



# ihi<sup>TM</sup> iWay<sup>®</sup> Service Manager

## Release Notes

Version 9.1.0 | March 2023



# Contents

---

<b>1. ibi™ iWay® Service Manager Version 9.1.0 Integration Release .....</b>	<b>7</b>
Overview .....	7
Installation Notes .....	8
Migration .....	9
Runtime.....	9
Design Time.....	10
Deprecated Components .....	10
Resolved Issues .....	11
Version 9.1.0.....	11
Version 9.0.0.....	14
Known Issues and Considerations .....	16
Upgrading iWay Service Manager From Older Releases Not Supported.....	16
Upgrading Integration Tools From Older Releases Not Supported.....	16
Multiple Hostnames Supported in the iWay Service Manager Administration Console.....	17
Migrated Process Flows Containing REST Objects With Query Parameters.....	17
Data Quality Services Plugin.....	17
Connector for SharePoint Multi-Factor Authentication.....	18
Git Support in ibi™ iWay® Service Manager Integration Tools.....	18
ibi™ iWay® Application Adapters.....	21
Deprecated.....	21
Updated Adapter Configuration Process.....	21
Available Application Adapters.....	26
Available Technology Adapters.....	27
Event Handling.....	27
iWay Business Activity Monitor.....	28
Process Flow Test-Run (z/OS).....	28
Updating the iWay Configuration File Metadata.....	29
ibi™ WebFOCUS® Support.....	30
Key Features .....	30
Version 9.1.0.....	30
Step Debugger.....	30
Debugging a Flow in Standalone Mode.....	31

Navigating and Using the Debug Configurations Dialog. . . . .	34
Starting a Debug Session. . . . .	35
Navigating and Using the Debug Perspective. . . . .	35
Variables Tab. . . . .	38
Document Tab. . . . .	39
Expressions Tab. . . . .	40
Setting Breakpoints. . . . .	40
Breakpoint Configuration Options. . . . .	42
Overriding Events. . . . .	43
Application Debug Mode. . . . .	45
Debugging Flows on Remote Servers. . . . .	45
Code Extensions and Extension Projects. . . . .	46
Creating an Extension Project. . . . .	46
Writing Agent Class Extensions. . . . .	48
Refreshing Metadata. . . . .	60
Debugging Agent Code. . . . .	63
Writing iWay Functional Language Class Extensions. . . . .	67
iWay Application Summary. . . . .	71
Defining a Scope for Delete Group Variables. . . . .	75
Connector for ibi™ Data Quality. . . . .	75
Connector for PowerShell. . . . .	77
Connector for Azure Storage Queue. . . . .	78
SharePoint Rest Connector for SharePoint Online. . . . .	79
Version 9.0.0. . . . .	82
Testing Platform. . . . .	82
Creating a Test Suite. . . . .	82
Navigating and Using the Test Case Editor. . . . .	84
Mock Tab. . . . .	87
Assert Tab. . . . .	89
Verify Tab. . . . .	91
Compare Tab. . . . .	92
Validate Schema Tab. . . . .	95
Outline View. . . . .	95

Navigating and Using the Test Suite Editor. . . . . 96  
 Running Tests. . . . . 97  
 Test Suite Run Configuration. . . . . 98  
 Test Execution View. . . . . 99  
 Running Tests Using Ant. . . . . 101  
 Connector for Microsoft Azure Service Bus Management. . . . . 102  
 Connector for Microsoft Office 365 Outlook. . . . . 104  
 Attachment Object. . . . . 108  
 Token Object. . . . . 108  
 SWIFT 2022. . . . . 108  
**Legal and Third-Party Notices . . . . . 109**



This document provides release information for ibi™ iWay® Service Manager version 9.1.0. It is intended for all levels of users, including system integrators, application developers, and administrators. For more information on specific features, refer to the online documentation.

**In this chapter:**

- [Overview](#)
  - [Installation Notes](#)
  - [Migration](#)
  - [Deprecated Components](#)
  - [Resolved Issues](#)
  - [Known Issues and Considerations](#)
  - [Key Features](#)
- 

## Overview

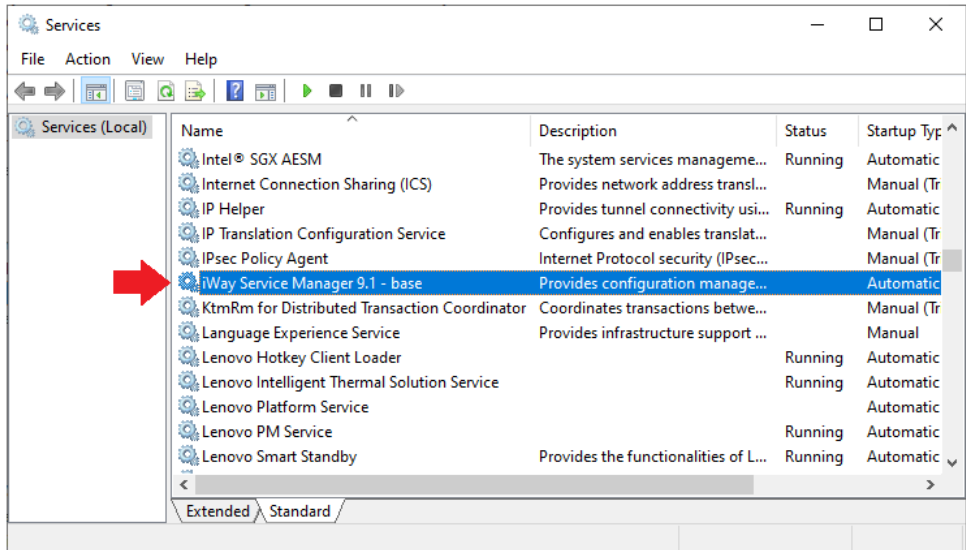
iWay Service Manager (iSM) is an integration server that ensures rapid access to timely, accurate data across all systems, processes and stakeholders – with unmatched interoperability between disparate systems and data. With iSM, all aspects of your existing infrastructure – every integration, application, and development platform – work in concert with modernized architectures to rapidly develop new business applications, and create powerful, reusable business services from existing applications. This support for modern architectures ensures a highly optimized development environment and rapid creation of internally and externally consumable services.

iSM offers end-to-end integration of the widest variety of sources, including real-time, batch, streaming, big data, structured and unstructured information, cloud-based sources, social network, and machine-generated data.

## Installation Notes

This section provides installation notes for ibi™ iWay® Service Manager version 9.1.0.

- ❑ Major releases of iWay Service Manager (iSM) version 9 and any service pack releases currently only support Java 2 Standard Edition (J2SE™) JDK version 11 as the required run-time environment.
- ❑ When updating ibi™ iWay® Service Manager Integration Tools (iIT) from version 9.0.0 to 9.1.0, iIT displays a message indicating that iIT must be restarted after updating to version 9.1.0. In addition, iIT must also be restarted a second time in order to start the embedded server.
- ❑ After installing iWay Service Manager version 9.1.0 on Windows or updating from version 9.0.0 to 9.1.0, note that the name of the deployed service has changed. The name of the service is now **iWay Service Manager 9.1** and also includes the name of the deployed configuration (for example, base), as shown in the following image.



- ❑ Major releases of iSM version 9 and any service pack releases require Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files, which can be obtained from the Oracle Java download.



- ❑ iSM version 9 can be installed on the same environment where previous versions of iSM (e.g., 9.x) have been installed. You must follow the proper procedure to migrate an application from a previous version to iSM version 9. iSM version 9 cannot be overlaid on top of an older version, and must be installed in parallel into an empty location (directory) on the file system.
- ❑ iSM version 9 does not provide a 32-bit installation, only a 64-bit installation is supported. The installation requires administrative rights for installation and runtime purposes. iSM version 9 requires full access to the sub-directories of its installation. As a result, ensure that the end-user has full access rights.
- ❑ On Windows platforms, a deployed iWay Integration Application (iIA) will create its own corresponding Windows service. This Windows service is not recreated with each iIA deployment and will retain its properties.
- ❑ The release level of iSM must match the release level of ibi™ iWay® Service Manager Integration Tools (iT). For example, if you upgrade iT, then you must upgrade iSM to the corresponding version. iSM version 9 releases no longer facilitate the use of core services, which were present in the older iSM version 7 framework. This is due to simplification and to ensure that the user always has matching design time and runtime environments to get the benefits of new features.

## Migration

ibi™ iWay® Service Manager Version 9.1.0 allows you to migrate iWay Integration Applications (iIAs) that were developed in iWay Service Manager (iSM) 8.x environments to an iSM 9 environment. When discussing migration, it is important to distinguish between *runtime* and *design time*, as described further in this section.

## Runtime

If you are not planning to make any modifications to an existing iIA, then you can simply export this iIA from your iSM 8.x runtime environment using the iSM Administration Console. You can then import the iIA into your iSM 9 runtime environment using the iSM Administration Console. This enables you to import the iIA and the corresponding deployment template. You can then deploy the iIA. The iIA will continue to run the same in your iSM 9 environment as it did in your iSM 8.x environment.

### Design Time

If you need to update an existing iIA for maintenance or review, you can do so using ibi™ iWay® Service Manager Integration Tools (iIT). Export the corresponding Integration Project from iSM 8.x to the file system or source management repository. You can then import or check-out the Integration Project into the iIT workspace of your iSM 9 environment. When the Integration Project is imported, the components will be automatically converted into iSM 9 format. Since objects have changed between iSM 8.x and iSM 9, the best attempt has been made to convert the objects accordingly. However, you may notice that some of the iSM 8.x objects show up as Service nodes in the iSM 9 process flows. However, aside from the visuals, the migrated Integration Project is fully functional. You can then build and deploy your iIA as required.

It is recommended that you review any iIAs developed using iSM 8.x to determine if the design is utilizing all of the latest features available in later releases. It is very common to discover that some parts of the iIA (even though it will continue to function as is), might benefit from a simplification as part of the update, instead of a direct migration.

### Deprecated Components

This section provides a summary of deprecated components in ibi™ iWay® Service Manager Version 9.1.0.

- ❑ MQSI components have been removed from the product and are no longer supported. If you have a requirement for continuing support for this and related components, please contact Support.
- ❑ SOAP over JMS/MQ has been removed from the product and is no longer supported. For alternative approaches, you may use a direct connection to the JMS/MQ components for data processing.
- ❑ WAR-based deployment has been deprecated. The ability to create WAR packages has been removed from the iWay Service Manager Administration Console. You can build Service Manager WAR packages using the iWay Service Manager Software Development Kit (SDK). If you require WAR-based deployment support, please provide your use case to Support.
- ❑ The Log Event Adapter for Microsoft SQL Server has been deprecated.
- ❑ The Emitter object has been deprecated in ibi™ iWay® Service Manager Integration Tools. Emitters can be used as part of the channel itself. If you have a use case for accessing Emitters in process flows, open a case with Support and provide full details.

- The following product components have been deprecated and removed from distribution. There are alternatives approaches for achieving the required functionality or the components are deemed no longer viable.
  - Corba
  - Fix
  - Clarify
  - Tuxedo
  - BEA JDBC
  - BEA PS
  - Validation (an older implementation of the QA Service)
  - CS3
  - Lawson Preparser
  - Manugistics Preparser
  - CDF/CSV (an updated version is available for flat file processing)

## Resolved Issues

This section summarizes resolved customer cases in ibi™ iWay® Service Manager Version 9.1.0 and 9.0.0.

### Version 9.1.0

This section provides a reference to the resolved customer cases in iWay Service Manager Version 9.1.0.

Customer Support Case	Summary
02156888	SAMBA: The Rename action generates an exception.
02146166	API entry is added to the Bundle list every time an Ant build is executed.

<b>Customer Support Case</b>	<b>Summary</b>
02145152	COTY - NFR - Provision to provide a description for the iWay API Listener Object as this gets displayed on the Monitoring page in the iWay Service Manager Administration Console.
02141521	iFL function that will calculate the last day of a month (LDM).
02138946	SFTP agent log to include additional details about the client connection (i.e., list of ciphers used by client and server).
02138376	An iSM configuration created that is based on the Raw configuration will not allow iIT / Library / Deploy to work.
02135453	Fix connection pooling and caching to avoid socket depletion.
02133982	NFR - iIT Build Reports list all of the objects, SREGs, and the description in each process flow.
02132794	iIT 807 - A SocketTimeoutException error is generated with long-running process flows.
02132161	Support for the latest JSON (2020-12) schemas.
02126308	iIT 8.0.4- The com.ibi.agents.XDDQBatchExec service does not display input parameters from earlier releases of iIT.
02123579	In 8.0.2 the SQL POST Query listener password defined as an expression using IFL is stored as ENCR() when deployed to runtime.

Customer Support Case	Summary
02113987	Remove the requirement to provide a value for the Value field when the Delete Variable(s) option is used.
02104387	Schema validator returning unexpected edges.
02098834	iWay Functional Language (iFL) date functions not working in iIT.
02096344	iIT - Need the ability to make the annotation edit box larger.
02095387	The Payload agent does not reset the bus to XML like the old Constant agent.
02090879	Element (item) is not included in the XML to CSV Transform.
02089031	The SFTPDirectFileTransfer agent increments the Extract count on the SFTP server prior to file download.
02084635	The _fileinfo('isdir', 'path') function returns an error if false.
02056300	NFR: The ability to set a default size of agents in a process flow in iIT.
02055660	iIT Performance - It can take up to two minutes to add or remove process flows from iWay Integration Applications (iIAs).
02048518	Annotation boxes behave differently when using 2540x1440 and 1920x1080 screen resolutions.
02036385	iWay BAM using an Oracle repository with NLS_TIMESTAMP_FORMAT DD-MON-RR HH24.MI.SSXF is getting error ORA-01830.

<b>Customer Support Case</b>	<b>Summary</b>
02035574	iIT is mapping only for string values when using a JSON transformation with a template containing multivalued values in the field.
02028288	NFR: Generic solutions to GET files by REST API.
02027584	com.ibi.agnets.XDOAuth2Agent with Google's G Suite Directory API.

### Version 9.0.0

This section provides a reference to the resolved customer cases in iWay Service Manager Version 9.0.0.

<b>Customer Support Case</b>	<b>Summary</b>
02108800	Changing the iIT perspective corrupts iSM application project.
02108484	Problem with iIT 'IF' function.
02108341	Error in open Mapping Builder in the JDBC Data Source for transformations.
02102533	Azure Data Lake Connector evaluation bug for a few fields and missing password decryption for account key.
02101047	The EDIBatchSplitter Preparser causes a JNPE when BAM is active.
02096322	Support channel Auto Start flag at Deployment Template level.
02095410	iIT deleting flow test results prompts "are you sure you want to delete xxxx" after deletion has happened.

Customer Support Case	Summary
02095391	Copying and pasting a disabled agent grays out the agent icon.
02094486	Copying and pasting a resized agent sets the pasted agent back to the default size.
02094480	Azure Blob Storage agent should create a target directory if it does not exist and overwrite existing file option.
02091523	Variables object cannot delete global variables.
02090198	EmailEmitAgent not working with TLS version 1.2.
02084635	The _fileinfo('isdir', 'path') function returns an error if false.
02080948	Process flow transforms no longer add a stylesheet declaration/instruction tag after the XML declaration.
02077102	Transformer replicates group with elements distinguished by attributes.
02071279 02071748	Validation errors when using a decision switch or iterator.
02056293	UC - iIT805 -> iIT806 import: Annotation/Comment missing.
02035544	Error when transforming JSON to other formats when the value contains "18\u001F1\u001F".
02032502	Azure Service Bus Management Connector

<b>Customer Support Case</b>	<b>Summary</b>
02030473	Importing process flows with a Count iterator shows invalid warnings until the agent is clicked on (selected).
02030427 02108690	_xquery missing from iFL expression tester and does not appear to work in process flows.
02027584	com.ibi.agents.XDOAuth2Agent with Google's G Suite Directory API.

## Known Issues and Considerations

This section describes known issues and considerations in ibi™ iWay® Service Manager Version 9.1.0.

### Upgrading iWay Service Manager From Older Releases Not Supported

Older installations of iWay Service Manager (for example, 7.0.x or 8.0.x) cannot be upgraded to version 9 (9.0.0).

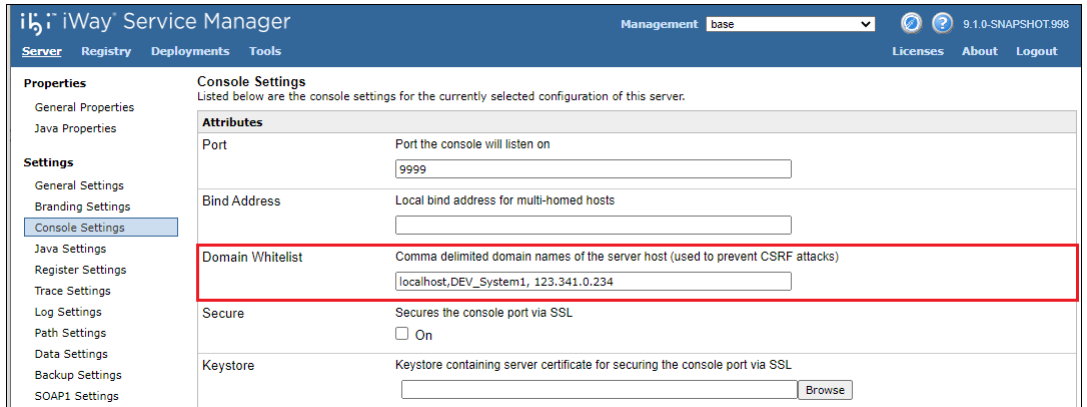
### Upgrading Integration Tools From Older Releases Not Supported

iWay Service Manager version 9 (9.0.0) uses a new Eclipse framework (version 4.19) for ibi™ iWay® Service Manager Integration Tools (iIT). Older installations of iIT cannot be upgraded to version 9.



## Multiple Hostnames Supported in the iWay Service Manager Administration Console

The new *Domain Whitelist* parameter allows you to add a comma delimited list of hostnames to access the iWay Service Manager Administration Console. This parameter is located in the Console Settings section of the iWay Service Manager Administration Console, as shown in the following image.



After configuring the *Domain Whitelist* parameter, scroll down the page and click *Update*, and then restart iWay Service Manager for these changes to take effect.

## Migrated Process Flows Containing REST Objects With Query Parameters

If you are importing an application project that contains a process flow with a REST object and query parameters from a previous version (for example, version 8.0.x) to ibi™ iWay® Service Manager Integration Tools version 9, the query parameters in the REST object are missing.

As a workaround, you will need to manually configure the query parameters for the REST object that has been imported in your process flow.

## Data Quality Services Plugin

Data Quality Services (DQS) version 13.7.1 is now used by ibi™ iWay® Service Manager Integration Tools in version 9.

The URL that must be used to access the DQS plugin is:

<https://update.tibco.com/eclipse/dqs-ide/4.19/>

## Connector for SharePoint Multi-Factor Authentication

As is typical with server-to-server applications, the Connector for SharePoint does not support Multi-Factor Authentication. The Security Defaults feature of Azure Active Directory enforces Multi-Factor Authentication and is therefore incompatible with the connector. Azure Active Directory Conditional Access security without Multi-Factor Authentication must be used, instead.

The following procedure describes how to disable Security Defaults in the Azure Active Directory portal:

1. Sign in to the Azure Portal from <https://aad.portal.azure.com/>.
2. From the left pane, select *Azure Active Directory* and then select *Properties*.
3. From the right pane, select *Manage Security defaults*.
4. Change the Enable Security defaults setting to *No*.
5. Click *Save*.

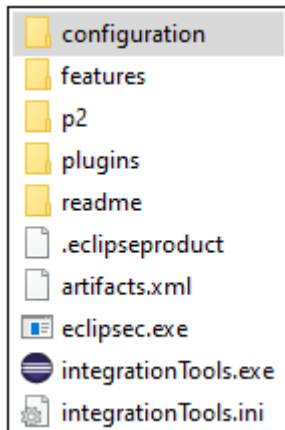
## Git Support in ibi™ iWay® Service Manager Integration Tools

ibi™ iWay® Service Manager Integration Tools (iT) enables you to install and integrate the Git open source distributed version control system. Due to a bug in the Git plugin for Eclipse, after installing iT, you must perform the following steps to enable integration with Git:

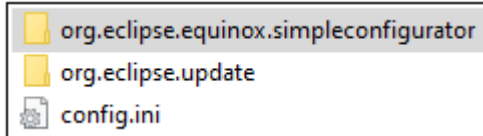
1. Close iT.
2. Navigate to the location on your file system where iT is installed. For example:

`C:\iIT_900`

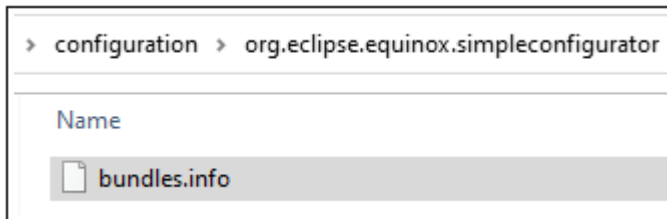
3. Open the *configuration* subfolder, as shown in the following image.



4. Open the *org.eclipse.equinox.simpleconfigurator* subfolder, as shown in the following image.



The *org.eclipse.equinox.simpleconfigurator* subfolder contains the *bundles.info* file, as shown in the following image.



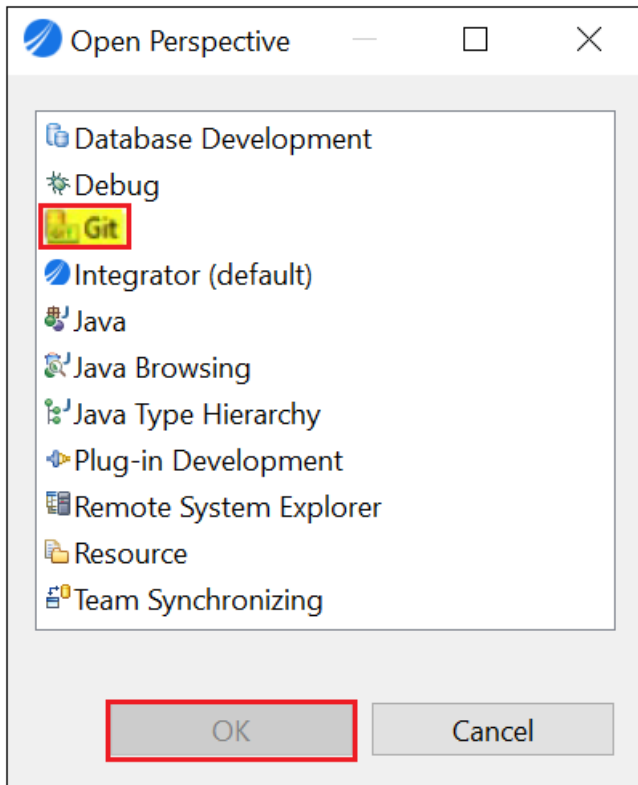
5. Edit the *bundles.info* file using a text editor (for example, Notepad).
6. Search for the following line in the *bundles.info* file:  

```
org.slf4j.api,1.7.2.v20121108-1250,plugins/  
org.slf4j.api_1.7.2.v20121108-1250.jar,4,false
```
7. Delete this line.
8. Save the *bundles.info* file.
9. Open iIT.

10. Click the *Open Perspective* icon on the toolbar, as shown in the following image.

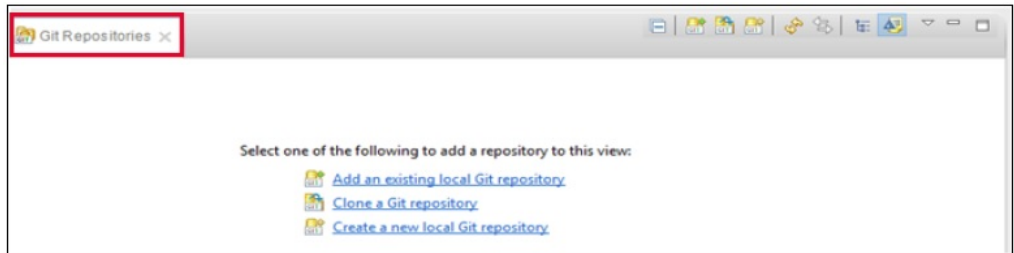


The *Open Perspective* dialog box opens, as shown in the following image.



11. Select *Git* from the list of available perspectives, and then click *OK*.

You are returned to iIT, where the Git perspective is now implemented, from which you can select and work with your Git repositories, as shown in the following image.



## ibi™ iWay® Application Adapters

This section describes known issues and considerations for ibi™ iWay® Application Adapters.

### Deprecated

- ❑ The Microsoft CRM Application Adapter 2011 is deprecated and will be removed in the next release of iWay Service Manager Version 9, as the underlying application has been removed from extended support by Microsoft. Event functionality has also been deprecated as this ability has been removed by Microsoft. If you have a requirement for continuing support for this adapter and related components, please contact Support.
- ❑ Microsoft Exchange supports Exchange Web Services at the Exchange 2010 level. This adapter is deprecated and will be removed in the next release of iWay Service Manager Version 9, as the underlying application has been removed from extended support by Microsoft. The Exchange Web Services API .jar file is no longer available from Microsoft, but can be found on GitHub. If you have a requirement for continuing support for this adapter at the Exchange 2010 support level and related components, please contact Support.

