCO Runtime Agent, TIBCO Designer, TIBCO Business Studio™ for BusinessWorks™, and Two-Second Advantage are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Enterprise Java Beans (EJB), Java Platform Enterprise Edition (Java EE), Java 2 Platform Enterprise Edition (J2EE), and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle Corporation in the U.S. and other countries.

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

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Please see the readme.txt file for the availability of this software version on a specific operating system platform.

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

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

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

**Figures** ............................................................................................ 5  
**TIBCO Documentation and Support Services** ........................................................ 6  
**Orientation** ........................................................................................ 7  
  TIBCO Business Studio Container Edition ........................................................... 7  
  Application Development ........................................................................................... 8  
  Web Services ........................................................................................................... 8  
  Shared Resources ..................................................................................................... 9  
  REST Support ........................................................................................................ 9  
    REST Documerter and Tester .................................................................................. 10  
    Discovering API Models from TIBCO Business Studio™ for BusinessWorks™ ........................................ 11  
    Importing an API Model into your Workspace .................................................... 12  
  Archive Files ......................................................................................................... 13  
  Debugger ............................................................................................................... 13  
  Runtime .................................................................................................................. 15  
  Changing Help Preferences ................................................................................... 15  
**REST Service Tutorial** ............................................................................ 17  
  Creating a Service Instance of Cloud Foundry managed PostgreSQL Database Service ........................................................................................................... 17  
  Importing a Process Package ................................................................................ 17  
  Building a REST Service ...................................................................................... 20  
  Testing the REST Service in Cloud Foundry ....................................................... 23  
    Testing the POST and GET Operations ............................................................ 25  
  Testing the REST Service in OpenShift .............................................................. 26  
  Troubleshooting .................................................................................................. 29  
**REST Reference Tutorial** ................................................................. 30
Figures

Shared Resources Folders in Project Explorer ................................................................. 9
REST Service Wizard ................................................................................................... 10
Swagger UI .................................................................................................................. 10
Export Project Dialog .................................................................................................. 13
Debug Perspective ....................................................................................................... 14
Console View ............................................................................................................... 14
Stop Icon in Debug Perspective .................................................................................. 14
Console View ............................................................................................................... 15
How to Access TIBCO Documentation

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

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

Product-Specific Documentation

Documentation for TIBCO products is not bundled with the software. Instead, it is available on the TIBCO Documentation site.

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

- Concepts
- Installation
- Getting Started
- Application Development
- Bindings and Palettes Reference
- Samples
- Error Codes
- Migration
- REST Implementation
- Application Monitoring and Troubleshooting
- Performance Benchmarking and Tuning

How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

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

How to Join TIBCO Community

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

TIBCO BusinessWorks™ Container Edition is an integration product suite for enterprise, web, and mobile applications.

The software allows you to create services and integrate applications using a visual, model-driven development environment, and then deploy them in the TIBCO BusinessWorks™ Container Edition runtime.

This product uses Eclipse-based graphical user interface (GUI) provided by TIBCO Business Studio™ Container Edition to define business processes and generate deployable artifacts in the form of archive files. These deployable artifacts can be:

- deployed and run in the product runtime, and
- managed using the administration command line console, bwadmin, or the web-based Admin UI.

For information about developing applications and TIBCO Business Studio™ Container Edition, see the following guides in this documentation set:

- Application Development
- Palette Reference
- Samples

**TIBCO Business Studio Container Edition**

TIBCO Business Studio™ Container Edition is the design-time IDE (based on Eclipse) where you create and test TIBCO BusinessWorks™ Container Edition processes.

You use TIBCO Business Studio™ Container Edition for end-to-end application development. You can create new services, orchestrate business process, and integrate applications in a short time. A model-driven development approach is supported, with a rich set of palettes for process design. These palettes can be used to visually create and test business processes that connect to various technologies such as database, messaging servers, and so on.


To open TIBCO Business Studio™ Container Edition:

- On Unix: Select the TIBCO Business Studio™ Container Edition executable located at: 
  
  \$TIBCO_HOME/studio/4.0/eclipse/

- On Windows: Start > All Programs > TIBCO > TIBCO_HOME > TIBCO Business Studio Container Edition for Designers

On the Workspace Launcher dialog, accept the default workspace or browse to create a new workspace, and then click OK. TIBCO Business Studio™ Container Edition is started and the default
Development environment, called a *workbench*, appears. The user can access the samples by clicking the listed samples on the welcome screen.

