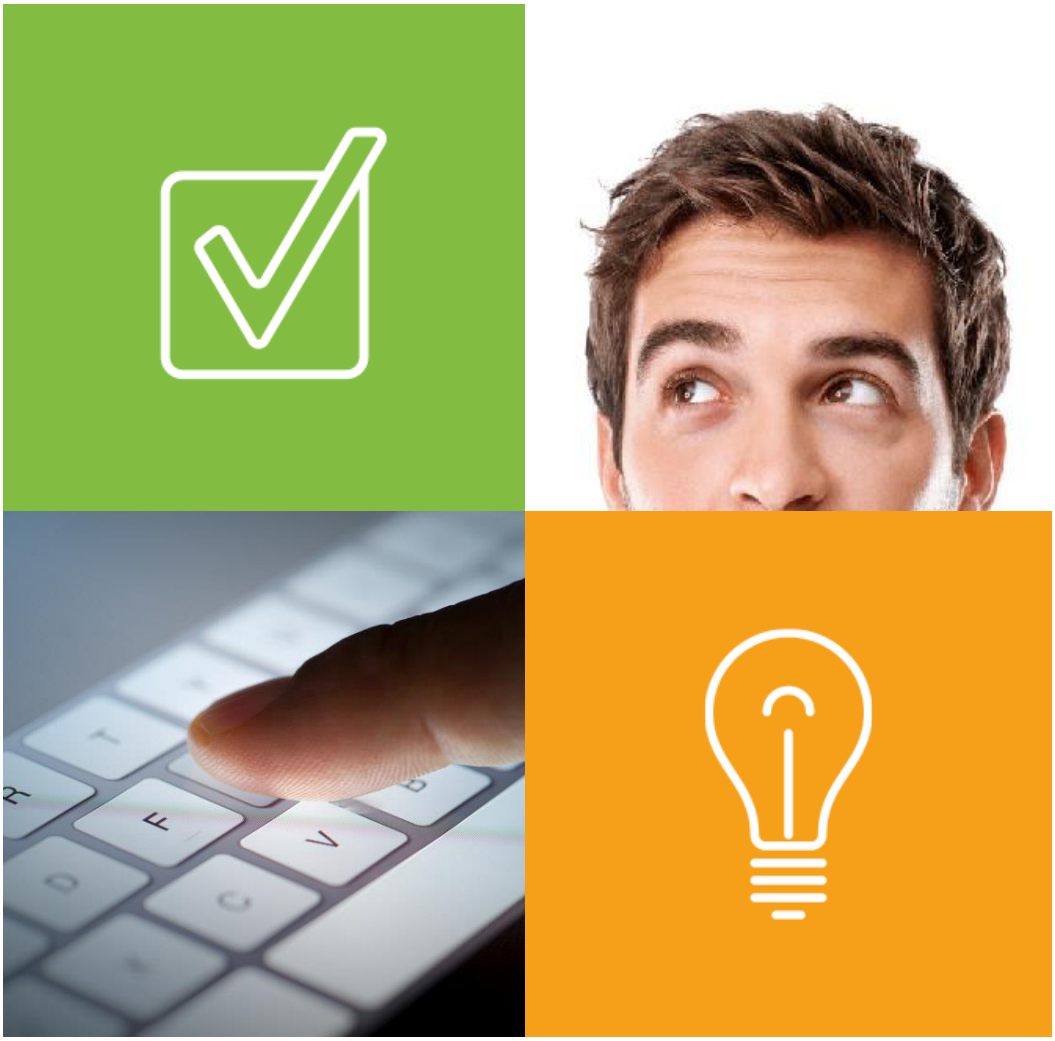
workday. EDUCATION



Introduction to Workday Studio

COURSE MANUAL AND ACTIVITY GUIDE

workday. | EDUCATION

Training the way you want it, within your budget

We offer a variety of learning delivery options, ranging from accreditation to self-paced independent learning offerings.

Workday Pro
This customer-focused accreditation program helps drive greater Workday competency and self-sufficiency for your organization.

Workday Touchpoints Kit
The Workday Touchpoints Kit helps you identify cross-functional impacts across the full suite. By providing a macro view of the Workday components and the relationships among them, it ensures higher quality and smarter implementations.

Adoption Kit
This collection of templates and resources accelerates student training and your Workday rollout. It includes a combination of videos, job aids, and facilitation and marketing materials. All content can be tailored to your needs, or used as-is.

The Next Level
A series of online demonstrations that show Workday in action and give you tips on deploying features.

Enablement Workshops
In-person training with hands-on configuration in your own sandbox tenant.

Learn Independent

This learning experience combines videos, interactive exercises, quizzes, and tests into a comprehensive, online learning curriculum that students can complete at their own pace. Students also experience hands-on activities in a Workday tenant.

Learn Virtual

Our virtual classroom offers the advantages of live instructors without the expense and time associated with travel. Students connect to our training environment and participate remotely, complete hands-on activities, and interact with instructors and other students.

Learn On-Demand

As a supplement to instructor-led offerings, this training provides immediate access to specific courses and includes short, topic-specific videos and job aids.

Learn In-Person

This instructor-led, in-classroom training prepares students to meet their job requirements. It combines lectures, social learning, product demonstrations, and hands-on activities.

Learn more about our training opportunities on Community:

https://community.workday.com/training/km

Workday Pro is a customer-focused accreditation program. Each Pro track consists of relevant courses, plus an online multiple-choice written test. Based on the training you are enrolled in, there is a check mark below to indicate your progress in the specific Workday Pro track(s). For more information on WD Pro, please visit us at: ***community.Workday.com/pro***

Integrations

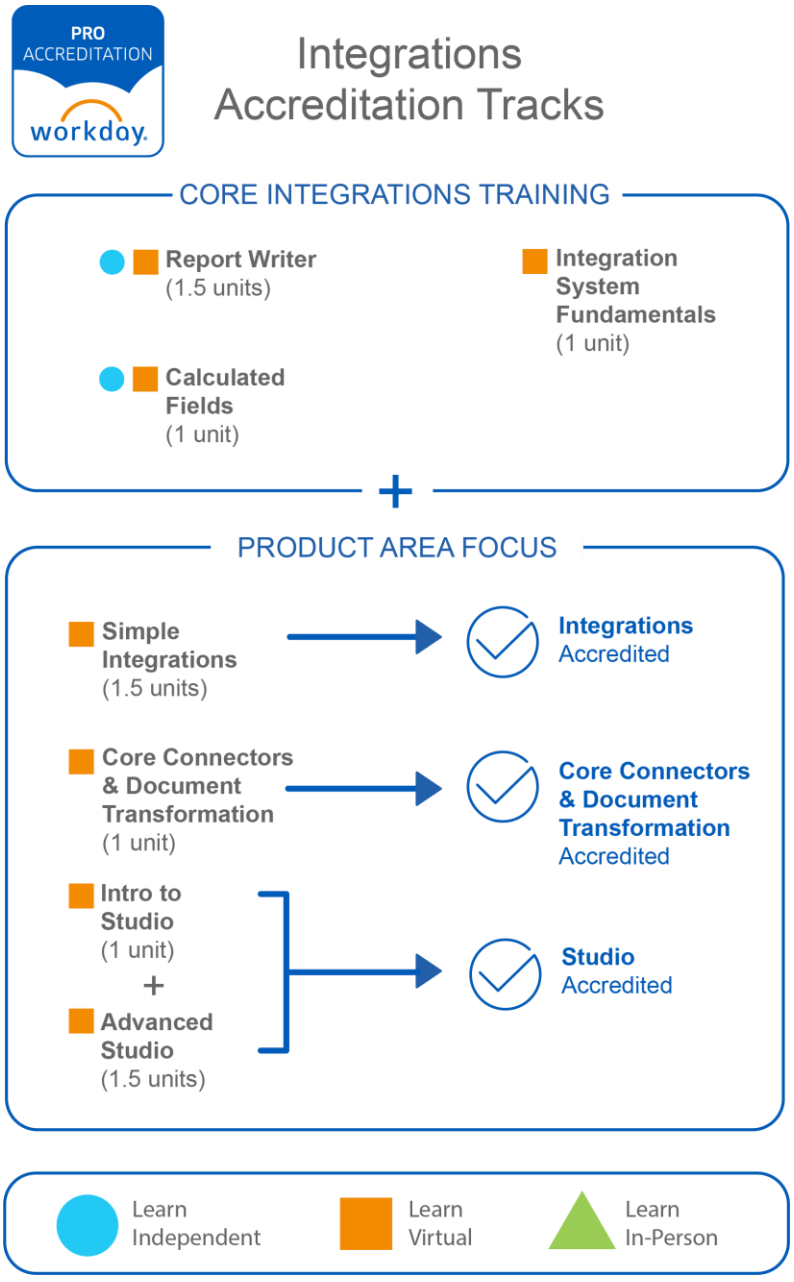
Workday Pro Integrations	Workday Pro Studio	Workday Pro CC & DT
Program Fee 1 Unit	Program Fee 1 Unit	Program Fee 1 Unit
Report Writer 1.5 Units ☑	Report Writer 1.5 Units ☑	Report Writer 1.5 Units ☑
Calculated Fields 1 Unit	Calculated Fields 1 Unit	Calculated Fields 1 Unit
Integration System Fund. 1 Unit ☑	Integration System Fund. 1 Unit ☑	Integration System Fund. 1 Unit ☑
Simple Integrations 1.5 Units	Intro to Workday Studio 1 Unit ☑	Core Connectors & Document Transformation 1 Unit
	Advanced Workday Studio 1.5 Units	
Workday Pro Integrations Written Test	Workday Pro Studio Written Test	Workday Pro CC & DT Written Test

Introduction To Workday Studio Contents

WORKDAY PRO

CUSTOMER ACCREDITATION PROGRAM

Workday Pro is a customer-focused accreditation program targeted at customers who want to actively engage and work side-by-side with the ecosystem on a path to develop a similar level of knowledge and expertise. It consists of several tracks, each with relevant courses, plus a written test.



Learn more: community.workday.com/pro

INTRODUCTION TO WORKDAY STUDIO

DESCRIPTION

Workday Studio is a unified Eclipse-based environment that allows Workday customers and third parties to develop, deploy, debug, and support their own complex integrations running in the Workday Cloud. This course will orient you to the Workday Studio development environment and show you how to begin working with Assemblies. Through hands-on activities and demonstrations, the course will introduce building Workday Studio Assembly projects and using Assembly Components and Steps.

GOAL & OBJECTIVES

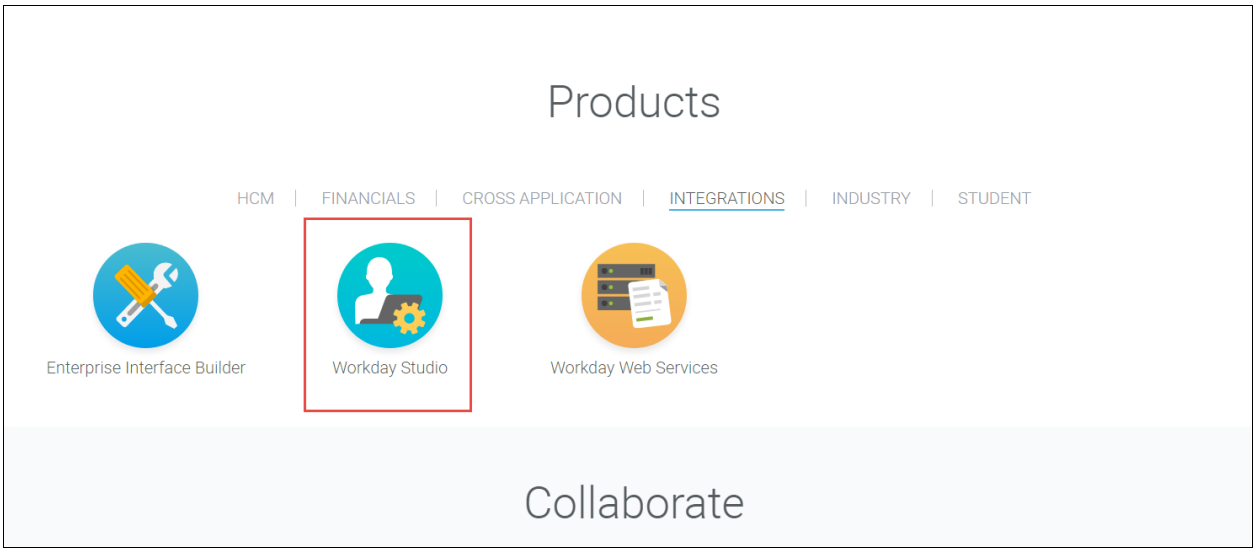
- Features of Workday Studio
- Workday Studio Assemblies (integration systems) Components
- Patterns for Common Integration Solutions
- Basic Flow of Control Between Assembly Components
- Basics of Web Services, XSLT, and MVEL

AGENDA

- Review Integration Architecture and Tools
- What is Workday Studio?
- Workday Studio Platform and Samples
- Investigate a Workday Studio Assembly
    - Deploy Hello Cloud
    - Hello Cloud Sample Investigation
- Build a Studio Assembly
- Context Mediation: Message Path Through an Assembly
- Review Workday Web Services
- Using Studio's Web Service Tester to call Workday Web Service
- Hello WWS Sample
- Build a Studio Assembly that calls a Web Service
- Build a Studio Assembly that calls a custom report Outputting a File to Workday in the Assembly
- Using XSLT in an Assembly
- Consolidated Report Viewer

WELCOME TO WORKDAY STUDIO

As part of our Integration Cloud Platform, Workday offers a set of easy-to-use integration tools designed to solve many of the common integration use cases faced today. Workday Studio is a powerful development tool enabling customers and partners to build sophisticated integrations to and from Workday.



Aimed at skilled developers and offered as a set of plug-ins to the Eclipse IDE, Workday Studio offers a rich, graphical development environment in which a user can drag and drop a variety of reusable components that handle the "plumbing" aspects of integration building, freeing you to focus on the critical business logic. Workday has been using this valuable tool for years to deliver all of our packaged integrations and connectors as well as the EIB. Workday Studio has been available to customer since Workday 12.

WHAT IS WORKDAY STUDIO

Workday Studio is a unified Eclipse-based environment that allows Workday customers and 3rd parties to develop, deploy, debug and support their own complex integrations running in the Workday Cloud.

You can reference samples, tutorials, user guides and reference guides in Workday Studio: **Help >> Help Contents**.

Training will focus on Introducing the Assembly Diagram and key components.

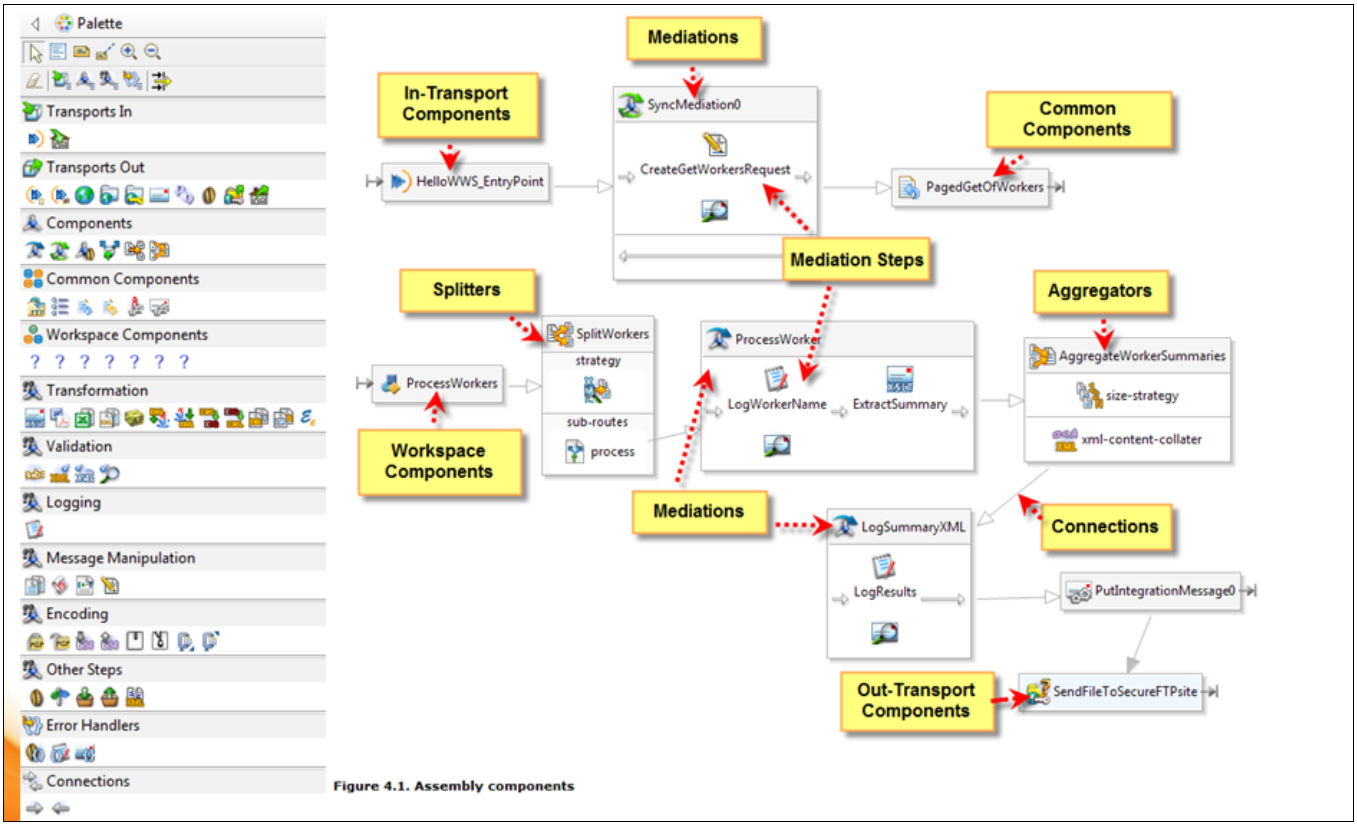
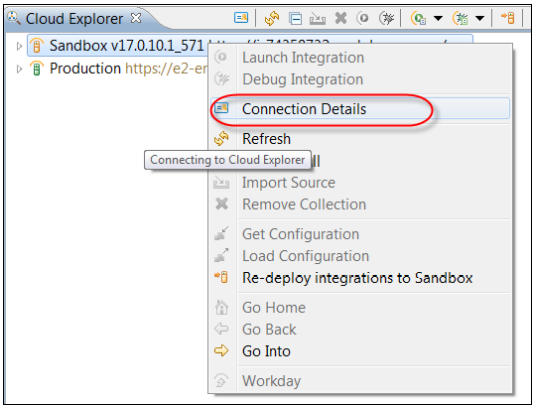
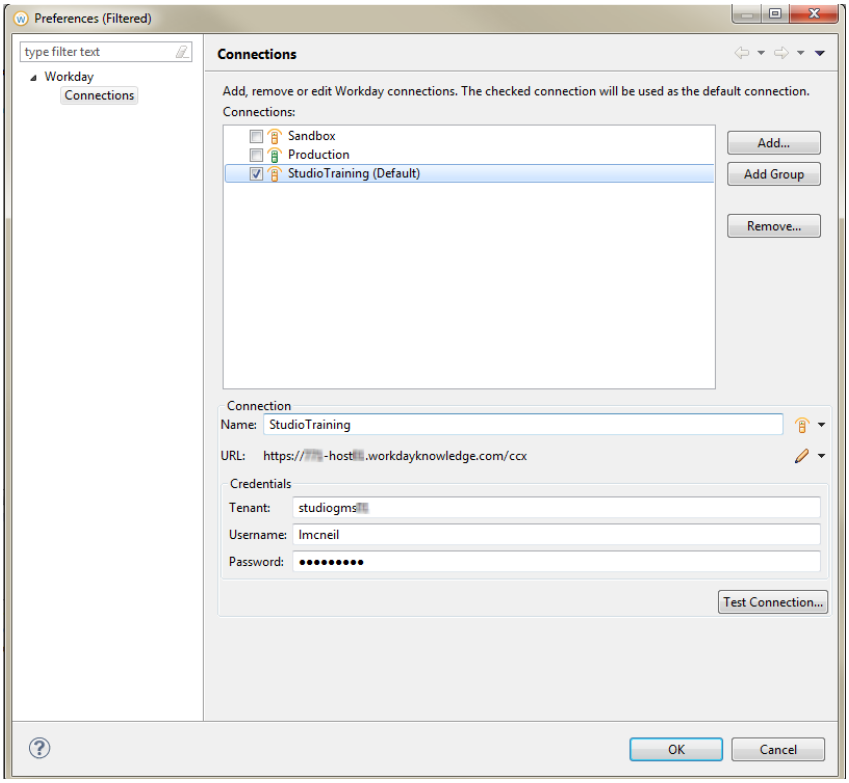


Figure 4.1. Assembly components

LAUNCHING AND CONNECTING TO WORKDAY STUDIO

Before you can deploy or launch a sample, you must connect to a Workday environment. The **Cloud Explorer** in Workday Studio provides an interface to the available Workday environments.





You can configure the existing Sandbox and Production connection or add a new connection with a name of your choice.

When you select the URL, you can choose based on the location of your data center for your Sandbox, Implementation or Production tenant. If you have a URL not listed, click the pencil icon to enter your Host name value manually.

(Find your Data Center and Host, excludes training SUVs) - https://community.workday.com/node/62993

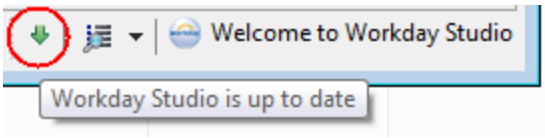
Since Workday Studio is installed locally on your computer and must connect to a Workday tenant, you may encounter network settings that prevent you from connecting. You may have to configure a proxy server or contact IT to help configure the network firewall.

For assistance with resolving connection issues, please refer to Workday Studio Help Contents or the Knowledge Base on Workday Community: https://community.workday.com/studio-kb.

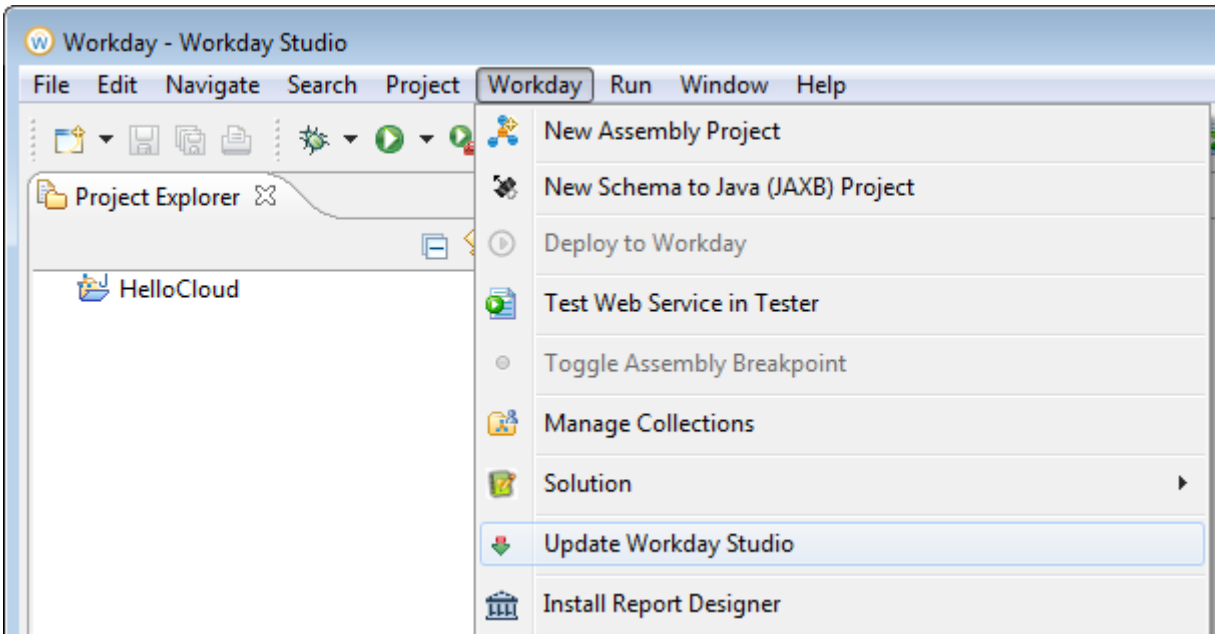
WORKDAY STUDIO UPDATES

Workday Studio will automatically detect that an update or patch has been released the next time you use Studio or when you attempt to interact with the Workday runtime. Your Studio installation must be up to date in order to interact with the Workday runtime. Studio will automatically prompt you when an update is required. Studio's *One-Click Update* feature means that when you attempt to connect to a Workday environment, Studio checks that it is up to date, and updating is just a matter of clicking a button.

The update-status icon in the status bar indicates whether your version is up to date. When your software version is up to date, the arrow icon is displayed in green. If your version is out of date, the arrow icon changes to red.



To update your software, select Update Workday Studio from the Workday menu.



If you are in the middle of something and do not need to connect to the Workday runtime, you can continue to use an out of date Workday Studio without updating or patching.

WORKDAY STUDIO SECURITY BASICS

Workday has separate security requirements to set up and launch integration systems (including EIBs) based on the integration type. Below are the integration-related security domains and what permissions are granted for each domain.