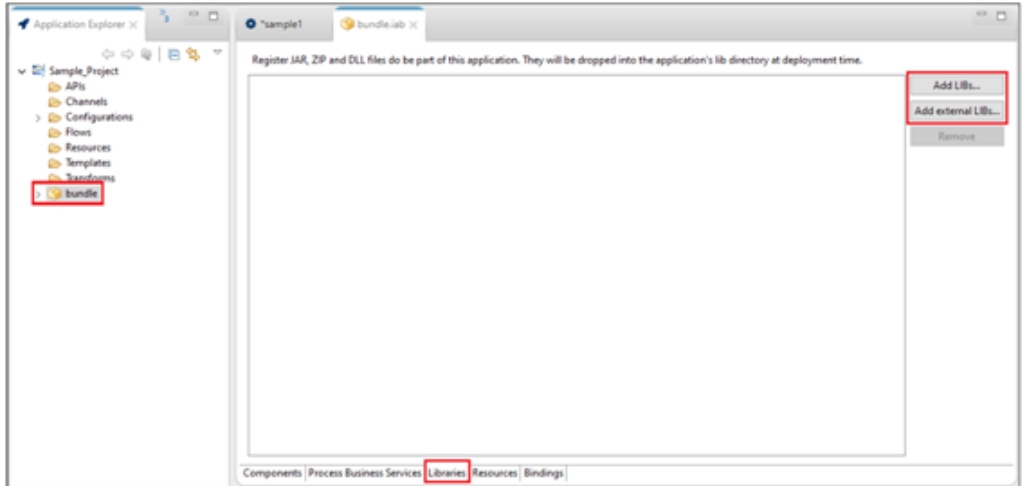
### Updated Adapter Configuration Process

As of iWay Service Manager Version 8, the concept of Application Projects is used as separate containers for adapter instances. Adapters are no longer installed as system-wide instances, rather Application Projects and their dependencies are designed in isolation and packaged together and deployed to the runtime. This provides improved isolation of applications and efficient use and allocation of system resources.

Applications can be tested and run from the design-time area without deployment during development, so iterative development can take place.

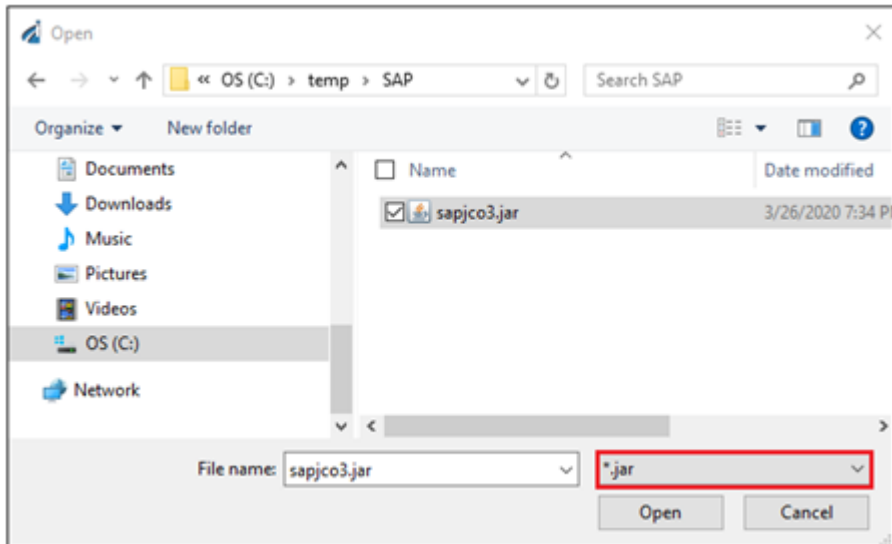
To configure an adapter in ibi™ iWay® Service Manager Integration Tools (iIT):

1. Create a new Application Project for an adapter to be used.
2. Double-click the *bundle* object in the Application Project hierarchy (Application Explorer tab), and then click the *Libraries* tab in the right pane, as shown in the following image.



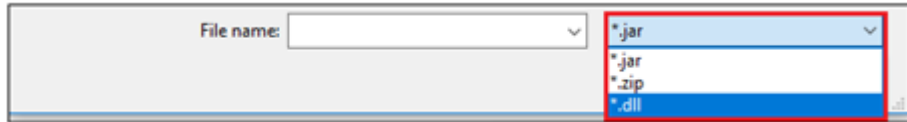
3. Click **Add external LIBs** to locate and select the third-party resource (.jar, .dll, .zip) that may be needed from your file system.

The Open dialog is displayed, as shown in the following image.

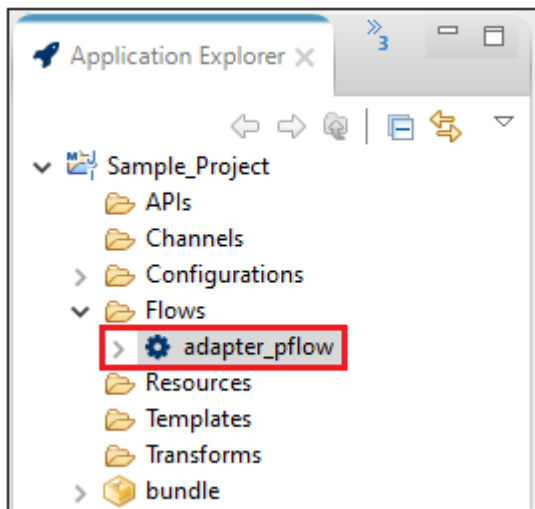


For example, for SAP ERP on Windows, you must add the *sapjco3.jar* and *sapjco3.dll* files to the Libraries tab, which will add these files to the Application Project's \lib folder during deployment.

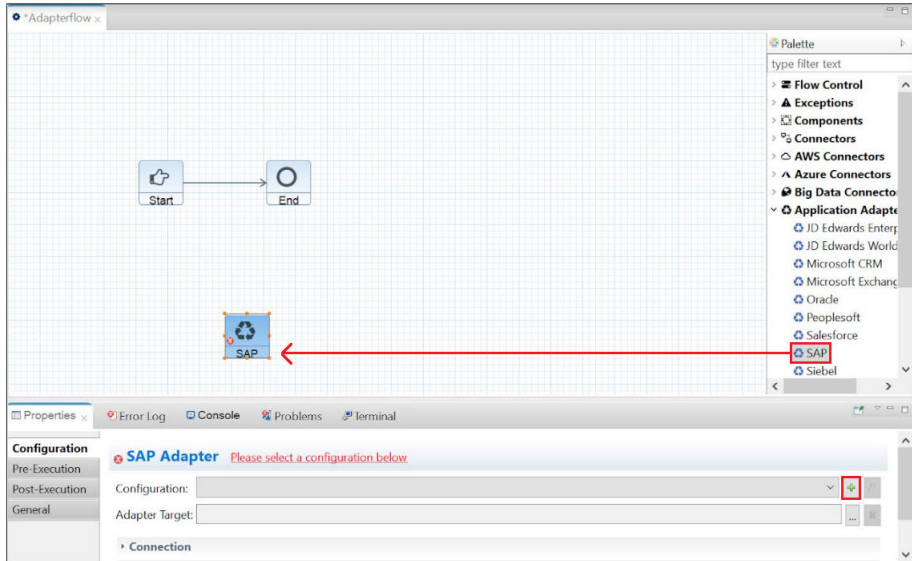
**Note:** The file type drop-down list in the Open dialog defaults to *.jar*. When adding dependencies that incorporate dynamic-link libraries (such as Windows *.dll* files), expand the file type drop-down list and select *.dll* or *.zip*, according to the appropriate type you require, as shown in the following image.



4. Create a new process flow in your Application Project (Flows subfolder), as shown in the following image.

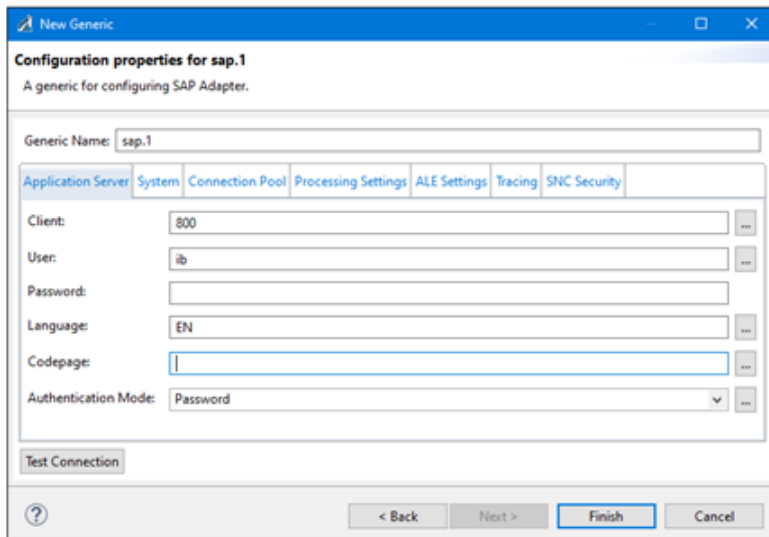


5. In the process flow, drag an Application Adapter (or Technology Adapter) onto the process flow canvas, as shown in the following image.



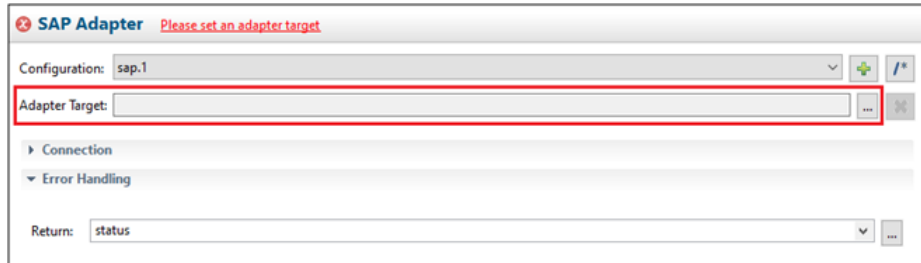
When the adapter object is added to the process flow canvas, the adapter configuration properties open with the name of the adapter as the header (for example, SAP Adapter).

6. Click the *green plus sign icon (Create a configuration)* to the right of the Configuration field. The New Generic dialog opens, as shown in the following image.





7. Enter the required connection information and login credentials for the system you are connecting to (for example, SAP ERP).
8. Click *Test Connection* to validate, and then click *Finish*.
9. Set an adapter target to use and explore based on the system you have connected (for example, SAP ERP). Click the *ellipsis* button (...) to the right of the Adapter Target field, as shown in the following image.

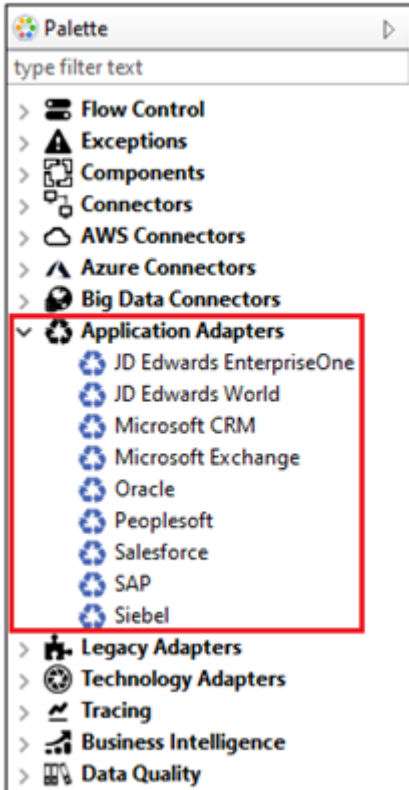


10. Select an available remote object (function or module) to be invoked on the target host.
11. Once your adapter target configuration is completed, return to the process flow.

For more information about testing, running, and deploying process flows and applications, see the *iWay Integration Tools* documentation.

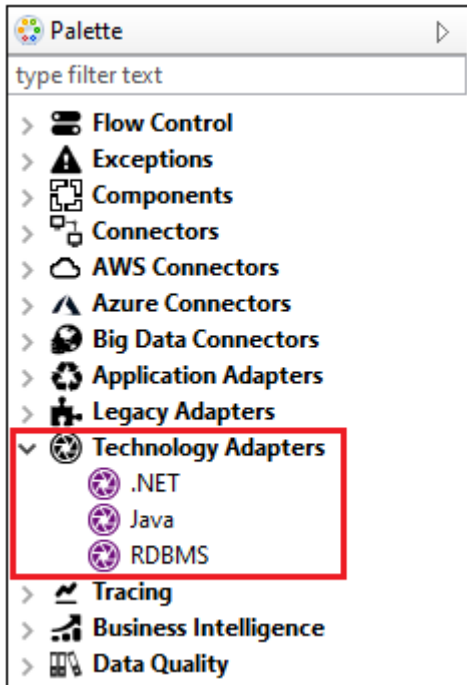
## Available Application Adapters

The following *Application Adapters* are available from the Palette in ibi™ iWay® Service Manager Integration Tools (iIT), which you can add to process flows in your Application Project.



## Available Technology Adapters

The following *Technology Adapters* are available from the Palette in ibi™ iWay® Service Manager Integration Tools (iT), which you can add to process flows in your Application Project.



## Event Handling

Adapter-specific Ports and Channels are deprecated in iWay 8 and have been removed from the product. For the following adapters, events are captured by using event listeners in the adapter configuration. They are added using Channels in the Application Project that trigger a process flow. Create a Channel and select one of the following adapters as a listener:

- ConnectDirect
- SAP (SAP ERP)
- LDAP
- MSMQ

For the following adapters, use an RDBMS, HTTP, or SOAP Listener in the channel to capture application events:

- JD Edwards EnterpriseOne
- JD Edwards World
- Siebel
- SalesForce
- PeopleSoft
- Oracle (Oracle Applications)
- RDBMS

### **iWay Business Activity Monitor**

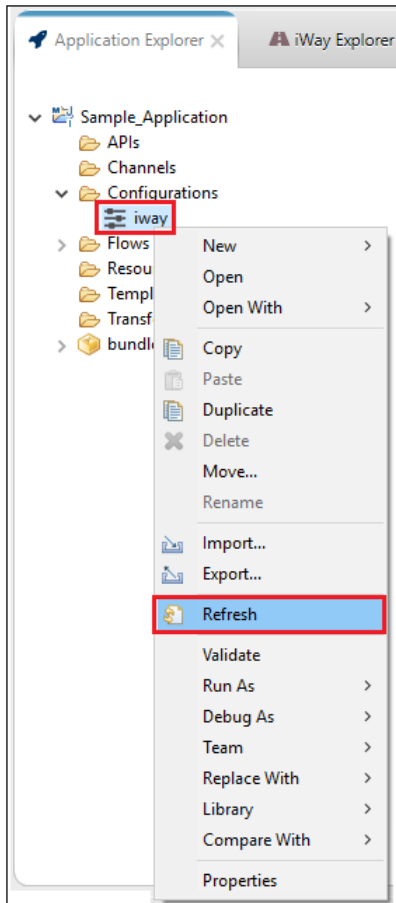
iWay Business Activity Monitor (BAM) requires the creation of a new database. An existing iWay BAM database (created in iWay Service Manager version 7.x) can be archived for future access through standard database utilities. There are known issues with the resubmit functionality of iWay BAM due to repackaging into the application structure. For more information or if you have any questions, contact Support.

### **Process Flow Test-Run (z/OS)**

On z/OS platforms, a *test run* for a process flow from ibi™ iWay® Service Manager Integration Tools (iIT) can be performed only against the local configuration or a remote configuration. The *test run* against a test server is not supported at this time.

## Updating the iWay Configuration File Metadata

Metadata for the iWay configuration file, which contains definitions for generics/configurations, can be updated to the latest iWay version using the *Refresh* option. Doing so will bring in the options for newer components that are available in the latest iWay version. To update your metadata using iIT, expand the *Configurations* subfolder in your application project, right-click the iWay configuration file (*iway* node), and select *Refresh* from the context menu, as shown in the following image.



If you create a new configuration from within a process flow, then this step is not required. In this case, metadata for the iWay configuration file will be refreshed automatically for any components that are used.

### ibi™ WebFOCUS® Support

Due to ibi WebFOCUS security model changes, including internal API changes, you must disable the new WebFOCUS security model to enable existing iWay applications to run against WebFOCUS. This will be addressed in a future iWay release, where the new WebFOCUS security model will be supported.

#### **iWay WebFOCUS Object.**

- Fully supports WebFOCUS version 80xx with no additional configuration required.
- If you are using this object with WebFOCUS version 82xx or 9.x, Application Folders are supported, but not Virtual folders.

#### **ReportCaster Object.**

- Fully supports WebFOCUS version 80xx with no additional configuration required.
- If you are using this object with WebFOCUS version 82xx or 9.x, you must uncheck the *Cross Site Request Forgery Protection* setting in the WebFOCUS Administration Console.

- ETL Object.** Supports WebFOCUS version 80xx, 82xx, and 9.x.

## Key Features

This section describes key features for ibi™ iWay® Service Manager and ibi™ iWay® Service Manager Integration Tools in version 9.1.0 and 9.0.0.

### Version 9.1.0

This section provides a summary of key features available in ibi™ iWay® Service Manager (iSM) and ibi™ iWay® Service Manager Integration Tools (iIT) in version 9.1.0.

#### Step Debugger

iWay Service Manager version 9.1.0 provides a set of tools, which allows you to debug your process flows within the iIT design time environment in real time.

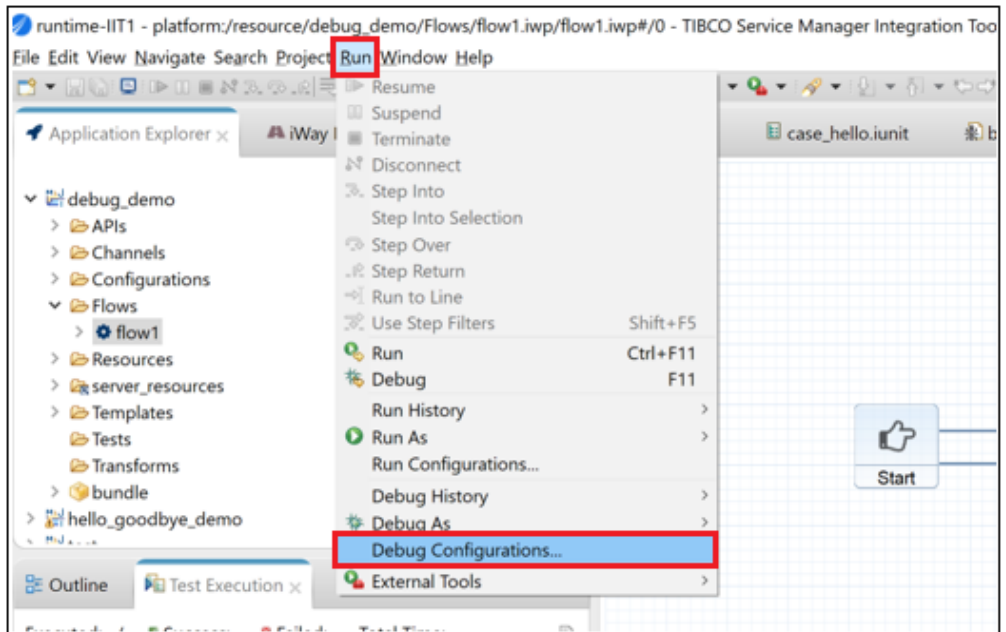
You can launch flows in Debug mode, step from one object to another inspecting contents of the document, as well as the variables (SREGs) currently in scope. You can also set breakpoints, override variable values, change the current document, among other functions.

Debugging flows can also be performed within the context of an application. In this scenario, a Debug session is started for the entire application, and the Step Debugger reacts to the breakpoints being hit on flows as a result of the flow being invoked from a channel, an API endpoint, or a web service call.

## Debugging a Flow in Standalone Mode

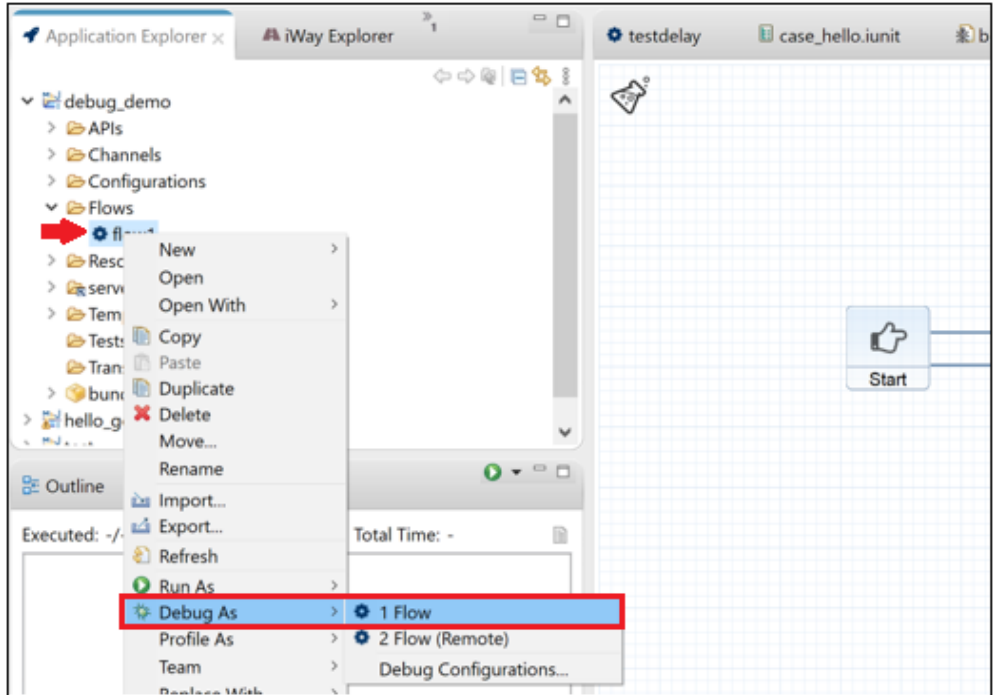
A *Debug Configuration* must first be created before you can debug a flow. There are several ways to do this in the iIT design time environment.

1. Click *Run* from the menu bar and select *Debug Configurations...* from the context menu, as shown in the following image.



The Debug Configurations dialog opens, where you can create your debug configuration.

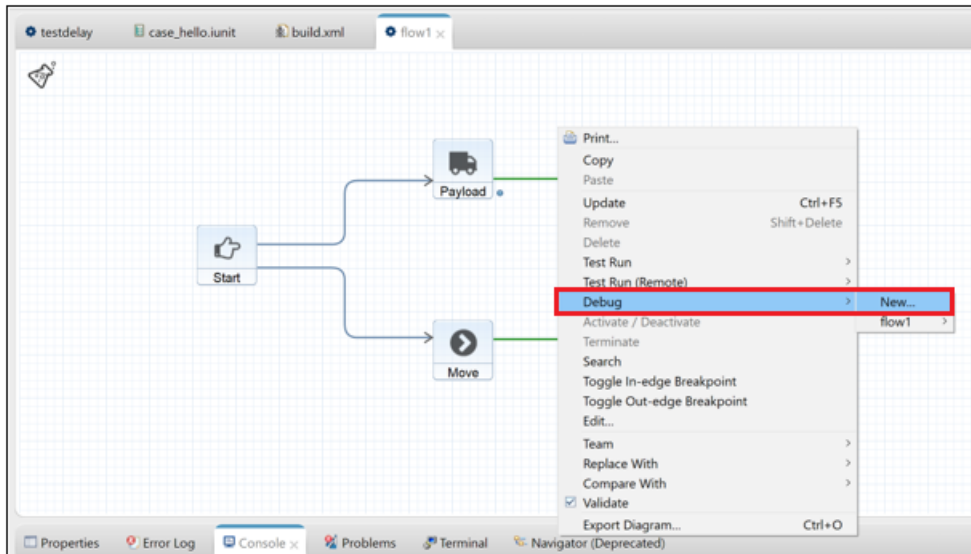
2. You can also use the *Debug As* option. Right-click a flow resource from the Application Explorer, select *Debug As*, and then click *Flow*, as shown in the following image.



The Debug Configurations dialog opens if no debug configurations exist for the flow, or if one is available, will launch the previously configured debug configuration against the flow. Select *Debug As*, and then click *Debug Configuration* to create a new debug configuration directly.



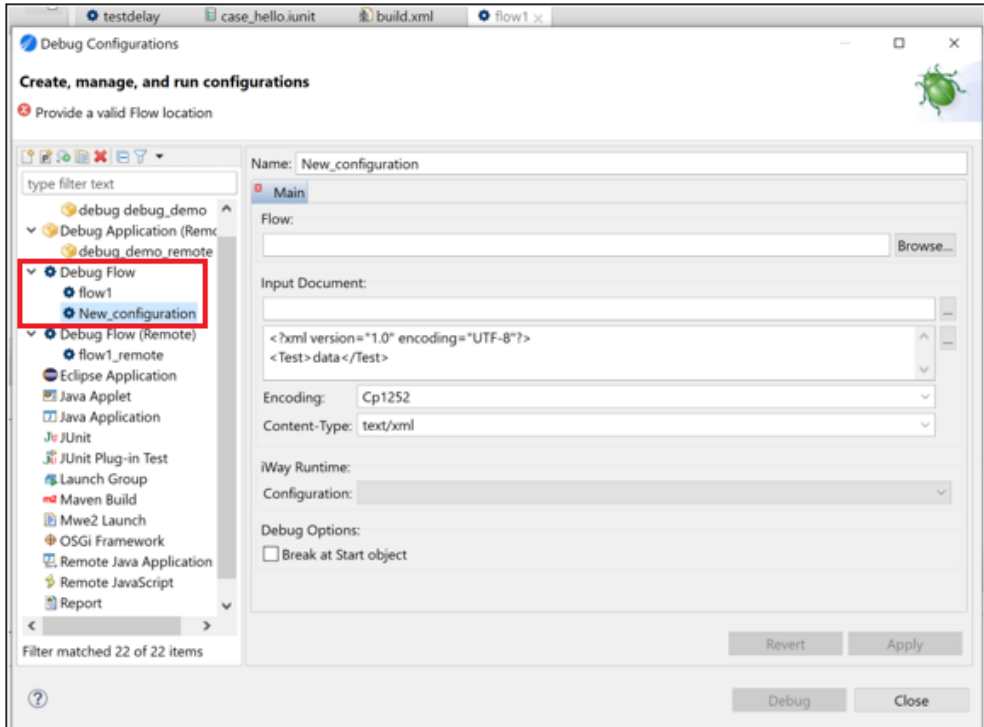
3. Finally, debug configurations can be managed or launched through the context menu of the flow editor. Right-click anywhere within the flow editor canvas, select *Debug* and click *New*, as shown in the following image.