For more information see the *Application Development* guide.

**Application Development**

Applications solve integration problems of varying complexity. In TIBCO BusinessWorks™ Container Edition, applications can be developed using an application-oriented integration style or a service-oriented integration style. How you design your application's integration style will depend on the following factors:

- Speed of integration
- Data abstraction
- Richness of operation primitives
- Typical endpoints

For more information about an application's integration style and other application design considerations, see the *Application Development* guide.

Processes allow you to implement business logic that can obtain and manage the flow of information in an enterprise between a source and different destinations. In process-driven design, the business processes or integration flows are first realized and captured. For more information about process design, see the *Application Development* guide.

Processes are developed in TIBCO Business Studio™ Container Edition and are saved in application modules. Application modules are equivalent to projects and are saved to folders on the disk. The TIBCO Business Studio™ Container Edition workspace contains one or more application modules.

- An application module contains one or more BusinessWorks packages
- A BusinessWorks package contains one or more BusinessWorks processes, which in turn are main processes or subprocesses
- A process is stored as a single file with a .bwp extension

An application module contains a special folder called *Processes*. This folder contains the user created processes. In addition, an application module also contains special folders to store WSDL files, schemas, and shared resources.

A package should follow the Java naming convention.

Processes are designed in the *Process Editor*. Activities and shared resources help you rapidly design business processes. An activity is the individual unit of work in a process. There are multiple ways to add an activity to a process: from the right-click menu on the *Process Editor*, from the palettes, and from the *File Explorer* or *Project Explorer*. To add an activity from the palette, select it and drop it on the *Process Editor*.

Implemented services are shown as chevrons on the left side of the *Process Editor*. Any references that are invoked are shown on the right side of the *Process Editor*. For a simple process, services and references are optional.

**Web Services**

Web services are application components that communicate using open standard protocols. You can develop SOAP-based web services using the Generate Concrete WSDL Wizard. The wizard generates a WSDL file and the appropriate response activities. You can develop REST-based web services using the REST Service Wizard in TIBCO BusinessWorks™ Container Edition.

Select a WSDL file in the *Project Explorer* and drop it on the *Process Editor* to implement a web service. Dropping the WSDL file displays a menu for creating services or implementing operations. Response activities are automatically generated.
To create a REST service, select a path under the `.json` file in the Service Descriptors folder and drop it on the Process Editor to implement a web service. When you drop the path, it displays a menu with an option to create a service or a reference.

**Shared Resources**

Shared resources are configurations that are shared among activities. These are resources such as database, JMS and HTTP connections, and connections to other servers. Resources are added to special folders in the Project Explorer. The following image shows these folder in the Project Explorer.

*Shared Resources Folders in Project Explorer*

The following types of folders for shared resources can exist in a project.

- **Resources**: Contains shared resources used by activities in a process.
- **Schemas**: Stores XSD (schema) files.
- **Service Descriptors**: Stores WSDL and JSON files.

**REST Support**

The REST Service wizard is used to build RESTful services.

When you create a REST service, make sure to edit the Default Host field in the HTTP Connection Resource to reflect the actual host name. By default, the Default Host field is set to localhost.
REST Service Wizard

Developing a RESTful service is a simple three step process:

1. Name the REST resource.
2. Choose the resource definition (the XSD schema).
3. Choose the REST operations to implement.

The input and output messages for the operations are automatically generated along with a Response activity. An HTTP shared resource is also generated with the default configuration. You can then add activities and implement the business logic for each operation in the process.

REST Documenter and Tester

A REST documenter and tester is automatically generated for a REST resource. Refer to OSGi commands to List REST URLs in the REST Implementation guide. The documentation is based on the Swagger specification and is rendered using the Swagger UI.

Swagger UI

This Swagger based interface provides a convenient and quick way to:

- View REST endpoints and operations implemented by the REST resource service.
- Examine the inputs and outputs for each operation in JSON format.
- Enable Input fields to specify JSON or XML input for each operation.
- Invoke an operation and receive a live response for the input supplied.
Discovering API Models from TIBCO Business Studio™ for BusinessWorks™

You can use the API Explorer view in the TIBCO Business Studio for BusinessWorks to view the APIs that reside on your local machine or on a remote server.