Task	Required Domain Access
Configure an EIB.	Integration Configure
Configure one or more integration services.	Integration Configure
Create or edit an EIB.	Integration Build
Create or edit an integration system.	Integration Build
Deploy a Workday Studio integration.	Integration Build
Debug a Workday Studio integration.	Integration Debug
Edit an integration's subscriptions.	Integration Subscriptions
Launch an EIB.	▪ Put permission on a Workday Web Service operation that an inbound integration uses as a transport protocol (for example, the *Submit Customer Invoice* web service). ▪ Get permission on a Workday Web Service operation that an outbound integration uses as a data source (for example, the *Get Compensation Grades* web service). ▪ Access to a custom report that an outbound integration uses as a data source (unless a security proxy exists to run the report as another user). ▪ Integrations: EIBs ▪ Integration Event
Launch an integration.	Integration Event
Set up an EIB.	Integrations: EIBs
Schedule an EIB	▪ Put permission on a Workday Web Service operation that an inbound integration uses as a transport protocol (for example, the *Submit Customer Invoice* web service). ▪ Get permission on a Workday Web Service operation that an outbound integration uses as a data source (for example, the *Get Compensation Grades* web service). ▪ Access to a custom report that an outbound integration uses as a data source (unless a security proxy exists to run the report as another user). ▪ Integrations: EIBs ▪ Integration Event
Schedule an integration	Integration Event
View an integration's subscriptions.	Integration Subscriptions
View integration reports, including reports for integration events, exception audits, messages, and integration IDs.	Integration Reports
View resulting integration events, including integration output documents.	Integration Event

For more information on configuring security for Workday Studio and enabling the correct domain policies, please see the Knowledge Base of the Developer Network on Workday Community.

https://community.workday.com/studio-kb

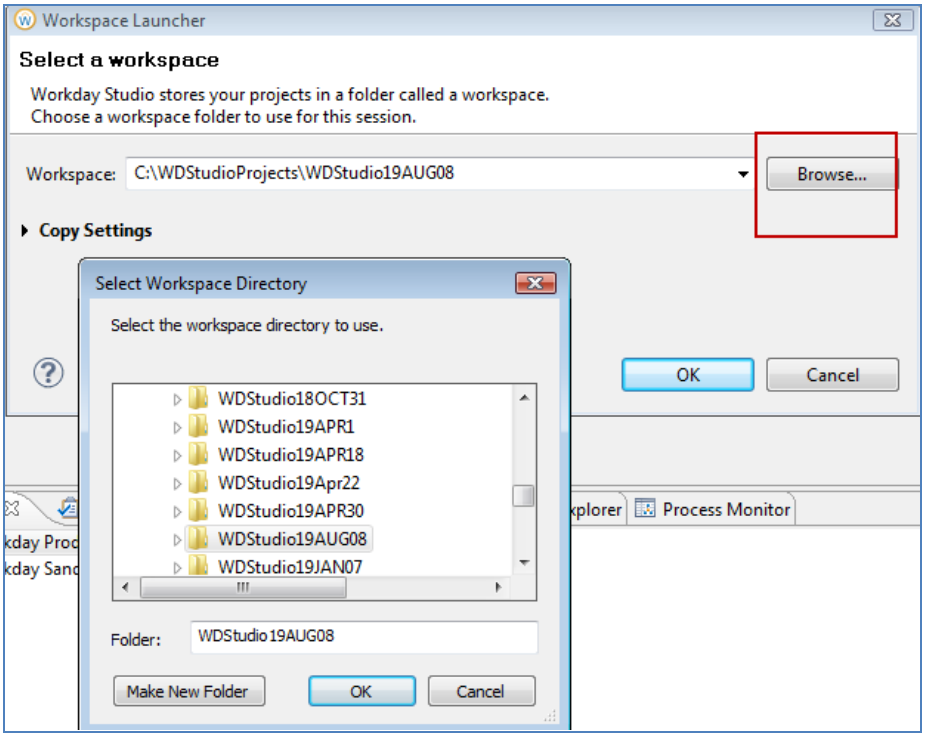
WORKDAY STUDIO NAVIGATION

WORKSPACE

A workspace is a logical collection of projects. A workspace is a directory on your hard drive where Studio stores the projects that you define to it.

Your workspace path should not include spaces because the Studio Debugger will not work. Connection Details are associated with your current workspace. If you switch or create a new workspace, you will need to configure your connection details.

To change your workspace, or create a new one, navigate to **File >> Switch Workspace >> Other**. The Workspace Launcher dialog box will open and allow you to browse for or manually enter a new workspace location.

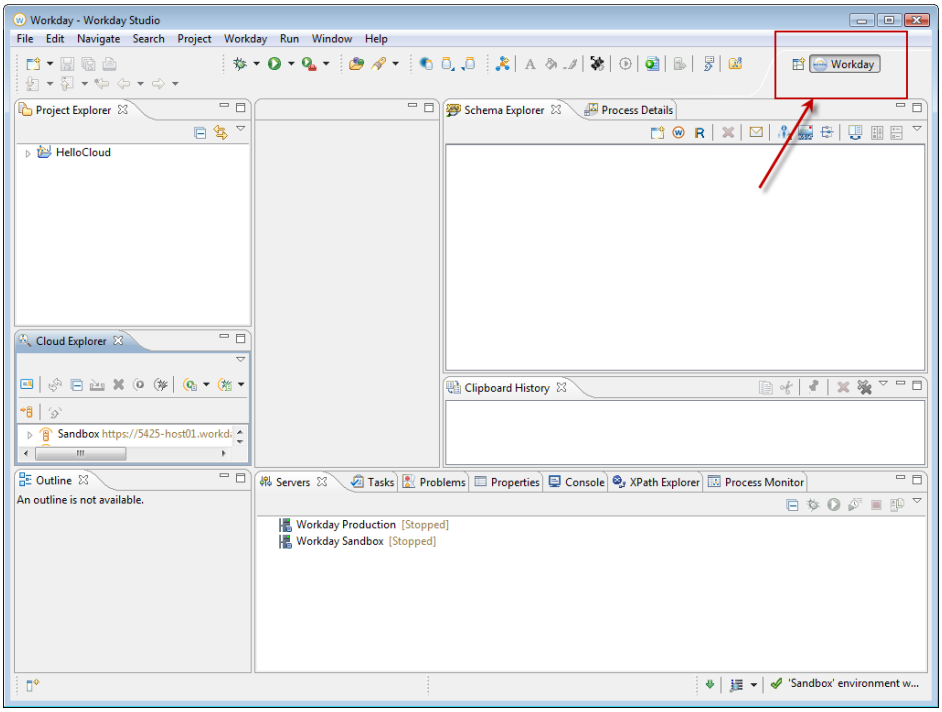


PERSPECTIVE

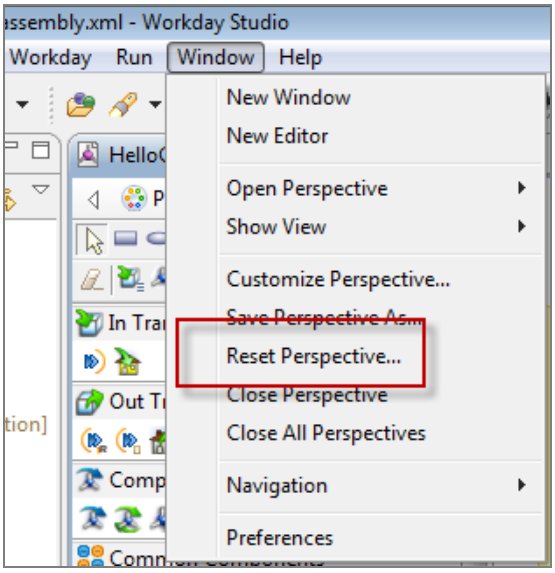
Each Workspace window contains one or more perspectives. A perspective defines the initial set and layout of views in the Workspace window. Within the window, each perspective shares the same set of editors. Each perspective provides a set of functionality aimed at accomplishing a specific type of task or works with specific types of resources.

Perspectives control what appears in certain menus and toolbars. They define visible action sets, which you can change to customize a perspective. You can save a perspective that you build in this manner, making your own custom perspective that you can open again later.

Workday delivers a custom perspective that defaults when you open Workday Studio. The Workday perspective provides one-click access to pages within the Workday application that are relevant for integration development, in general, and for specific Workday Studio Cloud integrations. The first time that you open the Workday view within a Studio session, the Workday view displays the Workday login page. After you enter your login details successfully, on all subsequent launches within the same session, the Workday view provides one-click access directly to the relevant pages.



If you need to reset back to default, navigate to **Window >> Reset Perspective...**



VIEWS

Views support editors and provide alternative presentations as well as ways to navigate the information in your Workspace. For example, the Project Explorer and other navigation views display projects and other resources that you are working with. Perspectives offer pre-defined combinations of views and editors.

To open a view that is not included in the current perspective, select **Window > Show View** from the main menu bar.

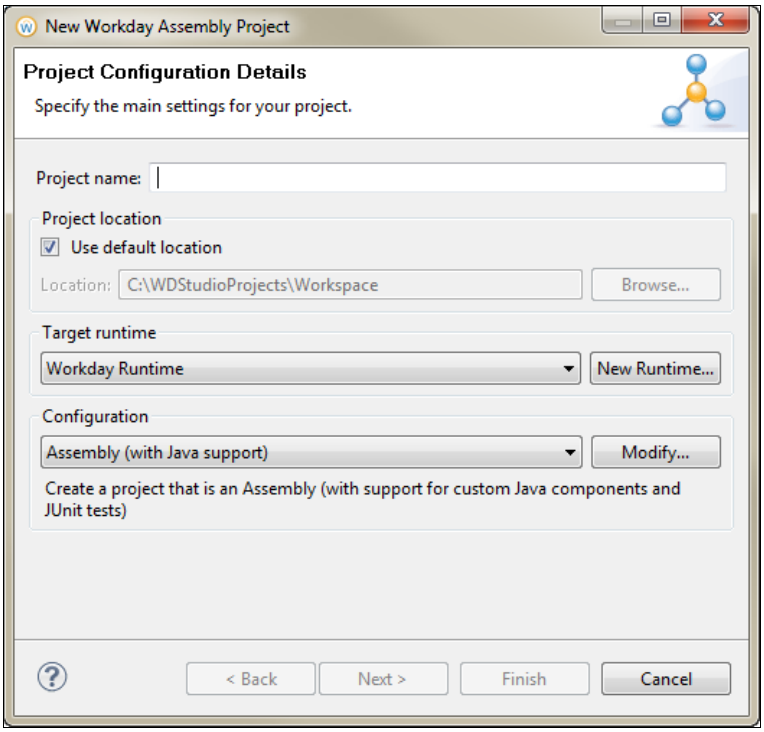
PROJECTS

Projects contain the files and resources needed to build and deploy a Workday Studio Integration System.

An Assembly defines the message-based business processes consisting of transports, processing steps, and routing components. The Assembly is defined within Spring-based files called assembly.xml which reside within assembly projects. The easiest way to create an assembly is by using Workday Studio's Assembly Editor.

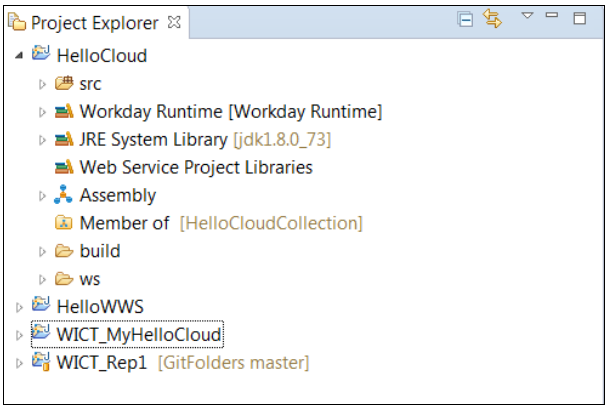
The Assembly Project wizard helps users quickly generate an assembly project. The generated project will contain a basic assembly.xml file, which will be your starting point. All required resources and Workday libraries are automatically included in the project.

From the **File** menu, select **New>Workday Assembly Project**. The **New Workday Assembly Project** wizard is displayed as follows:

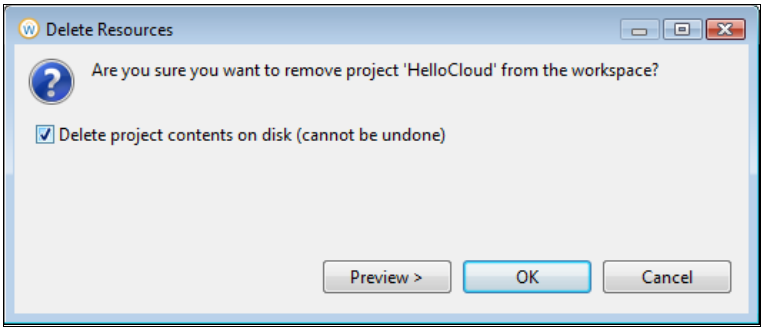


PROJECT EXPLORER VIEW

The **Project Explorer** is the default project navigator. It displays the assembly at the top level of the project structure.

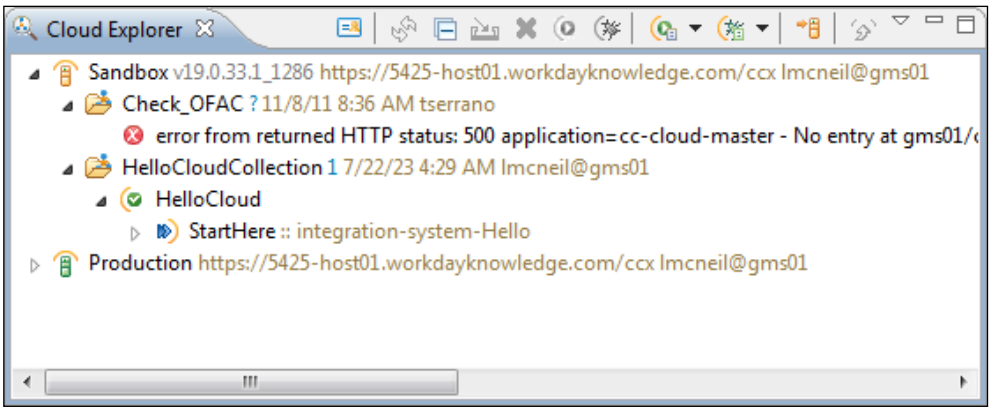


When you are working with the project in the Project Explorer, you are working on your local machine. The projects in the Project Explorer are pointers to files stored on your hard drive. If you delete from the Project Explorer, the files will remain on your local drive unless you check the box to delete the contents on the disk.



CLOUD EXPLORER VIEW

The Cloud Explorer is a "view" into your tenant. What you see in this view is what is deployed in the tenant identified in the connection details. Everything is secured so if you don't have access to an integration system, you will not be able to launch it.

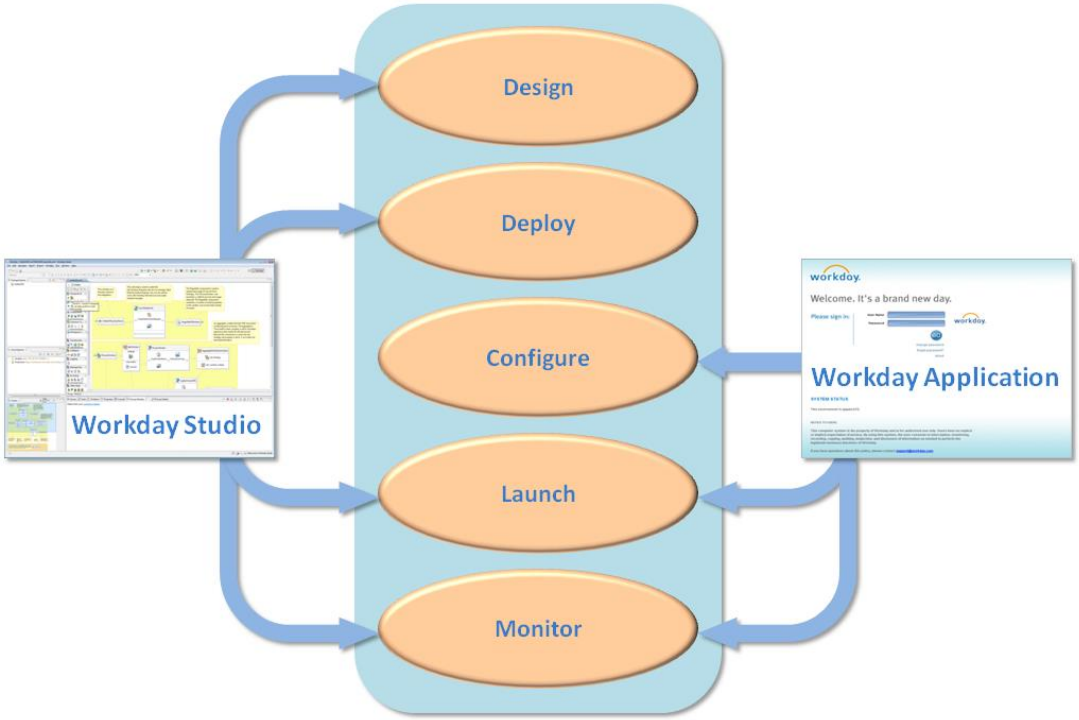


WORKDAY STUDIO PLATFORM

Studio provides an IDE (Integrated Development Environment) that helps developers design and build integrations. It is built on top of the open-source Eclipse IDE. With this development tool, you create an *assembly* from a selection of reusable components that include transport, routing, configuration, and mediation steps. These assemblies form the basis of your integration.

Depending on the integration phase, you can use either Workday Studio or the Workday application to work with the integration. Studio integrations have the following distinct phases:

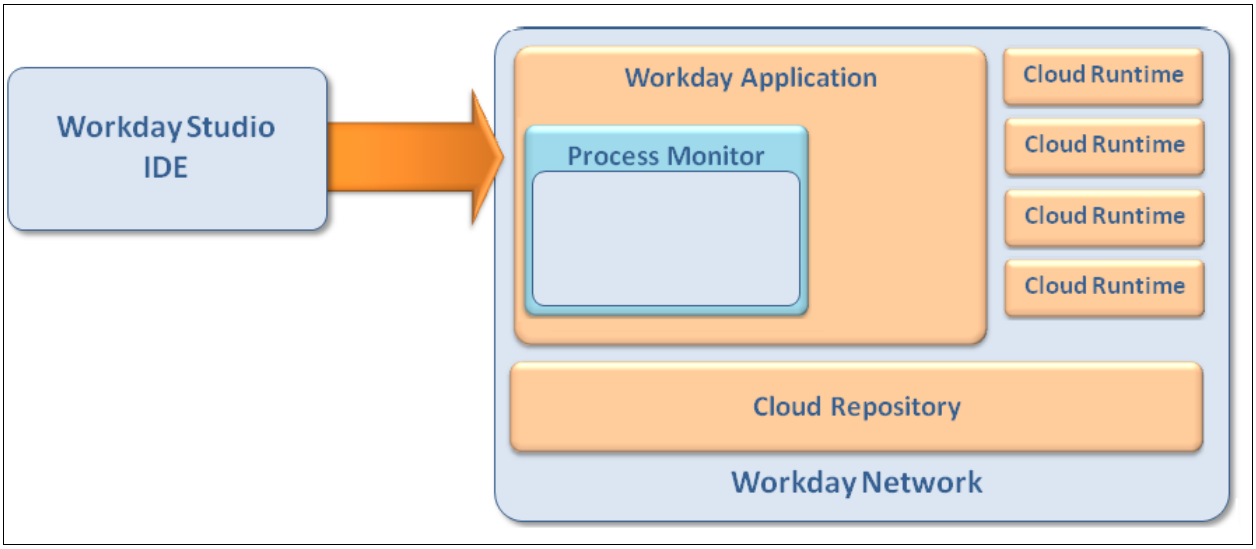
- **Design:** Using Studio's design-time environment, you can develop the assemblies that form your integration.
- **Deploy:** From Studio, you can connect to the Cloud Repository and deploy your integration.
- **Configure:** Within the Workday application, you can configure certain aspects of your integration, including user permissions and notifications.
- **Launch:** You can launch the integration in one of several ways, using either Studio or the Workday application.
- **Monitor:** After launching, you can track the progress of the integration from Studio or the Workday application. You can also view a consolidated report of the integration process after it has completed.



Integrations, collections, and the Cloud Repository: Integrations are applications designed to integrate and run within Workday's Web service environment. Studio organizes integrations into groups called collections and deploys the integrations to Workday's *Cloud Repository*. Workday's network hosts the Cloud Repository.

The Workday application: The Workday application maintains information about the integrations that you deploy to the Cloud Repository in an abstraction called the integration system. This allows you to configure various aspects of the integration such as security and setup data from within the Workday User Interface (UI). The Workday application also maintains a Process Monitor, which you use to track the execution history of integrations.

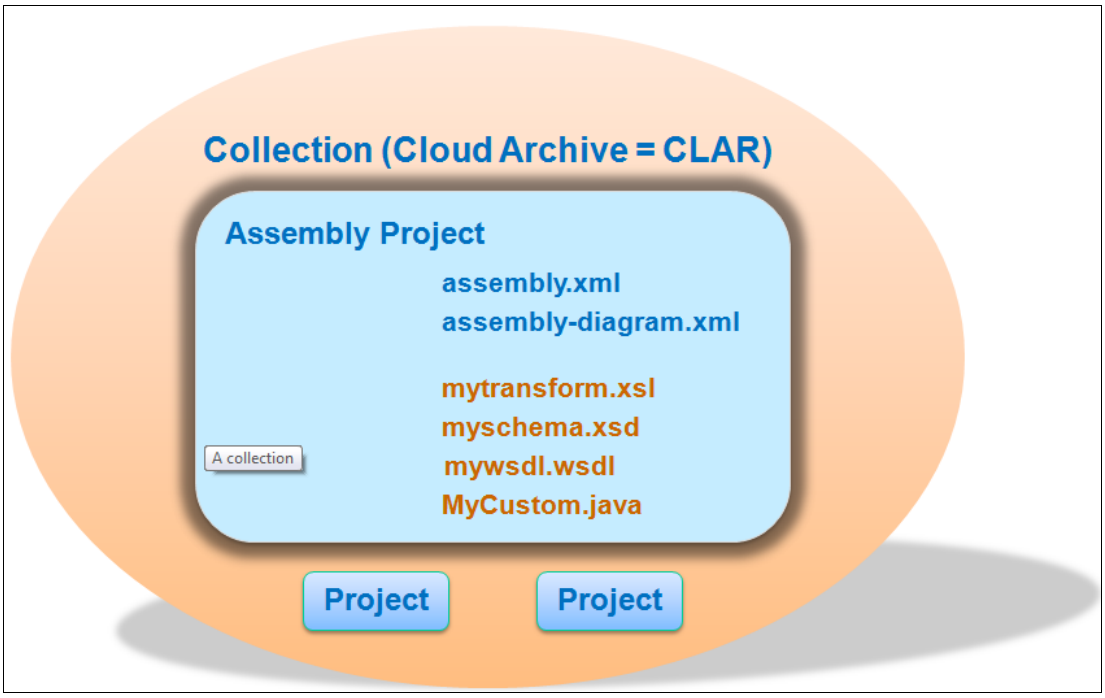
Cloud Runtimes: When the Workday Studio platform launches an integration, it allocates a *Cloud Runtime* to the integration. This acts to ensure that integrations are protected from each other. Instead of executing integration events immediately, the Workday network server places each event in a queue, awaiting access to an available Cloud Runtime. When one becomes free, it retrieves the integration from the repository and deploys it to the Cloud Runtime for execution.



INTEGRATIONS, COLLECTIONS, AND THE CLOUD REPOSITORY

Assemblies provide a simple framework for implementing integrations. Within Workday Studio, collections are a deployable unit and are the basis for deployment to the Cloud Repository. A collection can contain one or more integrations.

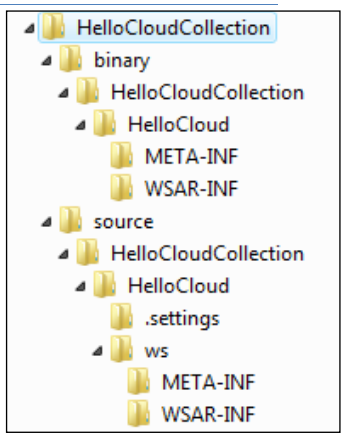
A collection is a packaging concept that Studio uses to construct a binary deployment artifact called a **CLAR** (*CLoud ARchive*). A **CLAR** contains all of the artifacts from the integration projects, for example, the assembly.xml file, any XSLT, and any Text Schema files. It also includes compiled Java code and any third-party Java libraries that developers have added to the integration projects. Collections also allow developers to organize their own reusable assembly components into common projects and share those between integrations.



Studio allows developers to deploy CLARs to Workday's Cloud Repository. This is a resource within Workday's network where Workday stores your integration. When deployed, users with the appropriate privileges can further configure the integration in the Workday application, launch it, or schedule it to launch, so that it can execute.

Because the Cloud Repository stores binary CLARs only, CLARs do not include source-code artifacts such as Java source files. You should ensure that your integration projects are properly backed-up. The Eclipse-based Studio provides support for a number of different version-control systems that developers can use to version their integration projects.

The Cloud Repository only maintains the last deployed copy of a CLAR. In addition, when you un-deploy a CLAR using Studio, Studio also removes the CLAR from the Cloud Repository.



The CLAR archive file extracts into the following folder structure:

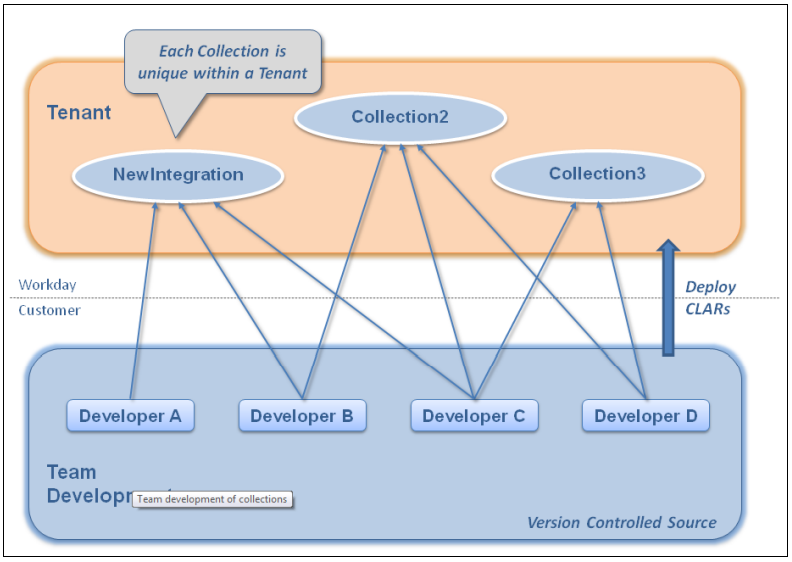
Binary: This folder contains a collection folder, which contains all of the binary artifacts for the collection's projects that are required for deployment to the Workday environment.