You can also select an existing debug configuration (for example, *flow1*) from the context menu to run against your flow.

### Navigating and Using the Debug Configurations Dialog

The Debug Configurations dialog allows you to create and manage your flow debug configurations. Expand *Debug Flow* in the left pane and click *New Configuration*, as shown in the following image.



To select an available flow resource for which you want to create a debug configuration, click *Browse* to the right of the *Flow* field.

The **Input Document** section allows you to load a file from the workspace or your local file system (upper text field). Alternatively, the document may be typed directly into the text area (lower text field). The upper text field takes priority if both are populated.

**Encoding** must be specified when the document is provided through a path and the file contains a document that is not binary and is not in XML format.

The **Content-Type** field specifies the document format.

Select an **iWay Runtime Configuration** to use for starting an embedded server.

When the **Break at Start** object debug option is selected, the debug session will start and the execution will automatically pause (suspend) at the Start object. This is useful if you would just like to "step" through the flow (inspecting data at each point on the flow), without having to previously set any breakpoints.

### Starting a Debug Session

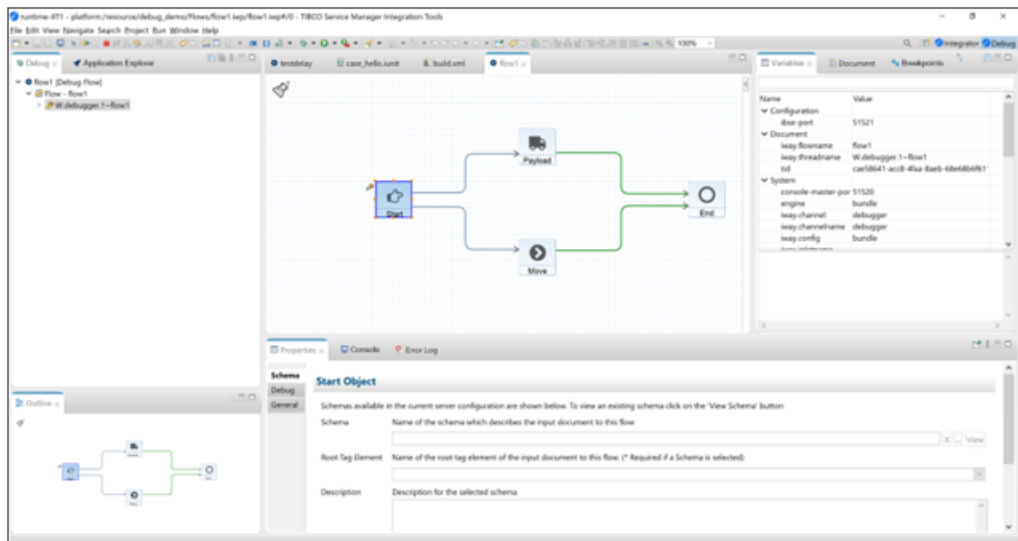
After all of the debug configuration parameters and options have been set, click *Debug* and a debug session will be started against the selected flow.


During the start of a debug session, the process will check whether there is an embedded server (iWay Service Manager) that is started and running for the project associated with the flow. If there is no embedded server currently running, one will be started using the *iWay Runtime Configuration* parameter described in the previous section.

ibi™ iWay® Service Manager Integration Tools (iIT) will automatically switch to the *Debug Perspective*. The flow being debugged will be opened in an editor if not already opened, and execution will pause at the Start object (if the *Break at Start object* debug option was selected during the debug configuration).

### Navigating and Using the Debug Perspective

The *Debug Perspective* consists of the Debug tab (view) located in the left pane, which lists the processes currently running and debug sessions, the flow editor (center pane), and several tabs (views) supporting debug operations, such as *Variables*, *Document*, *Breakpoints*, and *Expressions*.




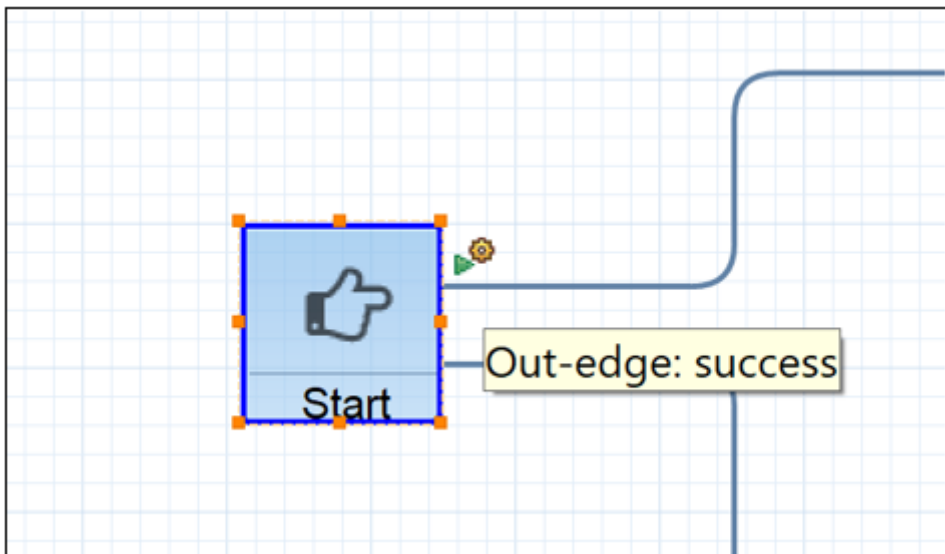
The above screenshot shows the Debug Perspective in iIT after a debug session has been started and execution is suspended at the Start object. In the Debug tab, you can see that the flow has a single thread (*W.debugger.1~flow1*) that is currently running. In the flow editor, you can see that execution is paused at the Start object. This is conveyed by a blue border around the object and the  icon, which appears in the upper-left corner of the object and indicates that execution is paused and waiting just before the document enters the Start object. The Variables tab in the right pane displays the current special registers and their values.

The main window's toolbar provides several actions that can be used to control flow execution:

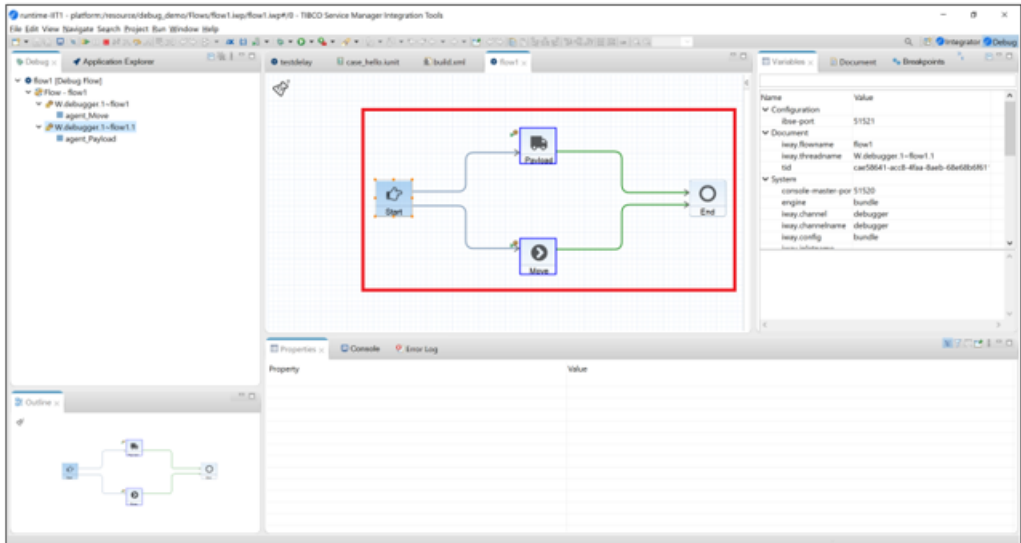


Clicking *Skip All Breakpoints* will ignore all breakpoints during the debug session that have been set. Clicking *Resume* continues execution of the flow. Clicking *Stop* terminates the debug session. Clicking *Step Over* moves to the next spot on the flow where execution may be suspended. In the example above, clicking *Step Over* once moves execution to the point right after execution of the Start object completes and before the document flows to the next

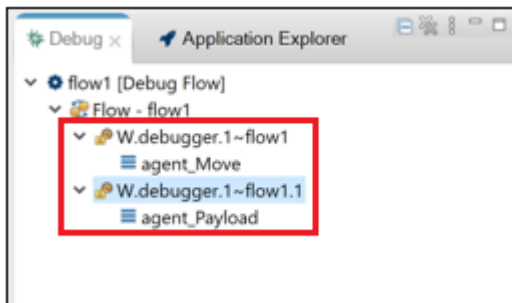
object(s) on the flow. Hovering over the  icon shows events returned by the object. For example:



Clicking *Step Over* one more time splits the execution into two branches, and you can now see two active threads, as shown in the following image.



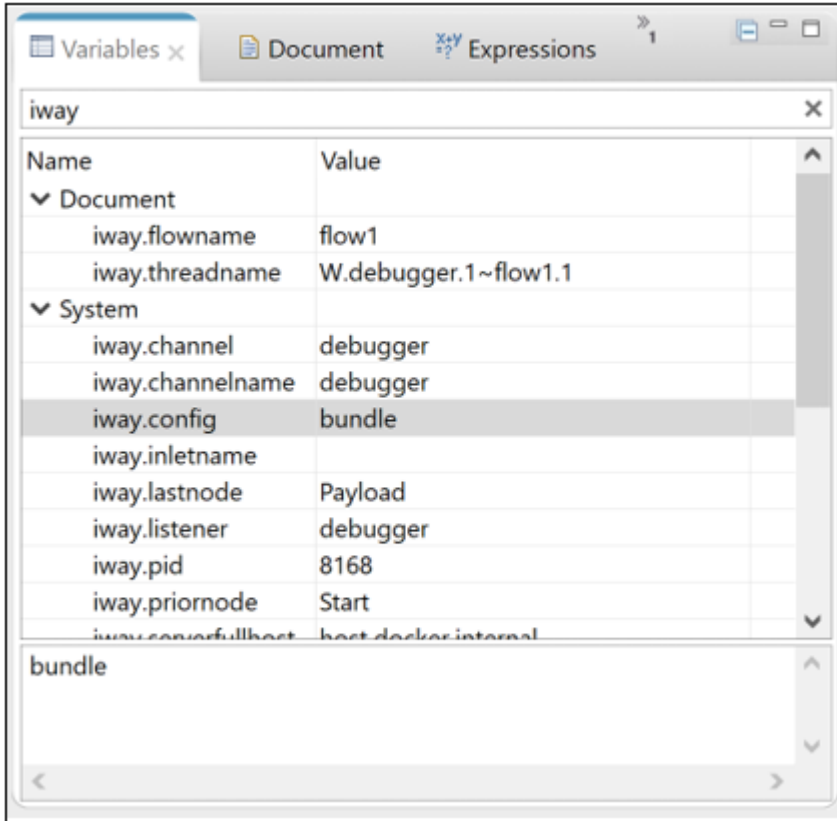
The Debug tab now shows two running threads.



Expanding the thread nodes shows the objects where execution is currently suspended. Select one of the threads or objects in the Debug tab or one of the highlighted objects on the canvas to control what is shown in the Variables, Document, and Expressions tabs.

*Variables Tab*

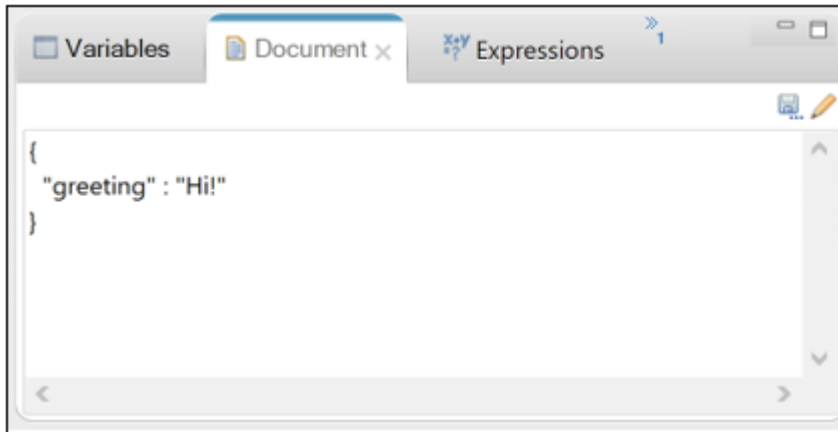
The Variables tab displays special registers (SREGs) that are currently available on the selected flow thread.





The registers are grouped by category. Selecting a variable and then clicking into the value cell allows you to update the current value of the register. Typing into the text field above the table allows you to narrow down the variables list.

### Document Tab

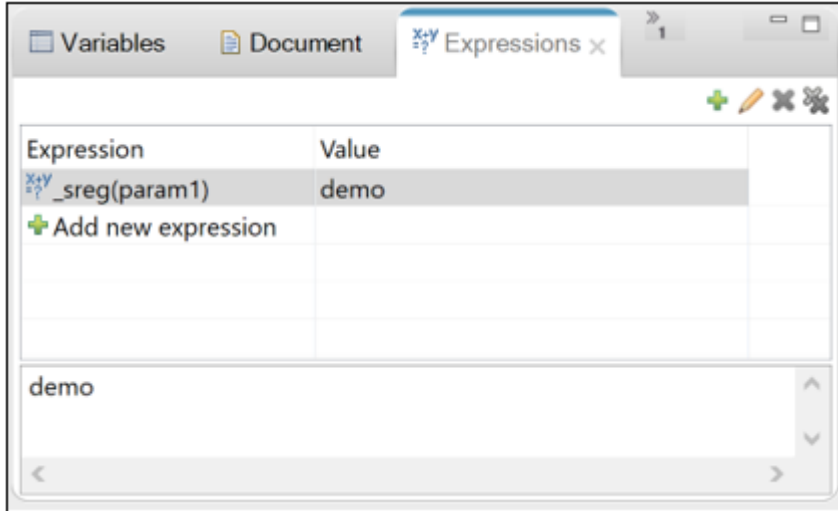
The Document tab shows the document arriving into or leaving from the object.



The Save icon  can be used to save the document into the workspace. The Override document  icon can be used to override the contents of the document on the currently selected thread.

### Expressions Tab

The Expressions tab allows you to add iWay Functional Language (iFL) expressions to be evaluated at the point where a flow's execution is currently suspended.



To add a new expression, click the *Add new expression* cell inside the table or click the icon from the toolbar. To edit an expression, select it first, and then click into the expression cell.

Alternatively, click the icon to open an edit dialog.

Click the Resume icon, allowing the flow run to completion.

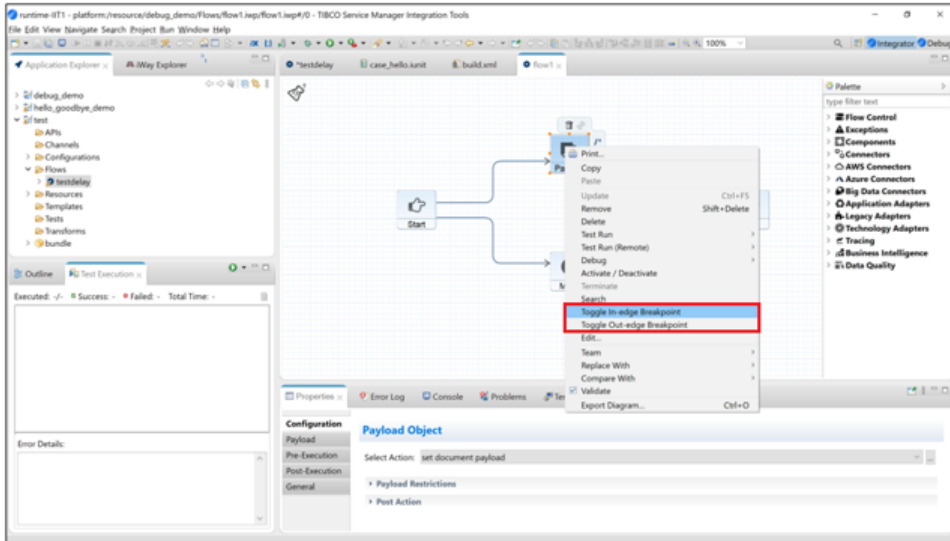
### Setting Breakpoints


Breakpoints can be set on flow objects at two locations:

- Before an object starts its execution (*In-edge Breakpoint*).
- After an object has completed its execution (*Out-edge Breakpoint*).



To set a breakpoint, right click on an object and select *Toggle In-edge Breakpoint* or *Toggle Out-edge Breakpoint* from the context menu, as shown in the following image.

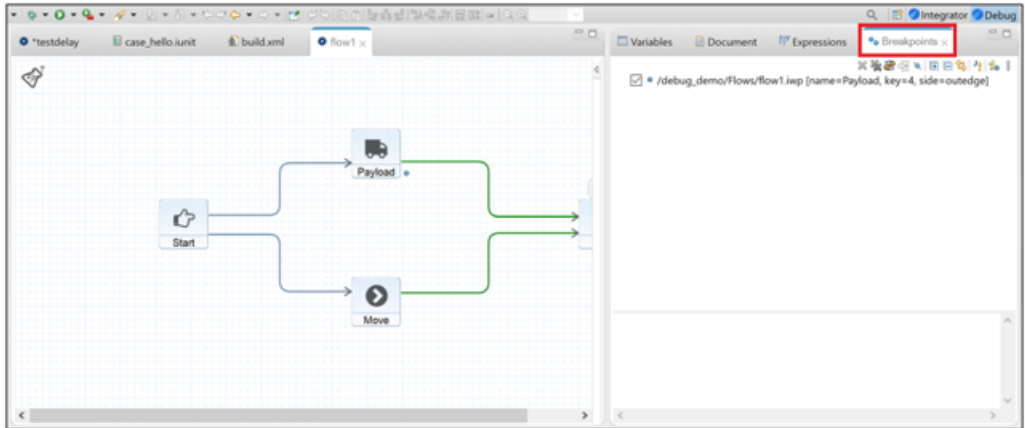


When an object has a breakpoint set, it is marked with the  icon. It appears on the lower-left or lower-right of an object, depending on whether it is an *in* or *out* edge breakpoint.

Once you have a breakpoint set, reopen the debug configuration you have created in previous steps, and uncheck the *Break at Start* object option in the Debug Configurations dialog. Click Debug. A debug session will start and the flow execution will pause at the object that has the breakpoint set.

## Breakpoint Configuration Options

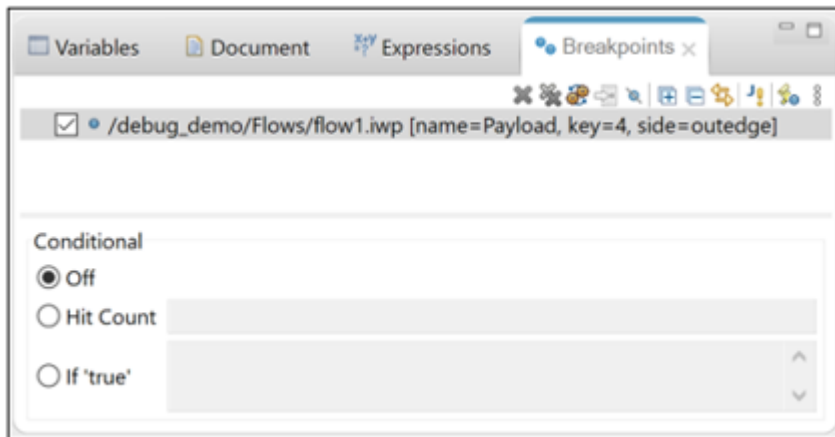
Once you have some breakpoints set, make sure you are in the Debug Perspective. Click the *Breakpoints* tab, as shown in the following image.



The Breakpoints tab displays all of the breakpoints that are currently set across all of the resources in the workspace.

Deselecting the check box next to a breakpoint disables that breakpoint (the breakpoint will be ignored during debug execution). Objects on the flow that have disabled breakpoints are marked with the  icon.

Selecting a breakpoint in the Breakpoints tab displays details for that breakpoint, as shown in the following image.

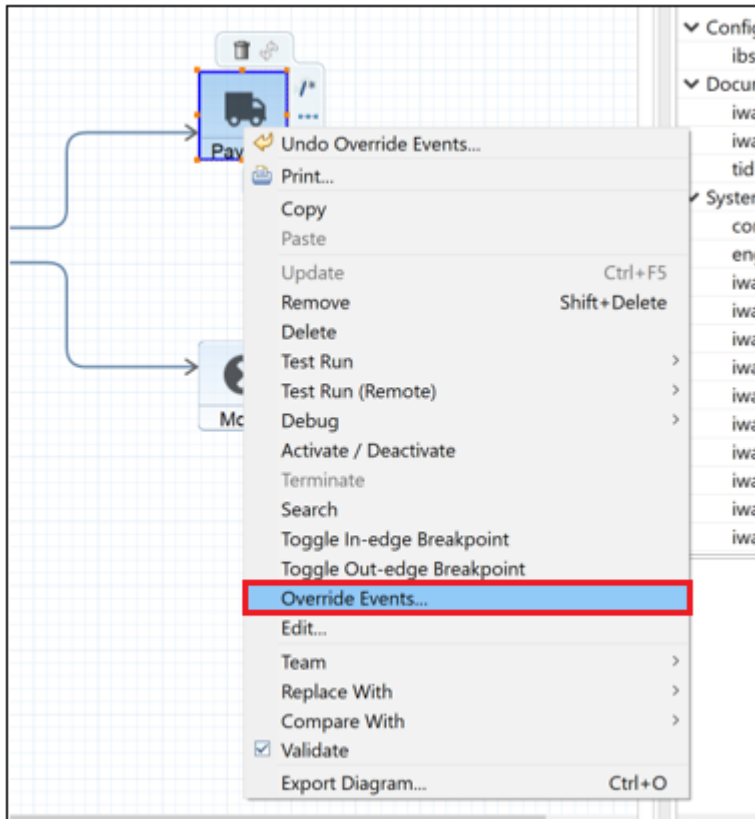


Breakpoint details allow you to control breakpoint execution.

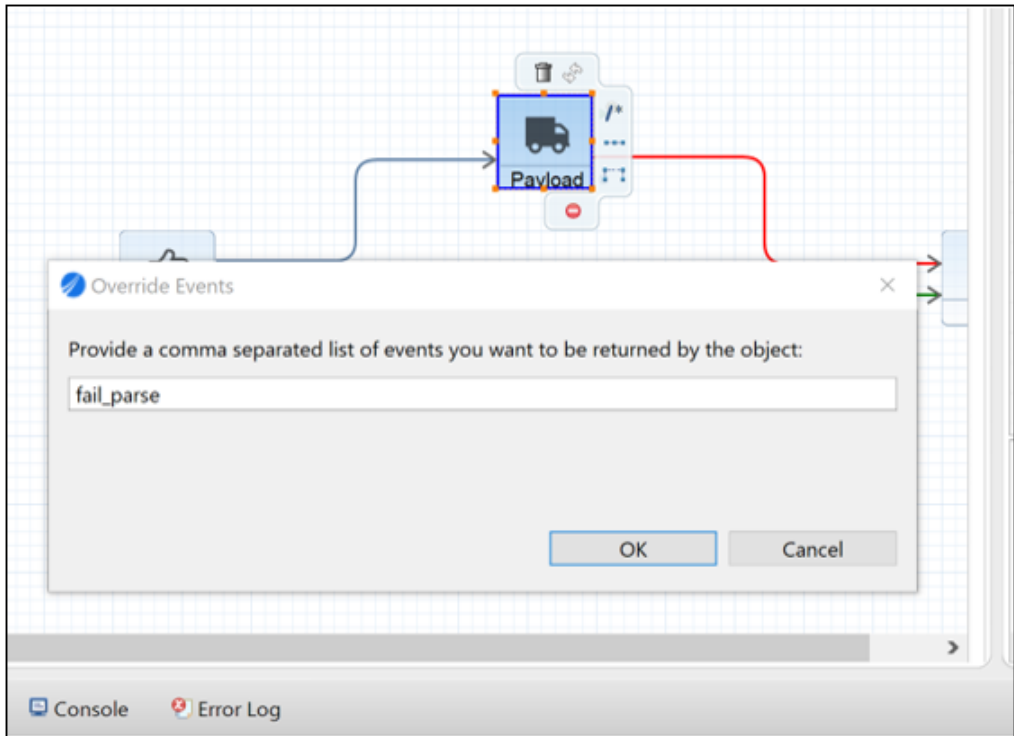
- Hit Count.** Execution will only be suspended after the breakpoint has reached the specified number of times.
- If 'true'.** Execution will only be suspended if the specified iFL expression evaluates as true.