**Prerequisites**

For the API Explorer to discover the APIs residing on a remote server, the remote server must be up and running.

You can set up the locations to which you want the API Explorer to connect and look for the APIs. To do so, follow the steps below.

**Procedure**

1. In TIBCO Business Studio for BusinessWorks, go to the API Explorer view.
2. In the button bar within the API Explorer tab, click the View Menu downward-facing triangle icon (itim) and select Settings. The Settings dialog will open.

   The registries for the TIBCO BusinessWorks Container Edition - API Modeler and the samples folder installed on your local machine are configured and appear in the API registry configurations box by default. In this dialog, you can specify how the discovered APIs will appear in the API Explorer:

   - **API Presentation** - specifies how the APIs will appear in the API Explorer
     - **Flat** - displays the APIs as a flat list with each API's version number displayed next to its name in parenthesis. If there are multiple versions of the same API, each version will be shown as a separate API, hence multiple APIs with the same name but different version numbers.
     - **Hierarchical** - displays every API as a hierarchy of API name label with version number folder under it and the actual API under the version folder. If there are multiple versions for an API, each version will be listed in its own separate folder under the API name label.
     - **Latest Version** - displays only the latest version of the API, even though there might be multiple versions available.

   - **Group by API registry** - groups the APIs according to the registry from which they were discovered

   - **API registry configurations** - displays the list of API registries that are currently configured in your TIBCO Business Studio for BusinessWorks installation.

     Select the API registry check boxes to display the APIs.

     You can edit an existing registry by clicking the Edit button, delete the registry configuration by clicking Remove, or changing the order in which the registries show up in the API Explorer by using the Up and Down button. These button get activated when you click on an API registry name.

3. Click New to add a new registry.
4. In the Create new API Registry client configuration dialog do the following:
   a) Enter a name for the API registry that you will be mapping to in the Name text box.
   b) Select the Local radio button to map a location where the APIs are stored on your local machine's hard drive and navigate to the location using the Browse button. Alternatively, select the Remote radio button if you want to map to a remote server that contains the APIs and enter the URL for the server in the URL text box.

5. Click Finish.

   You should now see the APIs displayed in the API Explorer in the format that you specified in the Settings dialog. Expanding an API will show you its version, the resource path, and the operations you can perform on that resource.
The API Explorer view has the following quick-access buttons that you can use to format the way the APIs are listed:

- ![Refresh](image)
- ![Expand All](image)
- ![Collapse All](image)
- ![Group by API Registry](image)
- ![API Presentation](image)
- ![API Registries](image)

Selecting a registry from this drop-down list toggles between displaying and hiding the registry in the API Explorer.

Use the search filter that appears at the bottom of the API Explorer view to search for API names that match the string that you enter in the Filter text box. You can search by typing in the version number, the full API name, or a full word within an API name. Wildcards are not supported. The search is case insensitive.

## Importing an API Model into your Workspace

The APIs that are discovered from local and remote servers are displayed in the API Explorer tab of the TIBCO BusinessWorks™ Container Edition. You can use these APIs in your project by importing them into the Service Descriptors folder of the project. The .json file for the API gets copied into the application module.

To import the APIs from the API Explorer into your project follow these steps.

### Procedure

1. Right-click on one or more API names in the API Explorer and select Import. The Import API dialog opens.
Every API you selected in the API Explorer is listed in this dialog. If an API has multiple versions, all versions are listed. By default, all APIs listed here are selected. You can deselect APIs that you do not want to import by clearing its check box.

2. Select the Import to project radio button to import the API into an existing project and browse to the project using the Browse button. To create a new project and import the API into that project select the Create a new project and import API to the new project radio button. This option walks you through the new project wizard, and after you go through the wizard will create a new project and import the API into its Service Descriptors folder.

3. Select the API or the appropriate version of the API should there be multiple versions of the API available in the API list to import box.

4. Click Finish.
You should see the API(s) under the Service Descriptors folder of the project. You can create sub-folders under the Service Descriptors folder and drag-and-drop APIs into them if you prefer to organize the APIs into a meaningful folder structure.