- **Collection-name**: This folder contains a clar.xml file and one or more projects.
    - **Project-name(s):** Each project contains the following sub-folders:
        - META-INF, which contains the MANIFEST.MF file. This file provides information about the files packaged in the archive file, for example, the manifest version and classpath details.
        - WSAR-INF, which contains the assembly.xml and assembly-diagram.xml files and any additional files that the assembly references, for example, XSL and text schemas, and any required libraries.
- **Source**: This folder contains a collection folder, which contains the integration source code, that is, the original Workday Studio project from which the CLAR was exported and its associated settings.
    - **Collection-name**: This folder is a container folder for one or more Workday Studio projects, with the following sub-folders:
    - **Project-name(s):** Each project contains the following sub-folders and the project's .classpath and .project files:
        - .settings, which contains project settings and facet configuration files.
        - ws, which contains:
            - META-INF
            - WSAR-INF

* See Studio Help for more information on building the manifest file.

DEVELOPING INTEGRATION COLLECTIONS

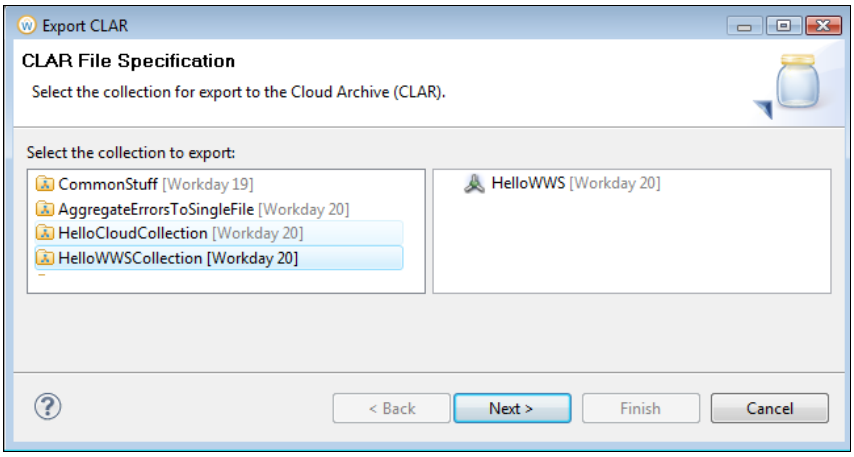
Collections are unique within each Workday tenant, however, you can co-develop a collection in a version-controlled, team-development environment. This means that developers can collaborate in the development of integrations that form a collection to be deployed to a tenant. Any developer with appropriate privileges can deploy a newly developed collection to the Workday Cloud Repository. It is important to note that only one instance of a particular collection can run at any time. This means that if you modify an assembly project within a collection and redeploy the project, it replaces any previously deployed version.



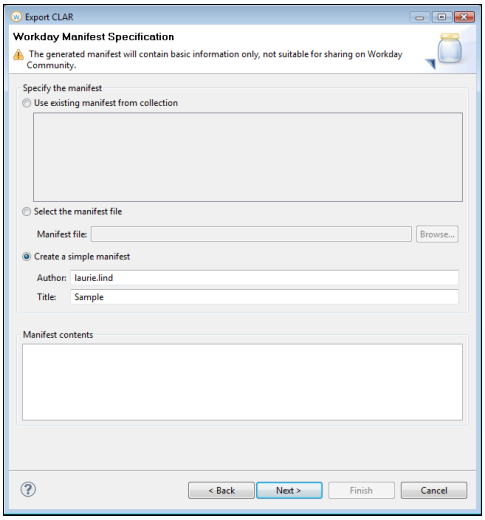
EXPORT

You can export a collection and its associated solution to a CLAR without deploying it to Workday.

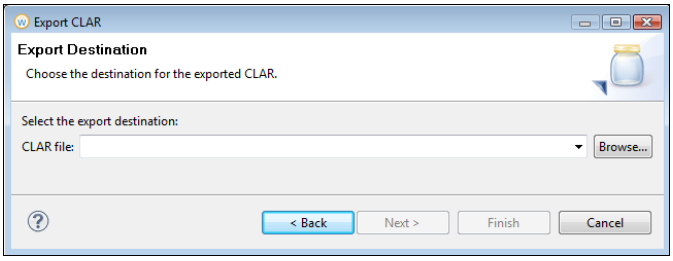
Using the Workday menu, you can **select Solution >> Export** then select the collection you wish to export.



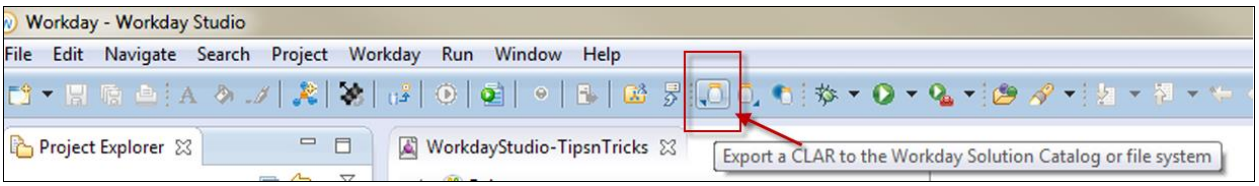
You can select an existing manifest file, or create a simple manifest. A simple generated manifest will contain basic information only, not suitable for sharing on Workday Community. Please see Workday Studio Help for more information on creating a manifest file.



Finally, select the export directory where you want to save your clar file.



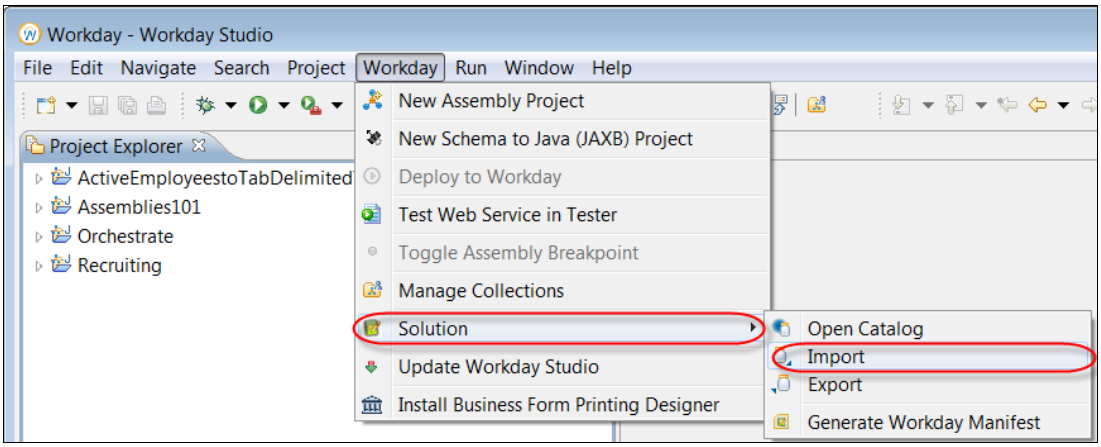
You can also click the **Export** a CLAR to the Workday Solution Catalog or file button in the Studio toolbar or right click on the project in the **Project Explorer** view.



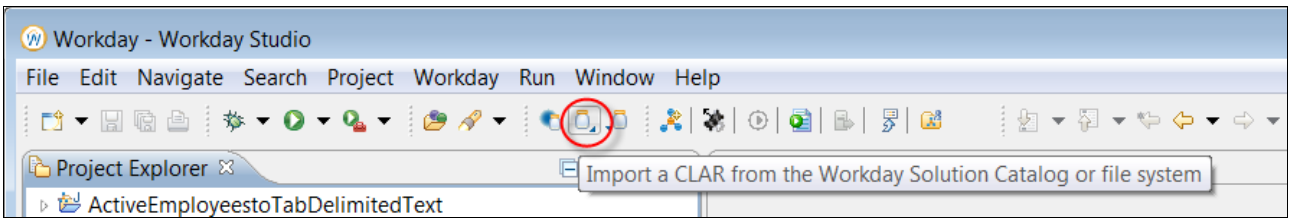
IMPORT SOLUTIONS

You can import a CLAR in any of the following ways:

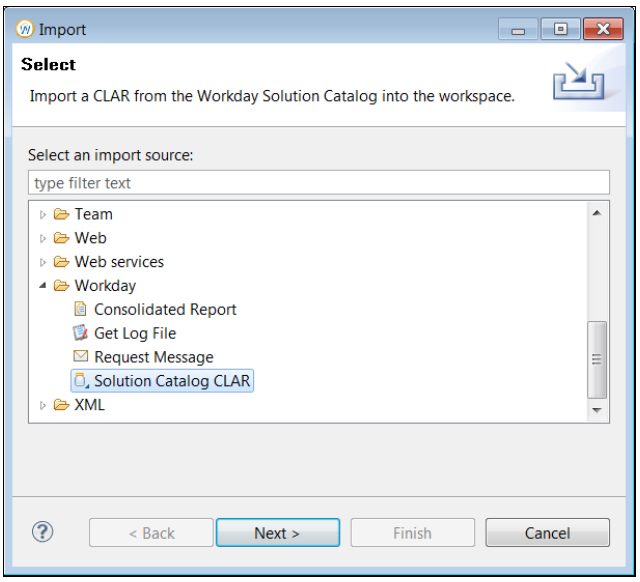
From the Studio menu bar, select **Workday> Solution> Import.**



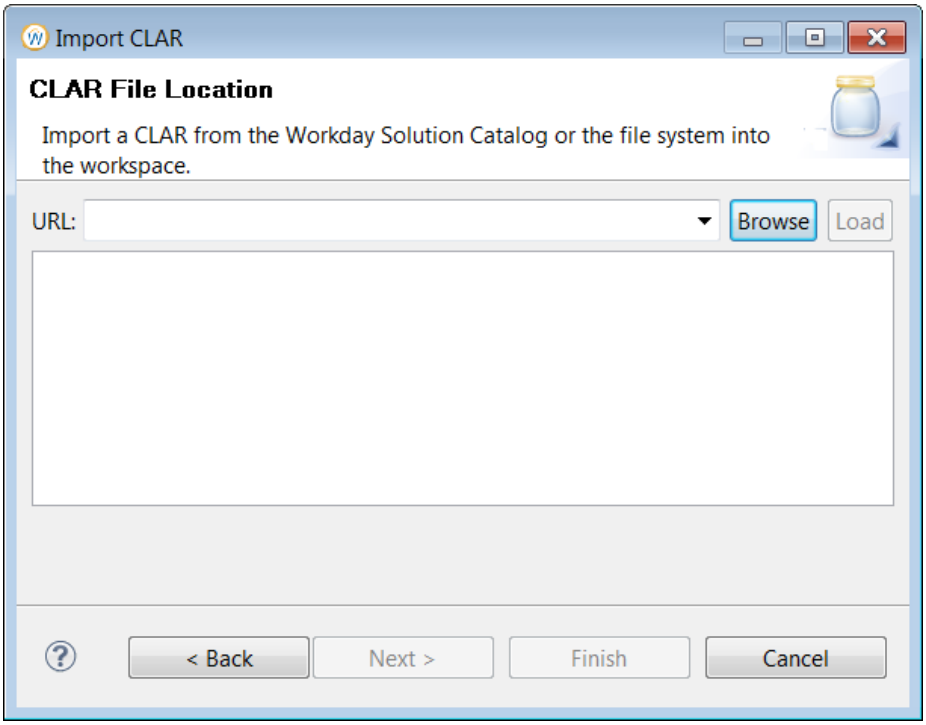
You can also click the Import a CLAR from the Workday Solution Catalog or file button in the Studio toolbar.



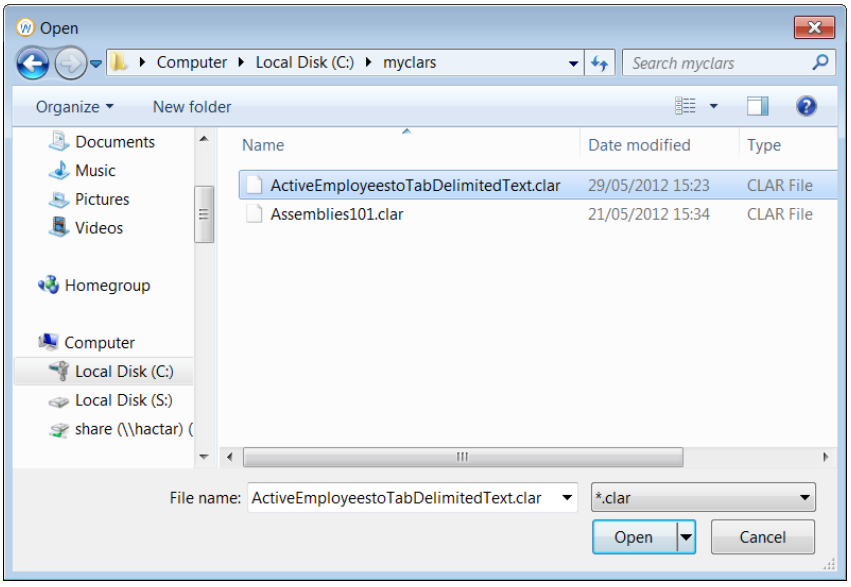
Alternatively, from the **File** menu, select **Import...**. The **Import** wizard is displayed. Expand the **Workday** folder, and then select **Solution Catalog CLAR**.



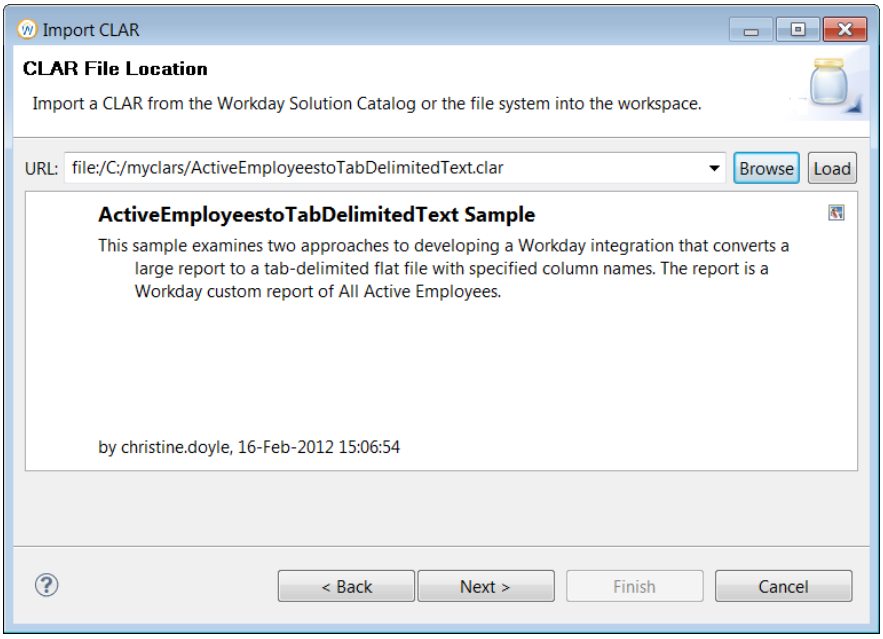
The **Import CLAR** wizard opens. On the **CLAR File Location** screen, you can browse to import a CLAR from the Solution Catalog or your file system into the workspace.



Click **Browse** to select the CLAR file, and then click **Open**.

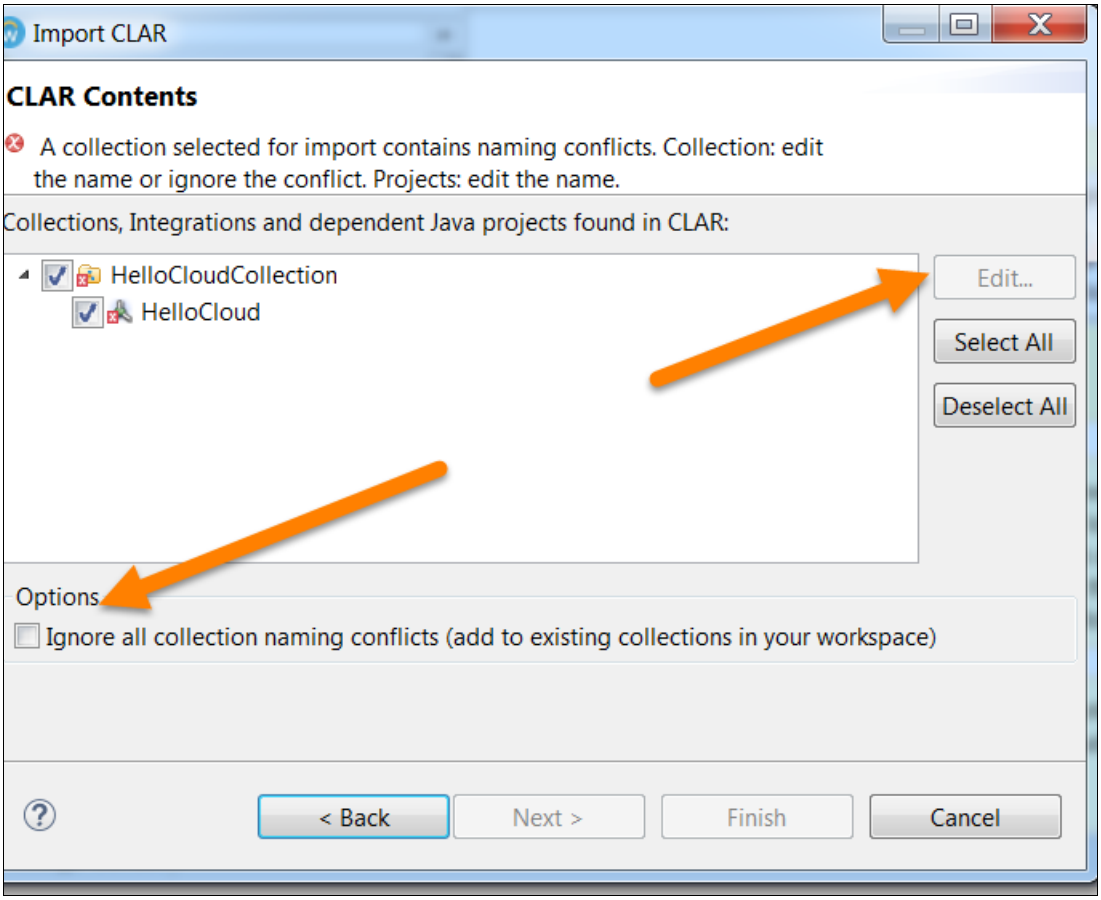


Click **Load**. Studio loads the file and displays the contents of the manifest file, such as the title, description, and icon in the **Import CLAR** wizard.



Click **Next**. The **CLAR Details** page is displayed.

If a project with the same name already exists in the workspace, Studio displays a message to indicate that you may either; rename the project that you are importing, or ignore the conflict and overwrite the existing CLAR on your workspace. This prevents you accidentally replacing existing projects in your workspace that may be under source control, for example, if you are using applications such as Subversion (SVN) for team development.



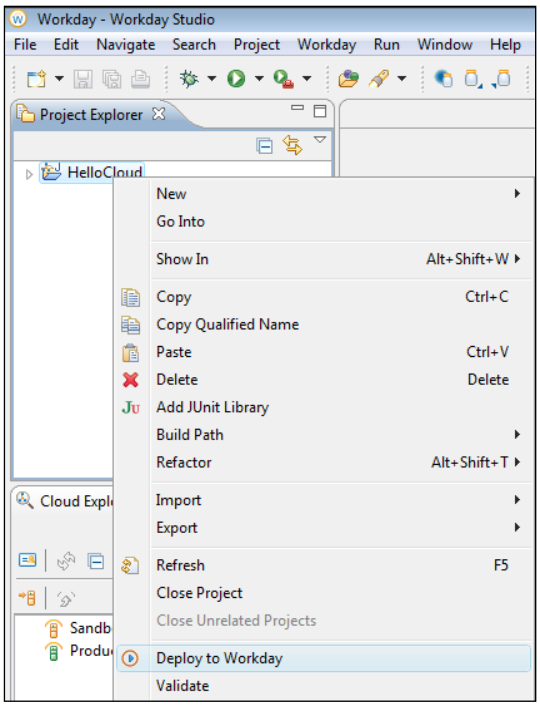
To rename a project or collection, select the items to rename and click Edit.

If the project does not already exist in the workspace, a message is not displayed and you can continue.

Click **Finish**. Studio imports the project from the CLAR and displays it in the Project Explorer. You are now free to work with the project in the same way as any Workday Studio integration project.

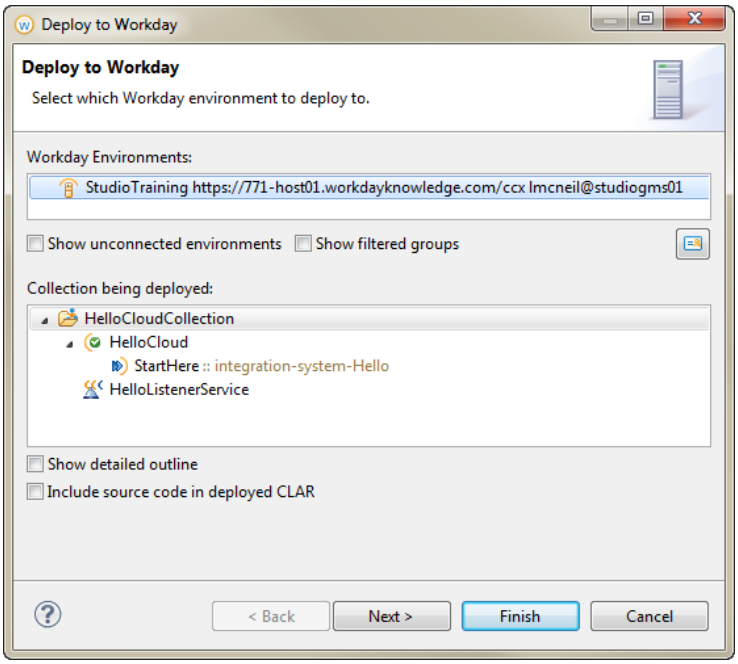
DEPLOYING STUDIO PROJECTS TO WORKDAY

When you select a project to deploy, Studio deploys all other projects that are members of the same collection. To deploy a collection of projects, follow these steps:

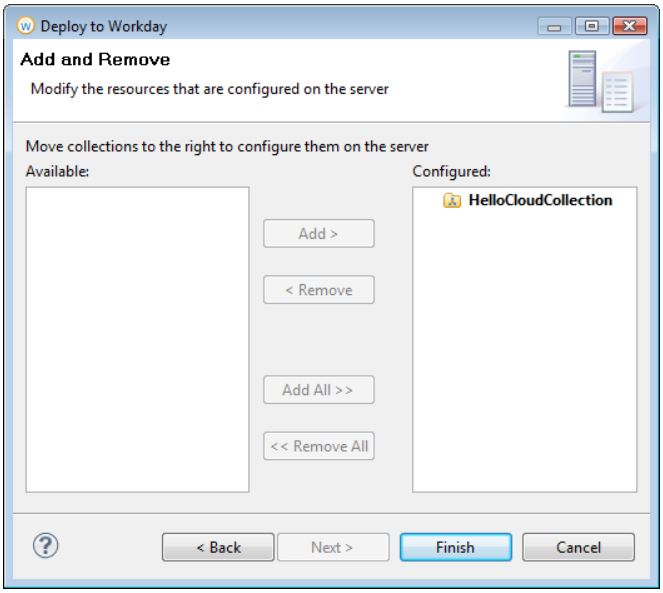


Right-click the sample project folder in the **Project Explorer** view, then select **Deploy to Workday**.

The **Deploy to Workday** wizard opens. By default, the **Workday Environments** pane shows all Workday environments that you are currently connected to as part of your Workday Studio session.

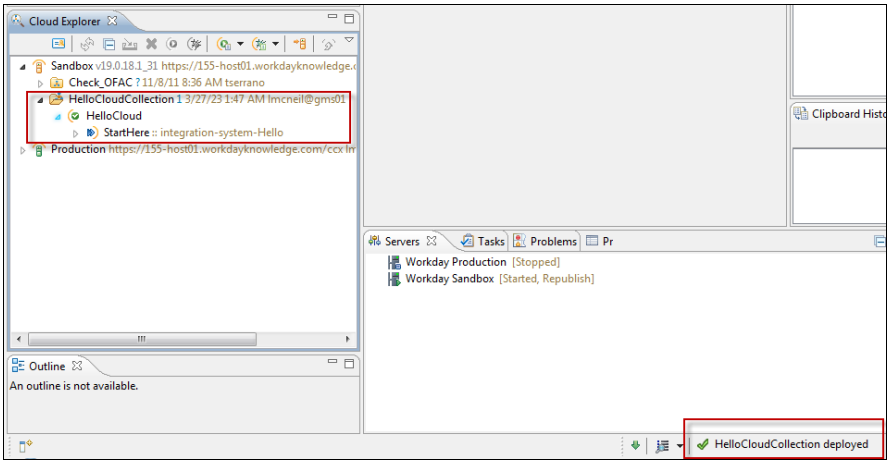


Click **Next** to continue through the deployment wizard to the **Add and Remove** screen.



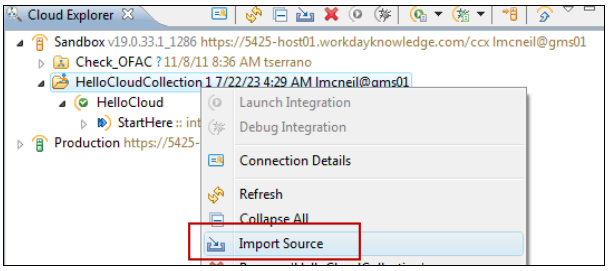
Because the assembly project is a member of a collection, the collection is automatically added for deployment; however, you can use the **Add** buttons to select additional available collections, if any, to deploy. Note that in the **Configured** list, the collection that the project belongs to is highlighted in bold, because the collection is required.

When a collection is deployed to the selected Workday environment, a message is displayed in the **Status Bar**.