## Overriding Events

When execution is suspended on the *out-edge* of an object, you can override events returned by the object. To do this, ensure flow execution is currently suspended on the out-edge of one of the objects. Right-click on the object and select *Override Events...*, as shown in the following image.



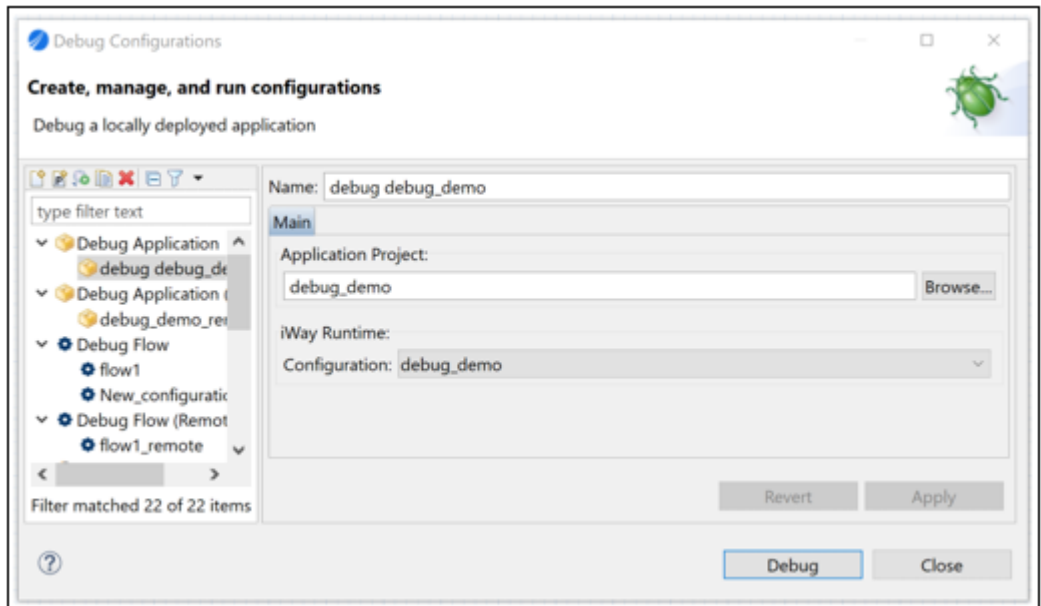
The Override Events dialog opens, displaying a comma-separated list of events that were returned by the object, as shown in the following image.



Make any required changes and then click *OK*. Continue debugging and observe the flow reacting to your changes.

## Application Debug Mode

As previously mentioned, flows may be debugged in the context of applications. To do this, first ensure you have an application project with a channel or an API with some configured endpoints. Then, open the Debug Configurations dialog and configure a new *Debug Application* configuration, as shown in the following image.



Click *Debug* when you have finished configuring the Debug Application. A new debug session starts.

Add some breakpoints to the flow(s) in the application project. Trigger those breakpoints by either invoking an API endpoint or submitting a document to the listener. When finished, remember to stop the debug session. After making changes to any flow(s) being debugged, ensure to restart the runtime configuration to make sure the flow sources in iIT are in sync with the flows deployed to the embedded iWay runtime.

## Debugging Flows on Remote Servers

You can also debug against an external iWay server (iSM) using test servers. This is similar to test running flows against a remote server. To support this functionality, two additional debug configurations are provided:

- Debug Application (Remote)

❑ Debug Flow (Remote)

Instead of specifying an iWay Runtime Configuration, you supply connection information for an external iSM server. The Application project is built and deployed as a test server instance on the external iSM server, and the debug process proceeds similarly to debugging against an embedded server.

## Code Extensions and Extension Projects

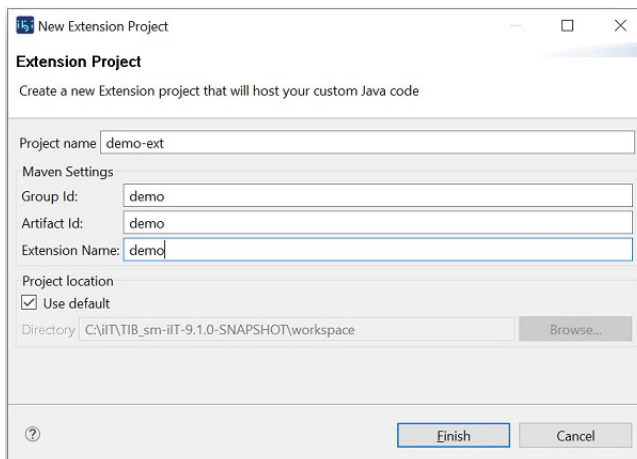
iWay Service Manager (iSM) version 9.1.0 provides a new Java SDK for extending iSM that allows you to write code extensions, and then run and debug those code extensions within ibi™ iWay® Service Manager Integration Tools (iIT). Code extensions are hosted within the newly introduced Extension projects. Each Extension project is Maven-enabled. When built using Maven commands, an extension .jar file is generated. Extension .jar files can be referenced by your Application project(s) to make the extensions immediately accessible when designing applications and also executing process flows. iIT includes component wizards for Agents, Functions, and Activity Drivers.

## Creating an Extension Project

To create an Extension project:

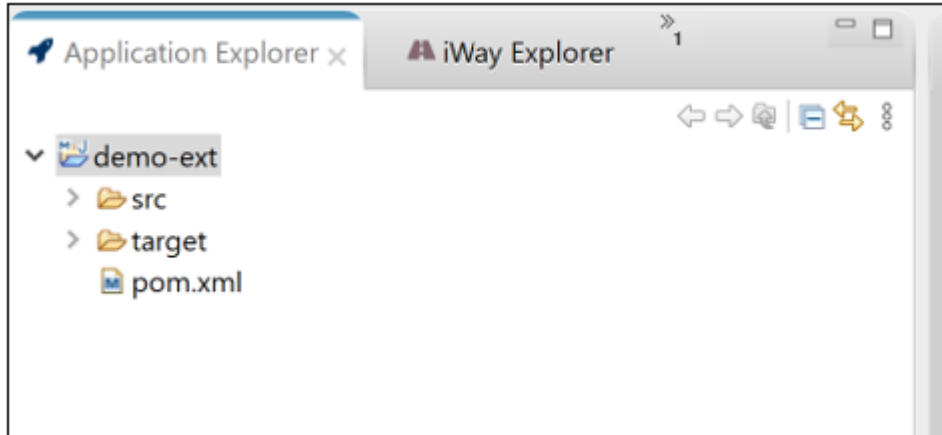
1. From the main menu, click *File*, select *New*, and then click *Extension Project*.

The New Extension Project dialog opens, as shown in the following image.



2. Provide values for the Project name, Group Id, Artifact Id, and Extension Name fields.
3. Click *Finish*.

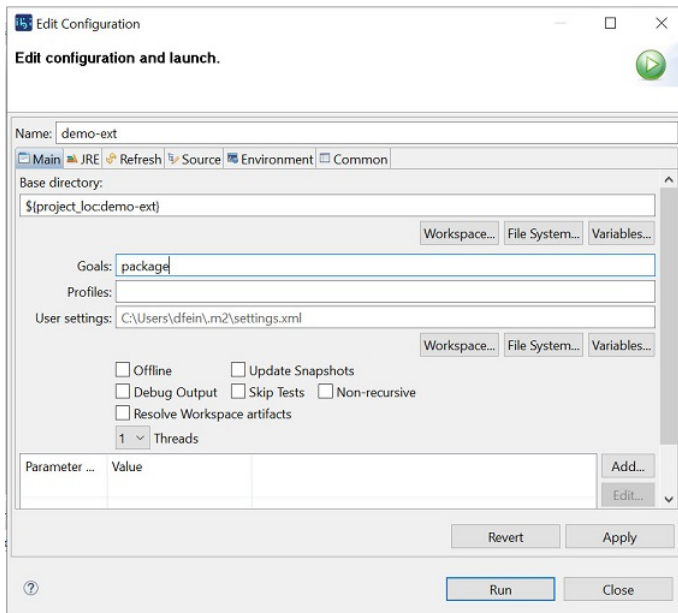
A new Extension project is created in the workspace, as shown in the following image.



The Extension project's build path is automatically initialized with everything required to compile iWay extensions code.

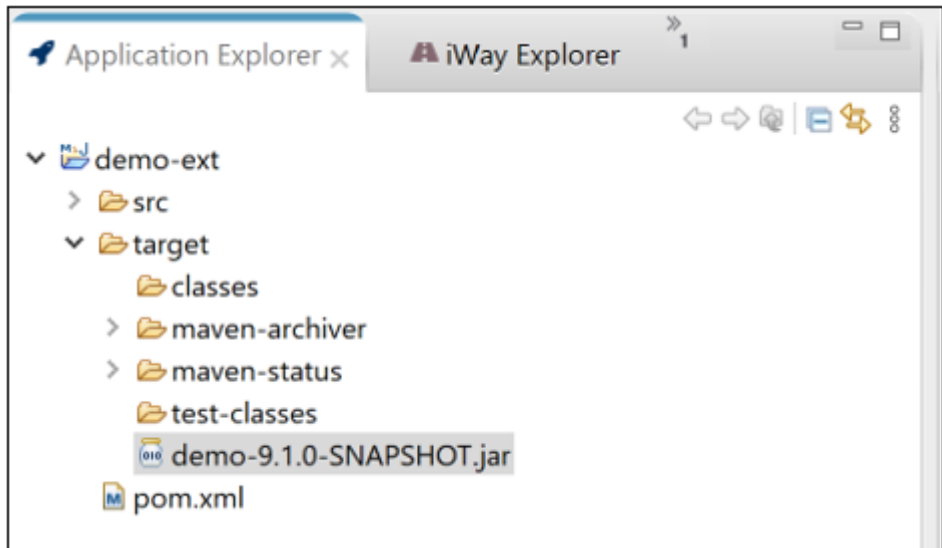
4. Right-click the *pom.xml* file, select *Run As*, and then click *Maven build*.

The Edit Configuration dialog opens, as shown in the following image.



5. Type *package* in the Goal field and click *Run*.

The Extension project is built and a corresponding extension .jar file is created in the target folder, as shown in the following image.



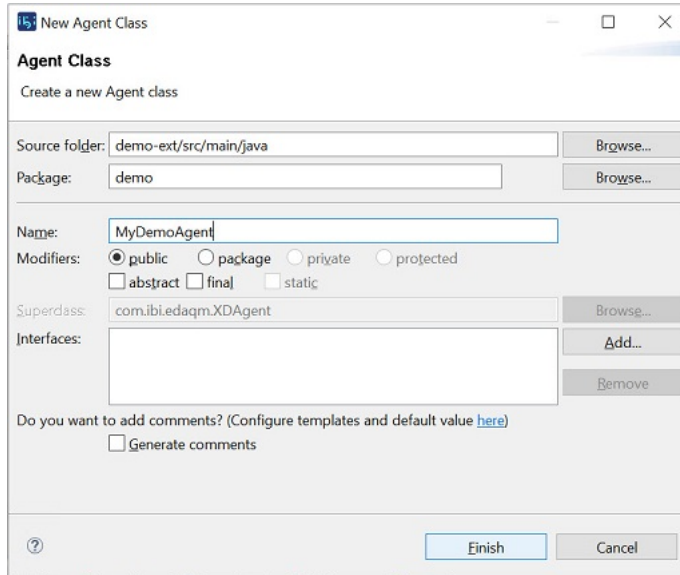
### Writing Agent Class Extensions

This section describes how to implement a new Agent class, which will be used later in a process flow.

1. Right-click an Extension project (for example, *demo-ext*), select *New*, and then click *Agent Class*.

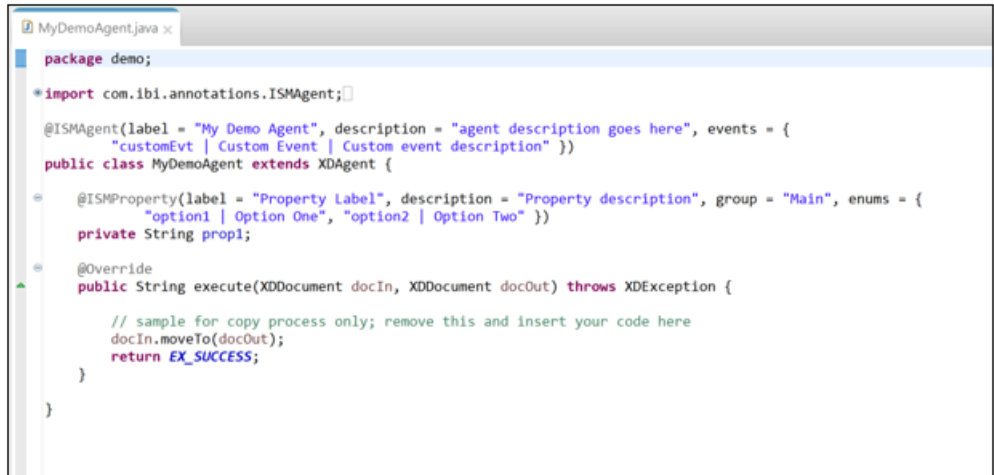


The New Agent Class dialog opens, as shown in the following image.



2. Provide a package name and a name for the new agent and press *Finish*.

A new class is created and automatically opens in a class editor, as shown in the following image.



The class is automatically populated with a sample placeholder property, and metadata defined via annotations.

The `@ISMAgent` annotation supports a number of different attributes that describe and/or control the behavior of an agent. The following table lists and describes the supported attributes.

Attribute	Description
categories	Provides a list of one or more category names, which allow for grouping agents.
deprecated	Marks the agent as deprecated.
description	Provides a summary of the agent's functionality.
events	Supplies an array of agent-specific events that the agent is capable of returning. Each event is represented by a string in the format "value   label   description". Value is the actual string that is returned by the agent at runtime, while label and description are used to display events at design time.
iterator	Marks the agent as an iterator.
label	Provides the agent display name.
supportsUserParams	Signals that the agent supports user-defined name/value pairs of parameters.
visible	Used to hide certain agents in the UI.

Agents may declare one to  $n$  number of properties, which will be automatically initialized during runtime. Each of these properties must be annotated with `@ISMProperty`. The following table lists and describes the attributes supported by `@ISMProperty`.

Attribute	Description
description	Provides a paragraph describing the property's purpose and functionality.

Attribute	Description
display	Enables special property rendering. For example, to render a password control, specify <code>display = Display.PASSWORD</code> .
enums	Supplies an array of possible values the property may be initialized with. Used to render a drop-down control. Each enum value is represented by a string in the format "value   label".
group	Used to group properties into property sections.
label	Provides the property display name.
required	Specifies whether the property is required.
value	Provides the default property value.

Let's implement an agent whose configuration consists of a single string property, providing a system path location. The agent will test for the existence of the resource at the specified location, and also will test whether it is a file or a directory. It will return one of three possible events *unresolved*, *file*, or *dir*. The agent will also construct an XML output document in the form `<PathTest path="some path" status="status"/>`, where *status* can be *unresolved*, *file*, or *dir*.

```

MyDemoAgent.java x
package demo;

import java.io.File;

@ISMAgent(label = "Test Path Agent", description = "tests system path", events = {
    "unresolved | Unresolved Path | The path cannot be resolved", "file | File Path | Path is a file",
    "dir | Directory Path | Path is a directory" })
public class MyDemoAgent extends XDAgent {
    private static final String UNRESOLVED = "unresolved";
    private static final String FILE = "file";
    private static final String DIR = "dir";

    @ISMProperty(label = "Path", description = "System path to test", group = "Main", required = true)
    private String path;

    @Override
    public String execute(XDDocument docIn, XDDocument docOut) throws XDEException {
        XDNode test = XDNode.newNode("PathTest");
        test.setAttribute("path", path);
        docOut.setRoot(test);

        File file = new File(path);
        if (!file.exists()) {
            test.setAttribute("status", UNRESOLVED);
            return UNRESOLVED;
        } else if (file.isFile()) {
            test.setAttribute("status", FILE);
            return FILE;
        } else {
            test.setAttribute("status", DIR);
            return DIR;
        }
    }
}

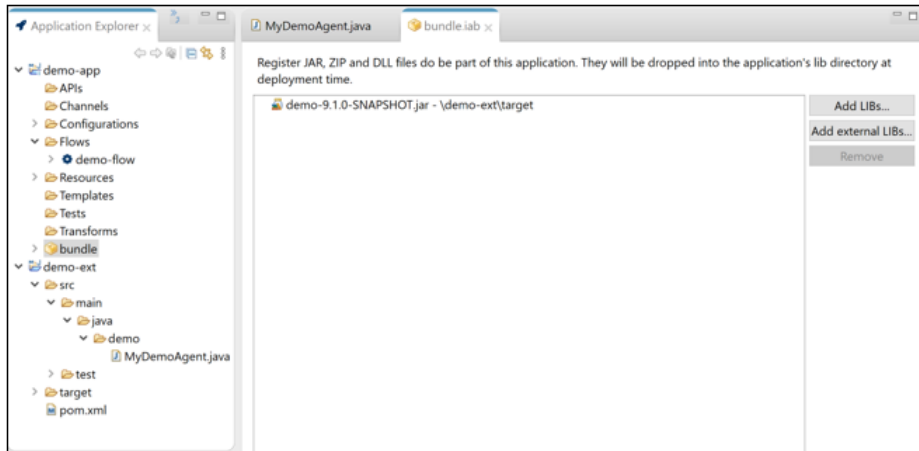
```

1. Right-click the *pom.xml* file, select *Run As*, and click *Maven build* to rebuild the extension *.jar* file.

Now let's call our new agent class from a process flow. First, create a new application project called *demo-app*.

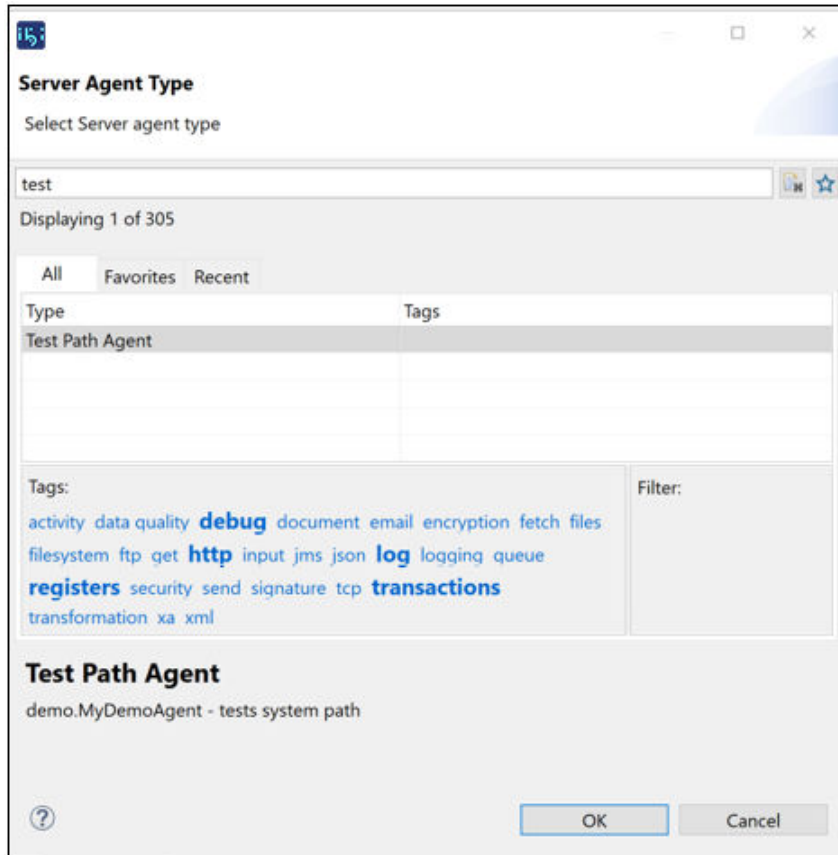
2. From the main menu, click *File*, select *New*, and then click *Application Project*.
3. Enter *demo-app* for the Project Name and click *Finish*.
4. Open the new application project's bundle resource in an editor, navigate to the *Libraries* tab, and click *Add LIBs*.
5. Select the extension *.jar* file from the *demo-ext* project you configured earlier and click *OK*.

This will make our extension .jar file a dependency of the *demo-app* project, exposing its functionality at design time and also runtime.



6. Save and close the bundle editor.
7. Create a new process flow called *demo-flow*.
8. Drag and drop the *Server Agent* object from the palette, and click the ellipsis button (...) to select an agent to configure.

The Server Agent Type dialog opens, as shown in the following image.

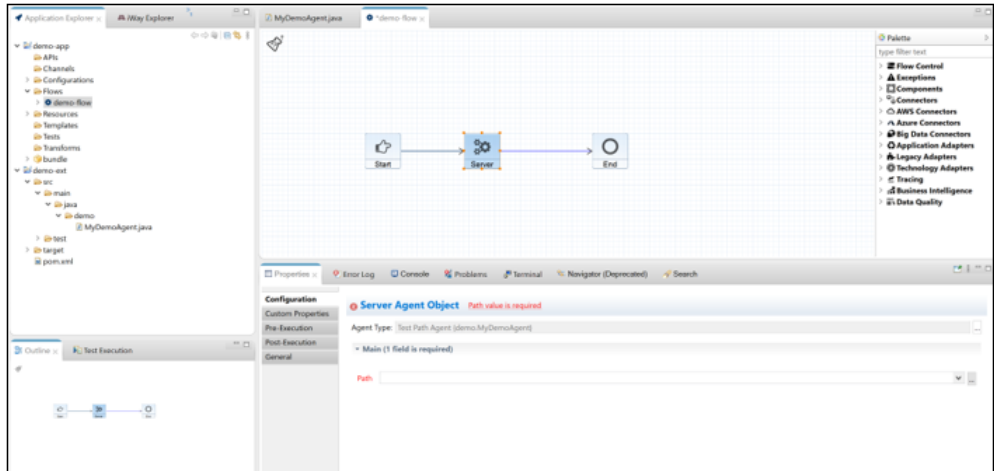


9. Type the word `test` to filter the list of agents.

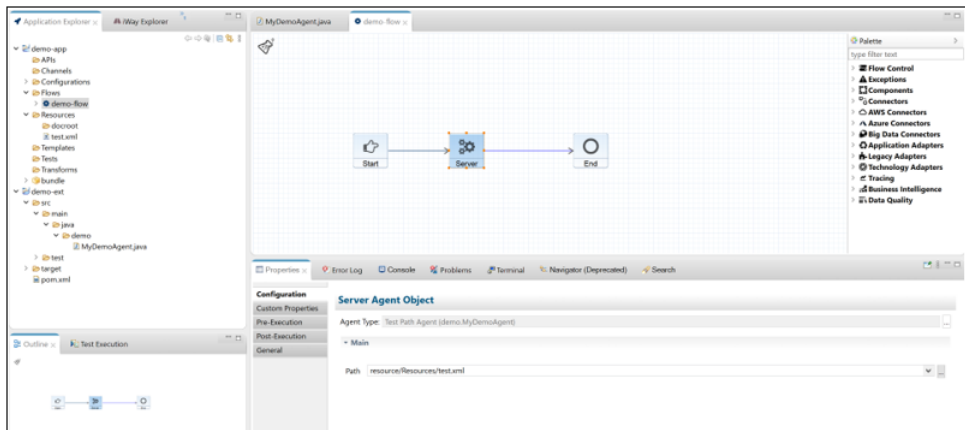
The new agent you recently created is found in the list.

10. Click `OK`.

Your process flow should now appear as shown in the following image.



11. Create a file under `demo-app/Resources/test.xml`, then enter `resource/Resources/test.xml` into the Path text field to finish configuring the Server object.



12. Drop two more End objects on the canvas.

Connect and configure as shown in the series of three screenshots that follow. Ensure to select the connection Event types as shown.

### Screenshot 1: File Path

The screenshot shows an IDE window with a flow diagram and a configuration panel. The flow diagram consists of a 'Start' node (blue square with a hand icon) connected to a 'Server' node (blue square with a gear icon). From the 'Server' node, three arrows branch out: a blue arrow to an 'Is File' node (blue square with a circle icon), a purple arrow to an 'Is Dir' node (blue square with a circle icon), and a green arrow to an 'Unresolved' node (blue square with a circle icon). The configuration panel, titled 'Execution Path', contains a table of events with checkboxes. The 'File Path' checkbox is checked.

**Execution Path**

*The execution path will always be followed unless configured to only execute if one or more events are selected.*

Events:

Name	Description
<input type="checkbox"/> OnSuccess	The operation was succesful.
<input type="checkbox"/> OnFailure	A fail condition occurred during execution.
<input type="checkbox"/> OnError	An exception occurred during execution.
<input type="checkbox"/> Unresolved Path	The path cannot be resolved
<input checked="" type="checkbox"/> File Path	Path is a file
<input type="checkbox"/> Directory Path	Path is a directory



### Screenshot 2: Directory Path

The screenshot displays the iWay Service Manager interface. At the top, there are tabs for 'MyDemoAgent.java' and '\*demo-flow x'. The main workspace shows a flow diagram on a grid background. The flow starts with a 'Start' node (hand icon), followed by a 'Server' node (gears icon). From the 'Server' node, three paths emerge: a solid blue line to an 'Is File' node, a dashed orange line to an 'Is Dir' node, and a solid green line to an 'Unresolved' node. Below the workspace is a toolbar with icons for Properties, Error Log, Console, Problems, Terminal, Navigator (Deprecated), and Search.

The 'Configuration' panel on the left shows the 'Execution Path' settings. The 'Directory Path' checkbox is checked, while others are unchecked.