As an alternative to the above procedure, you can also drag and drop the API from the API Explorer into the project’s Service Descriptors folder.

APIs that were created using a Swagger file must be implemented exactly as defined by the Swagger file. TIBCO Business Studio for BusinessWorks allows you to only view the parameters and operations that are defined in the Swagger file. You cannot create any new parameters or operations for such applications.

Archive Files

After completing an application module, you must define an application to build a deployment archive file. An application defines all the processes, properties, and resources that must be included in the archive file. By default, all processes are included.

To create an archive, choose one of the following:

- Right-click the project in the Project Explorer and choose Export > Studio Projects to Archive. The Export Project dialog is displayed.

  Export Project Dialog

- Drag the project from the Project Explorer and drop it on a folder in the File Explorer. In both scenarios, and archive file is created with all required processes, properties, and resources. In the first scenario, you can name the archive file, select the format, and select the resources to include. In the second scenario, the archive is created for you in the format appropriate for your operating environment. All required elements are included.

Debugger

The TIBCO Business Studio™ Container Edition debugger is used to test processes during the process development stage. Starting the debugger brings up the Debug perspective. This perspective can be
used to set breakpoints, steps through processes, examine job variables, and activity input/output at each step.

**Debug Perspective**

The **Console** view displays the messages and errors returned by the runtime.

**Console View**

Start the debugger with the **Run > Debug** command. To stop the debugger, press the **Stop** icon on the **Debug** perspective toolbar.

**Stop Icon in Debug Perspective**
Runtime

You can run applications in TIBCO Business Studio™ Container Edition and test them in a runtime environment, which consists of a domain, an AppSpace, and an AppNode on your local machine. These runtime entities were created when you installed TIBCO BusinessWorks™ Container Edition. For more information about runtime entities, see the Concepts guide.

To run an application in TIBCO Business Studio™ Container Edition, choose the Run > Run command. (Applications can also be run with the Run > Run Configurations command. This option allows you to manage and launch run configurations.) The Run command opens the Console view where progress messages and errors are displayed.

Console View

Click the Businessworks Jobs view in the top left to see the jobs created for the process. To stop the current job, click the Stop button on the Console view toolbar.

From the Console view, you can use OSGi commands to monitor the running AppNode and gather metrics about your application. For information about OSGi commands, press Enter in the Console view to display the <@BWEclipseAppNode> prompt. Type help to get a list of commands.

The scope is indicated along with the command. Commands with the scope bw return information about the running application. Type a command name followed by -h for information about the command. For example, the command help bw:dsr returns:

dsr - Diagnoses Shared Resource issues
  scope: bw
  parameters:
  String Partial or full name of a Shared Resource. Case is ignored.

Changing Help Preferences

By default, documentation access from TIBCO Business Studio™ for BusinessWorks™ is online, through the TIBCO Product Documentation site (Doc site) at https://docs.tibco.com/ which contains the latest version of the documentation. Check the Doc site frequently for updates. To access the product documentation offline, download the documentation to a local directory or an internal web server and then change the help preferences in TIBCO Business Studio for BusinessWorks.

Prerequisites

Before changing the help preferences to access documentation locally or from an internal web server, download documentation from https://docs.tibco.com/.

1. Go to: https://docs.tibco.com/
3. Select the TIBCO BusinessWorks Container Edition product from the list. This opens the product documentation page for the latest version.
4. Click Download All.
5. A zip file containing the latest documentation downloads to your web browser’s default download location. Copy the zip file to a local directory or to an internal web server and then unzip the file.

To change help preferences on the Preferences dialog to access the documentation from a custom location:

**Procedure**

1. In TIBCO Business Studio for BusinessWorks, click **Window > Preferences**. On Mac OS X, click **TIBCO Business Studio BusinessWorks > Preferences**.
2. In the Preferences dialog, click **BusinessWorks > Help**.
3. Click **Custom Location** and then click **Browse** to select the `html` directory in the folder where you unzipped the documentation, or provide the URL to the `html` directory on your internal web server.
4. Click **Apply** and then click **OK**.
REST Service Tutorial

The REST Bookstore sample lets you explore the REST tooling in TIBCO Business Studio™ Container Edition. You can import this sample into TIBCO Business Studio™ Container Edition through File Explorer and examine the project and the solution implemented by it.