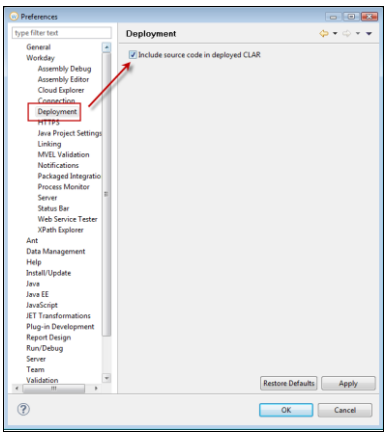


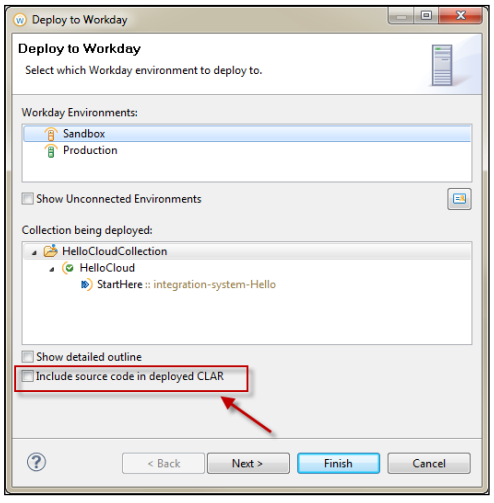
SOURCE CODE FOR DEPLOYED INTEGRATION IN TENANT

If you select the **Include source code** in deployed CLAR option when you deploy to Workday, you can import the source to a project in your workspace.



You can either change your preferences or check the box when you deploy to Workday. Under **Window >> Preferences**, you can navigate to **Workday > Deployment**. When you deploy the project to Workday, you also have the option to select (or deselect) deploy with source code.





VERSION CONTROL

Workday and Workday Studio are not version control systems. However, Workday Studio helps development teams design and build integrations. The Workday Studio platform is an Eclipse-based IDE, which installs Eclipse, as well as several Studio plug-ins. As with every development environment, it requires a version-control system to allow developers to collaborate and work effectively in a team environment, by keeping track of changes in a set of files. Eclipse supports several version-control systems, for example:

- Subversion
- Concurrent Versions System (CVS)
- Git
- Mercurial
- Bazaar

Before using a version-control system, such as SVN, CVS, Git, Mercurial, or Bazaar, you need a repository, which is a central location for managing your files using your version-control system. Your company may already have one. Check with your IT department or System Administrator to obtain access to the URL and user login credentials.

Workday provides an introduction tutorial to version-control systems. It demonstrates how to add Workday Studio Assembly project files to a repository, check out projects, edit your local copy, commit changes, revert your changes, and commit reverted changes back to the repository.

Disclaimer: This tutorial is provided for informational purposes only and does not endorse or recommend using one version-control system, client application, repository tool, or service over another for commercial purposes or otherwise with Workday Studio.

ACTIVITY 1 – MANAGE WORKDAY STUDIO WORKSPACE

In this activity you will open Workday Studio and practice managing your Workspace by Exporting, Deleting and Importing Solution Catalog CLARs.

Steps:

1) Open Workday Studio. If you have already configured your connection details, enter the **username** and **password** in the splash screen. Otherwise, you will need to access the connection details once Studio opens to enter in the **username** and **password**.

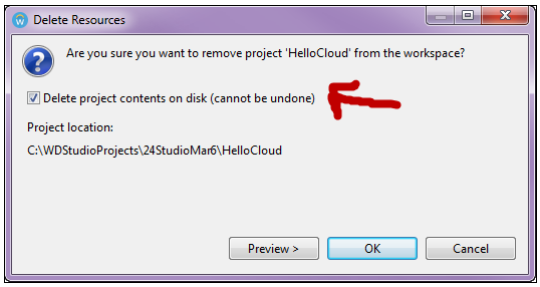
2) Locate and make a note of your Workspace path. **File >> Switch Workspace >> Other**

   _____

   * Make sure there are no spaces in the entire path. Spaces will prevent the Studio Debugger from working.

3) Export HelloCloud as a *Solution Catalog Clar*.
   a) Choose to create a Simple Manifest file
   b) Browse to a location on your hard drive and name the solution **<your initials>_HelloCloudSolution.clar**
   c) Select **Save** then **Finish**

4) Delete the existing HelloCloud project from your Project Explorer and make sure to delete it from the disk.



5) Import HelloCloud as a *Solution Catalog Clar*.

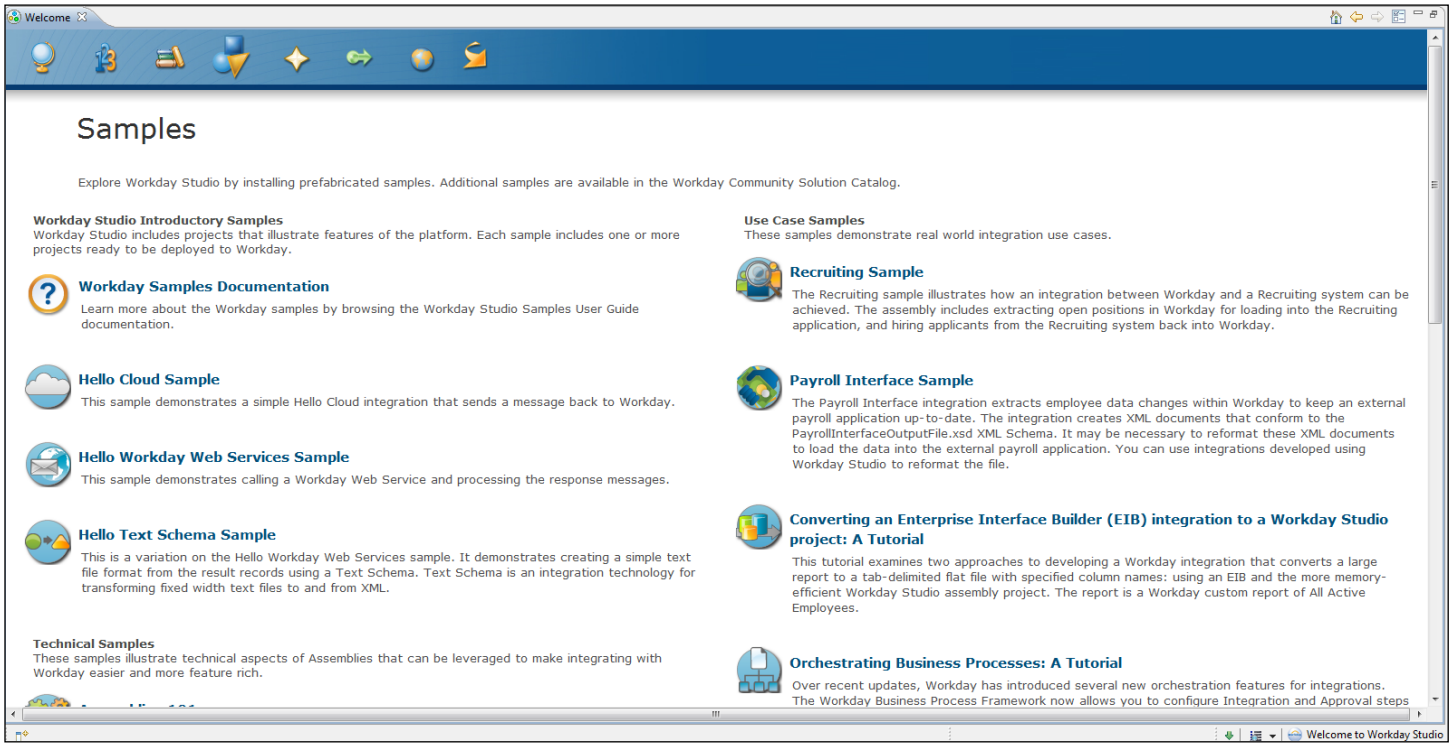
6) Deploy HelloCloud to Workday and check the box "include source code".

7) Delete the existing HelloCloud project from your **Project Explorer** view and make sure to delete it from the disk.

8) Right click on the HelloCloudCollection in the **Cloud Explorer** view and select **Import Source** then follow the prompts.

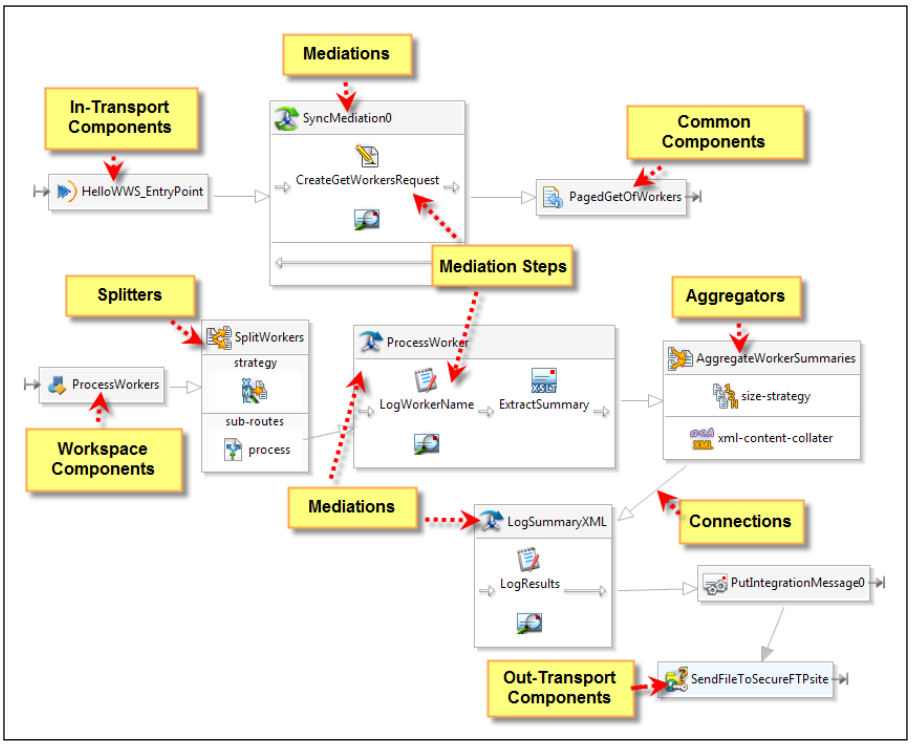
WORKDAY STUDIO SAMPLES

Workday Studio has many samples and tutorials available to help guide you through learning the different components. You can find samples under the **Help >> Welcome** menu once you have launched Studio.



ASSEMBLIES

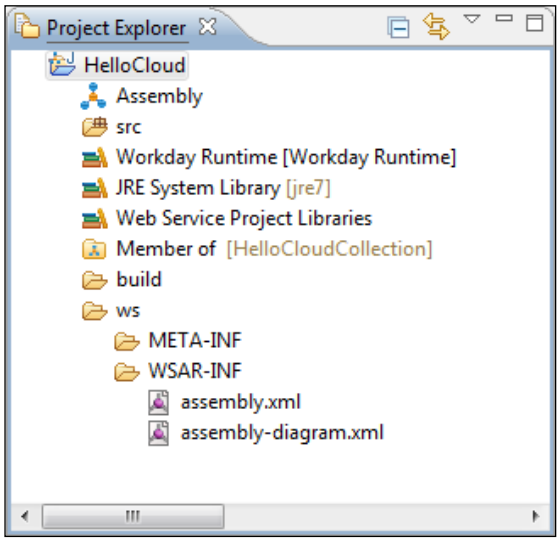
Assemblies provide a unified environment for all the core functions of Workday Integration System (transportation, routing, transformation, and error handling). This functionality is provided in the form of individual components that are connected or *wired* together to form processing chains. You have complete freedom in how you choose to wire chains and components together.



You design assemblies using Studio's visual Assembly Editor. This is a graphical editor that allows you to define a flow of messages exchanged between Workday and other systems. The Assembly Editor provides an extensive range of packaged components for manipulating messages and communicating with other systems over various protocols such as FTP, SFTP, SMTP (e-mail), HTTP, and HTTPS. It supports message transformation using XSLT (Extensible Stylesheet Language Transformations), a form of XML-to-Text mapping called Text Schema, and message construction using the powerful write step.

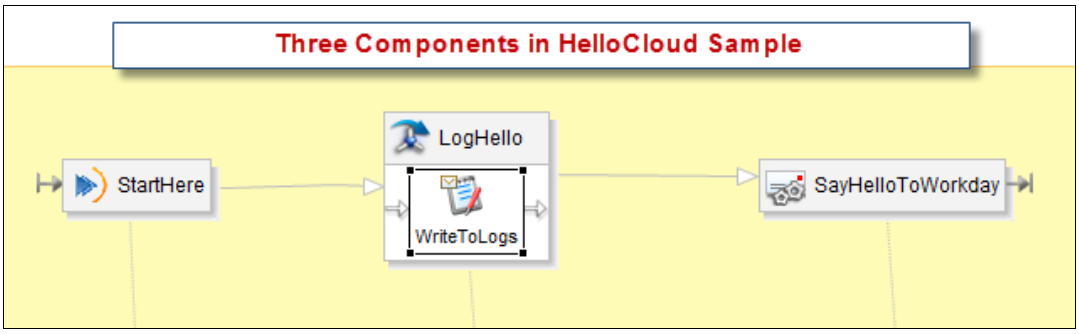
Studio is designed specifically to facilitate and expedite integration with Workday. The purpose-built Workday transports, combined with the Workday Public API, remove much of the effort normally associated with system integrations.

To open the **Assembly Diagram**, expand the project in the Project Explorer and double click Assembly to open it in the Workspace area.



ASSEMBLY DIAGRAM

The Assembly Diagram consists of Components and Connections. Components are "wired" together with Connections which drive the message flow through the Assembly.



Assembly components are classed into the following categories:

Transports: The Assembly runtime supports several transports. There are two IN-TRANSPORT components: Workday (workday-in) and local (local-in). In-transport components receive messages from a Workday or local resource. There are multiple OUT-TRANSPORT components that can send messages to a resource, for example HTTP, FTP, email, custom, and so on. Transports can operate in request-only, or request-response modes.

Basic Mediation Components: Mediations are the main mechanism for modifying and transforming a message as it flows through an assembly. A mediation defines a sequential ordering of one or more individual processing steps that perform operations on a message as it is passed from one step to another within the mediation definition.

Mediations can either be synchronous or asynchronous. Synchronous mediations consist of two sub-elements, one for request steps and one for response steps. Asynchronous mediations define a single set of steps.

Advanced Assembly Components: Basic components provide a way to connect processing steps together into processing chains. The advanced components provide a way to dynamically control how the assembly runtime traverses the processing chain.

Advanced Mediation Components include:

- Route
- Splitter
- Aggregator

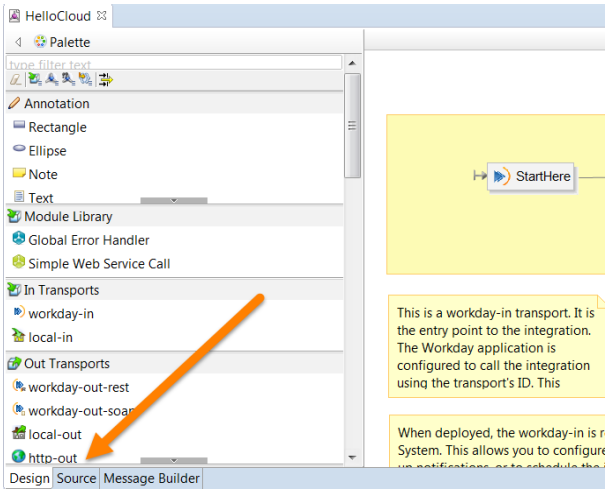
Common Components: Pre-packaged sub-assembly components delivered by Workday that you can use when developing Workday integrations.

Mediation Steps: Pre-configured steps that support operations such as transformation, validation, logging, identity, among others. You can also write custom steps using Spring beans.

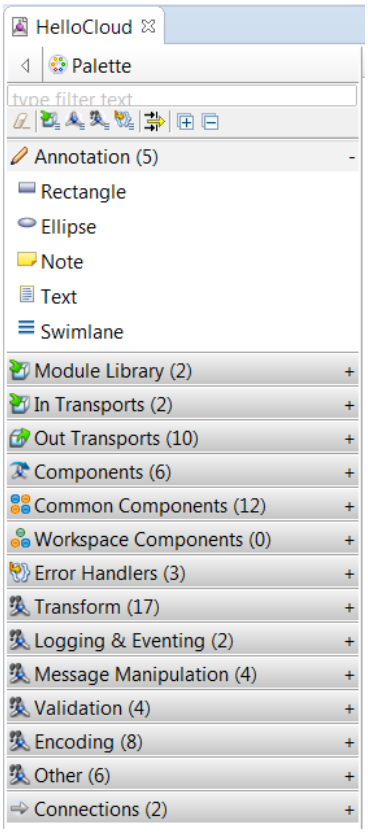
BASIC NAVIGATION

Using the Assembly Editor you can graphically build up your assembly by adding and linking transports, mediations, and steps together to define your integration.

There are two representations of an assembly. The first is a graphical representation of the underlying **assembly.xml** file. This is displayed in the default **Design** tab. A **Source** tab also exists and displays the actual Assembly XML. You should add and configure Assembly components using the editor and not by directly modifying component XML within the **Source** tab



The main area of the Assembly Editor is a diagrammatic representation of the transports, mediations, steps, and general assembly components that combine to define the overall business processing performed by the assembly. The **Palette** lists all of the assembly components. You can add components to the assembly diagram by dragging them over from the palette. You can select an element in the diagram to view and edit its properties in the **Properties** view.



ASSEMBLY TOOLS PALETTE

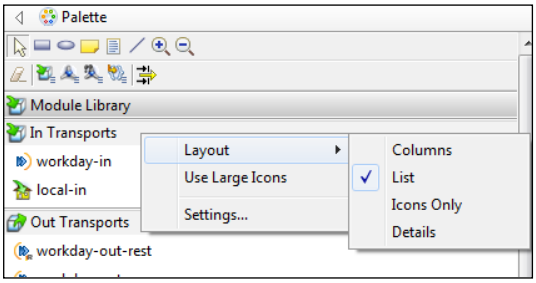
The **Palette**, which is located on the left-hand side of the main editor view within the workspace, displays tools for adding assembly components. Add components to the assembly diagram by dragging them over from the palette. Hover over an icon to get more information in a tooltip.

The palette groups components into a number of categories:

- Transports, both in and out
- Components, such as asynchronous and synchronous mediations, routes, aggregators, splitters, common components and workspace components.

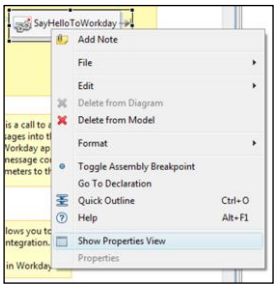
- Mediation steps that transform, manipulate, or log messages.
- Error Handlers

You can change the view of the icons on the palette by right clicking and select the Layout you prefer.

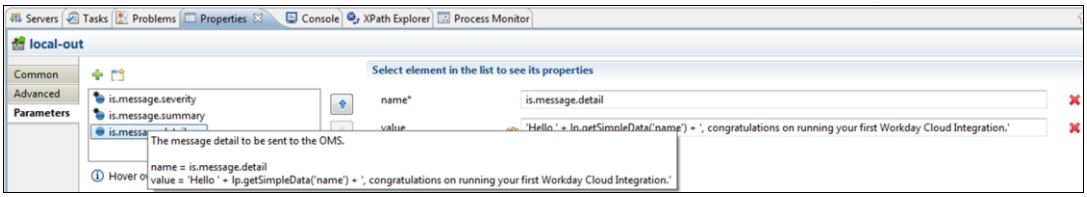


PROPERTIES

Each Component has properties that are configurable. Right-click component in the assembly diagram to Show Property View. The contents of the Properties tab will vary with each type of component.



Select an element in the diagram to view and edit its properties in the **Properties** view.

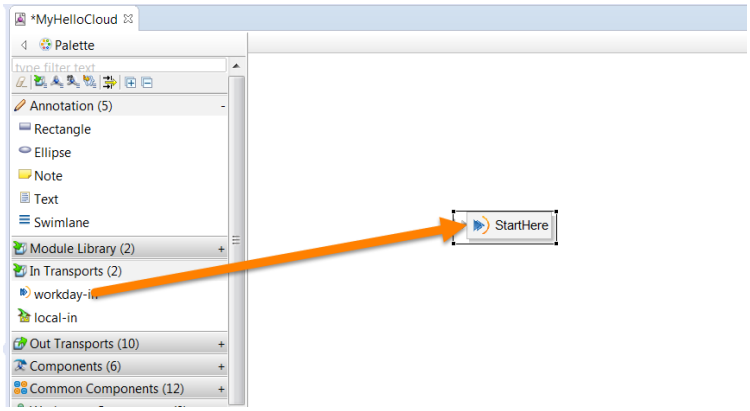


Some components have more than one item with properties. For example, the Asynchronous Mediation Component has properties. The Log Step inside of the mediation has its own set of properties. It is context sensitive when you right click to select to view the Properties tab.

WORKDAY-IN TRANSPORT

Every integration's assembly starts with a **workday-in** transport component that represents the integration system in the Workday OMS. The **workday-in** transport defines the attributes, maps, launch parameters, and services that are associated with the integration system.

The assemblies you create in Studio always contain one or more **workday-in** transports. The **workday-in** transport is designed specifically to facilitate and expedite integration with Workday's runtime environment.



WORKDAY-IN TRANSPORT PROPERTIES

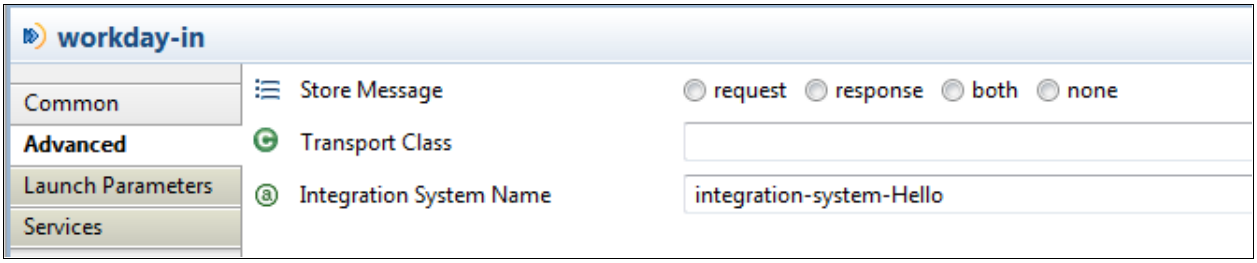
When you select the **workday-in** transport, the component properties are displayed in the **Properties** view below the assembly.

The **Common** tab allows you to configure the component's ID and routing details.

The **Advanced** tab allows you to configure the Integration System Name that will be seen in Workday.

The **Launch Parameters** tab to display the component's launch parameter details.

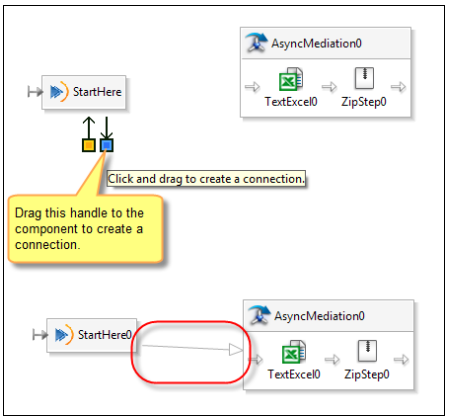
The **Service** tab is used to configure the integration services such as the Attribute Map Service, Sequence Generators, Report Service and Listener Service.



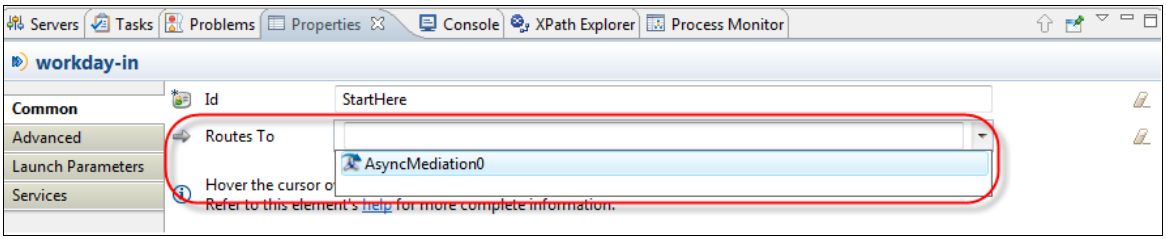
WIRING COMPONENTS

When a message arrives into an assembly on an in-transport, the path it takes through the assembly is defined by connectors between transports and message-consuming components such as mediations and routers. Components are "wired" or connected to one another using connection handles. The message output from each component becomes the input to the component it is connected to.

The routes-to and routes-response-to connectors are different colors, making it easy to distinguish them. Connect components by hovering over a source component until the connection handles display, then drag over to the target component.

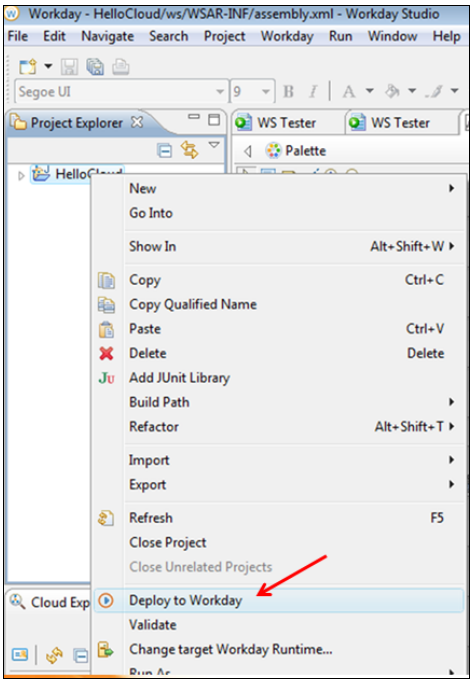


You can change where a connection starts or ends using drag and drop. You can also connect a component to another component in the assembly by selecting the target component from the **routes-to** drop-down list in the Properties of all the connectable components within the assembly:

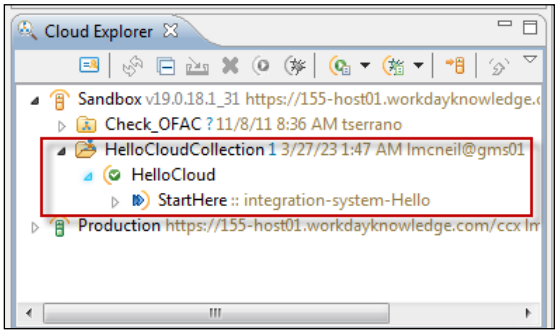


DEPLOYING TO WORKDAY

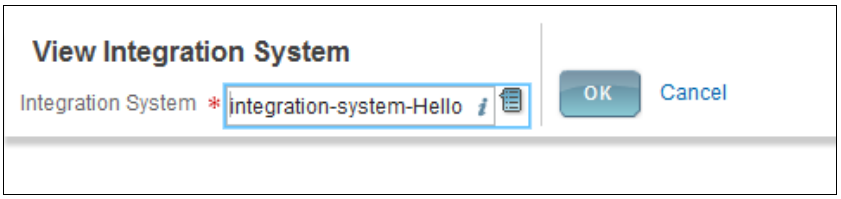
Once the Assembly is complete and saved, you can deploy it to Workday. To deploy, right click on the project in the Project Explorer and select Deploy to Workday.