Name	Description
<input type="checkbox"/> OnSuccess	The operation was successful.
<input type="checkbox"/> OnFailure	A fail condition occurred during execution.
<input type="checkbox"/> OnError	An exception occurred during execution.
<input type="checkbox"/> Unresolved Path	The path cannot be resolved
<input type="checkbox"/> File Path	Path is a file
<input checked="" type="checkbox"/> Directory Path	Path is a directory

**Screenshot 3: Unresolved Path**

The screenshot shows a process flow diagram in an IDE. The diagram consists of the following nodes and connections:

- Start** (thumbs up icon) → **Server** (gears icon)
- Server** → **Is File** (circle icon)
- Server** → **Is Dir** (circle icon)
- Server** → **Unresolved** (circle icon)

The 'Unresolved' node is highlighted with a dashed orange arrow, indicating it is the current focus. Below the diagram is a configuration panel for the 'Unresolved Path' event.

**Configuration**

**Execution Path**

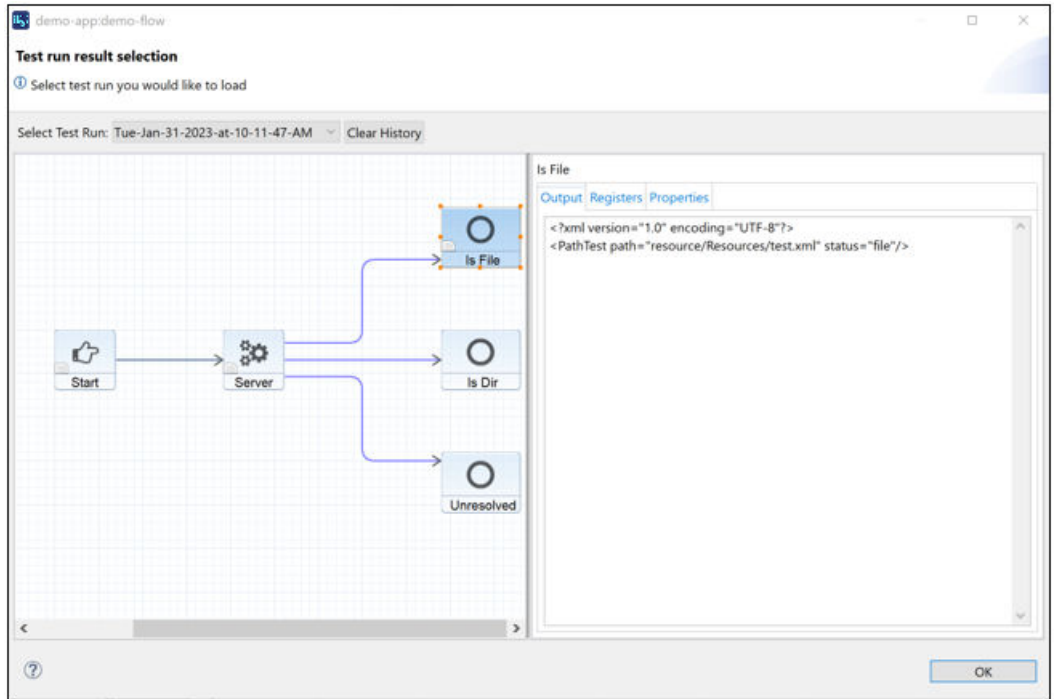
*The execution path will always be followed unless configured to only execute if one or more events are selected.*

Events:

Name	Description
<input type="checkbox"/> OnSuccess	The operation was succesful.
<input type="checkbox"/> OnFailure	A fail condition occurred during execution.
<input type="checkbox"/> OnError	An exception occurred during execution.
<input checked="" type="checkbox"/> Unresolved Path	The path cannot be resolved
<input type="checkbox"/> File Path	Path is a file
<input type="checkbox"/> Directory Path	Path is a directory

13. Test run the process flow.

The document reaches the *Is File* End object, with the output document containing  
`<PathTest path="resource/Resources/test.xml" status="file"/>`.



## Refreshing Metadata

Let's go back to our agent's source code and add another property. We will add a boolean property called *reportFileSize*. If the property is set to *true* and the path contains a file, then we will add a size attribute to the payload containing the size of the file.

```
MyDemoAgent.java x demo-flow

*import java.io.File;

@ISMAgent(label = "Test Path Agent", description = "tests system path", events = {
    "unresolved | Unresolved Path | The path cannot be resolved", "file | File Path | Path is a file",
    "dir | Directory Path | Path is a directory" })
public class MyDemoAgent extends XDAgent {
    private static final String UNRESOLVED = "unresolved";
    private static final String FILE = "file";
    private static final String DIR = "dir";

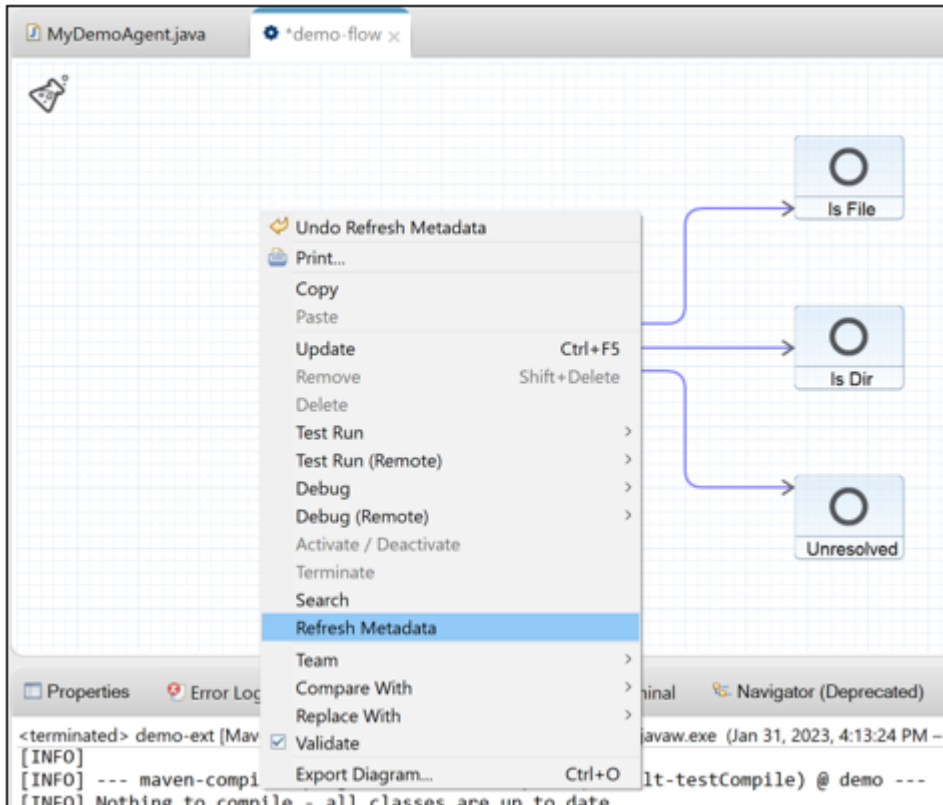
    @ISMProperty(label = "Path", description = "System path to test", group = "Main", required = true)
    private String path;

    @ISMProperty(label = "Report File Size", description = "If path contains a file, set attribute size", group = "Main")
    private boolean reportFileSize;

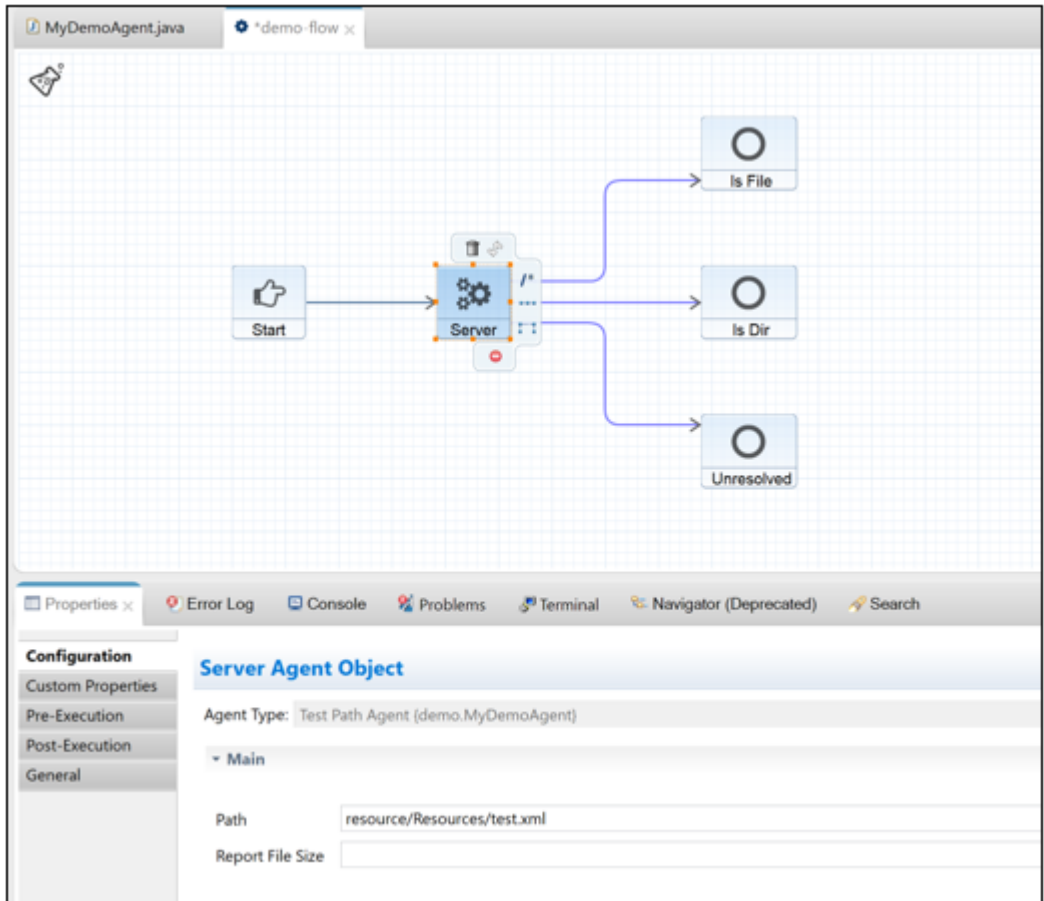
    @Override
    public String execute(XDDocument docIn, XDDocument docOut) throws XDEException {
        XDNode test = XDNode.newNode("PathTest");
        test.setAttribute("path", path);
        docOut.setRoot(test);

        File file = new File(path);
        if (!file.exists()) {
            test.setAttribute("status", UNRESOLVED);
            return UNRESOLVED;
        } else if (file.isFile()) {
            test.setAttribute("status", FILE);
            if (reportFileSize) {
                test.setAttribute("size", Long.toString(file.length()));
            }
            return FILE;
        } else {
            test.setAttribute("status", DIR);
            return DIR;
        }
    }
}
```

Save the class, then right-click *demo-ext/pom.xml*, select *Run As*, and click *Maven build* to rebuild the extension .jar file. Click the *demo-flow* editor tab. Right-click anywhere on the canvas and select *Refresh Metadata*, as shown in the following image.

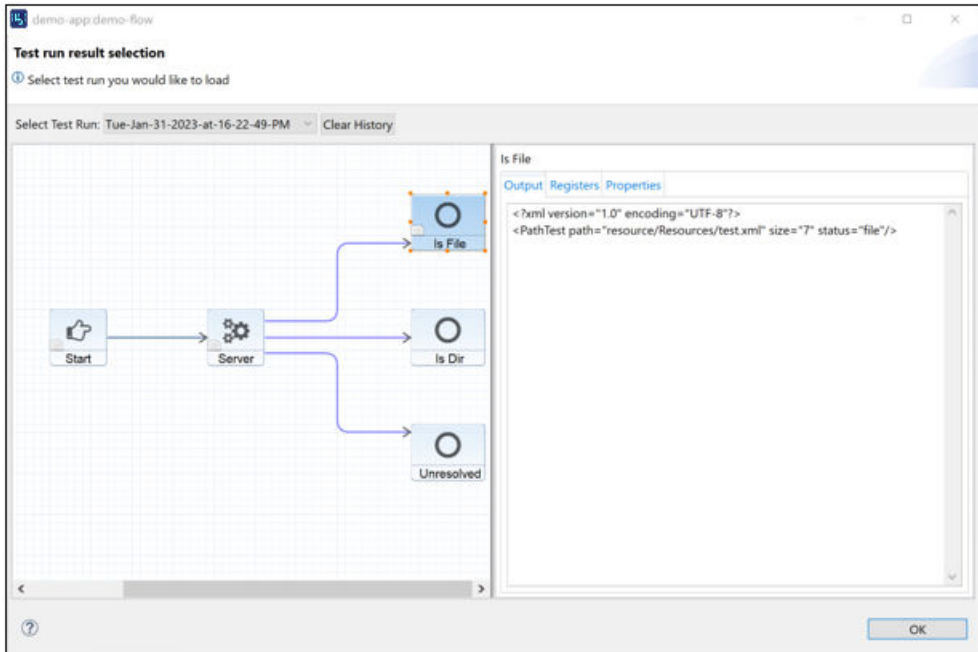


Reselect the *Server* object and notice the new property appear in the Properties tab.



If you do not see the new property, select a different object on the process flow and then reselect the *Server* object.

Set the *Report File Size* parameter to *true* and rerun the process flow, making sure to restart the embedded server. Review the output document, which now has the extra size attribute.

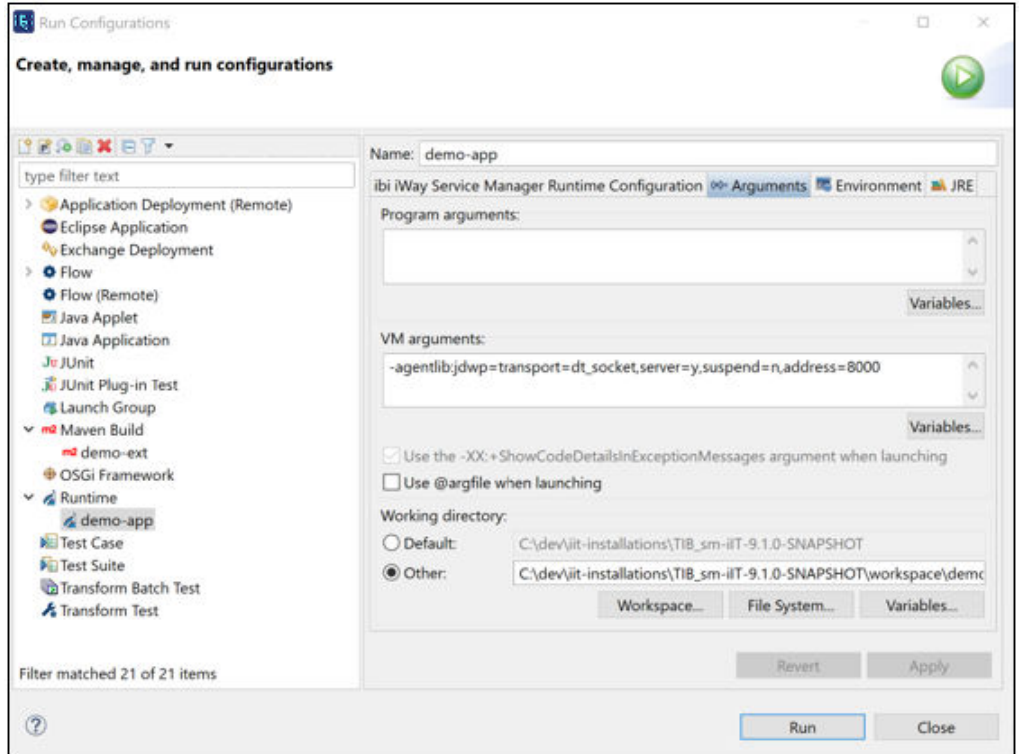


## Debugging Agent Code

It is possible to Step debug agent code without leaving iIT. You will need to edit the runtime configuration for the *demo-app* Application Project.

1. From the main menu, click *Run*, and then *Run Configurations*.

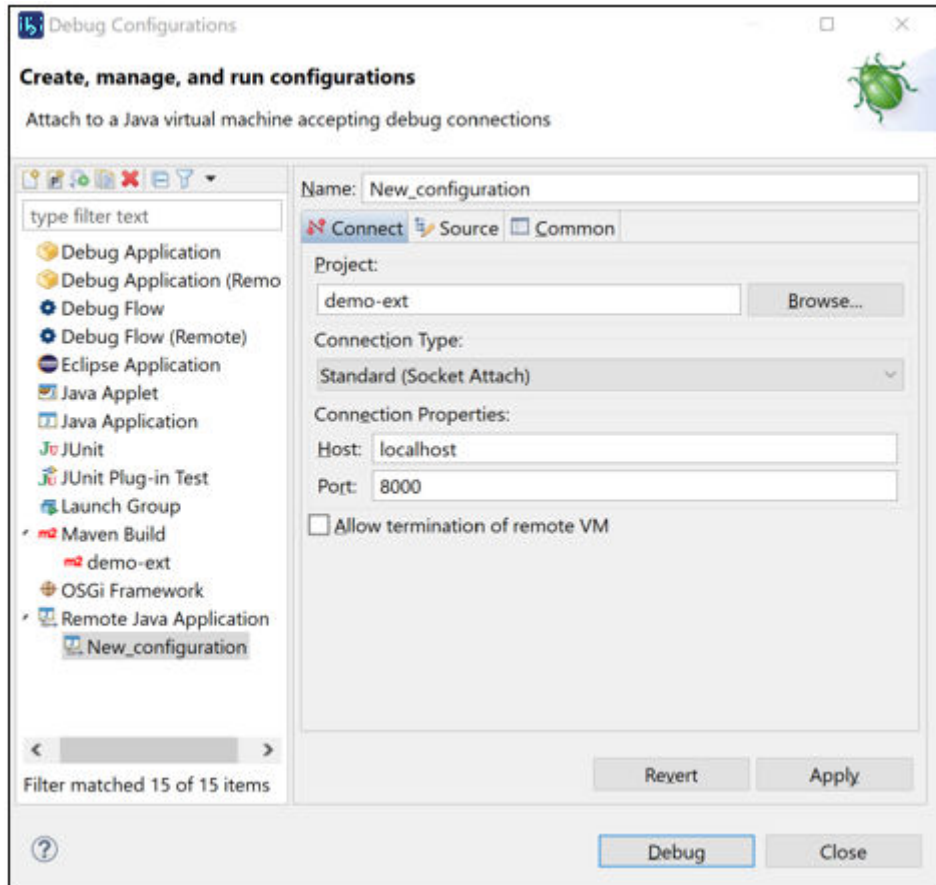
The Run Configurations dialog opens, as shown in the following image.



2. In the left pane, expand *Runtime* and select *demo-app*.
3. Click the *Arguments* tab and paste the following into the VM arguments field:  
`-agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=8000`
4. Click *Apply* then *Close*, and then restart the *demo-app* embedded server.  
The embedded server starts in debug mode listening on port 8000.
5. From the main menu, click *Run*, and then *Debug Configurations*.



The Debug Configurations dialog opens, as shown in the following image.



6. Expand *Remote Java Application* and select *New\_configuration*.
7. Specify *demo-ext* in the Project field.
8. Click *Debug*.

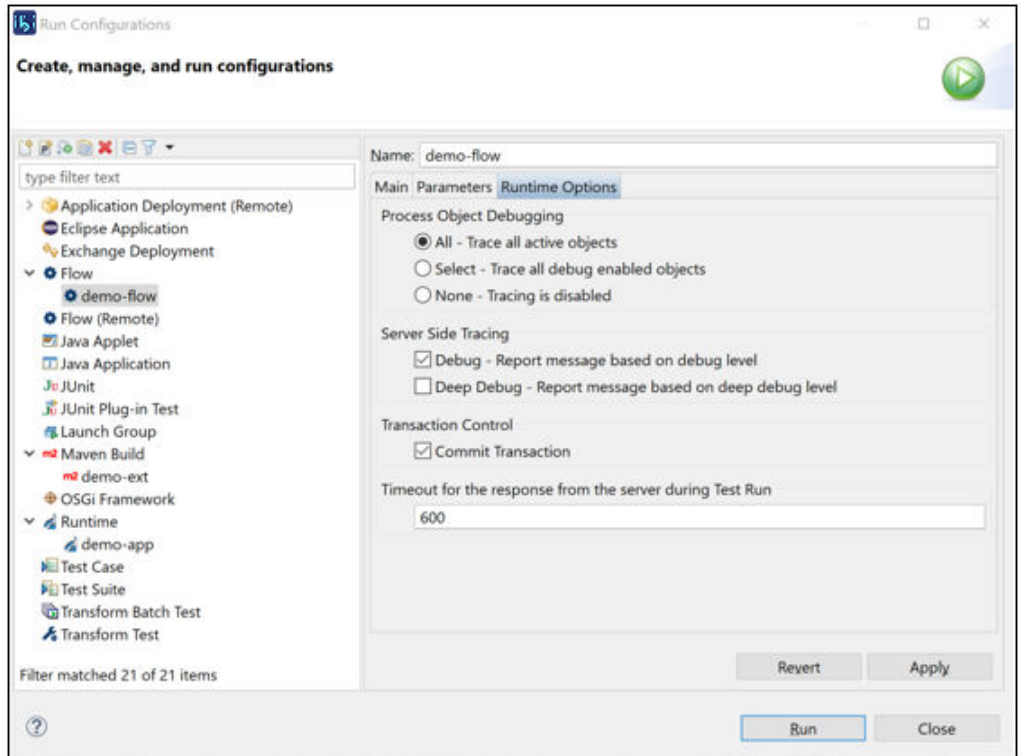
You now have a debug session started.

9. Open the agent class and set a breakpoint at the first line of the `execute()` method.
10. Trigger the breakpoint by invoking a test run.

First let's increase the flow timeout.

11. From the main menu, click *Run*, and then *Run Configurations*.

The Run Configurations dialog opens, as shown in the following image.

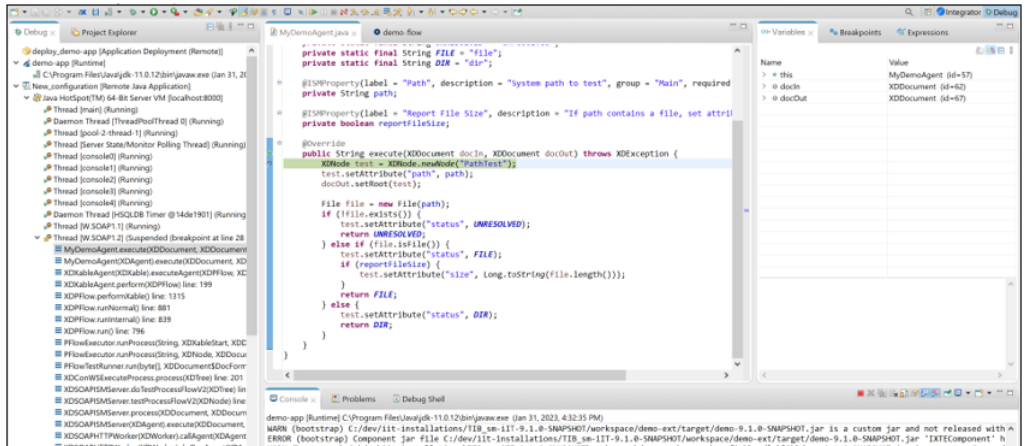


12. Expand *Flow* in the left pane and select *demo-flow*.

13. Click the *Runtime Options* tab, and set the Timeout value to 600.

14. Click *Run*.

The breakpoint is triggered, and you can step through the code (inspecting variables, etc.), as shown in the following image.

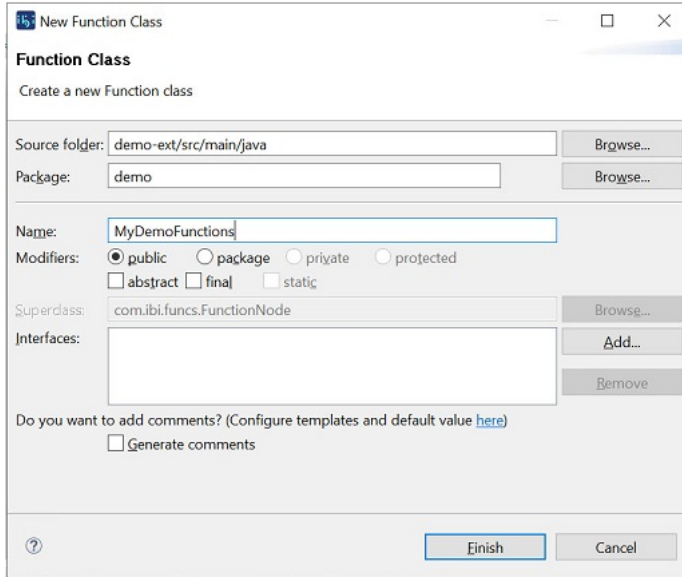


## Writing iWay Functional Language Class Extensions

This section describes how to write custom iWay Functional Language (iFL) functions, which can then be used in a process flow.