The processes in the sample implement different aspects of a bookstore, such as adding books, deleting a book, and getting a list of books or a single book by ISBN. For more information about the sample, see "Using REST to Manage Books for a Bookstore" in the Samples guide. This tutorial walks you through the steps to build an additional REST service for the sample and test it in the debugger. You can use the Swagger UI to invoke the operations for the REST resource.

Prerequisites

- Access to a locally running PostgreSQL database.
- The latest version of Google Chrome.

Creating a Service Instance of Cloud Foundry managed PostgreSQL Database Service

To bind an application to a managed PostgreSQL service already running in your Cloud Foundry environment complete the following steps:

Procedure

1. In your Cloud Foundry environment execute

   `cf services`

2. Execute

   `cf create-service postgresql default postgres`

Importing a Process Package

These steps show how to create a new process package.

Procedure

2. Open the Design perspective by clicking the icon in the upper right.
3. Click the File Explorer tab. If the tab is not visible, click Window > Show View > Other > FileSystem > File Explorer and click OK.
4. Click File > Switch Workspace and select or open a clean new workspace.
5. In the samples directory, select `cloudfoundry > binding > rest > Bookstore` and double-click `tibco.bwce.sample.binding.rest.BookStore.zip`. This opens the project in the Project Explorer.
6. In the Project Explorer, expand the `tibco.bwce.sample.binding.rest.BookStore` project.
7. You can also import the sample using the File > Import > General > Existing Studio Projects into Workspace > Select Archive File > Browse option.

8. The project is displayed in the Project Explorer panel on the left.

9. Expand the folders in the project to see all the project processes and resources. Refer to the Application Development guide for information about the folder structure.

10. Expand Processes and then expand tibco.bwce.sample.binding.rest.bookstore.db. You will see BooksDB.bwp.

    bwp is a BusinessWorks process.

11. Double-click BooksDB.bwp. The BusinessWorks process comprises:

    - Green chevron on the left indicates the service details.
    - addBooks, getAllBooks, and other operations implemented by this process.
    - Each operation is implemented separately.
12. Double-click an operation to display the process for example, **BooksPersist > addBooks**.
   a) In the addBooks operation, a JDBC activity is seen.
   b) The activity is repeated using a **ForEach** group.
   c) **addBooksOut** represents the **Response** to the web service request.

13. To add a new process package named **tibco.bwce.sample.rest**, right-click on **Processes** in the Project Explorer and select **New > BusinessWorks Package**.

14. In the BusinessWorks Package screen, specify **tibco.bwce.sample.rest** in the **Name** field.

15. Click **Finish** and verify that the new package **tibco.bwce.sample.rest** has been added in the Project Explorer.
Building a REST Service

This section details how to build a REST service.

Procedure

   The REST Service Wizard window opens.

2. Specify the following values in the REST Service Wizard window.
   a) **Resource Name**: MyBooks
   b) **Summary**: Summary about the new REST service. (default)
   c) **Resource Service Path**: Auto-filled
   e) **Operations**: Select POST and GET check boxes.
   f) **Implementation Data**: Accept the default value of Structured.

3. Click **Finish**.
   This creates a new process MyBooks.bwp process is opened in the Process Editor.

4. Open the tibco.bwce.sample.binding.rest.bookstore.db package in the Project Explorer and select the BooksDB.bwp process. Drag it to the Process Editor and drop it on the implemented POST operation.
   A menu is displayed with two options: Create Invoke Activity and Create Reference and Wire Process.

5. Select **Create References and Wire Process**.
The references are added to the process. The purple chevron indicates the service and its operations that can be referenced by the process.

6. To update the POST process to invoke the appropriate external service operation:
   a) Click the **addBooks** operation.
   b) Select and drag the operation to the left of the **postOut** activity and drop it. An Invoke process activity is created.

7. Click the newly added activity. Select the icon and connect **addBooks** to **postOut**.

8. Click the **getAllBooks** operation and select, drag, and drop the operation to the left of the **getOut** activity in the OUT process.

9. Connect **getAllBooks** to **getOut**.

10. Save your changes.

11. Click the **addBooks** activity and select Properties > Input.

12. Expand the data tree in the Data Source pane to locate the Book element.
13. Drag the Book element from the left to the Book* element on the right.