When you have deployed your integration, you can view it in the **Cloud Explorer**.

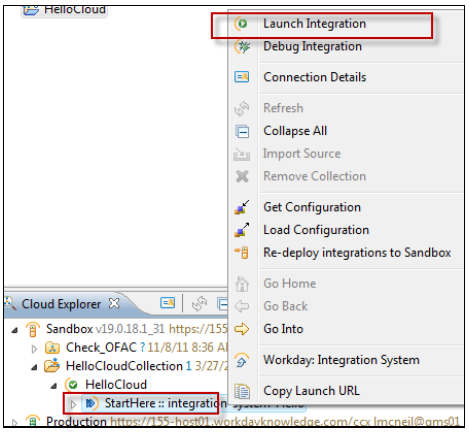


You can also view it in Workday. Search for the task **View Integration System** and select your integration.

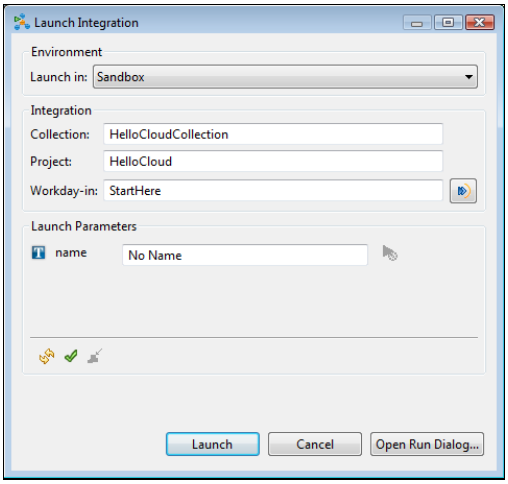


LAUNCHING AN INTEGRATION IN STUDIO

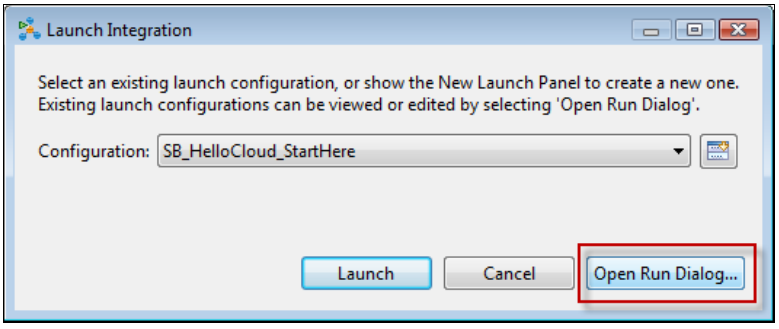
You can launch an integration system form Studio from the Cloud Explorer. Expand the Sandbox tenant until you reach "Start Here". Right click and launch the integration.



Review the Launch Integration Dialog Box and click **Launch**.

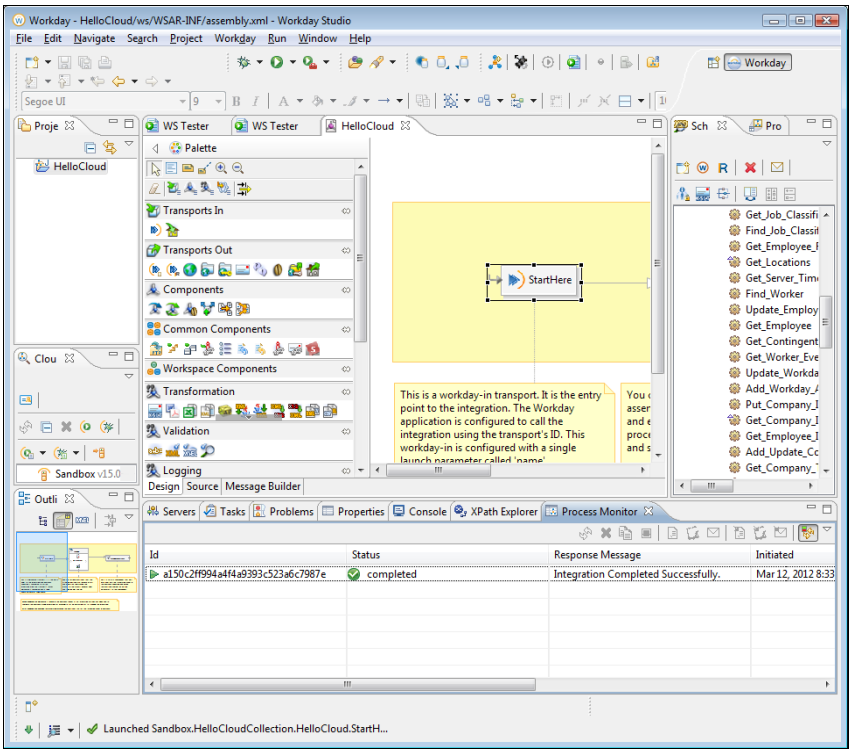


The first time you launch an integration system, the Launch Parameters are displayed. When you launch that same integration system from Studio again, the Launch Parameters are not displayed. To view the launch parameters, click on the **Open Run Dialog...** button.

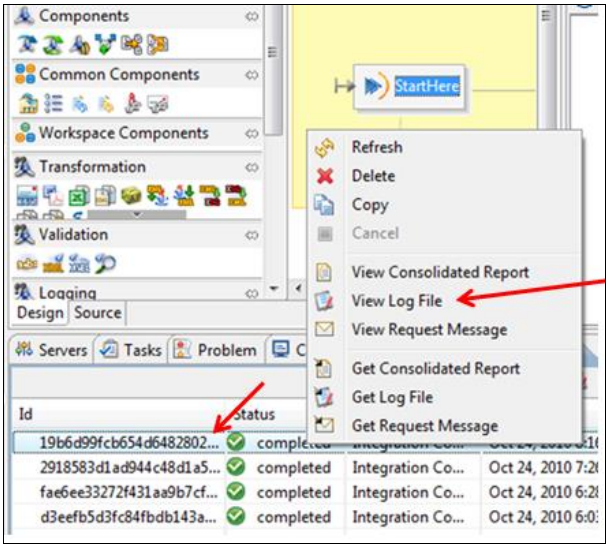


PROCESS MONITOR

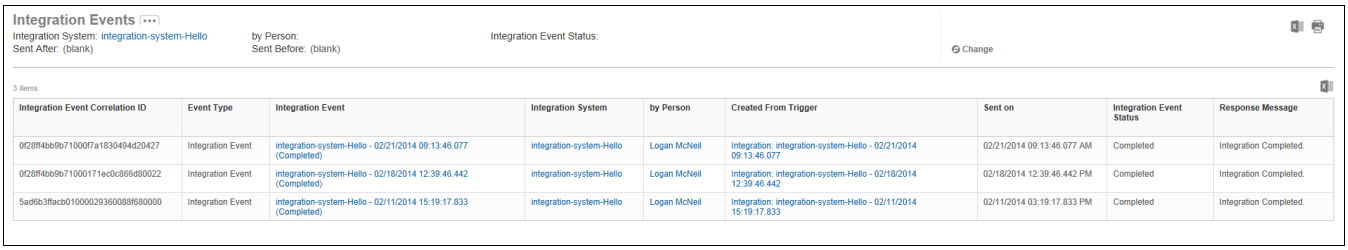
Studio contains a Process Monitor to review the results of the integration. The Process Monitor tab shows the status of the event which may be preparing, waiting, processing or completed. The status should automatically update but you may right click on the event and select **Refresh**.



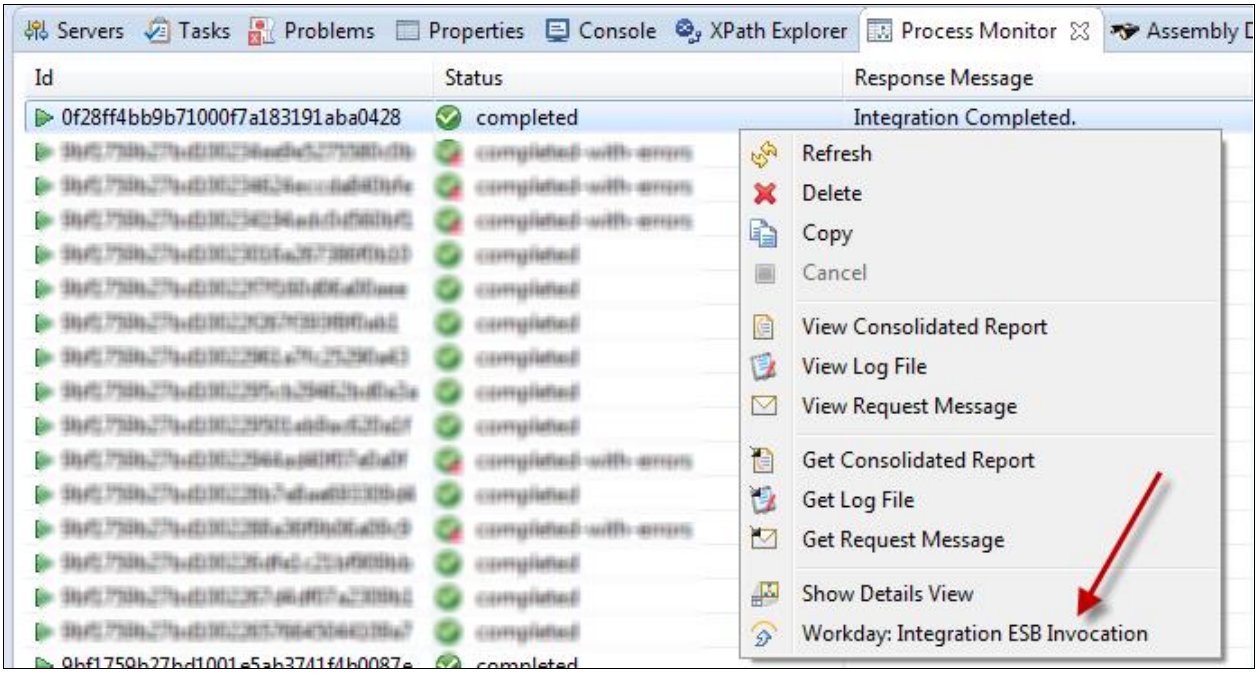
If you have included a Logging step in the integration, it is written to a log file viewable only in Studio. To view this log, right click on the event and select **View Log File**.



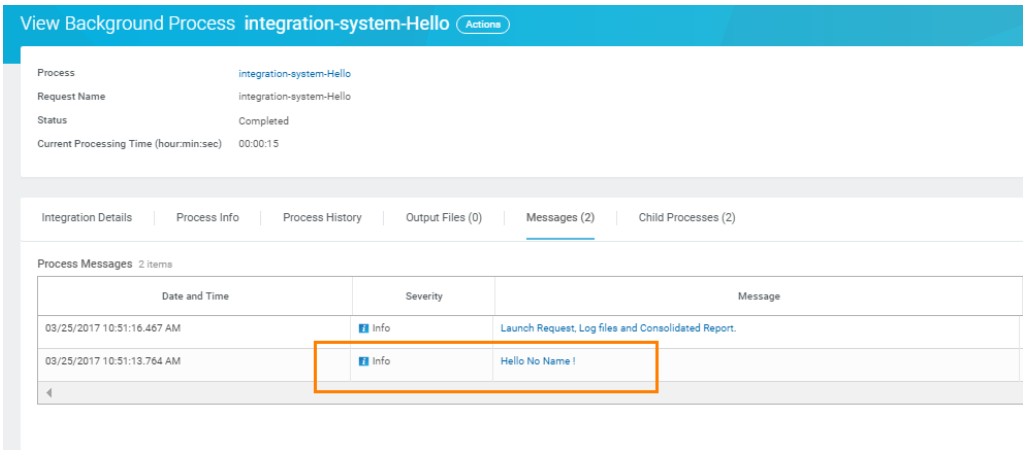
Each Integration Event is viewable in the Workday tenant. You can view the integration event in the tenant by running the **Integration Events** report.



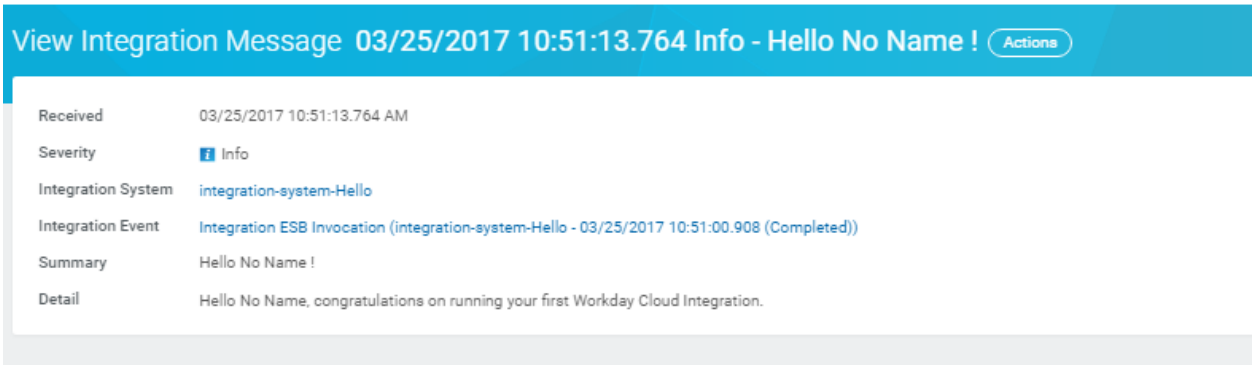
You may also right click on the event in Studio's Process Monitor view to access the event in the tenant.

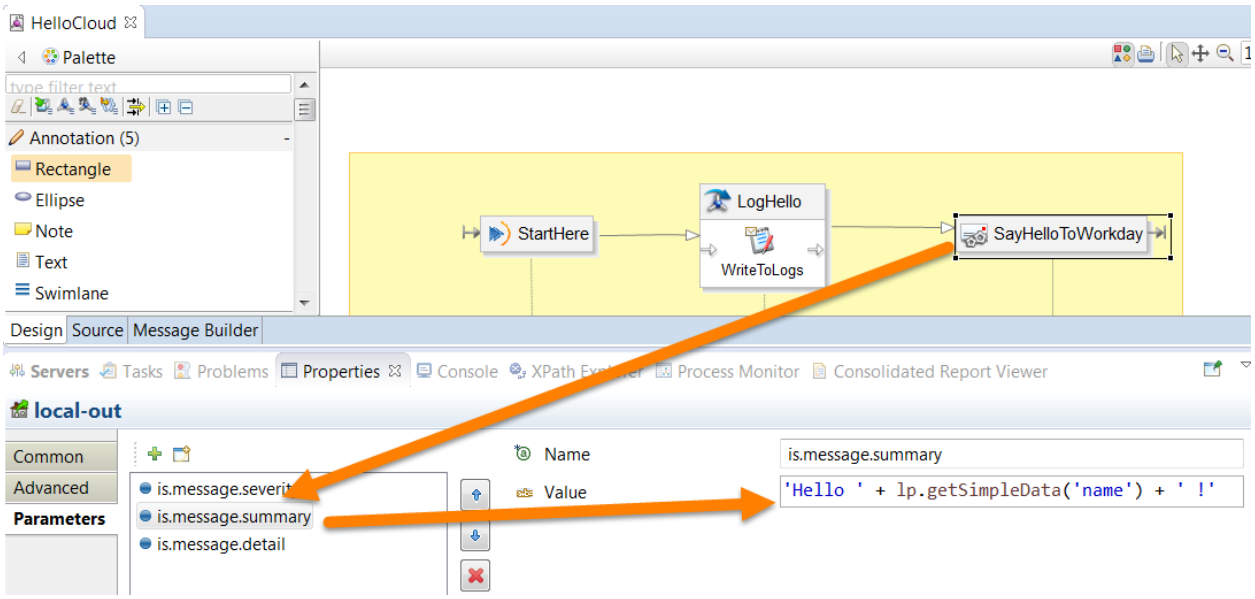


When you view the Integration Event in the tenant contains all the information about the event, including the messages written to the event through the Put Integration Message Component is studio. These message are viewable through the integration event only.



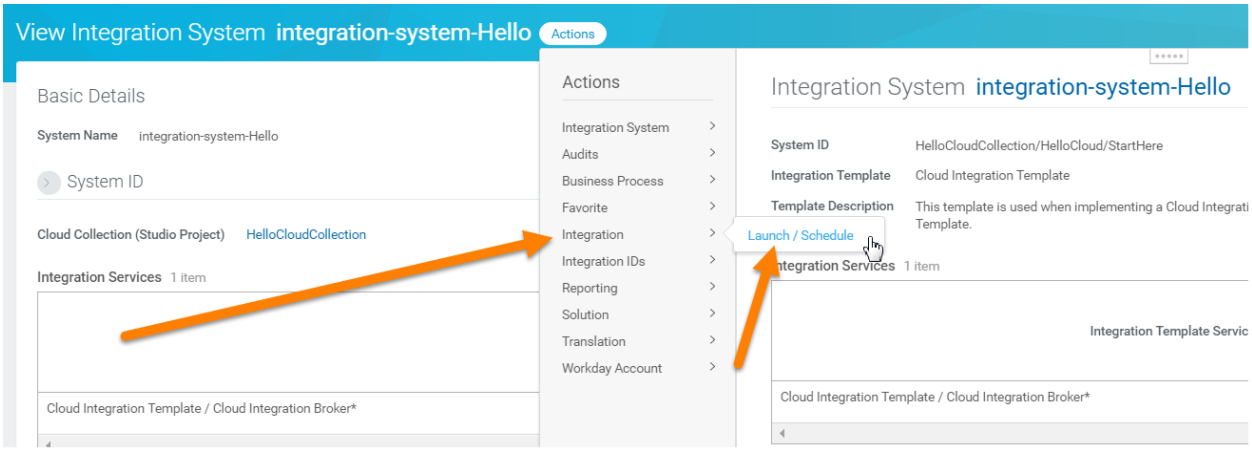
These messages are defined in the parameters in the properties of the **Put Integration Message** component.





LAUNCH INTEGRATIONS FROM TENANT

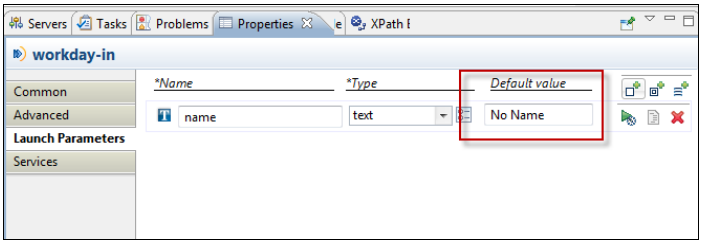
Once the Integration System is deployed, you can also launch it from the tenant. By navigating to the integration system in the tenant, you may use the related actions menu to Launch or Schedule the integration system.



Please note that you may see all launches of Workday Studio integrations, regardless of where they were initiated in the Integration Events report on the tenant. This is in contrast to the Process Monitor on the Workday Studio client which will only report the launches started within the Workday Studio Client

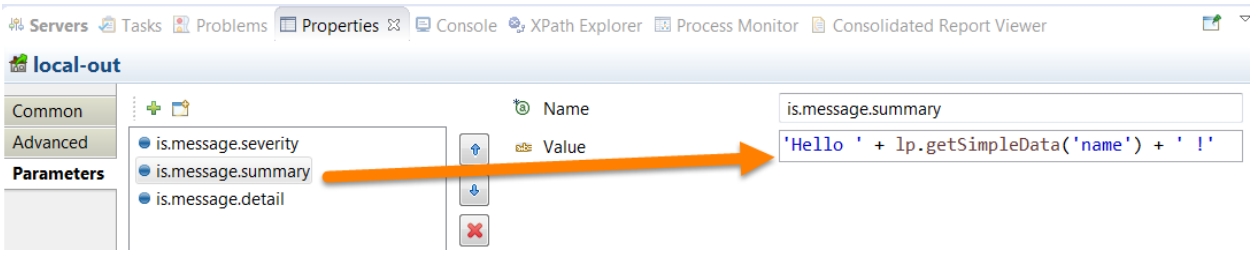
MODIFYING HELLOCLOUD COMPONENTS

To change the default value of the launch parameter, select the **Workday In** Transport and view the properties. Use the **Launch Parameters** tab to change the default value. *Creating Launch Parameters is covered in Advanced Studio training.*



PUT INTEGRATION MESSAGE

The **PutIntegrationMessage(PIM)** Component sends update integration messages to Workday. It can also add documents to the W: drive for the integration event covered later in this class.

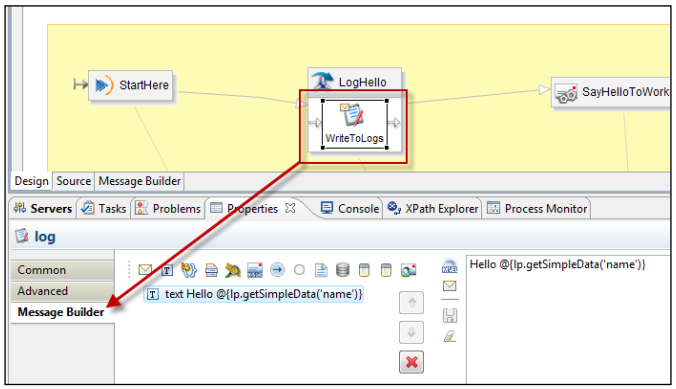


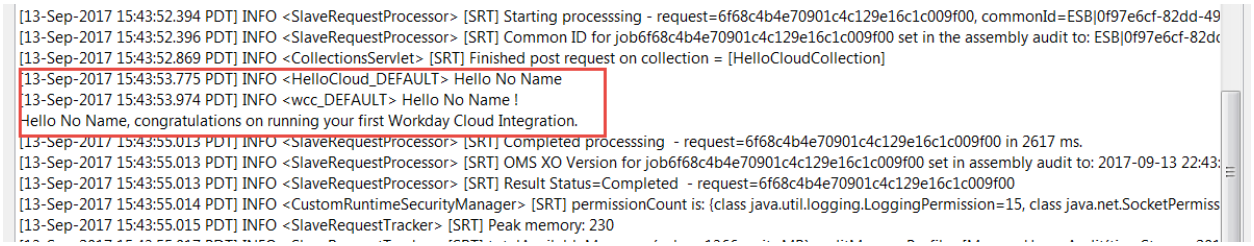
When entering values in the **PIM** parameters, you must include 'literal text' in single quotes.

LOG STEP

This step is used to create a customized string message that is logged at **INFO** level with log4j. The default *log4j appender* used writes the message out to the console in Studio.

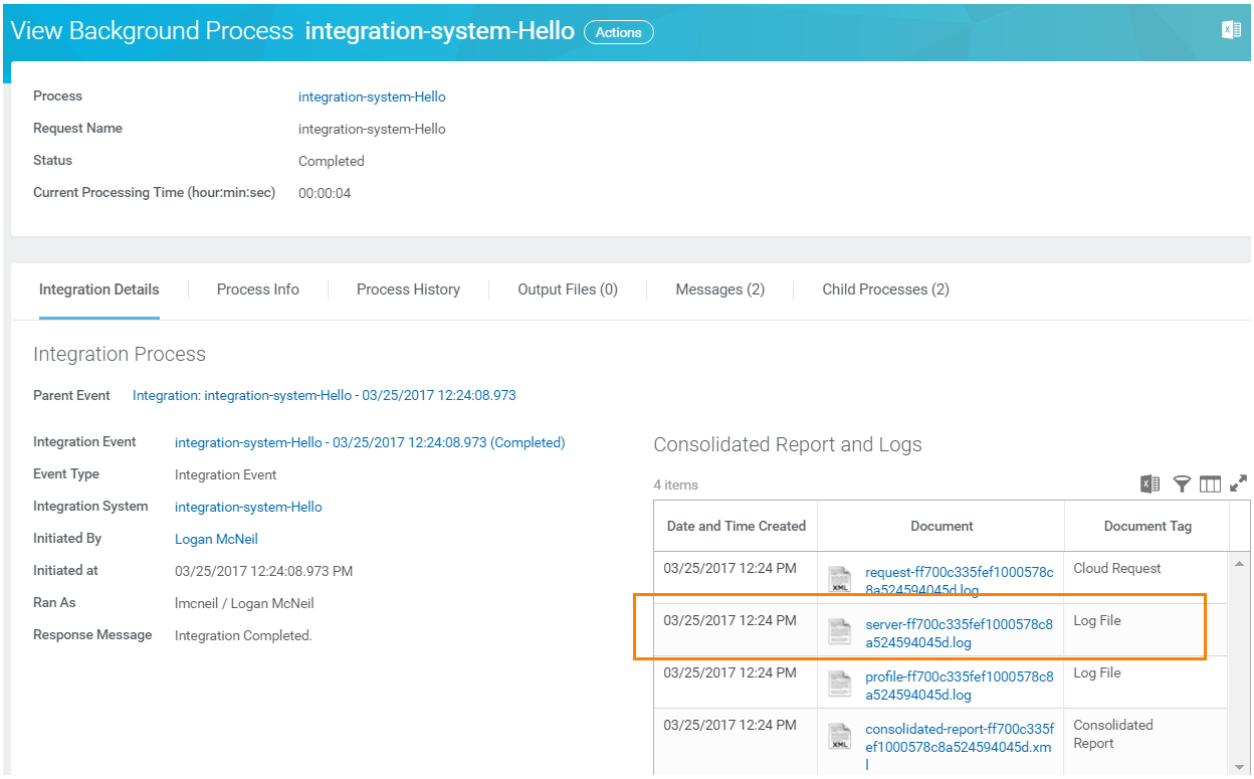
Select the Log icon to view the Properties tab and click on the Message Builder.