1. Right-click the `demo-ext` Extension project, select `New`, and then click `Function Class`.

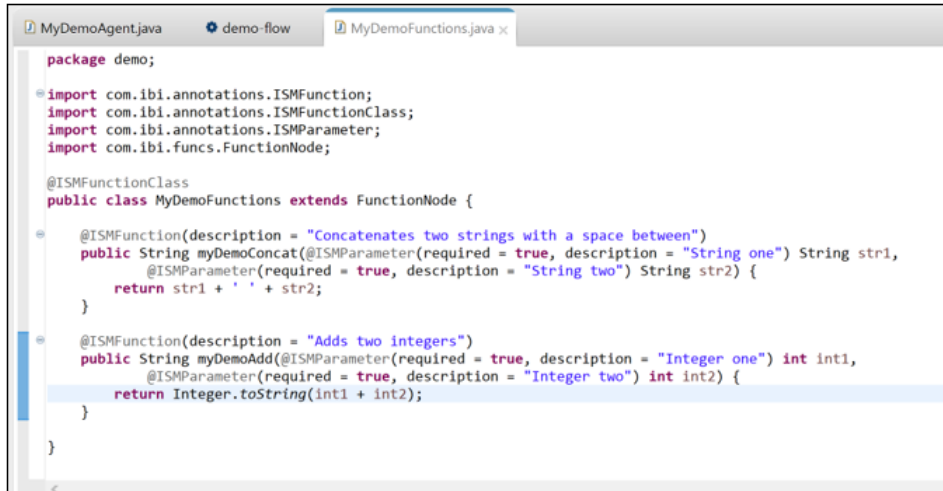
The New Function Class dialog opens, as shown in the following image.



2. Provide a class name and click *Finish*.

Every iFL Function class must be annotated with `@ISMFunctionClass` and must extend `com.ibi.funcs.FunctionNode`. It also needs to declare one to  $n$  number of methods annotated with `@ISMFunction`, and whose return type is `String`. Each of these functions may have 0 to  $n$  number of parameters. Declare the first parameter to be of type `com.ibi.funcs.IFunctionContext` if your function needs access to some server state or services (for example, document, special registers, logger). The `IFunctionContext` parameter is optional. Additionally, you may declare 1 to  $n$  number of parameters annotated with `@ISMParameter`. They may be of any primitive type, such as `int`, `String`, `float`, `boolean`, etc.

Let's implement two functions. The first function will concatenate two string arguments with a space between, and the second function will add two integers, returning the result.



```

package demo;

import com.ibi.annotations.ISMFunction;
import com.ibi.annotations.ISMFunctionClass;
import com.ibi.annotations.ISMParameter;
import com.ibi.funcs.FunctionNode;

@ISMFunctionClass
public class MyDemoFunctions extends FunctionNode {

    @ISMFunction(description = "Concatenates two strings with a space between")
    public String myDemoConcat(@ISMParameter(required = true, description = "String one") String str1,
        @ISMParameter(required = true, description = "String two") String str2) {
        return str1 + ' ' + str2;
    }

    @ISMFunction(description = "Adds two integers")
    public String myDemoAdd(@ISMParameter(required = true, description = "Integer one") int int1,
        @ISMParameter(required = true, description = "Integer two") int int2) {
        return Integer.toString(int1 + int2);
    }
}

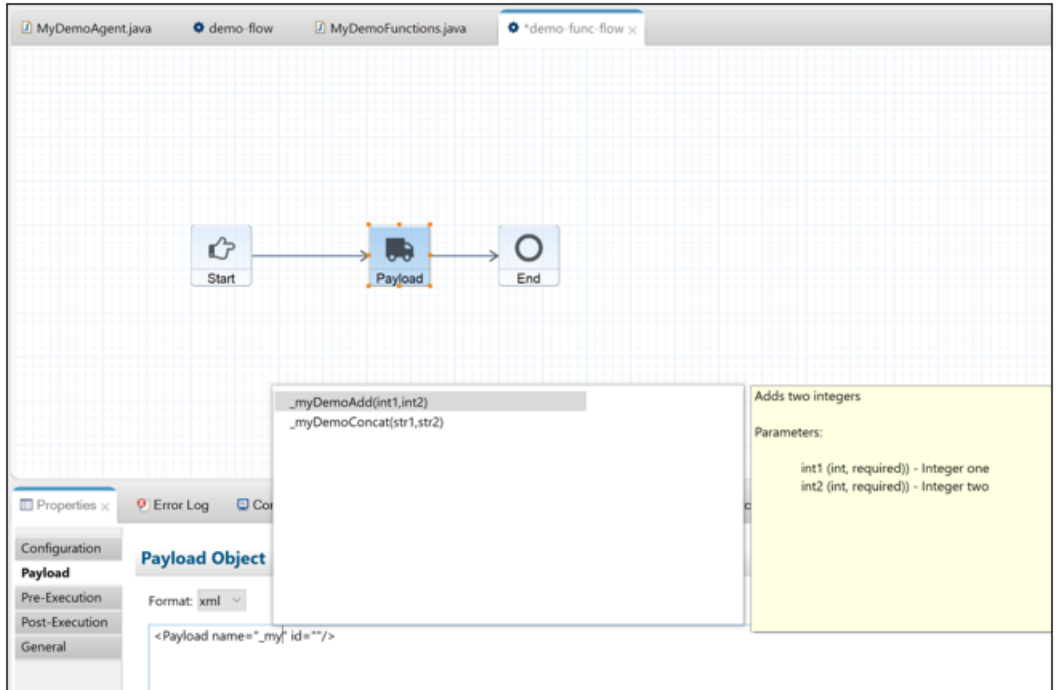
```

3. Save the class.
4. Right-click *demo-ext/pom.xml*, select *Run As*, and then click *Maven build* to rebuild the extension *.jar* file.
5. Create a new process flow called *demo-func-flow*.
6. Drag and drop the *Payload* object on the line.
7. In the *Payload* properties tab, paste the following XML:
 

```
<Payload name="" id="" />
```
8. Place the cursor inside the quotes for the value of the *name* attribute, and then type the underscore (`_`) character.
 

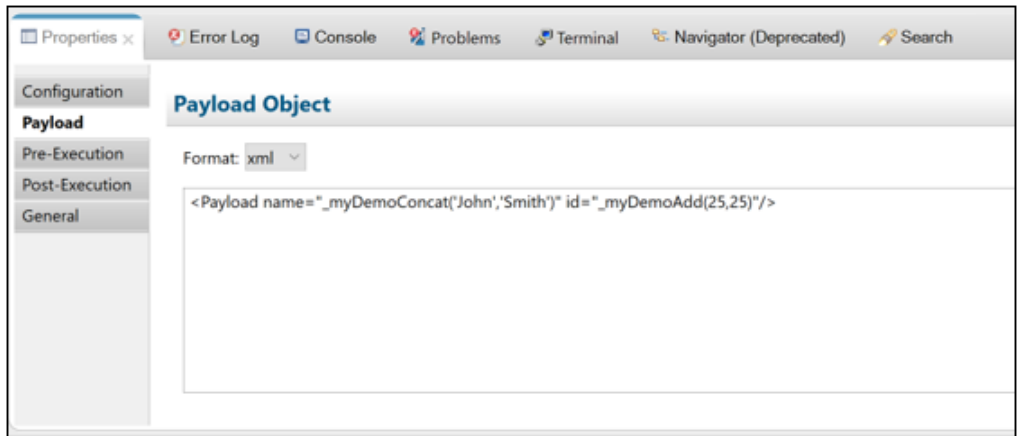
The autocomplete dialog opens.
9. Start typing *myDemo* and see the list of iFL functions narrow down until the two functions you implemented previously appear.

10. Navigate with up/down arrows to inspect the auto-generated function doc.



11. Double-click the `_myDemoConcat` function to select and provide argument values.

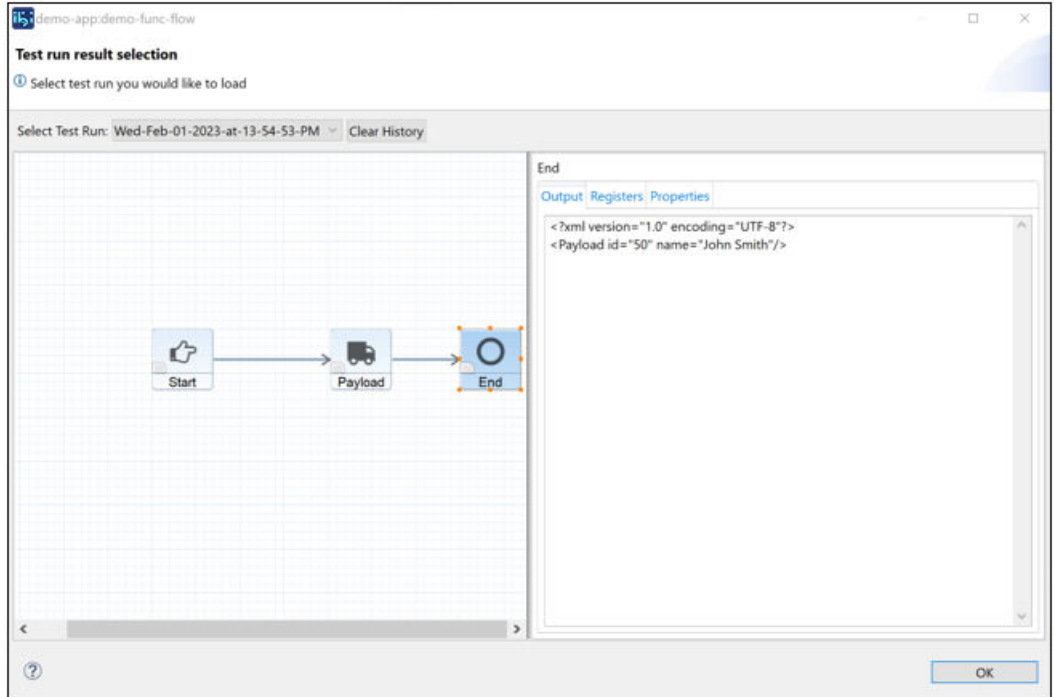
12. Repeat the steps for `id`, this time using the `_myDemoAdd` function.



13. Test run the process flow, making sure to restart the embedded server.

**Note:** Since the extension .jar file was updated, the server must be restarted in order to have the latest classpath.

14. Inspect the resulting document.

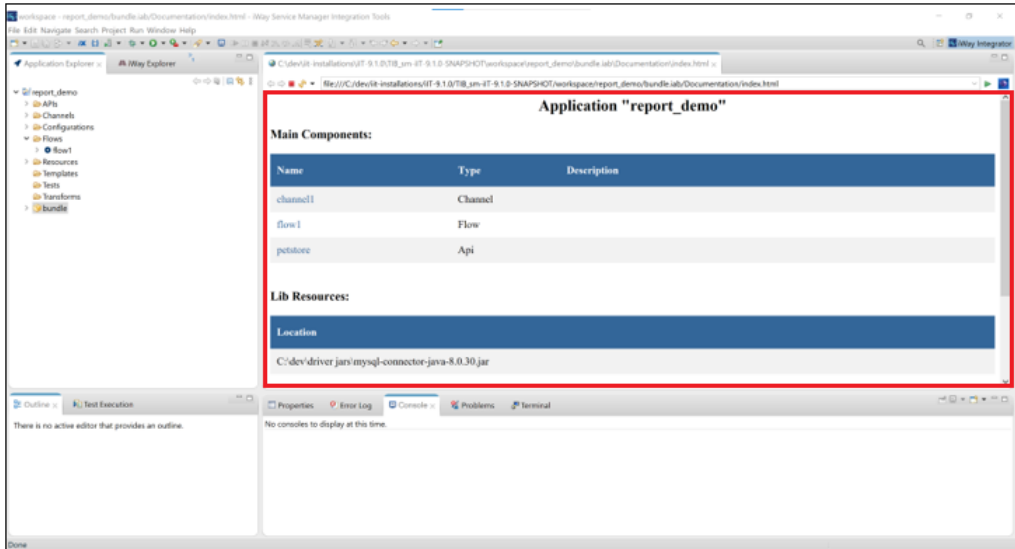


## iWay Application Summary

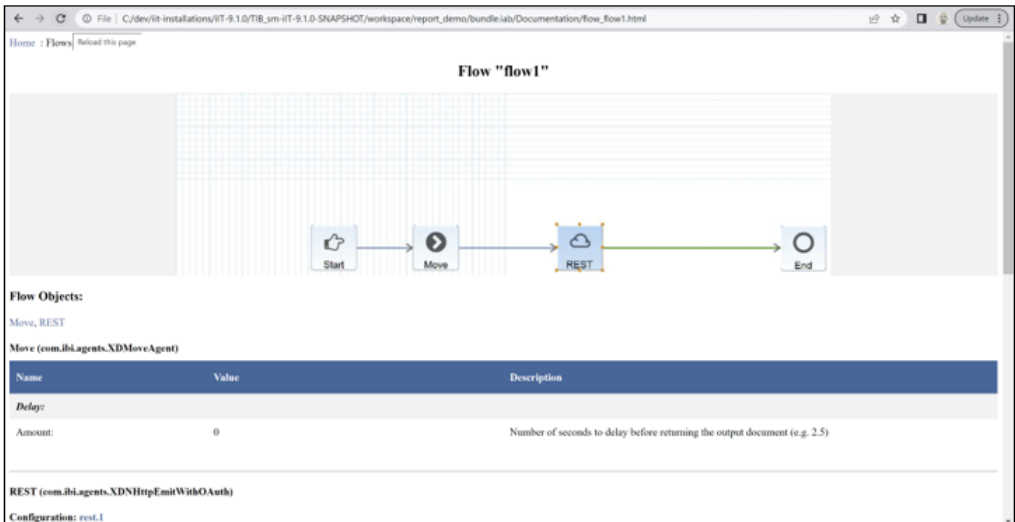
You can now generate a report for an Application Project in ibi™ iWay® Service Manager Integration Tools (iIT) and also view a report that was packaged as part of an application archive in the iWay Service Manager Administration Console. The report documents all of the components that are included in the Application Project, such as process flows, channels, APIs, and Special Registers (SREGs).

## Key Features

The following image shows a generated report in iT for an Application Project named *report\_demo*.

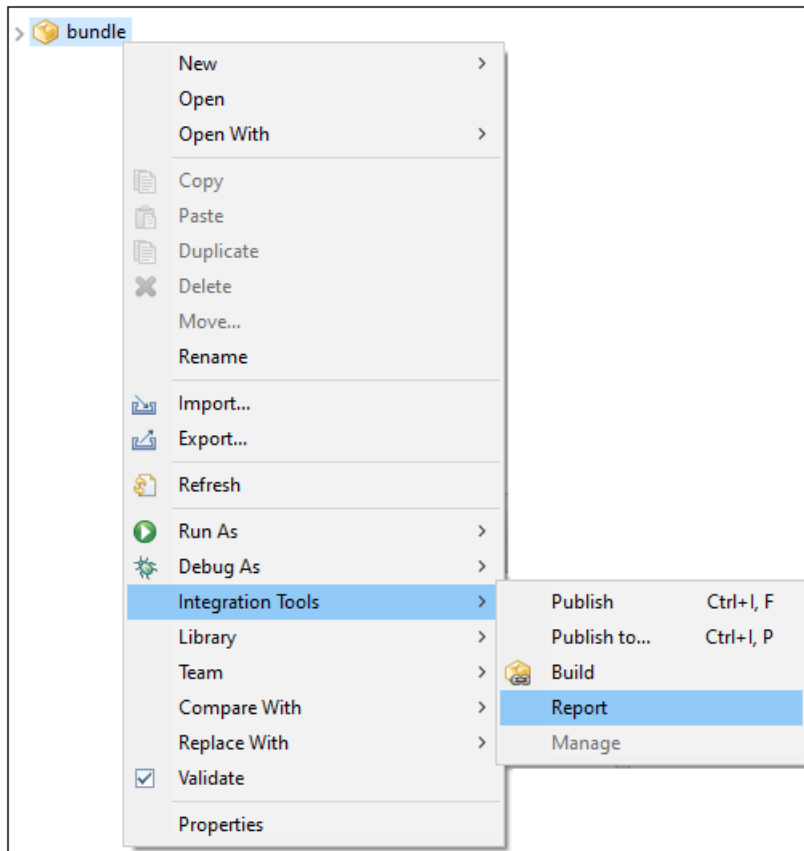


Each component in the report is listed as a hyperlink in the Main Components section. Clicking on the name of a specific component in this section of the report (for example, *flow1*), provides additional details for the selected component, as shown in the following image.

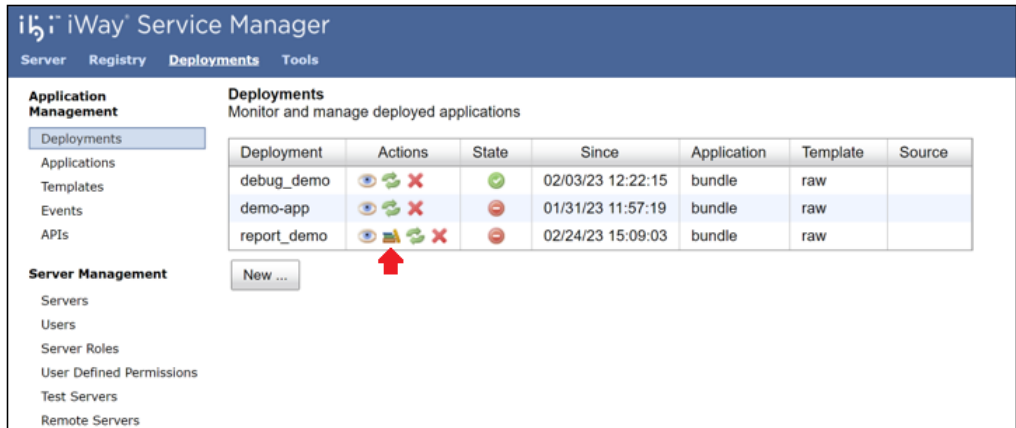

















To generate a report in iIT, right-click the *bundle* folder of an Application Project, select *Integration Tools*, and then click *Report*, as shown in the following image.



You can also access the report from the iWay Service Manager Administration Console. A new Documentation icon is now available for a deployed Application Project that includes a report as part of an application archive, as shown in the following image.



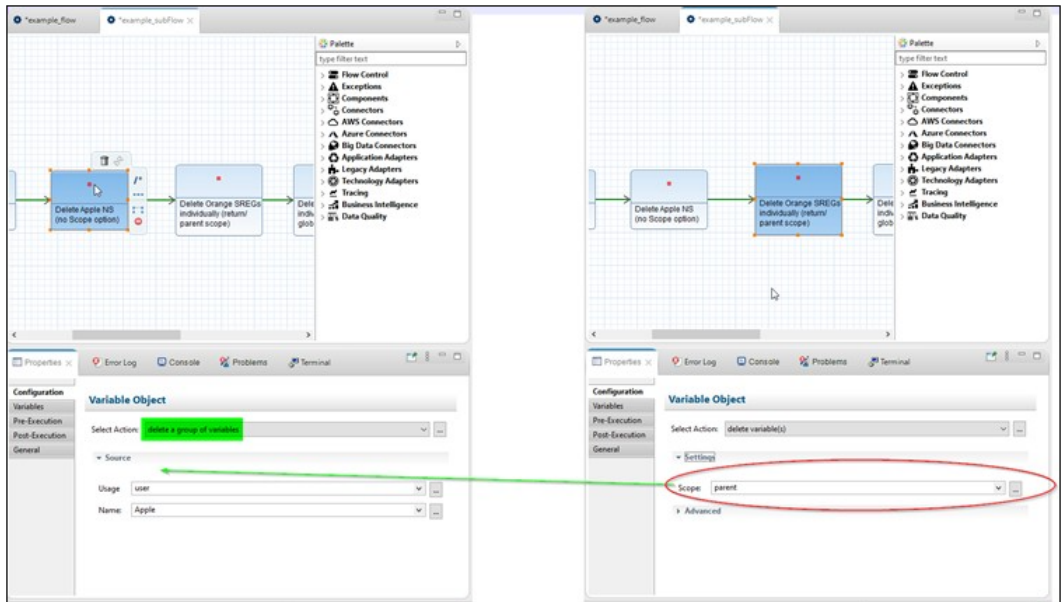
The screenshot displays the iWay Service Manager Administration Console interface. The main content area is titled "Deployments" and contains a table of deployed applications. A red arrow points to the Documentation icon (represented by a book) in the Actions column for the "report\_demo" deployment.

Deployment	Actions	State	Since	Application	Template	Source
debug_demo	  		02/03/23 12:22:15	bundle	raw	
demo-app	  		01/31/23 11:57:19	bundle	raw	
report_demo	   		02/24/23 15:09:03	bundle	raw	

Clicking the Documentation icon opens the report in a new browser tab.

## Defining a Scope for Delete Group Variables

In iIT, Delete Group Variables now includes an option to define the scope, similar to Delete Variable.



## Connector for ibi™ Data Quality

ibi™ Data Quality (DQ) connects “data people” with measurable, reportable, and actionable facts about their data. Users who rely on data to make everyday business decisions often deal with poor quality or untrusted data. ibi DQ provides these users with a complete view of the trustworthiness of their data, improves the quality of the data that drives business outcomes, delivers a rich user experience for technical and non-technical users, and enables seamless communication between the different systems and stakeholders involved in the data value chain.

The Connector for ibi DQ extends iWay Service Manager to integrate with the ibi DQ Rest API and enables users to access ibi DQ service resources. Users can retrieve uploaded datasets and rules, analyze a dataset, download analysis results for a dataset, and profile data, along with additional actions.

The following actions are currently supported by the Connector for ibi DQ:

- upload data set with input
- profile data

- deduplicate
- get data sets
- run numeric analysis
- run correlation analysis
- run KMeans clustering analysis
- get profile
- get correlation or KMeans clustering analysis
- upload data set from file
- get rules
- get matched rules
- run dq analysis with rules
- get data set analysis
- get data set analysis summary
- delete dq analysis
- delete data set
- run transactional request with input
- get analysis id for the last analyze request
- get all analysis ids
- get analysis ids for a data set
- get rule by id
- get data classes
- get services
- get service by id
- run transactional request with file
- create service

- update service
- delete service
- create rule
- update rule
- delete rule
- export data set by id
- get data set variables by id

### Connector for PowerShell

The Connector for PowerShell enables users to execute PowerShell commands and scripts. The connector integrates with Windows 10.11 PowerShell, which is a replacement for the Windows Console.

The connector session configuration is based on generics: *file-based configuration* and *parameter-based configuration*. The difference is the provenience of the session configuration parameters.

A *file-based* generic/configuration allows for two optional parameters:

- Specify session configuration file: json, .properties
- Specify powershell.exe location

A *parameter-based* generic/configuration allows for two optional parameters:

- Specify session configuration parameters as key-value pairs keyed through grid widget
- Specify powershell.exe location

A command can be passed through parameters. These are introduced as a separate parameter. Parameters can contain SREG values.

The PowerShell command/script execution can issue:

- An error (boolean)
- Timeout (boolean)
- Text output, which can be processed later by the flow

The text output of a command can be optionally stored in a node and/or a registry.

Possible outcomes are:

- OnSuccess
- OnFailure
- OnError
- OnShellNotAvailable (fail\_shell\_not\_available) - PowerShell.exe was not found
- OnShellExecFail (fail\_shell\_exec) - PowerShell command/script execution returned false
- OnFailNotFound (fail\_notfound) - Configuration file or script was not found
- OnFailTimeout (fail\_timeout) - Command/script execution timed out
- OnParseError (fail\_parse) - A parsing error of any of the input configuration documents failed

### **Connector for Azure Storage Queue**

The Connector for Azure Storage Queue provides cloud messaging between application components. Queue storage also supports managing asynchronous tasks and building process work flows. The following actions are currently supported by the Connector for Azure Storage Queue:

- build a client
- create a queue
- delete a queue
- list queues in account
- get properties in queue account
- set properties in queue account
- get queue service statistics
- put queue message into a queue
- update message in a queue
- peek at messages in a queue
- receive messages from a queue
- delete messages from a queue

- set a queue metadata
- get a queue properties

## SharePoint Rest Connector for SharePoint Online

The SharePoint Rest Connector now allows you to connect and integrate with SharePoint Online using a set of supported actions.

The following sets of actions are currently supported by the SharePoint Rest Connector for SharePoint Online:

### Attachments

- Attachment get all
- Attachment get content
- Attachment get metadata
- Attachment update content
- Attachment update content from document
- Attachment create from document
- Attachment create Attachment delete

### Files

- File create
- File get all
- File create from document
- File copy
- File get content
- File get metadata
- File move
- File delete
- File update content
- File update content from document

- File recycle
- File approve
- File deny approval
- File check out
- File undo check out
- File check in
- File publish for approval
- File unpublish for approval
- File move (Local and Remote)

### **Folder**

- Folder get all
- Folder get
- Folder create
- Folder delete
- Folder move
- Folder recycle

### **Lists**

- List get all
- List get
- List create
- List delete
- List get changes
- List update
- List recycle
- Check help texts for action 'list get changes'



### **List Items**

- List item get all
- List item get
- List item create
- List item update
- List item recycle
- List item delete

### **Recycle Bin**

- Recycle bin get all
- Recycle bin get
- Recycle bin restore

### **Sites**

- Site get all
- Site get
- Site get changes
- Check help texts for action 'site get change'
- Site delete

### **User-Defined Requests**

- Object get
- User-defined request - GET
- User-defined request - POST
- User-defined request - MERGE
- User-defined request - PUT
- User-defined request - DELETE

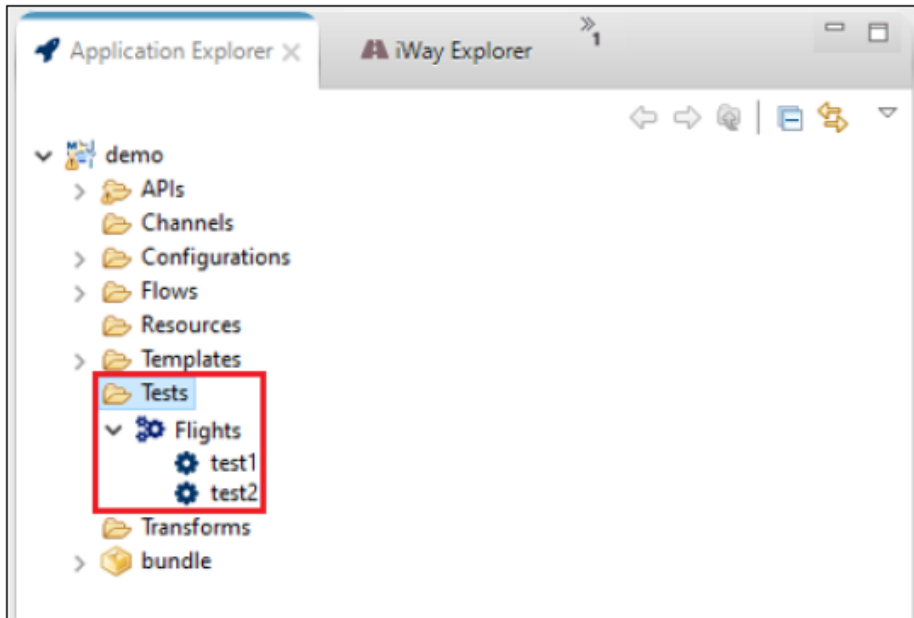
## Version 9.0.0

This section provides a summary of key features available in ibi™ iWay® Service Manager (iSM) and ibi™ iWay® Service Manager Integration Tools (iIT) in version 9.0.0.