14. In the pop-up window, select **Make a Copy of each "Book"** and click **Finish**.

   The **Input** tab will look like this:

15. Save your changes.

16. Click the **postOut** activity and open the Properties > Input tab. Expand the **post** activity and drag the Book* element from left to right.

17. In the pop-up window, select the **For each** option and click **Next**. Click **Finish** on the Auto-Map window: The Properties > Input tab will look similar to this:

18. Click **getAllBooks** and select Properties > Input.

19. In the XPath Expression pane, add a dummy value to the input element, such as, "Get All Books". The input must be in quotes.
20. Click the getOut activity in the Process Editor, and select the Properties > Input tab. Expand the getAllBooks activity and choose Book* to map the Book* element from left to right. In the pop-up window, choose Make a Copy of each "Book" and click Finish. The tab will look similar to this:

Result

Your project is complete without any errors.

Testing the REST Service in Cloud Foundry

You can now test the REST service using the built-in tester and the Swagger UI.

Procedure

1. In the Project Explorer, expand the tibco.bwce.sample.binding.rest.BookStore.application process and expand the Package Unit > Properties folder.

2. In the Properties window, expand the tibco.bwce.sample.binding.rest.BookStore.application and set the default Application Profile to PCF as shown in the next image. The bracketed profile in the column head is the one that is selected:

3. Expand the Package Unit and select Overview.
4. In the **Overview** window select **Export Application for Deployment**.
5. Enter the location of your **EAR** file (this would be the same directory as the **manifest.yml** file).
6. In your Cloud Foundry environment, navigate to the directory containing your EAR.
7. Execute

   
   cf push -f manifest.yml

   
   If the application deploys successfully, you will get a routable URL.
8. Launch the Google Chrome browser and open **http://<BWCE-APP-URL>/swagger**. Click **Books** or **Events** to see the operations. Click **MyBooks** to see the REST service operations you just added. See the section called **Testing the POST and GET Operations** for information.

   ![Swagger UI](image)

9. Expand the Books and Events headers, and test out the operations as listed below.

**Result**

Click **Books** or **Events** in the Swagger UI to view the following operations for Books and Events:

**Books**

- Post books
- GET books
- GET book by ISBN
- PUT book by ISBN
- DELETE book by ISBN

**Events**

- POST Events
- GET Events
- GET Event by EventID
- PUT Event by EventID
- DELETE Event by EventID

**GET books** returns an output similar to the following:

```json
{
  "Book": [
    {
      "isbn": "0061122416",
      "name": "The Alchemist",
      "description": "Every few decades a book is published that changes the lives of its readers forever. The Alchemist is such a book",
      "authorName": "Paul Coelho",
      "releaseDate": "2006-04-25",
    }...
```
Testing the POST and GET Operations

An available RESTful service displays the GET operation in the Swagger UI. The POST operation is tested using the JSON service. It is important to test these operations by doing some simple tasks. This section explains how to test the POST and GET operations you just added.

Procedure

1. Click MyBooks. It expands and displays the POST and GET operations.
2. Click the POST icon to display its details.
3. Provide values for the Books parameter.
4. Click the Try it out! button.
5. Now click the GET icon to display its details.
6. Click the Try it out! button.
   The response displays a list of books returned by the REST service from the database.
Testing the REST Service in OpenShift

You can deploy the REST service in OpenShift environment and test by using the built-in tester and the Swagger UI.

Prerequisites

Ensure that the OpenShift environment is set up to deploy the application.

Procedure

1. In the Project Explorer, expand the tibco.bwce.sample.binding.rest.BookStore.application process and expand the Package Unit > Properties folder.

2. In the Properties window, expand the tibco.bwce.sample.binding.rest.BookStore.application and set the default Application Profile to Docker as shown in the following image:

   ![Properties Window](image)

3. Expand the Package Unit and select Overview.
4. In the **Overview** window select **Export Application for Deployment**.