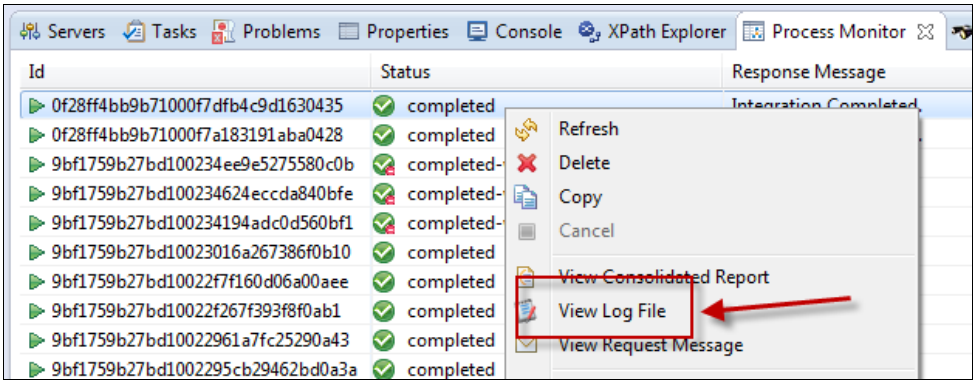


VIEWING THE LOG FILE

When you launch the integration from the tenant, you may generate the log file from the event.

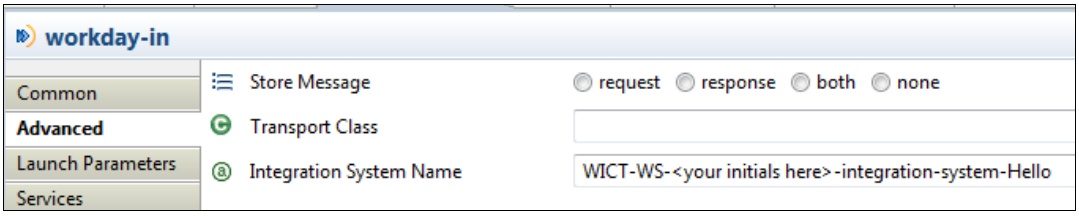


If you launch the integration system from Studio, you can view the log file from the Process Monitor tab in Studio (and generate it on the event).

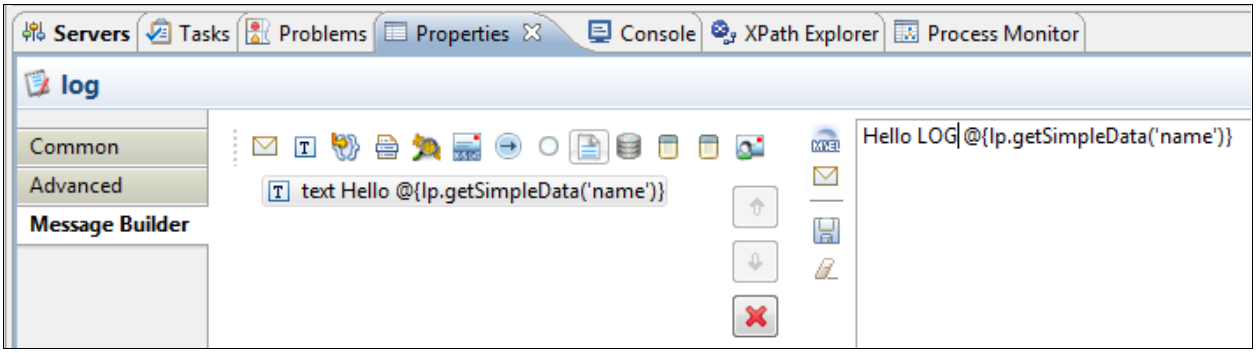


ACTIVITY 2 –MODIFY HELLOCLOUD COMPONENTS

In this activity you will modify the assembly in Studio by changing the name of the deployed integration. You will also change the log message from Hello<name> to Hello LOG <name>. Access the PutIntegrationMessage properties and change the Summary message from 'Hello'<name>'!' to 'Hello SUMMARY' <name>'!'.

1. Open the assembly diagram for *HelloCloud* in Studio
2. Right click on **Start Here** and select *Show Properties View*
3. On the **Advanced** tab, change the name of the integration system by adding a prefix of *WICT-WS-<your initials>-* to the existing name.
4. 

5. Select the Log Step in the **LogHello** Asynchronous Mediation and right click to select *Show Properties View*.
6. Change the log text literal from *Hello* to *Hello LOG*



   a. What are we logging in this step?

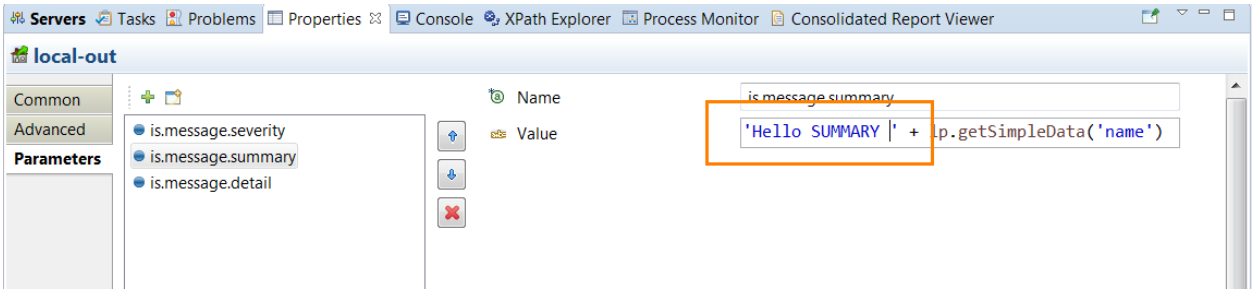
      _____

   b. Where do you expect to see that log?

      _____

7. Select the **PutIntegrationMessage** component and right click to select *Show Properties View*

8. On the Parameters tab, select *is.message.summary* then modify value to *Hello SUMMARY*



9. Select the **SayHelloToWorkday** Put Integration Message component and view the properties.

   a. What are the parameters?

   _____

   b. Where do you expect to see the output of this component?

   _____

10. Save the changes to the project and **Deploy to Workday**

11. Expand the *HelloCloudCollection* in your current Connection in cloud explorer until you see **StartHere**. Right click and launch the integration system.

12. In the Process Monitor, double click on the row representing this launch of the integration to see the log

Additional (Optional)

13. In the tenant, search for the most recent launch of your integration system *WICT-WS-<your initials>-integration-system-Hello using either the search function or the Integration Events Report*

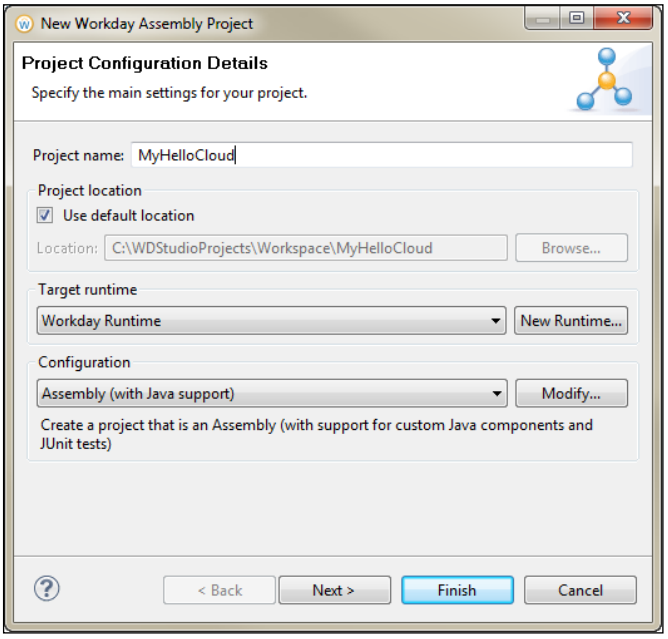
14. Once *Completed*, navigate to the **Messages** tab in the background process to view the messages.

15. Open the Log file and locate the change you made to the Log step (step 12 above) .

CREATING ASSEMBLY PROJECTS

The Assembly Project wizard helps users quickly generate an assembly project. The generated project will contain a basic *assembly.xml* file, which will be your starting point. Using the Assembly Editor you can graphically build up your assembly by adding and linking transports, mediations, and steps together to define your integration.

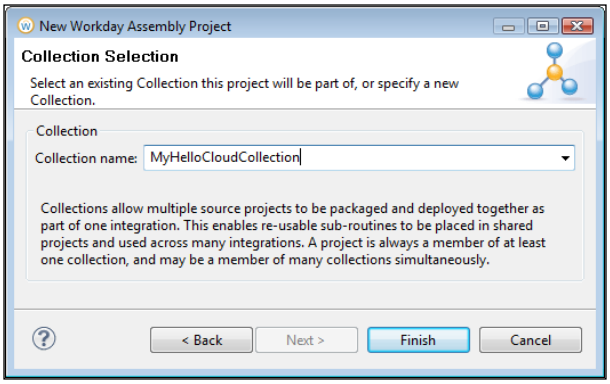
From the **File** menu, select **New>Workday Assembly Project**. In the **Project name** text box, enter a name for the project, for example, *MyHelloCloud*.



In the **Target Runtime** list, allow Workday Runtime to default to use for the project.

In the **Configurations** list, select **Assembly (with Java support)**. This creates a project that also contains the Java facet.

Click **Next** to configure the Collections. Enter a name that is different from your project name. For example, add the word Collection to the end.



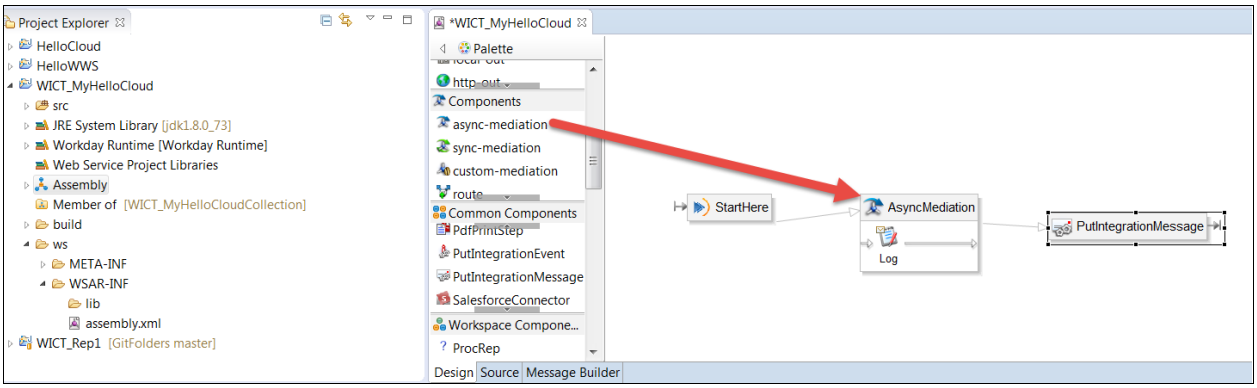
Click **Finish** to generate the project. The wizard will generate a project containing a number of assembly artifacts:

- assembly.xml
- assembly-diagram.xml

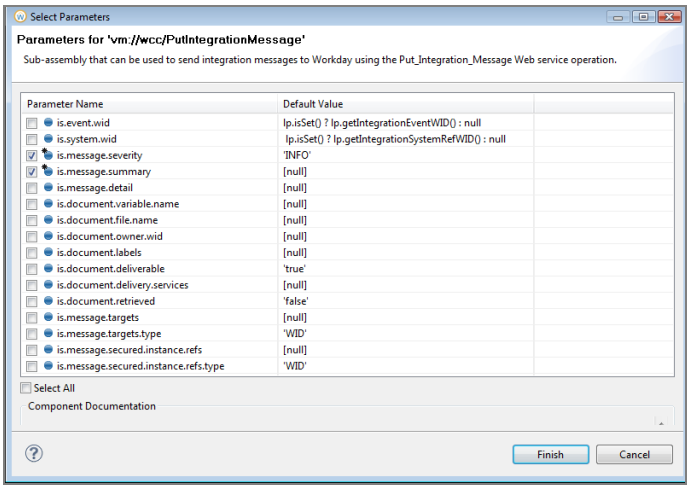
Double-click on the Assembly icon.xml file to open it in the Assembly Editor.

ADDING COMPONENTS TO THE ASSEMBLY

Add components by dragging them from the tools palette over onto the assembly diagram:



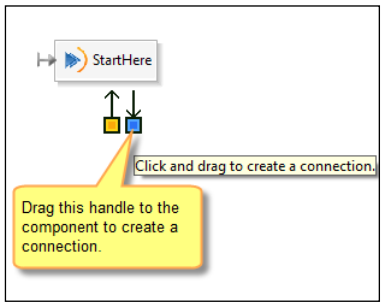
If a component has parameters, like the PutIntegrationMessage, when you drop it on to the diagram a wizard will display a list of those parameters.

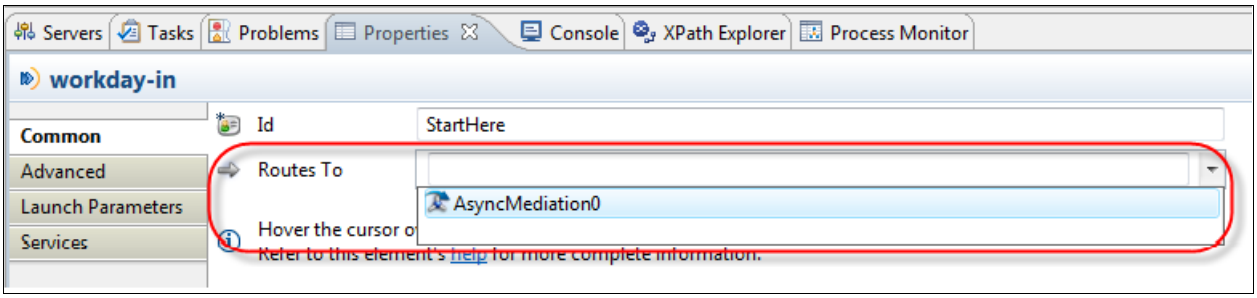


After you add a component to the assembly diagram, you can configure it by selecting the component and if the **Properties** view is not displayed, right-click the component, then select **Show Properties View**. The **Properties** view displays a list of all properties required to configure that component.

You may use the help links displayed at the bottom of the **Properties** view to find out more about each component and how to configure it.

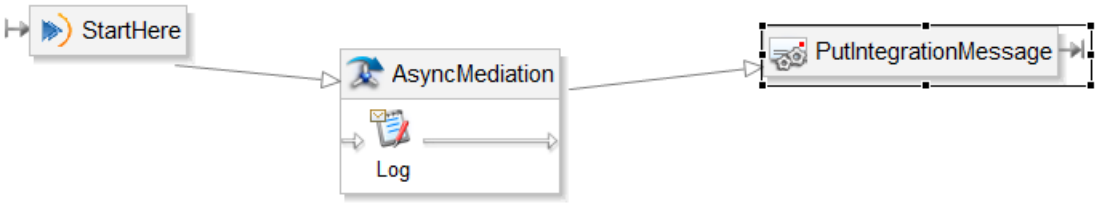
Finally, the components must be "wired" together to define the message path. You may use either the boxes to drag and drop or the properties to wire two components together.





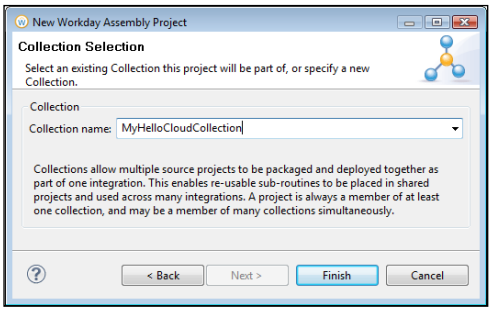
ACTIVITY 3 –"MYHELLOCLOUD" ASSEMBLY

In this activity you will build your own assembly from scratch. It will be similar to the HelloCloud sample and used in future activities. Your final assembly will look like this:



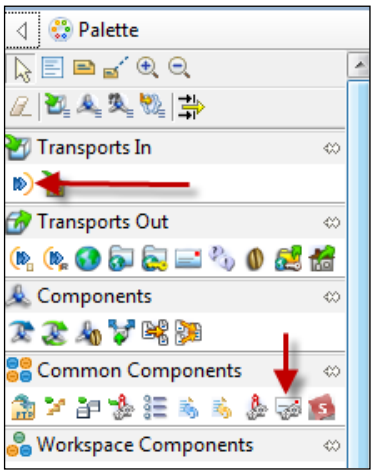
1. Create a new Assembly Project in Studio called **MyHelloCloud**

2. Click **Next** and change the collection name to **MyHelloCloudCollection**



3. Click **Finish**

On the blank Assembly Diagram, Drag and Drop from the Palette:



4. **Workday In** transport

5. **PutIntegrationMessage** (PIM) - when the "*Select Parameter*" dialog is displayed, click **Finish**.

6. Wire the two components together (from the **Workday In** to the **PIM**)

7. Access the Properties tab of the PIM and select Parameters. For the *is.message.summary,* enter a value such as 'My Hello Cloud' Don't forget single quotes.

8. Save the project, deploy and launch.

9. Do you see a log in the log message when you access the event on the **Process Monitor** tab, *View Log File*. _____

10. Add a Log Step to the Assembly Diagram.

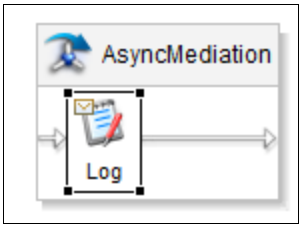
    a. Can you just drag and drop a step onto the Assembly Diagram? _____

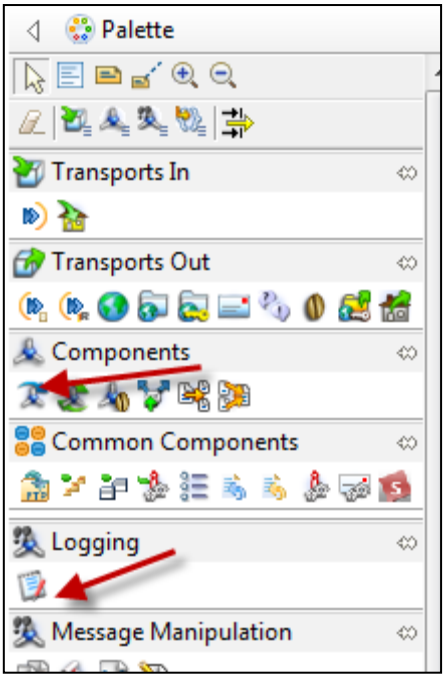
    b. What do you need to add to the diagram before you drag and drop the Logging Step? _____

11. Drag and drop an **AsyncMediation** onto the Assembly Diagram.

12. Drag and Drop a **Logging** step into the **AsyncMediation**.

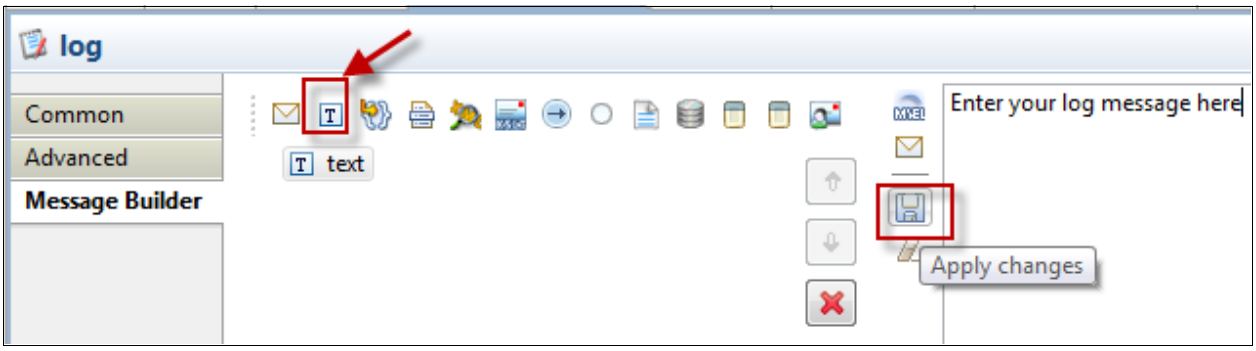
13. Select the **Log** step and access the properties to configure the *Message Builder*.





14. In the *Message Builder*, **remove the "message-content"** and add "**text**".

15. Enter a message in value and click the save button.



16. Re-wire the components as needed.

17. Save, deploy and launch to test.

MESSAGE FLOW THROUGH ASSEMBLIES

Messages originate at an in-transport component and follow the paths defined by connectors, which connect the in-transport to a variety of possible assembly components. Messages leave the assembly via an out-transport component.

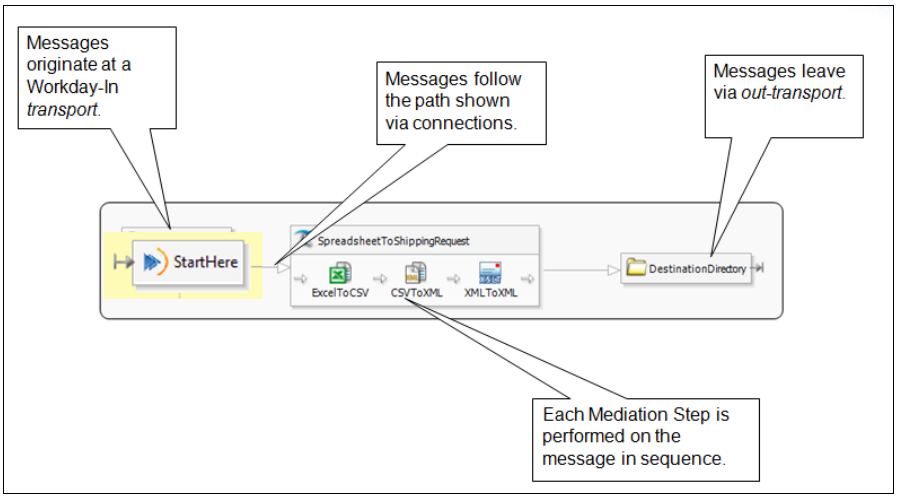
The behavior of a message based on whether transports are synchronous or asynchronous can be summarized as follows:

- In-transports: synchronous ones expect a response. Asynchronous ones do not and will discard any response message returned to them.
- Out-transports: synchronous ones will return a response. Asynchronous ones will not.
- Mediation Components: synchronous mediations have steps for the response message. Asynchronous mediations are directly equivalent to synchronous ones that don't have any response steps

ASSEMBLY RUNTIME

The assembly runtime has a simple execution model. It consists of transports and components that are linked together to form processing chains. Processing starts at a **workday-in** transport, which receives a document called a launch integration event. This document contains some data about the event, for example, the integration event ID that is used to track the integration. It also includes any parameters that were specified at launch time.

The diagram below shows a Simple Flow of a message through an assembly:

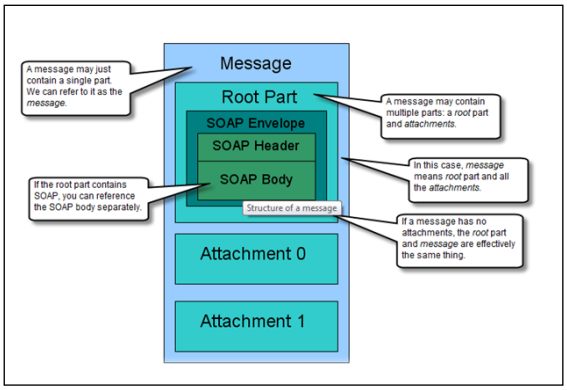


The assembly runtime supports several different types of components. Integration projects typically consist of a mix of re-usable services, routing components, transport configurations, and mediation steps, that is, steps that perform message validation, transformation, manipulation, logging, and so on.

The style of processing is primarily message-oriented: take in a message, process it, and route it on. Assemblies provide a way of describing such message-oriented processing chains. Messages arrive over a variety of transports and go through a series of mediation steps before being routed on over another transport or delivered locally to a Web service.

MEDIATION MESSAGE

An assembly processes data in the form of messages that contain the information payload. The entry point for a message within an assembly is an in-transport. Once a message has arrived into an assembly, it follows the path defined by connectors between transports and message-consuming components such as mediation steps and routes. Finally, messages leave an assembly through an out-transport.



A message can consist of one or more parts. For example, a message can be a single document, in which case it consists of one part. It is possible to manipulate the contents of a message by accessing the various parts using Studio assembly components.

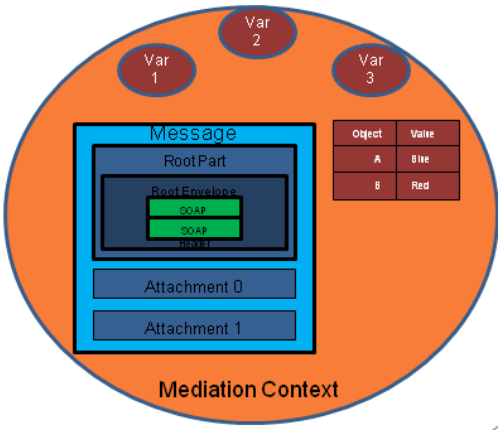
MEDIATION CONTEXT

The assembly framework uses instances of the **MediationMessageContext** class to pass all information between assembly components. When an in-transport creates a message, the assembly framework passes the message inside an instance of a **MediationMessageContext**. The **MediationMessageContext** class enables components in a processing chain to share state information.

This class also has a number of sub-components, which are defined as accessors.

These include:

- **MediationMessage:** This object represents a message. Messages can be single or multi-parted (MIME). Parts can contain either textual or binary data. A message is ultimately exchanged with an external system using an out-transport component, for example, http-out, sftp-out, and ftps-out.
- **Variables:** These provide temporary holders for message data. A common usage pattern is to copy the contents of a message to a variable for later processing in the assembly chain.
- **Properties:** This provides a Map where developers can place temporary Java Objects.



MVEL

MVEL is a powerful expression language for Java-based applications. It provides a plethora of features and is suited for everything from the smallest property binding and extraction, to full blown scripts.