### Testing Platform

A new Testing Platform is now available, which adds unit testing functionality to the iIT design time environment. The Testing Platform provides a set of tools that allow users to build tests for their process flows and then run those tests in an automated environment (design time and command line). Tests are composed of Test Suites and Test Cases. Each Test Suite is associated with one specific process flow. A Test Suite can contain one or more Test Cases.

In the Application Explorer tab, each new application project now includes a Tests subfolder where users can add their Test Suites and Test Cases.

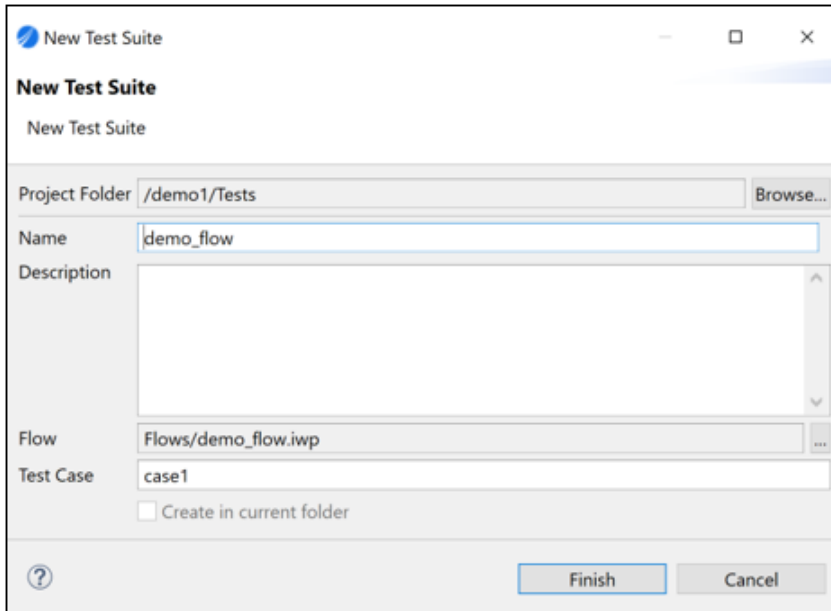


Each Test Case is stored in its own file resource and nested under the Test Suite resource. In this example being shown, there is one Test Suite called Flights with two Test Cases (test1 and test2).

### Creating a Test Suite

To create a Test Suite, right-click an available process flow in the Flows subfolder of your application project, select *New*, and then click *Test Suite*.

The New Test Suite dialog opens, as shown in the following image.



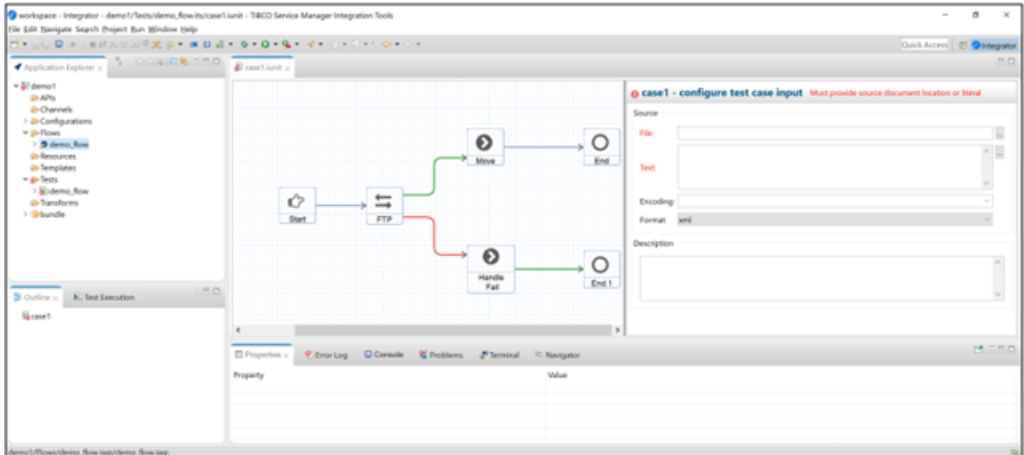
The image shows a "New Test Suite" dialog box. The title bar reads "New Test Suite". Below the title bar, the text "New Test Suite" is displayed. The dialog contains the following fields and controls:

- Project Folder:** A text field containing "/demo1/Tests" and a "Browse..." button to its right.
- Name:** A text field containing "demo\_flow".
- Description:** A large, empty text area.
- Flow:** A text field containing "Flows/demo\_flow.iwp" and a browse button to its right.
- Test Case:** A text field containing "case1".
- Create in current folder:** An unchecked checkbox.
- Buttons:** A help icon (question mark in a circle) on the left, and "Finish" and "Cancel" buttons on the right.

Specify a name for the new Test Suite in the Name field. The Flow field provides the location of the process flow being tested. Specify a name for the first Test Case to be added to the new Test Suite in the Test Case field.

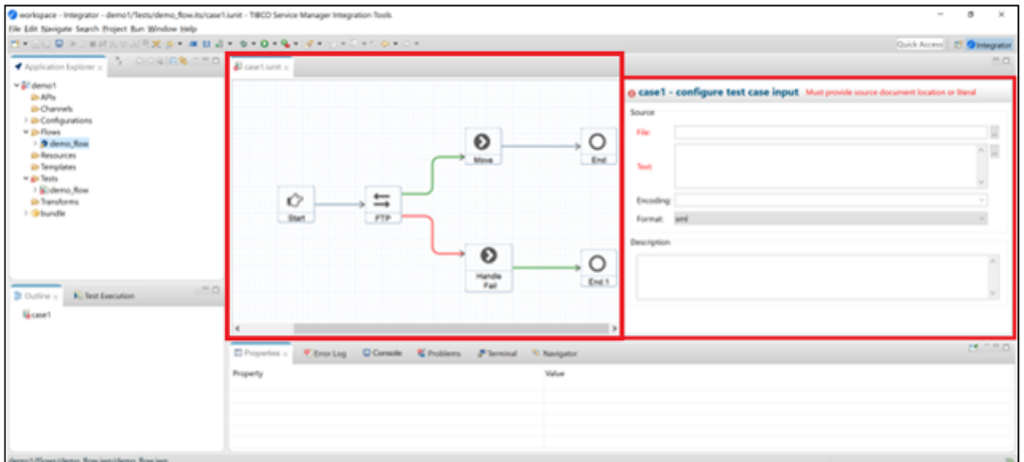
Click *Finish* to continue.

The new Test Case is created and opens in the Test Case editor, as shown in the following image.



### Navigating and Using the Test Case Editor

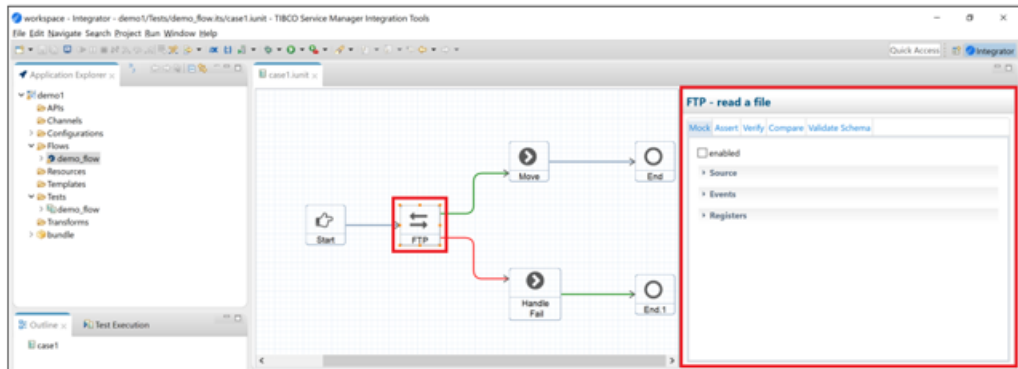
The Test Case editor consists of the flow canvas on the left-hand side and the details pane on the right-hand side, as shown in the following image.



Initially, the details pane displays the global settings for the Test Case. These include the required Source document configuration and an optional description.

These global settings can be accessed at any time by clicking inside the empty space within the flow canvas. Provide a Source document and click Save in the main toolbar to save your configuration or any changes.

Selecting an object on the flow canvas displays the details of the object's test configuration in the details pane, as shown in the following image.

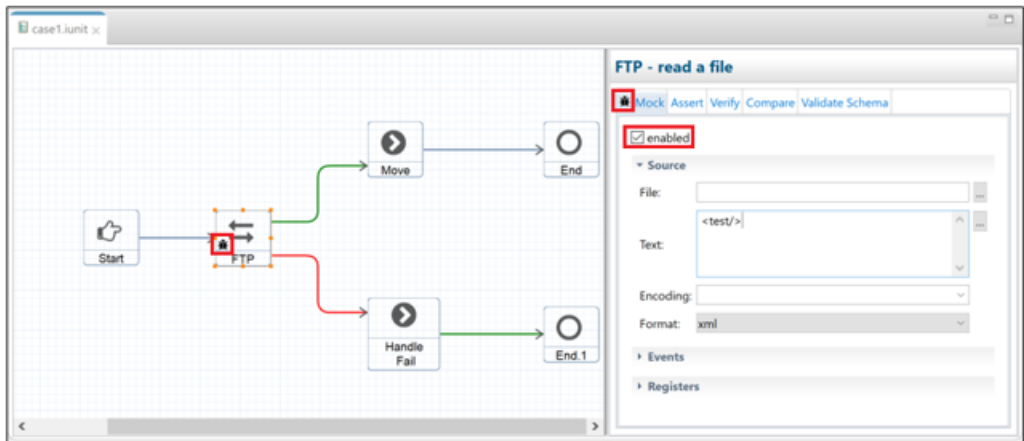


The details pane provides the following tabs, which represent a specific test action:

- Mock
- Assert

- Verify
- Compare
- Validate Schema

Each test action can be enabled or disabled by toggling the *enabled* check box in the details pane. Selecting the enabled check box adds an icon to the tab and also to the corresponding object on the flow canvas to indicate that one or more test actions have been configured. In the following image, the FTP object's Mock test action has been enabled. The object on the flow canvas and the Mock tab display the associated icon as a result.

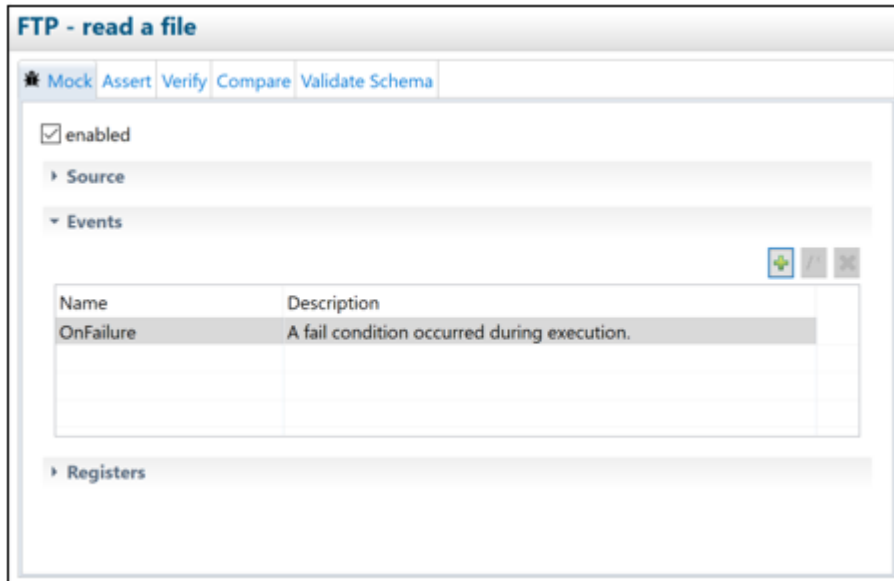


### Mock Tab

The Mock tab allows you to override the behavior of a selected object. Configuring the corresponding Source section will override the document that is returned from the object. In the following image, a mock document literal value of `<test/>` has been set.

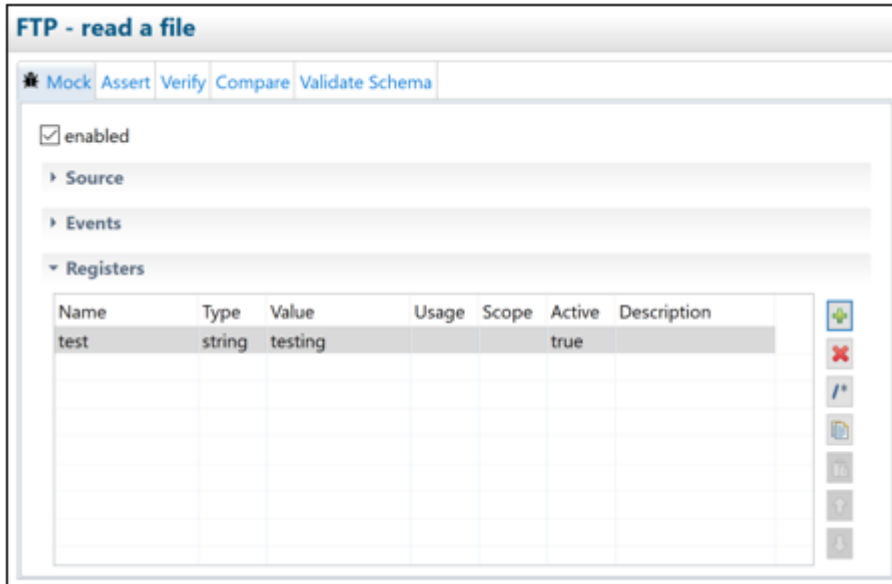
The screenshot shows the configuration interface for the 'Mock' tab of an object named 'FTP - read a file'. The interface includes a tab bar with 'Mock', 'Assert', 'Verify', 'Compare', and 'Validate Schema'. The 'Mock' tab is active, and a checkbox labeled 'enabled' is checked. Under the 'Source' section, there are three fields: 'File' (empty), 'Text' (containing the XML literal '<test/>'), and 'Encoding' (empty). Below these are two dropdown menus: 'Format' (set to 'xml') and 'Events' (empty). At the bottom, there are two expandable sections: 'Registers' and 'Events'.

The Events section allows you to specify which event(s) to emit so the execution is forced to follow a specific route in the process flow. In the following image, an event has been set to be emitted *OnFailure* to force the execution to follow the failure route.





The Registers section allows you to configure specific registers to be set by the object during the test's runtime, as shown in the following image.

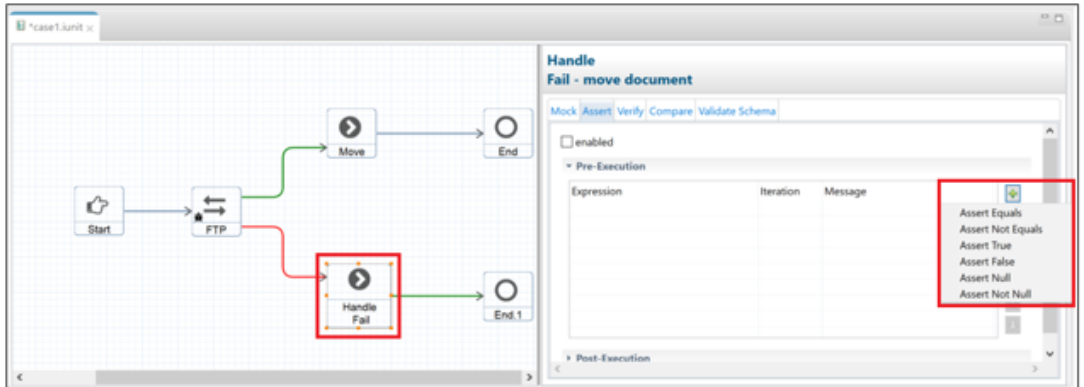


#### *Assert Tab*

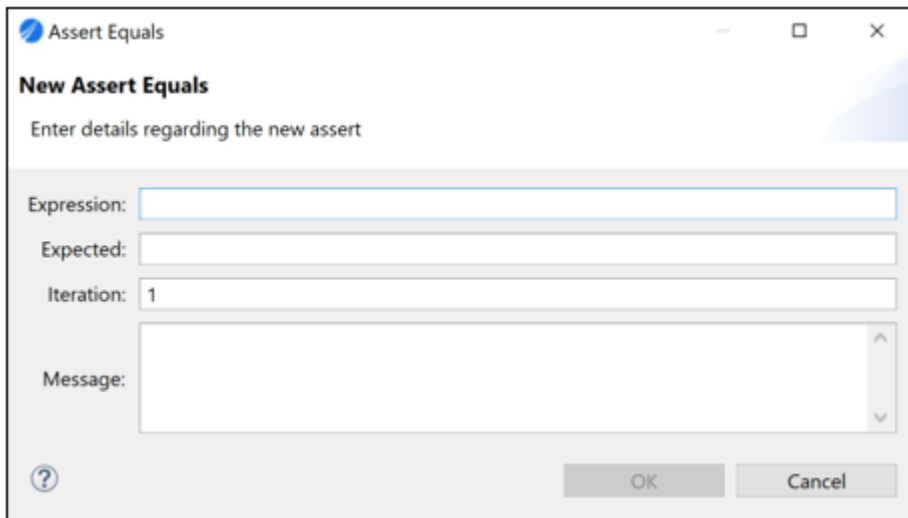
The Assert tab allows you to provide various expressions to be evaluated, tested for validity, and have results reported at the end of the test's execution.

## Key Features

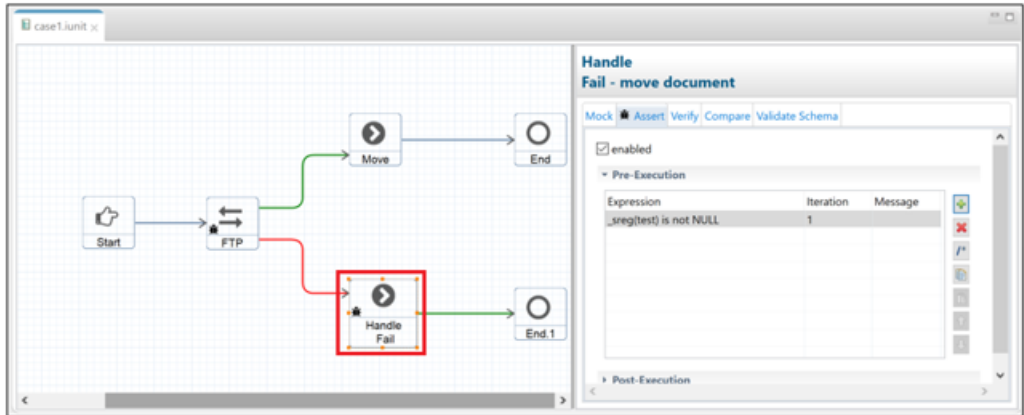
Asserts may be configured at a selected object's pre and/or post-execution stage. Asserts are added by clicking the plus (+) button. Six types of asserts are currently supported (Assert Equals, Assert Not Equals, Assert True, Assert False, Assert Null, Assert Not Null), as shown in the following image.



Selecting a specific assert type opens a corresponding dialog where the assert may be configured, as shown in the following image.



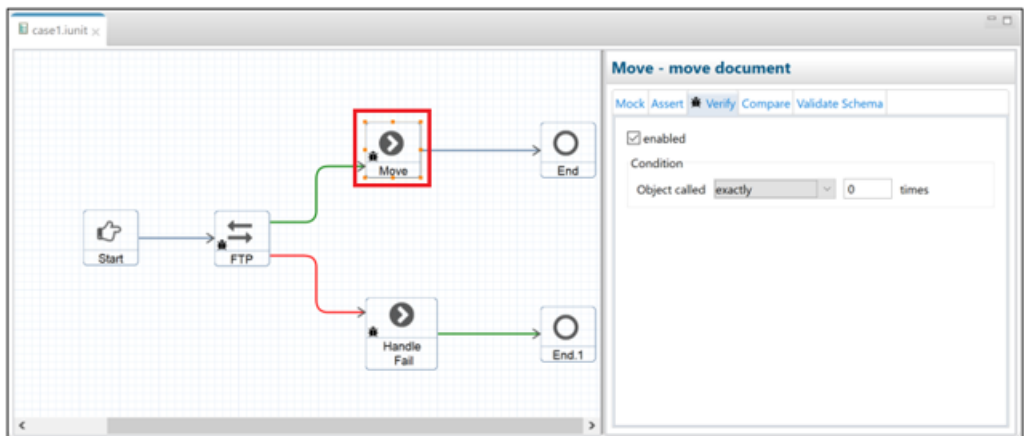
All assert types require an expression to be provided with Assert Equals and Assert Not Equals also requiring an expected value.



The Iteration field is used to specify the iteration at which to perform the assert. This is useful when working with iterators and loops in a process flow. The Message field is used to provide an optional user message to display in the event that the assert fails.

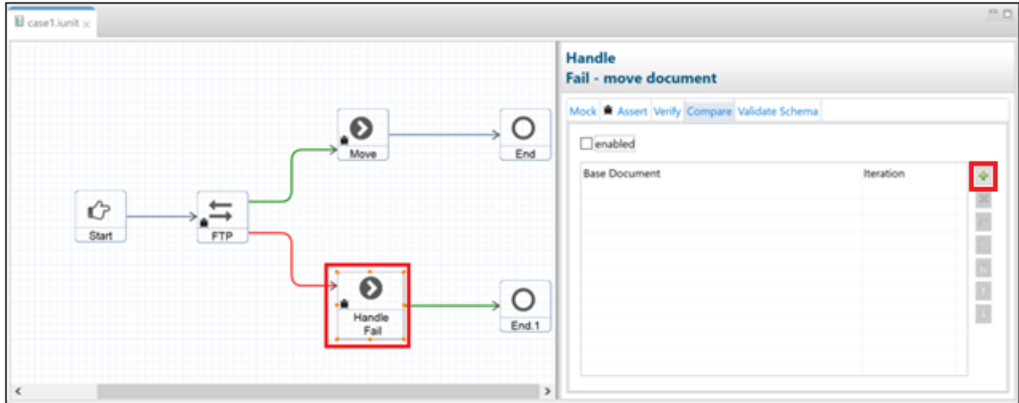
### Verify Tab

The Verify tab allows you to test for the number of invocations of a selected object. You can specify the number of times the object is expected to be called during the test's runtime, as shown in the following image.