5. In your Docker environment, navigate to the directory containing your EAR file.

6. To login to the OpenShift environment, run the following command:

   `oc login <OpenShift_Server_URL>`

7. If the project is not created, create an OpenShift project by running the following command:

   `oc new-project <PROJECT_NAME>`

8. To build an application docker image, run the following command:

   `docker build -t <APPLICATION_NAME>`

9. To tag an application docker image with your docker repository, run the following command:

   `docker tag <APPLICATION_IMAGE_NAME> <DOCKER_REPOSITORY_NAME>/<APPLICATION_IMAGE_NAME>`

10. To push a docker image to the docker repository, run the following command.

    `docker push <DOCKER_REPOSITORY_NAME>/<APPLICATION_IMAGE_NAME>`

    ![Image](image1.png)

    Here the docker hub registry is used to push the docker image. You can push images to OpenShift internal registry or any other public registry.

11. To use the OpenShift configuration file (YAML or JSON) and push the application on OpenShift, run the following command.

    `oc create -f <MANIFEST_FILENAME>`

```
apiVersion: v1
defaultService:
kind: Service
metadata:
  name: restapp

spec:
  type: LoadBalancer
  selector:
    app: restapp
  ports:
    - port: 80
      targetPort: 8080

apiVersion: v1
kind: ReplicationController
metadata:
  name: restapp

spec:
  replicas: 1
  selector:
    app: restapp
  template:
    metadata:
      name: Restapp
      labels:
        app: restapp
    spec:
      containers:
        - name: restapp
          image: <DOCKER_REPOSITORY_NAME>/<APPLICATION_IMAGE_NAME>
          imagePullPolicy: Always
          env:
            - name: BW_LOGLEVEL
              value: ERROR
            - name: DB_URL
              value: <Supported_database>://<username:password>@machine:port</database_name>
          ports:
            - containerPort: 8080
```

TIBCO BusinessWorks™ Container Edition Getting Started
12. To create a route for application, run the following command.

```
oc expose service <APPLICATION_SERVICE_NAME>
```

13. To view the route of an application, run the following command:

```
oc get route
```


15. In the Swagger UI, click **Books** or **Events** to see the operations and then click **MyBooks** to see the REST service operations you just added. For more information, see Testing the POST and GET Operations.

16. Expand the Books and Events headers, and test the operations as follows.

**Result**

In the Swagger UI, click **Books** or **Events** to view the following operations for the Books and Events headers:

**Books**

- POST books
- GET books
- GET book by ISBN
- PUT book by ISBN
- DELETE book by ISBN

**Events**

- POST Events
- GET Events
- GET Event by EventID
PUT Event by EventID
DELETE Event by EventID

**GET books** returns the following similar output:

```
{
  "Book": [
    {
      "isbn": "0061122416",
      "name": "The Alchemist",
      "description": "Every few decades a book is published that changes the lives of its readers forever. The Alchemist is such a book",
      "authorName": "Paul Coelho",
      "releaseDate": "2006-04-25",
      "vintage": true,
      "signed": true,
      "price": 11.9
    },
    {
      "isbn": "0071450149",
      "name": "The Power to Predict",
      "description": "How Real Time Businesses Anticipate Customer Needs, Create Opportunities, and Beat the Competition",
      "authorName": "Vivek Ranadive",
      "releaseDate": "2006-01-26",
      "vintage": false,
      "signed": true,
      "price": 15.999
    }
  ]
}
```

**GET books** by ISBN returns the following similar output for ISBN 0061122416:

```
{
  "isbn": "0061122416",
  "name": "The Alchemist",
  "description": "Every few decades a book is published that changes the lives of its readers forever. The Alchemist is such a book",
  "authorName": "Paul Coelho",
  "releaseDate": "2006-04-25",
  "vintage": true,
  "signed": true,
  "price": 11.9
}
```

**Troubleshooting**

You may encounter some errors while executing or running the process. The following are some of the possible errors you may encounter and their resolutions.

<table>
<thead>
<tr>
<th>Error Encountered</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>The REST Swagger UI page is not visible.</td>
<td>Verify that the application has started and that you are accessing the correct URL.</td>
</tr>
<tr>
<td>Problem markers are visible in the project.</td>
<td>Clean the project by invoking <strong>Project &gt; Clean</strong> or by switching to a clean new workspace.</td>
</tr>
<tr>
<td>The database and database tables are not created.</td>
<td>Ensure that the Cloud Foundry PostgreSQLService is configured correctly.</td>
</tr>
</tbody>
</table>
REST Reference Tutorial

The REST reference tutorial shows you how to create a simple REST Invoke to an existing REST Service defined by a Swagger specification.