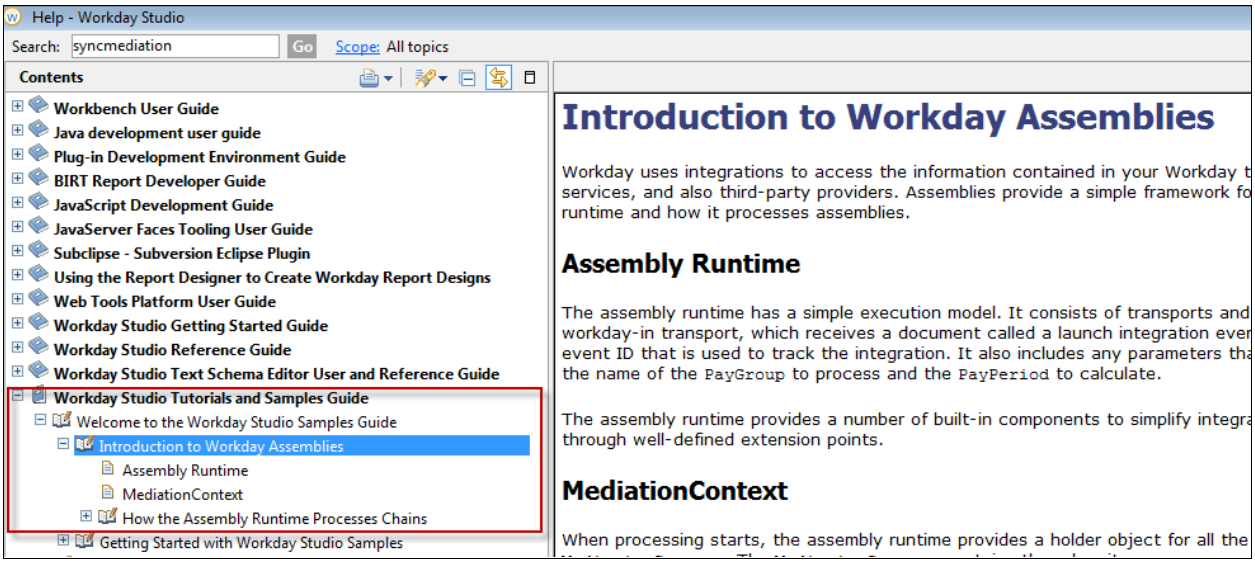
The assembly runtime uses MVEL pervasively to enable dynamic configuration of components. Below are some examples of the MVEL variables that Workday Studio and the assembly runtime supports:

- **parts**: provides access to parts of the MediationMessage, for example, parts[0].text gets the text of the root part of a message.
- **props**: provides access to the MediationContext properties.
- **vars**: provides access to variables in the MediationContext, for example, vars[‘fred’].text gets the text of the variable called 'fred'.
- **lp:** provides access to launch parameters, for example, lp.getParameterData(‘PayGroup’) gets the value for the parameter 'PayGroup' that was passed as part of the launch integration event.

SUGGESTED READING

It is recommended that you access the Help Contents of Workday Studio and review the Introduction to Workday Assemblies .

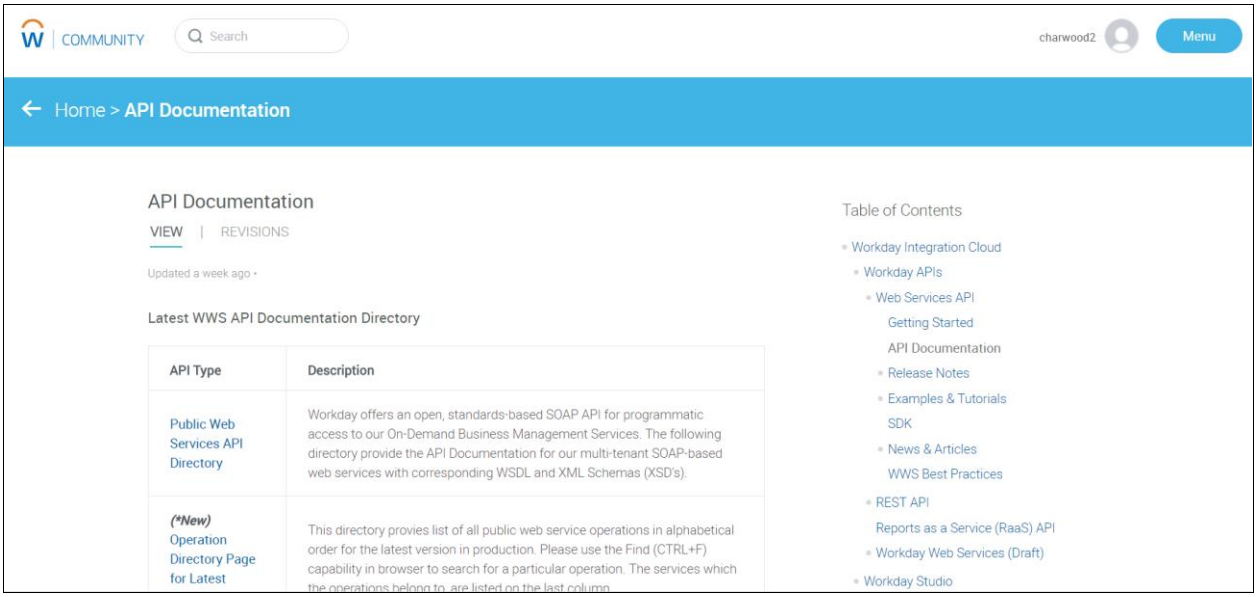
1. In Studio, navigate to **Help >> Help Contents**
2. Expand **Workday Studio Tutorials and Samples Guide**
3. Expand **Welcome to the Workday Studio Samples Guide**
4. Expand **Introduction to Workday Assemblies**
5. Review



WORKDAY PUBLIC WEB SERVICES

Workday offers an open, standards-based SOAP API for programmatic access to our On-Demand Business Management Services. The following directory provide the API Documentation for our multi-tenant SOAP-based web services with corresponding WSDL and XML Schemas (XSD's).

These services are available to the Workday Integration tools. Workday Studio includes a Web Services tester that can be used to connect to a specific web service operation to test the request and response messages. In order to invoke the services, security must be properly defined in the tenant.

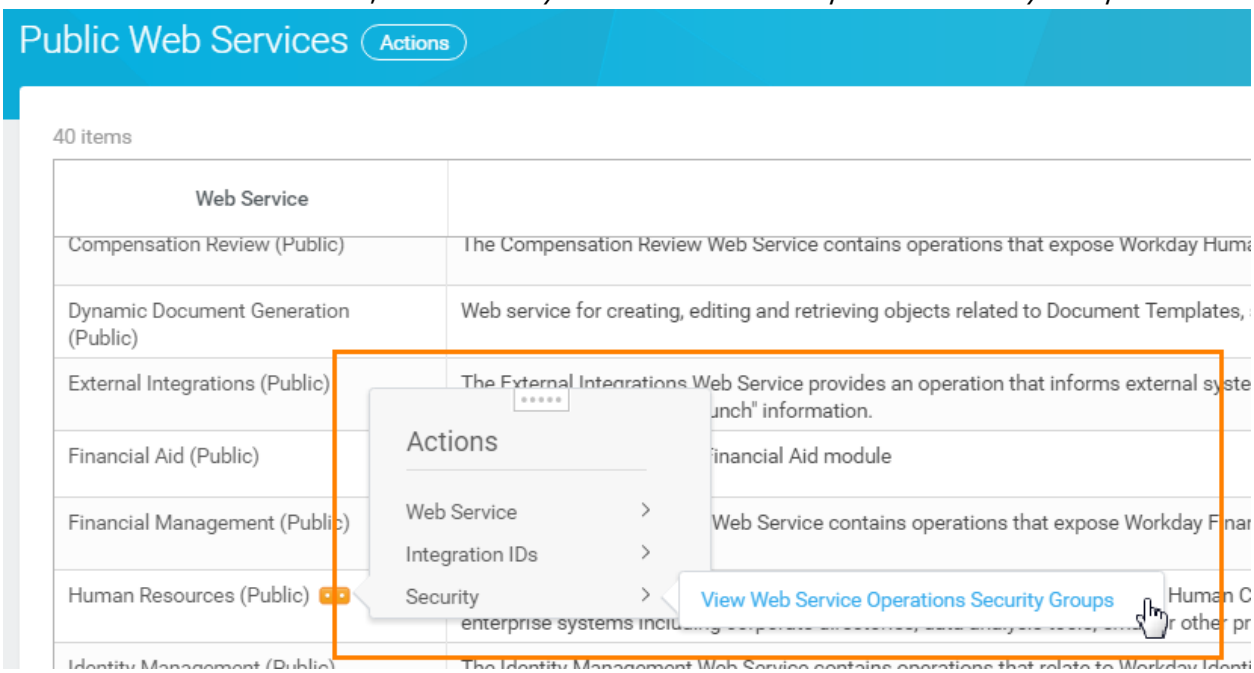


WEB SERVICES SECURITY

A Workday Web Service Operation is based on a business process or task. Each of these operations require a Security Group in order to initiate the business process or task.

To find out which Workday Web Service contains a particular operation and to verify which Security Groups have permission to initiate the operation, access the Public Web Services report in Workday. As a

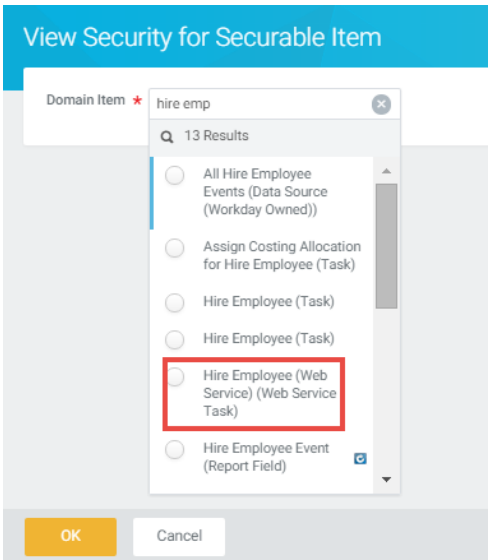
related action on a web service, select *Security > View Web Service Operations Security Groups.*



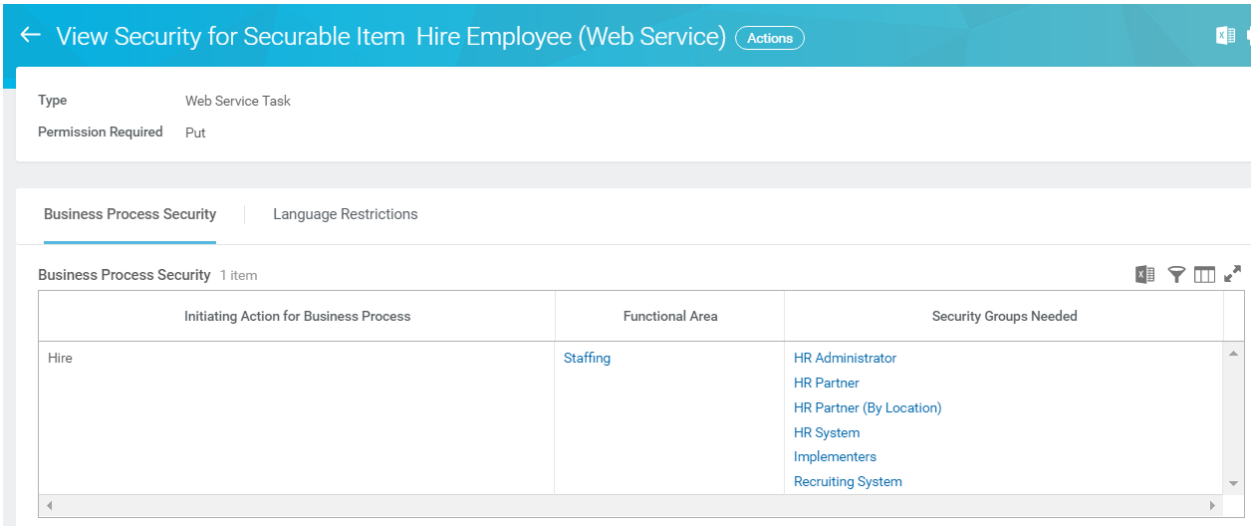
Workday does not assign default security groups to protect system integrity and to encourage caution when granting access to web service operations.

GRANTING SECURITY

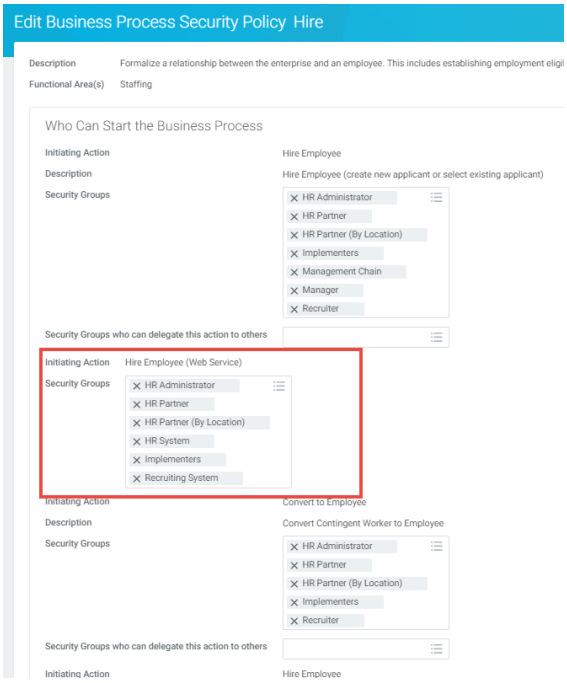
You can access the **View Security for Securable Item** report and look up a Domain Item to view which security groups have access.



For example, when the *Hire Employee(Web Service)(Web Service Task)* is selected, the security groups required to initiate the associated business process can be viewed.



By using the related action icon of the Hire Employee business process, you can edit the Business Process Policy and add/remove security groups.

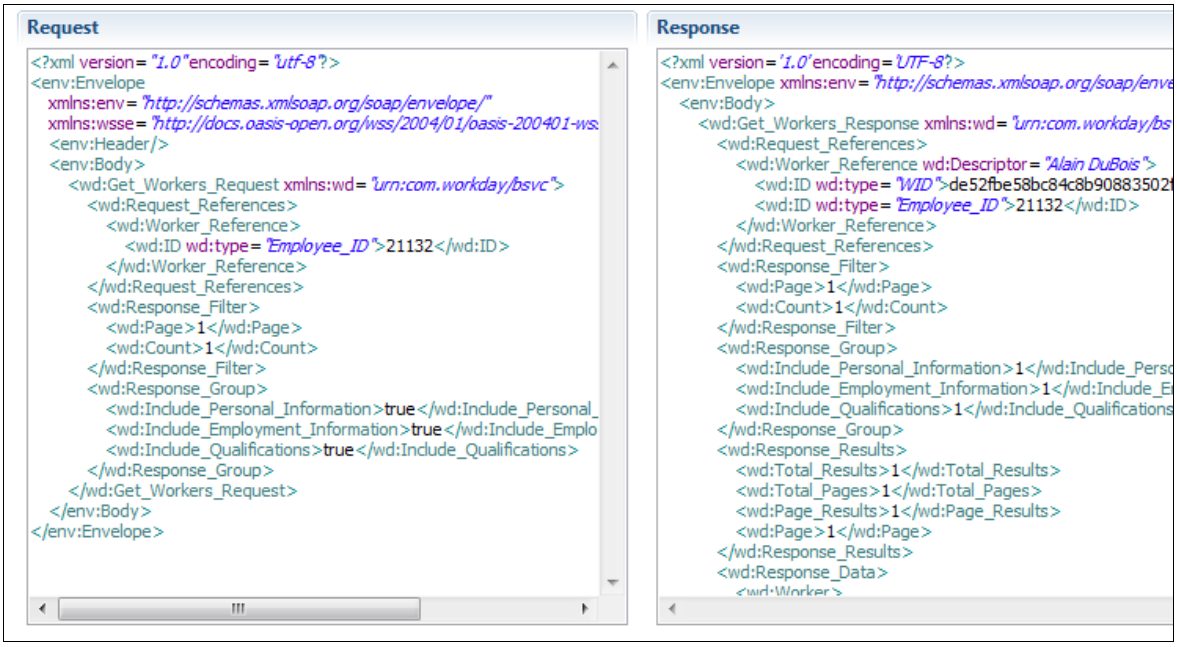


If a change is made to the Business Process Security Policy, the **Activate Current Security Policy Changes** task must be executed before the new changes will take effect.

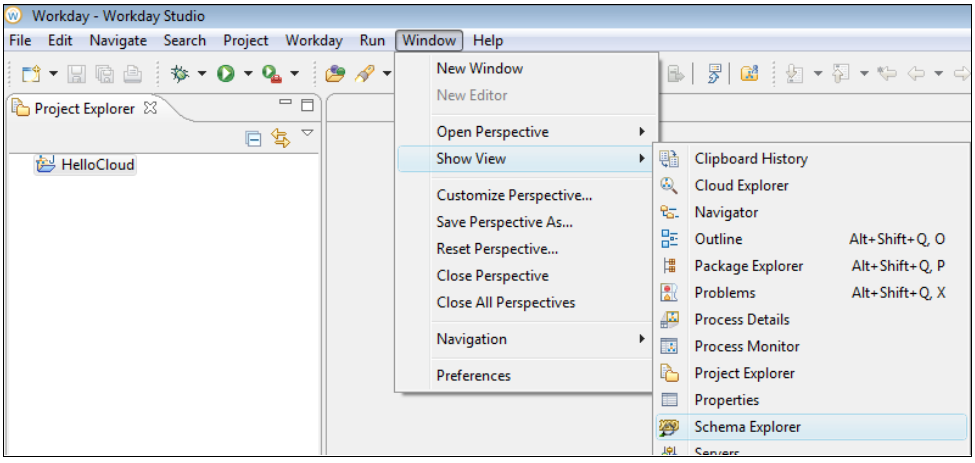
Note: Rather than change the security policy, you can also assign Security Roles and User-Based Security Groups to users.

WORKDAY STUDIO WEB SERVICE TESTER

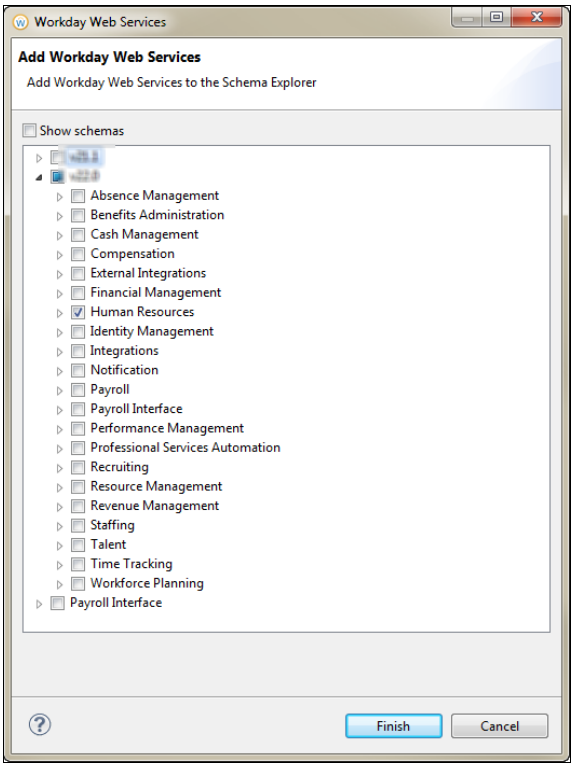
The **Schema Explorer** integrates with the Workday Web Services (WWS) API by providing direct access to the WWS WSDL and schema documents, allowing you to work with their operations, messages, element, and type definitions. The **Schema Explorer** view displays the structure of WSDL and schema files in a read-only, outline view. You cannot edit the contents of the file. The tester enables you to generate and send a Request message to an operation and view the response.



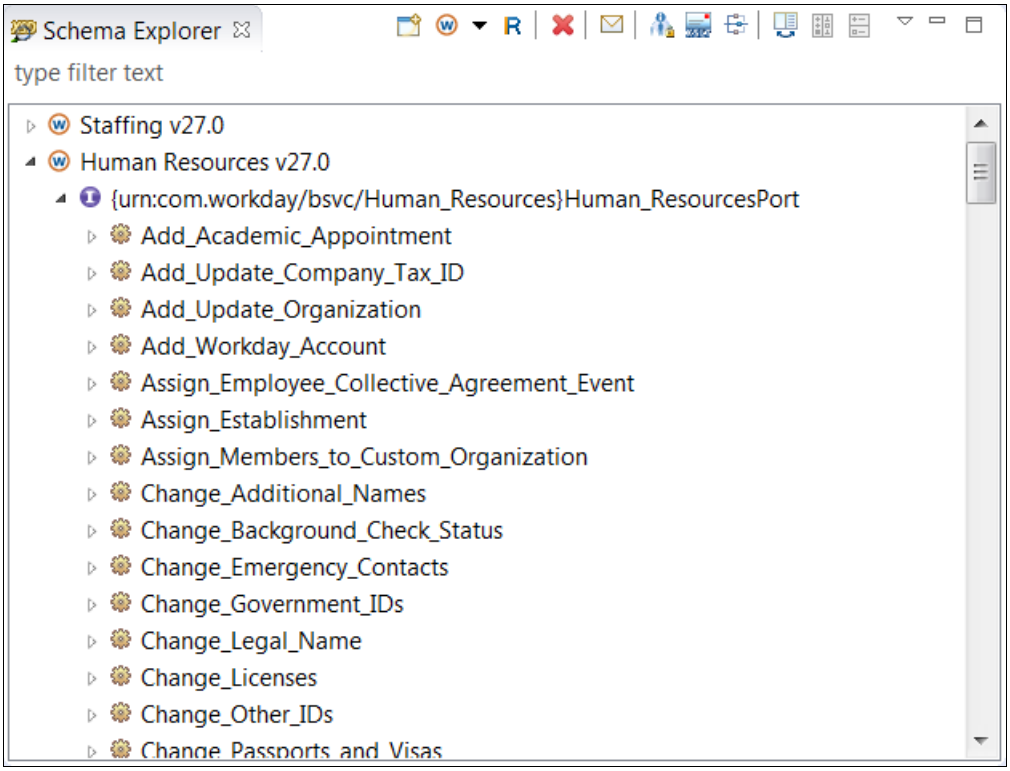
By default, in the **Workday Studio** perspective, the **Schema Explorer** is displayed along the top right-hand side of the screen, but as with all views in the Eclipse development environment, you can rearrange the views to create your preferred layout. If you do not see the Schema Explorer, you can open it by navigating to **Window >> Show View >> Schema Explorer**.



To view a list of web services, click on the "W" icon and select the service(s) you will to view in detail.

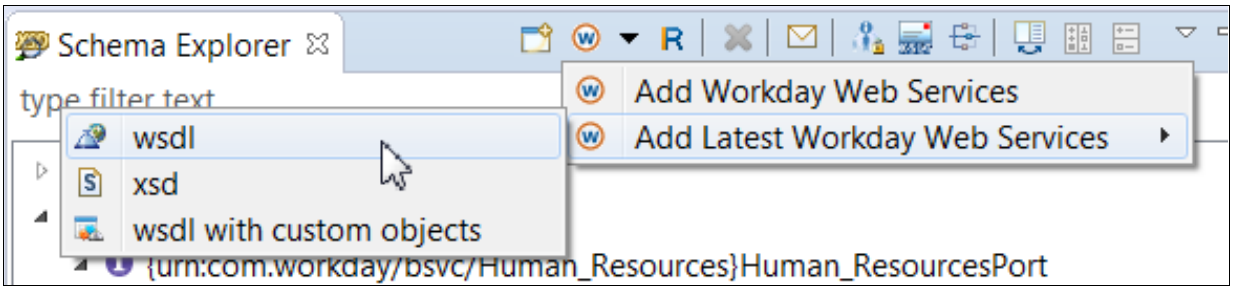


Once you click **Finish**, you can view a list of operations available to that service by expanding it.



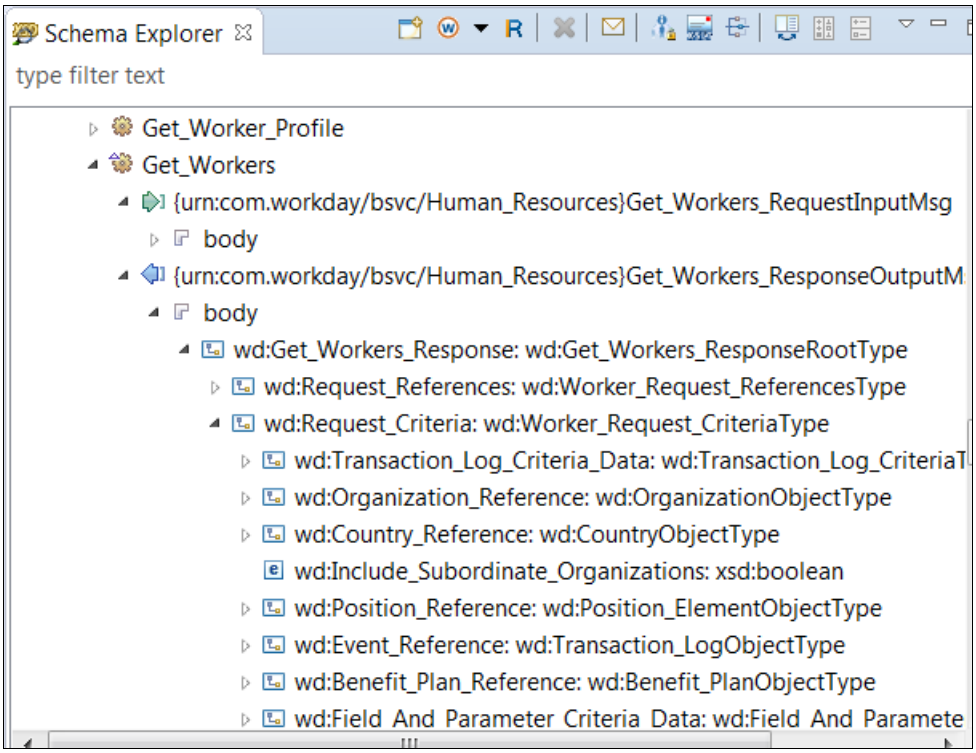
NOTE: Workday Studio patches do not always coincide with application patches. If the Workday Application patches to a newer version of Public Web Service API, Schema Explorer in Studio will not reflect that version until it is patched. You can still modify the request to call the latest version in the tenant.

You can access the latest version in the tenant by selecting **Add Latest Workday Web Services > wsdl**.