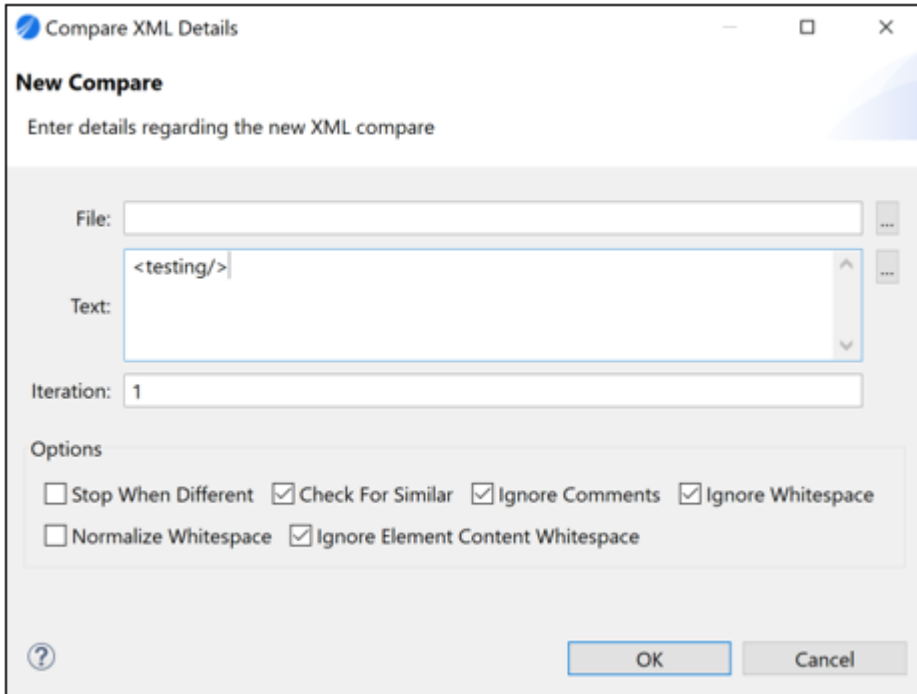


### Compare Tab

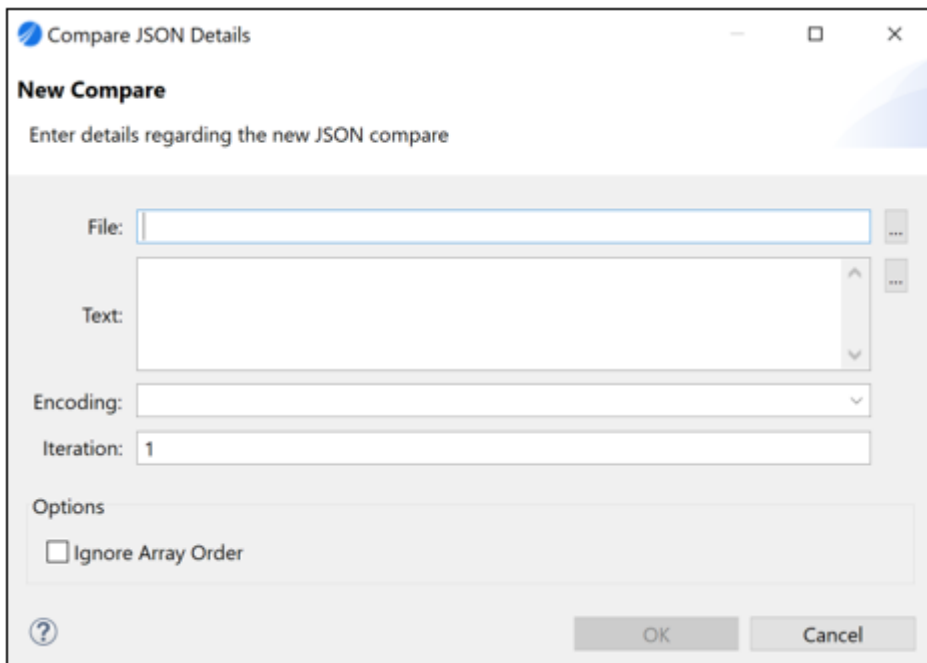
The Compare tab allows you to perform document comparisons between the document returned from the selected object at runtime against a base document provided by you during the configuration of the test case. A Compare test action is added by clicking the plus (+) button, as shown in the following image.



XML or JSON document comparison is currently supported.

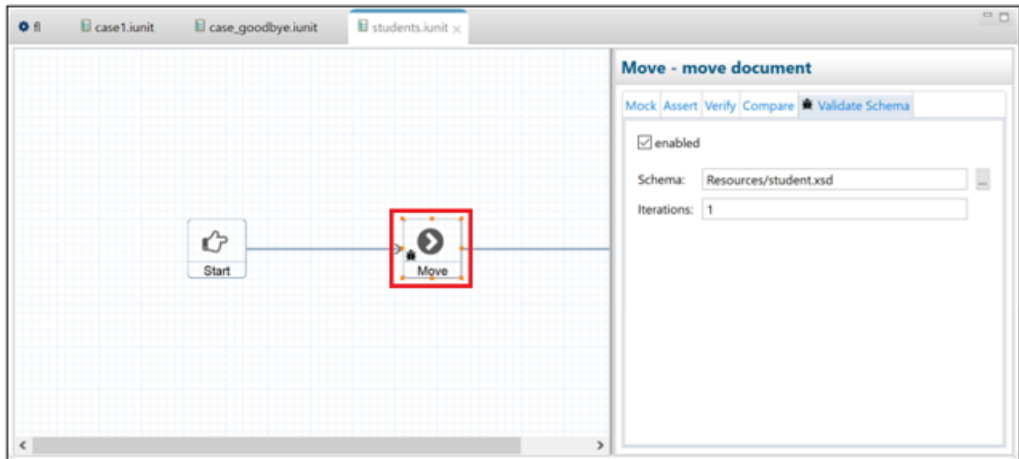


Various options are provided to allow you to adjust the comparison behavior, depending on the document type (XML or JSON).



## Validate Schema Tab

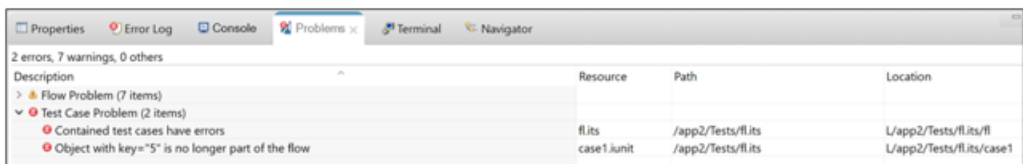
The Validate Schema tab allows you to validate the document returned from the selected object against an XML schema. The Schema field allows you to specify the location of the schema document relative to the local project. The Iterations field allows you to specify a comma-separated list of iterations at which validation should be performed. By entering an asterisk character (\*) in the field, the validation will be performed at all object iterations.



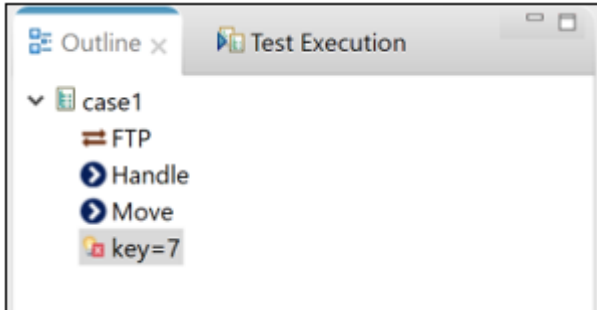
## Outline View

The Outline view displays as a tab and provides a compact view of the objects in a selected process flow that have test actions configured and enabled. Double-clicking an object in the Outline view automatically scrolls the canvas to reveal the object, selects the object, and displays the object's details.

In the event that a Test Case object with enabled test action(s) is no longer part of the process flow (for example, when it has been removed from the flow), the Test Case is no longer considered valid and error(s) are displayed in the Problems tab, as shown in the following image.



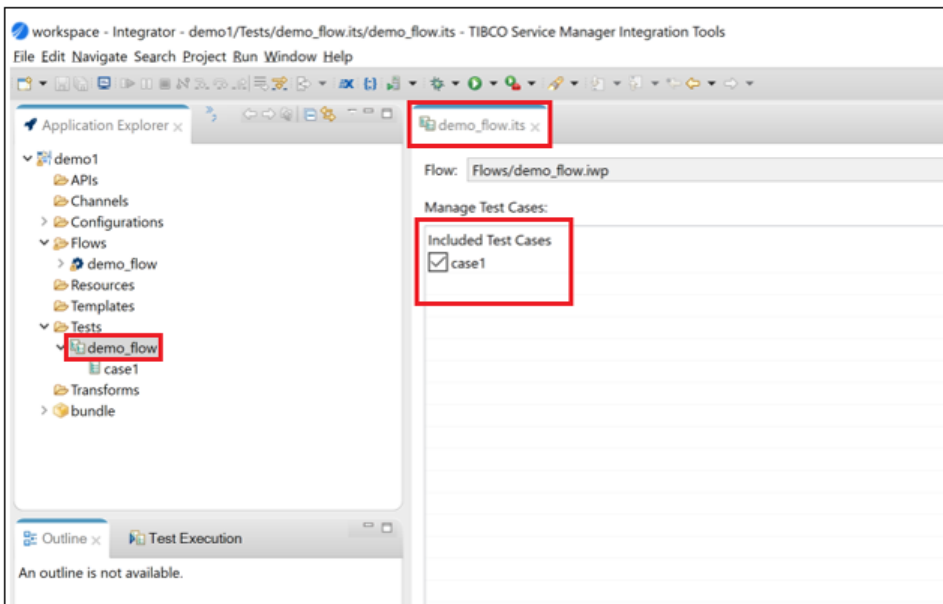
The Outline view displays an element in the form "key={object id}" with a red x next to it, as shown in the following image.



You can repair an invalid Test Case by deleting the element. Right-click the element and select *Delete* from the menu.

### Navigating and Using the Test Suite Editor

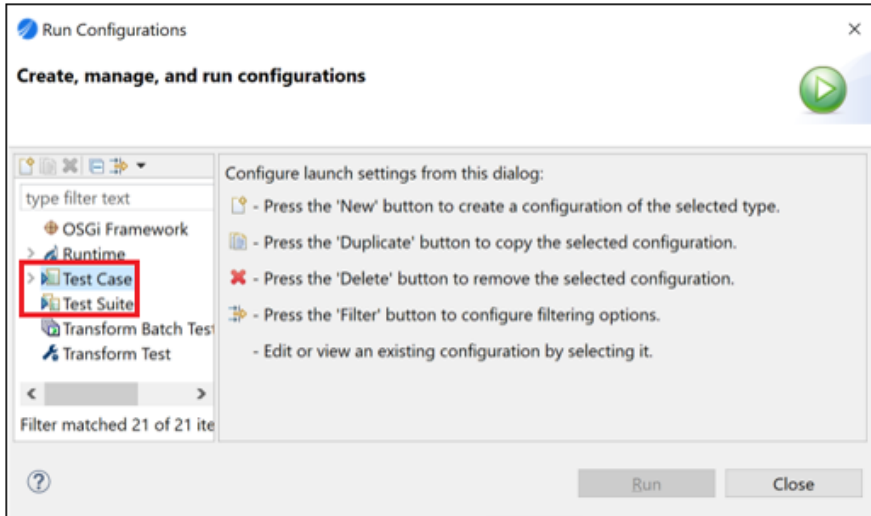
Opening a specific Test Suite displays as a tab, which shows the location of the process flow currently associated with the Test Suite and a list of associated Test Cases. Check boxes next to Test Case names allow you to skip certain Test Cases during runtime if required, as shown in the following image.





## Running Tests

Test Suites and Test Cases can be executed inside iIT using Eclipse Run configurations. Two types of Run configurations are supported for testing (*Test Suite* and *Test Case*).

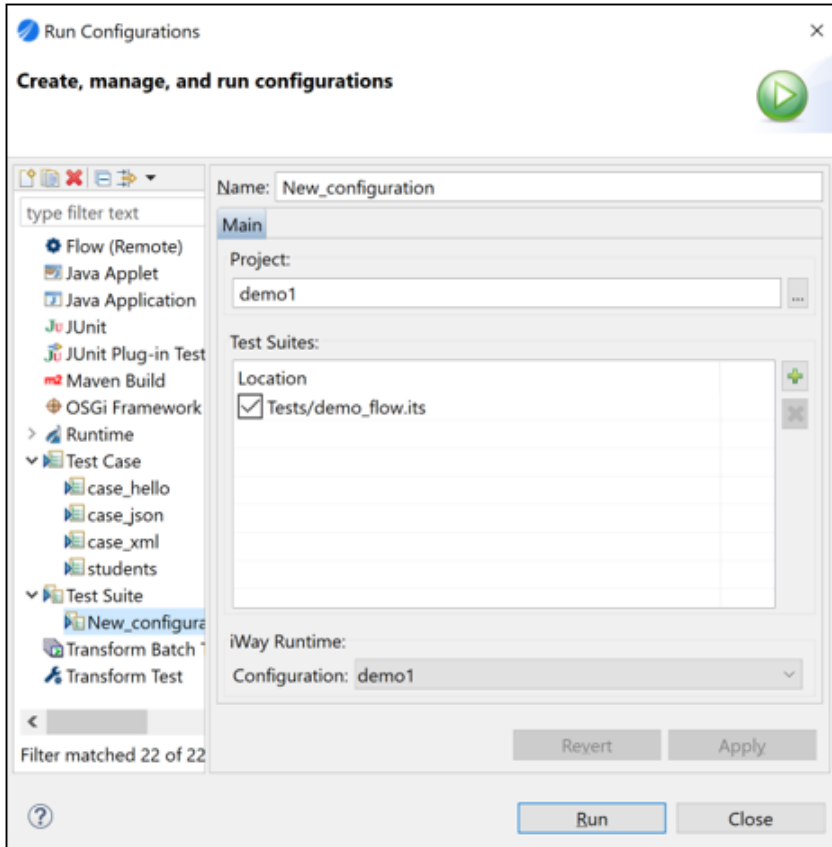


To run an individual Test Case, right-click on the flow canvas inside the Test Case editor, select *Run*, and then click *New*.

Alternately, you can right-click on a Test Case from the Application Explorer, select *Run As*, and click *Test Case*.

## Test Suite Run Configuration

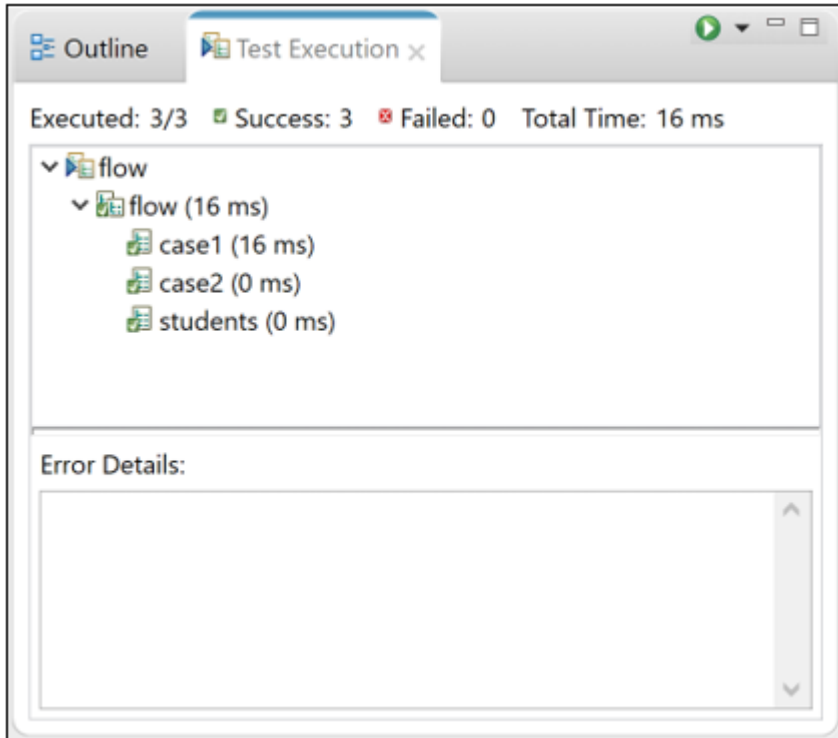
Test Suite Run configurations allow you to run one or more Test Suites. Each Test Suite configuration is bound to an application project, and may contain one or more Test Suites.




Unchecking the check box next to a specific Test Suite will skip its execution during run time.

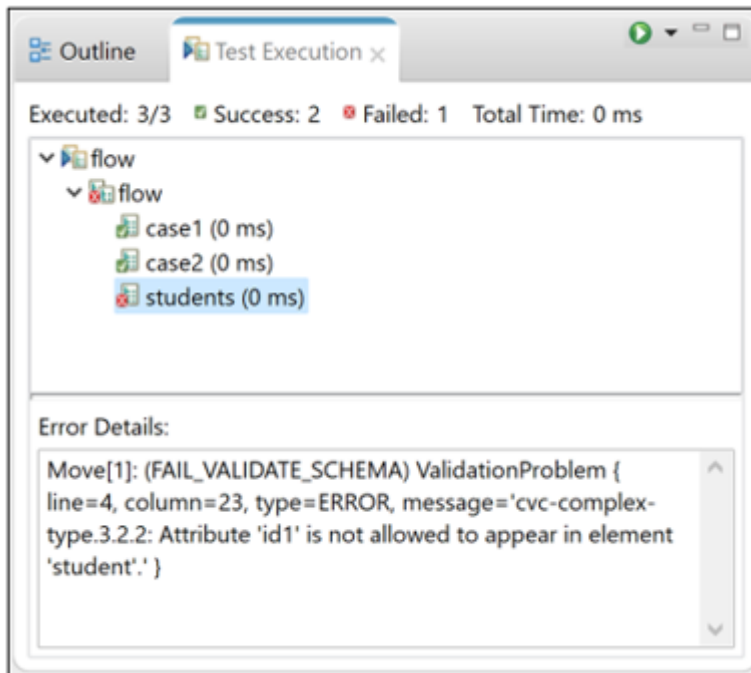
## Test Execution View

When running Test Suites or individual Test Cases, execution results are displayed in a dedicated Test Execution tab, as shown in the following image.



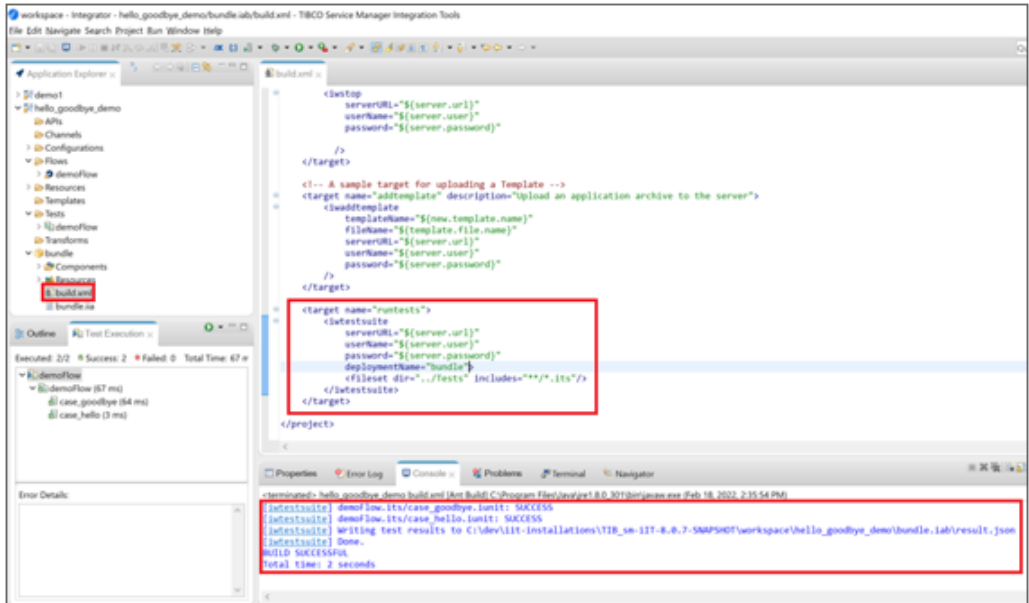
You can see execution time organized by Test Suites and Test Cases. You can also see the totals for successful/failed Test Cases. Clicking the green play icon  will rerun the last Run configuration. Clicking the down arrow next to the green play icon lists all of the existing Test Case and Test Suite Run configurations.

Selecting a failed Test Case in the tree displays the failure details, as shown in the following image.



## Running Tests Using Ant

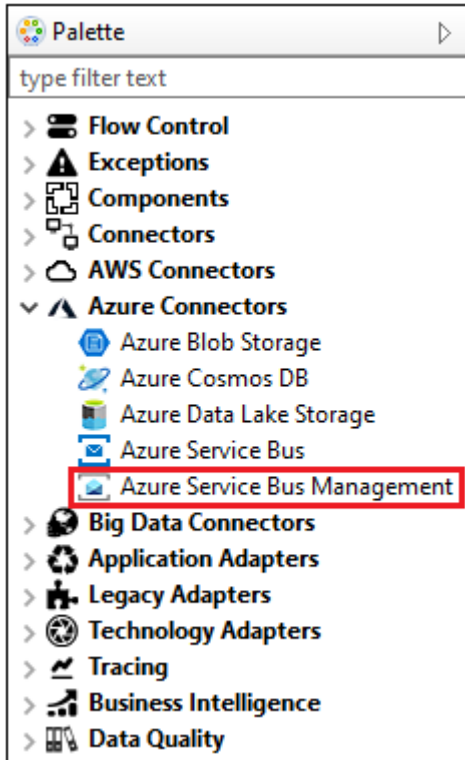
An Ant task is provided, which allows you to automate the process of executing Test Suites. Open the *build.xml* file, which is located in the *bundle* subfolder of your application project and look for the *runtests* target, as shown in the following image.



When you run the target, the execution status is printed to standard output, and a report in JSON format is stored in the application bundle folder.

## Connector for Microsoft Azure Service Bus Management

iWay Service Manager version 9 introduces the Connector for Microsoft Azure Service Bus Management. This connector enables integration with Azure Service Bus, which is a cloud-hosted messaging service between applications and services. Support for queues, topics, and event hubs is provided, with messages being sent and received.



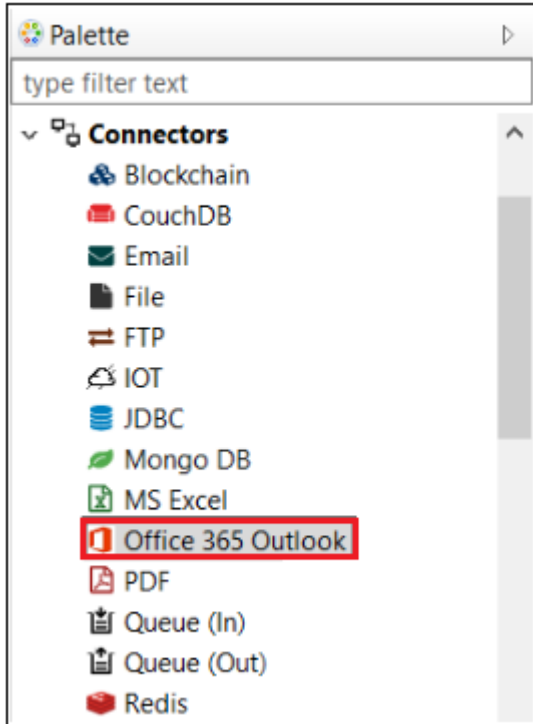
The ability to exchange data in the form of messages is the primary use case for using the Azure Service Bus. Microsoft Azure Service Bus Management implements Service Bus topology that allows you to integrate the queues, topics, subscriptions, and rules of the messaging entity. Each entity uses CREATE, DELETE, GET, LIST, and UPDATE actions that are issued through API calls. The following actions are supported by the Connector for Microsoft Azure Service Bus Management:

- Create Queue
- Create Rule
- Create Subscription

- Create Topic
- Delete Queue
- Delete Rule
- Delete Subscription
- Delete Topic
- Get Queue
- Get Rule
- Get Subscription
- Get Topic
- List Queue
- List Rule
- List Subscription
- List Topic
- Update Queue
- Update Rule
- Update Subscription
- Update Topic

## Connector for Microsoft Office 365 Outlook

iWay Service Manager version 9 introduces the Connector for Microsoft Office 365 Outlook, which allows you to connect to Microsoft Office 365 Outlook and work with Graph APIs to perform management-related tasks. The connector also provides an Office 365 Message Listener to receive messages from a specific Outlook account during a specified time and runtime.



The following actions are supported by the Connector for Microsoft Office 365 Outlook:

- List messages
- Get message
- Get message's MIME content
- Create a new message draft
- Update message
- Delete message



- Get message delta
- Send message
- Copy message
- Move message
- Create reply draft to message
- Reply to message
- Create replyAll draft to message
- Reply all to message
- Create draft to forward message
- Forward message
- Send mail
- List attachments for message
- Add file attachment to message
- Add message attachment to message
- Add event attachment to message
- Attach large file to message
- Get attachment for message
- Get attachment's raw contents for message
- Delete attachment on message
- List Outlook categories
- Create Outlook category
- Get Outlook category
- Update Outlook category
- Delete Outlook category
- List mail folders

- Get mail folder
- Create mail folder
- Update mail folder
- Delete mail folder
- Copy mail folder
- Create child folder
- Get mail folder delta
- List child mail folders
- Move mail folder
- List calendars
- List calendar groups
- Create Calendar
- Get calendar
- Update calendar
- Delete calendar
- List calendar view
- Create calendar group
- Get calendar group
- Update calendar group
- Delete calendar group
- List events
- Get event
- Delete event
- Get event delta
- Create event

- Update event
- Forward event
- Cancel event
- Accept event
- Tentatively accept event
- Decline event
- Dismiss reminder for event
- Snooze reminder for event
- List event instances
- List attachments for event
- Add file attachment to event
- Add event attachment to event
- Add message attachment to event
- Get attachment for event
- Get attachment's raw contents for event
- Delete attachment on event
- List event reminders in calendar
- Find meeting times
- Get Schedule
- List contacts
- List contact folders or child folders
- Get contact
- Get contact folder
- Delete contact
- Delete contact folder

- Get contact delta
- Get contact folder delta
- Create contact folder
- Update contact folder
- Create contact
- Update contact
- Attach large file to event

### **Attachment Object**

An Attachment object is now available in the iWay Service Manager Integration Tools design time environment for version 9 (9.0.0). This object allows you to group agents, which can then be added as an attachment to the bus. In previous versions, the attachment functionality within the Email object had to be used for this purpose.

### **Token Object**

A Token object is now available in the iWay Service Manager Integration Tools design time environment for version 9 (9.0.0). This object can now be used for all token generation functionality. In previous versions, only the Bearer token was available (`com.ibi.agents.XDBearerTokenAgent`).

### **SWIFT 2022**

iWay Service Manager version 9 supports SWIFT 2022. All documents and transaction sets in SWIFT 2022 are supported.

# Legal and Third-Party Notices

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

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

This document 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 Cloud Software Group, Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, ibi, ibi logo, ActiveMatrix BusinessWorks, TIBCO Administrator, BusinessConnect, TIBCO Designer, Enterprise Message Service, Hawk, and Maporama are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle Corporation and/or its affiliates.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

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

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme 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. CLOUD SOFTWARE GROUP, 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.

This and other products of Cloud Software Group, Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2023. Cloud Software Group, Inc. All Rights Reserved.