You cannot convert REST reference to SOAP or vice versa.

Prerequisites

The REST service which you want to consume has to be deployed in your Cloud Foundry environment.

Creating a New Application

2. Open the Design perspective by clicking the Design icon in the upper right corner.
4. Enter in the Project Name text box. Do not change the remaining default settings. Click Finish. This will create a new application module with an empty process.

Obtaining the REST Service Swagger File

The REST service for which you want to obtain the Swagger file must be running. In this example the tibco.bwce.sample.binding.rest.BookStore.application should be deployed in your Cloud Foundry environment.

To obtain a Swagger file for a REST service, do the following:

1. Append swagger to the routable URL to access the Swagger doc. The format is http://<routable url>/swagger.

Importing the JSON File into your Project

1. In the Project Explorer, expand tibco.bwce.sample_binding_rest application module.
2. Right-click Service Descriptors and select Import > Import... > General > File System and click Next.
3. In the File system dialog, click the Browse button and browse to the location of the Books .json file.
4. Select the check box next to Books.json in the left pane and click Finish.

Creating the REST Reference

1. In the Project Explorer, expand the tibco.bwce_sample_binding_rest Service Descriptors folder completely.
2. Select the /books under Books.json and drag and drop it to the right side of the process in the Process Editor. The references are added to the process. The purple chevron indicates the service and its operations.
3. In the **Process Editor**, right-click **Add Activity > General Activities > Timer**. Optionally, you can configure the **Sleep** activity with **IntervalInMilliseconds** as 3000 in a similar manner and connect the **Timer** with **Sleep**.

4. Drag the **get** operation under the purple chevron and drop it on the right of **Timer** activity (or **Sleep** if configured) and connect the **Timer** activity with the **get** activity.

5. Drag the **post** operation under the purple chevron and drop it on the right of the **get** activity, connect the **get** activity with the **post** activity.

6. Right-click the **get** activity select **Show Properties View**.

7. In the **Properties** view, select the **Input** tab and click **Show Check and Repair** icon in the icon bar on the upper right corner of the Properties view.

8. Select the check box under **Fix** and click **OK**.

9. Click **Show Check and Repair** icon again. Select the check box under **Fix** and click **OK**.

10. Select the **post** activity and right click and select **Show Properties View**. In the **Properties View**, select the **Input** tab and select **Data Source** tab.

11. Expand **$get** in the **Data Source** tab completely.

12. In the XPath Expression pane, expand the **post-input** completely.

13. Drag and drop **Book** from the **Data Source** tab to the **Book** under post-input in the XPath Expression pane.

14. In the Drop dialog, select **Make a copy of each “book”** radio button and click **Finish**.

15. Click **Show Check and Repair** icon in the icon bar on the upper right corner of the Properties view.

16. Select the check box under **Fix** and click **OK**.

17. Click **Show Check and Repair** icon again. Select the check box under **Fix** and click **OK**.

18. In the **Project Explorer**, select **Books.json** under **Service Descriptors** of **tibco_bwce_sample_binding_rest_basic** application module, and right click **Open With > Text Editor** and locate the “host” attribute. Make a note of the host name and port number.
19. Expand the Resources folder under the tibco_bwce_sample_binding_rest_basic application module completely.


21. In the HTTP Client section, change the Default Host and Default Port to the values in the Books.json file you obtained in step 18 above.

22. Click File > Save All.

**Testing the REST Reference**

You can now test the REST service using the built-in tester and the Swagger UI. To do so follow these steps:

1. Click Run > Debug Configuration.

2. In the left pane of the Debug Configuration wizard, expand BusinessWorks Application and select BWApplication.

3. Click the Applications tab and then click Deselect All if you have multiple applications. Select the check boxes next to tibco_bwce_sample_binding_rest_basic_application.
4. Click **Debug**. This runs the sample in debug mode. The Console view is opened and shows engine messages similar to: 

```
Started BW Application [ tibco_bwce_sample_binding_rest_basic_application:1.0]
```

5. In the **Debug** view, expand **BWApplication [BusinessWorks Application] > <launched> BWEclipseAppNode > tibco_bwce_sample_binding_rest_Process** and select **get**.

6. In the **JobData** view, you can see the job data of the **get** activity.