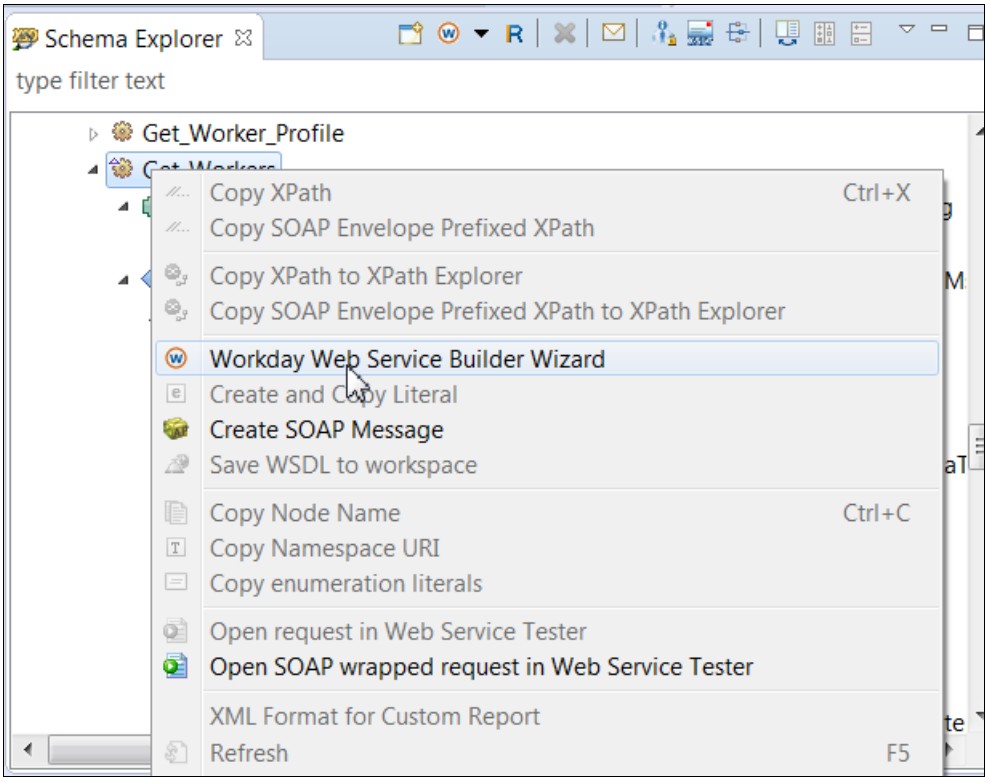


There are many ways to use the Schema Explorer to learn and test web services. Each operation has a request and response message and you can review those by expanding the operation.

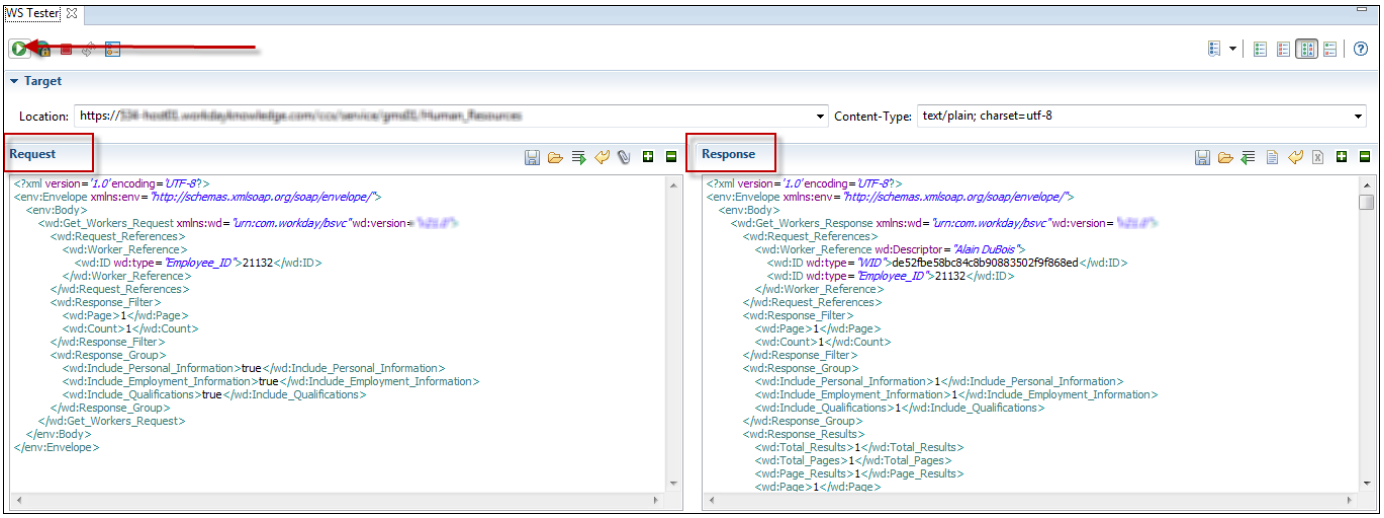
You can filter for keywords in the loaded list of services and operations. This is a very useful way to search for an operation.



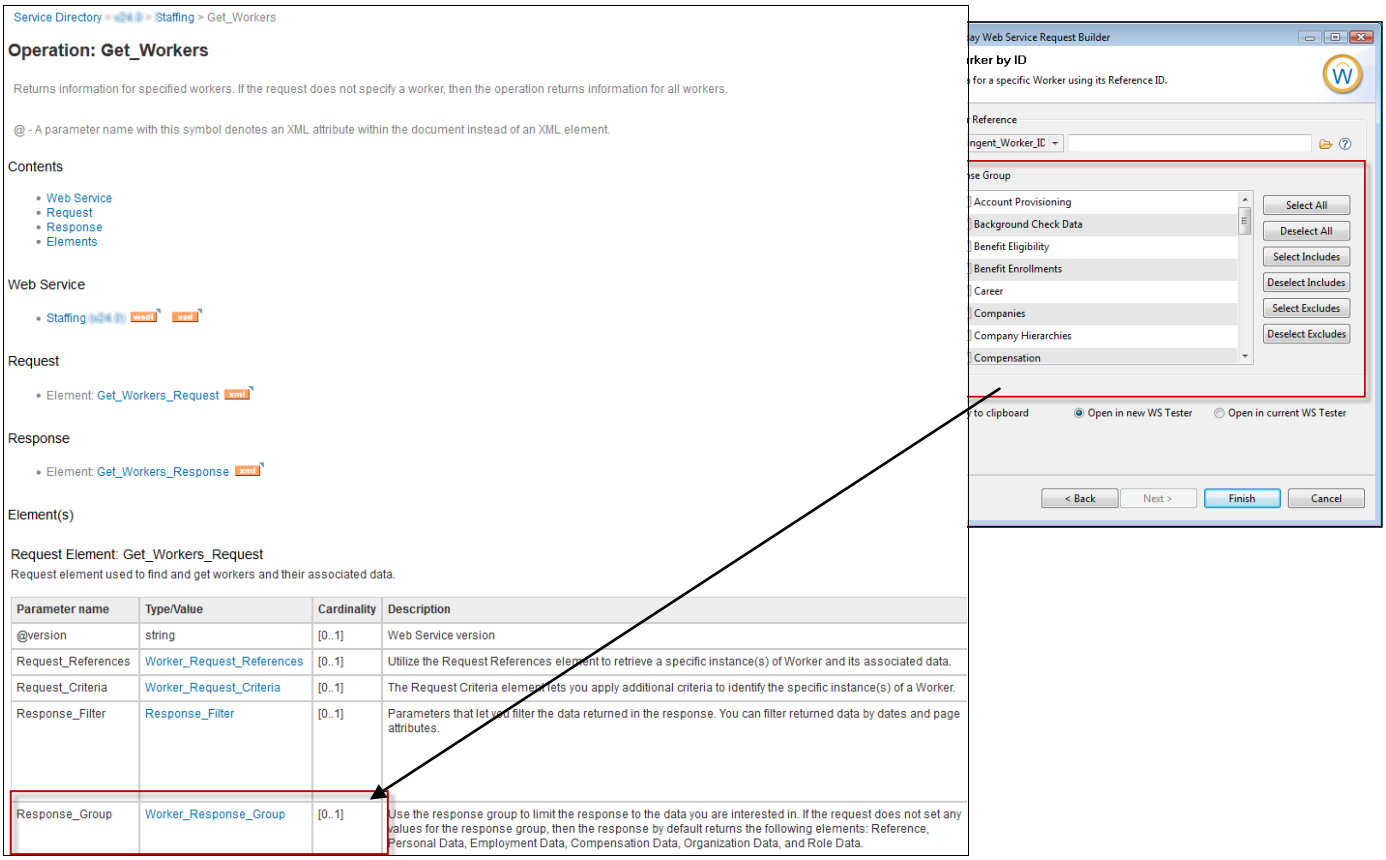
To generate a test message, right click on the name of the operation, for example, Get_Workers and select Workday Web Service Builder Wizard. This wizard will take you step by step to create the Request message to invoke the Get_Workers_Request.



Based on the specific operation selected, you will be presented with a series of questions. Once completed, the wizard will generate a request message and open a **WS Tester** window. You can use the toolbar to send the request to Workday and view the response message.



This tool presents opportunities to test and learn the Workday Web Services. For example, using the API documentation, you can learn the details about the **Response Group**.



Worker_Response_Group
part of: Get_Workers_Request, Get_Workers_Response
Use the response group to limit the response to the data you are interested in. If the request does not set any values for the response group, then the response by default returns the following elements: Reference, Personal Data, Employment Data, Compensation Data, Or

Parameter name	Type/Value	Cardinality	Description
Include_Reference	boolean	[0..1]	Indicates if the Reference element for the worker is included in the response.
Include_Personal_Information	boolean	[0..1]	Indicates if the Personal Data element is included in the response.
Include_Employment_Information	boolean	[0..1]	Indicates if the Employment Data element is included in the response.
Include_Compensation	boolean	[0..1]	Indicates if the Compensation Data element is included in the response.
Include_Organizations	boolean	[0..1]	Indicates if the Organization Data element is included in the response.
Exclude_Organization_Support_Role_Data	boolean	[0..1]	Excludes the supporting role information from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Location_Hierarchies	boolean	[0..1]	Excludes the location hierarchies from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Cost_Centers	boolean	[0..1]	Excludes the cost centers from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Cost_Center_Hierarchies	boolean	[0..1]	Excludes the cost center hierarchies from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Companies	boolean	[0..1]	Excludes the company organizations from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Company_Hierarchies	boolean	[0..1]	Excludes the company hierarchies from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Matrix_Organizations	boolean	[0..1]	Excludes the matrix organizations from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Pay_Groups	boolean	[0..1]	Excludes the pay groups from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Regions	boolean	[0..1]	Excludes the regions from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Region_Hierarchies	boolean	[0..1]	Excludes the region hierarchies from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Supervisory_Organizations	boolean	[0..1]	Excludes the supervisory organizations from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Teams	boolean	[0..1]	Excludes the teams from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Exclude_Custom_Organizations	boolean	[0..1]	Excludes the custom organizations from the Organization Data element response. This can only be selected when the Include Organization Data boolean is also selected.
Include_Roles	boolean	[0..1]	Indicates if the Role Data element is included in the response.
Include_Management_Chain_Data	boolean	[0..1]	Indicates if the Management Chain Data element is included in the response.
Include_Benefit_Enrollments	boolean	[0..1]	Indicates if the Benefit Election Data element is included in the response.

Once the WS Tester window is displayed, you cannot "re-open" the wizard. However, you can right click on the operation and use the wizard to generate another request that will open in a new WS Tester window.

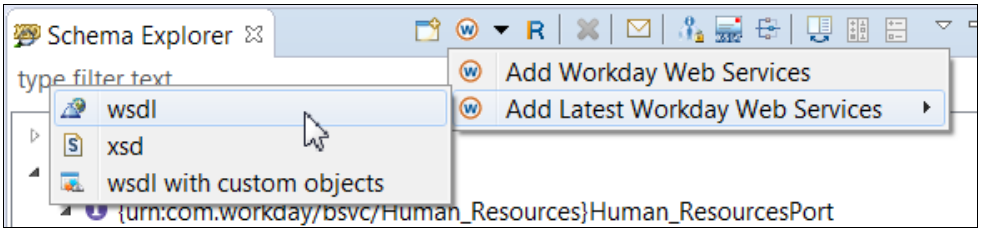
ACTIVITY 4 – WEB SERVICES TESTER

Use the Web Service Tester to create a SOAP request and send to Endpoint (Workday) and return a worker response (from Workday) using the Human Resources Service Operation, Get_Workers.

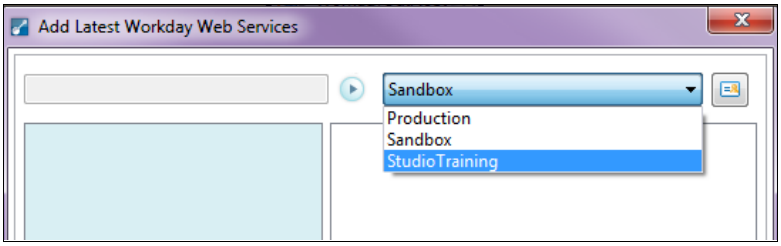
TASK 1

1. If you are unable to see Schema Explorer, navigate to *Window >> Show View >> Schema Explorer*

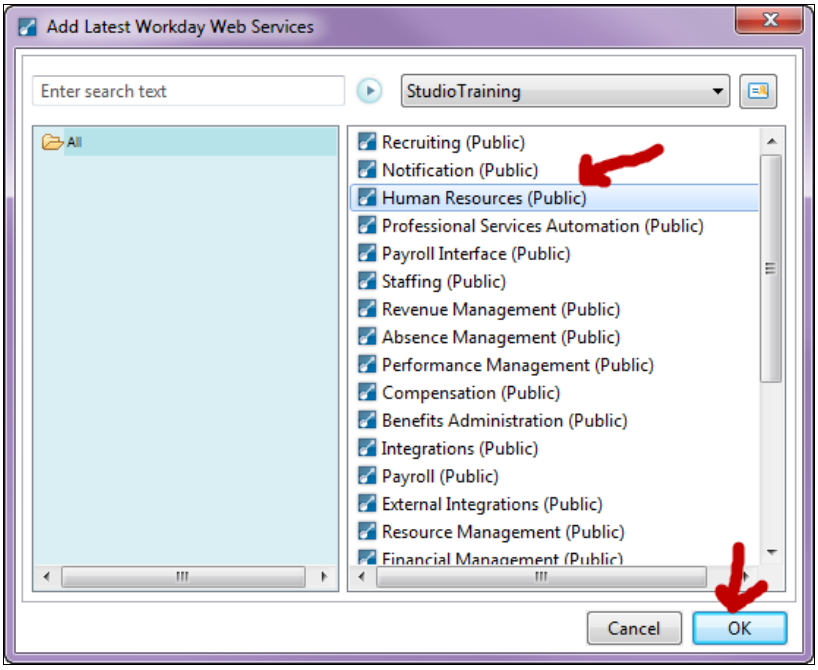
2. In the Schema Explorer, click on the arrow icon, select *Add Latest Workday Web Services > WSDL*.



3. Select your correct environment for this training class.

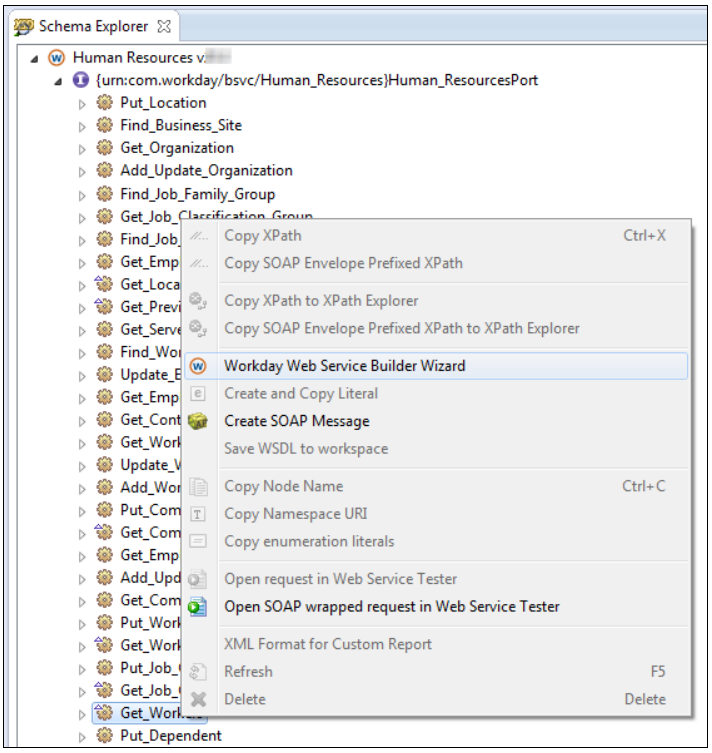


4. Select Human Resources from the list presented and then click OK.

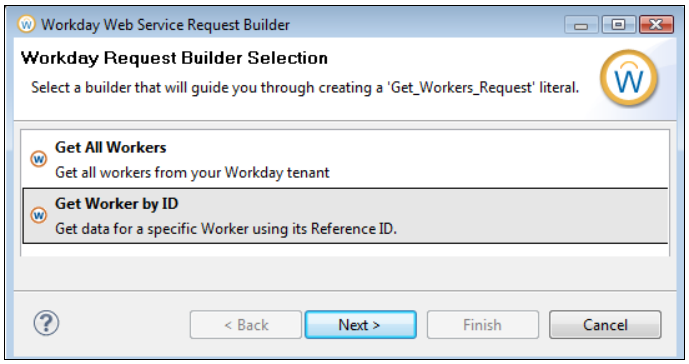


5. In the Schema Explorer, expand *Human Resources v2x.x >> {urn:com.workday/bsvc/Human Resources}Human_ResourcesPort*

6. Scroll through the list of operations and right click on *Get_Workers* and select *Workday Web Service Builder Wizard*

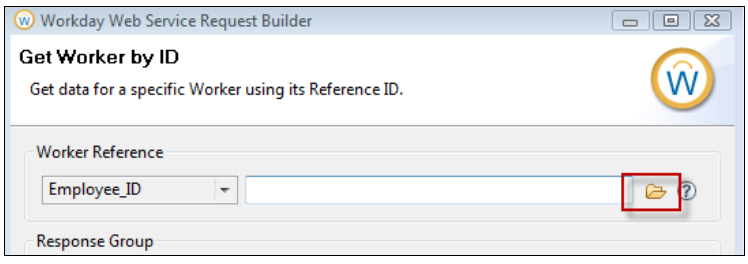


7. Select **Get Worker By ID** and click **Next**



6. Change the *Worker Reference* dropdown list value to **Employee_ID**

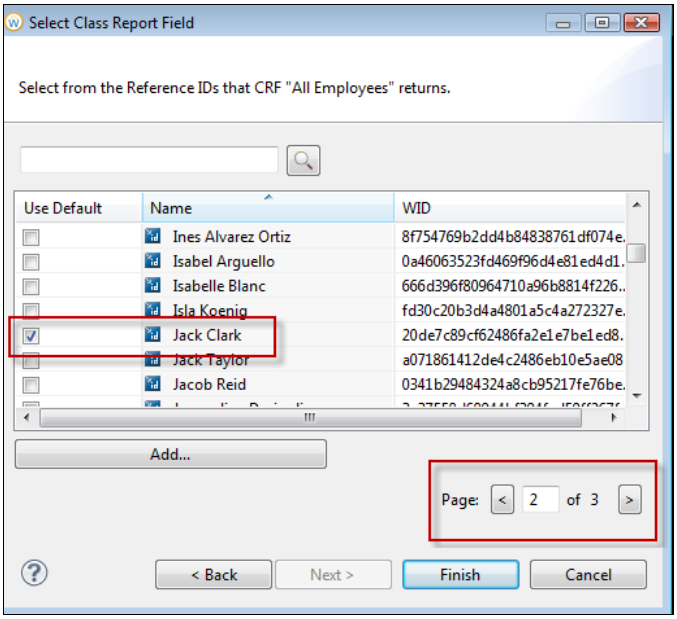
7. Click on the **folder** icon to browse for an Employee ID



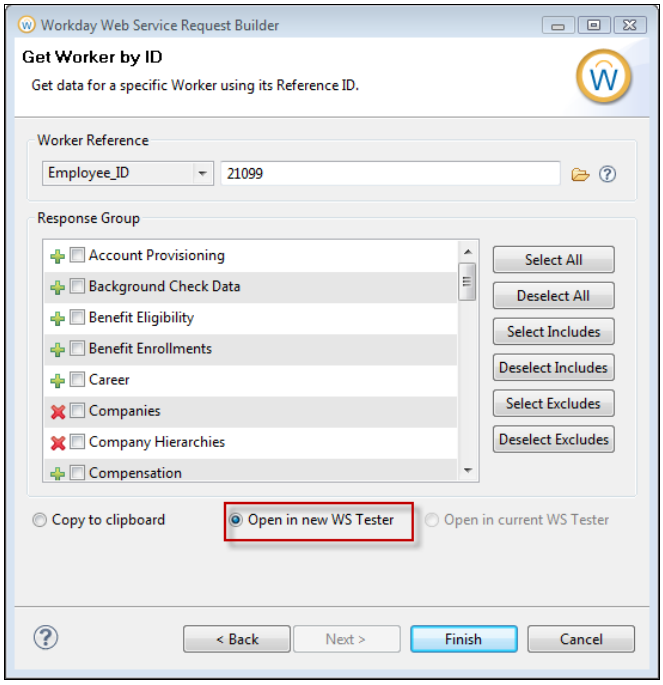
8. In the Select Class Report Field wizard, select **Next**

9. Select the class report field *All Employees* from the list presented and click **Next**

10. Go to Page 2 of the list of employees and select *Jack Clark* and click **Finish**

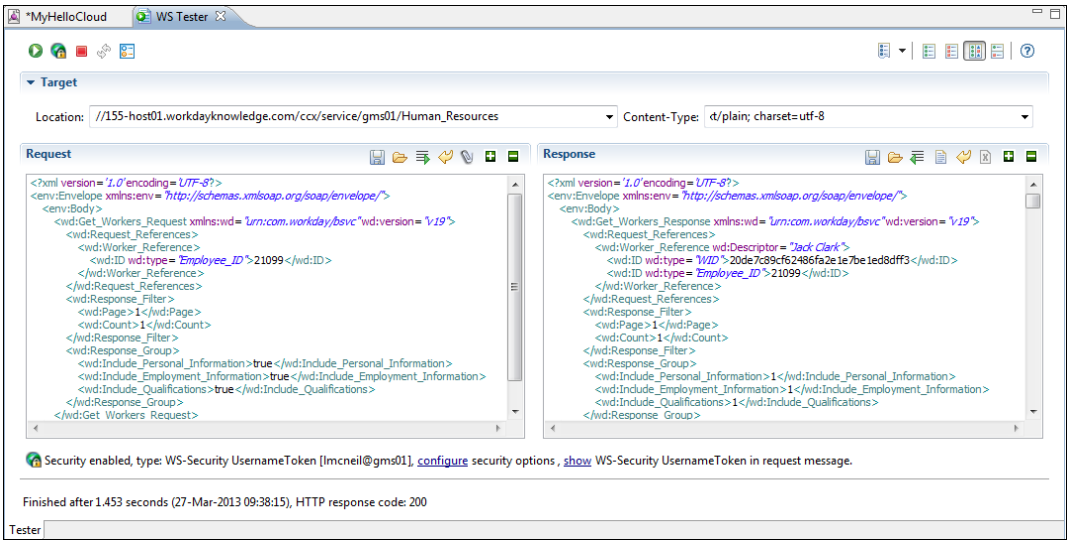


11. Verify the radio button *Open in new WS Tester* is selected and click **Finish**



12. The *WS Tester* window will open. Review the Request message and click **Send**.

NOTE: Please verify the **wd:version**. If it is not reflective of the most current WWS API version in the tenant, you may want to change it.



13. Review the Response

ADDITIONAL (OPTIONAL)

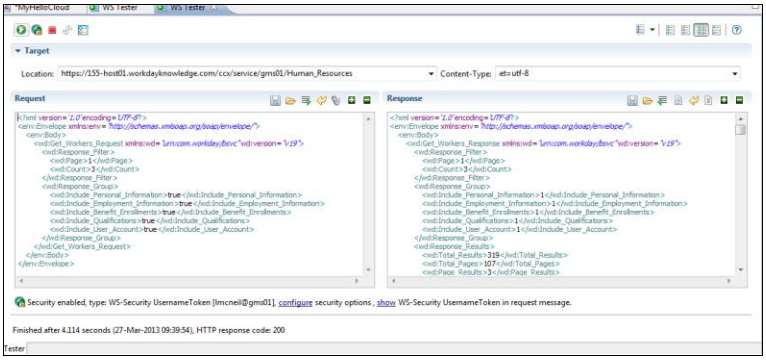
14. Create a new message using the *Get_Workers* operation *Workday Web Service Builder Wizard* (You cannot access the wizard to modify the message in the current *WS Tester* window)

15. Select Get All Workers in the first screen of the wizard

16. In addition to the existing Includes in the Response Group, select *Benefit Enrollments* and *User Account*

17. Return **1** page of **3** Workers

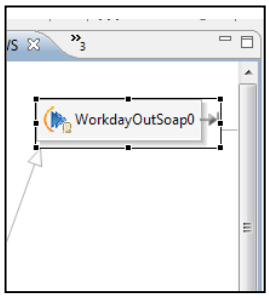
18. Generate the Request and review the new request and the response once it is received

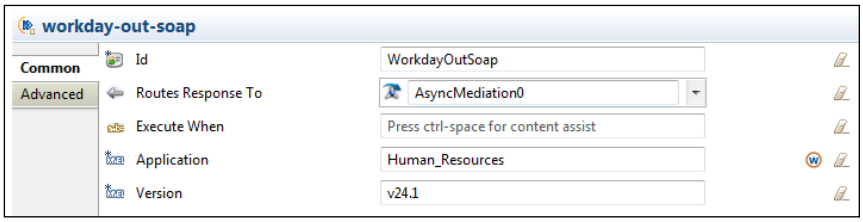


CALLING A WORKDAY WEB SERVICE OPERATION FROM THE ASSEMBLY

WORKDAY-OUT-SOAP TRANSPORT

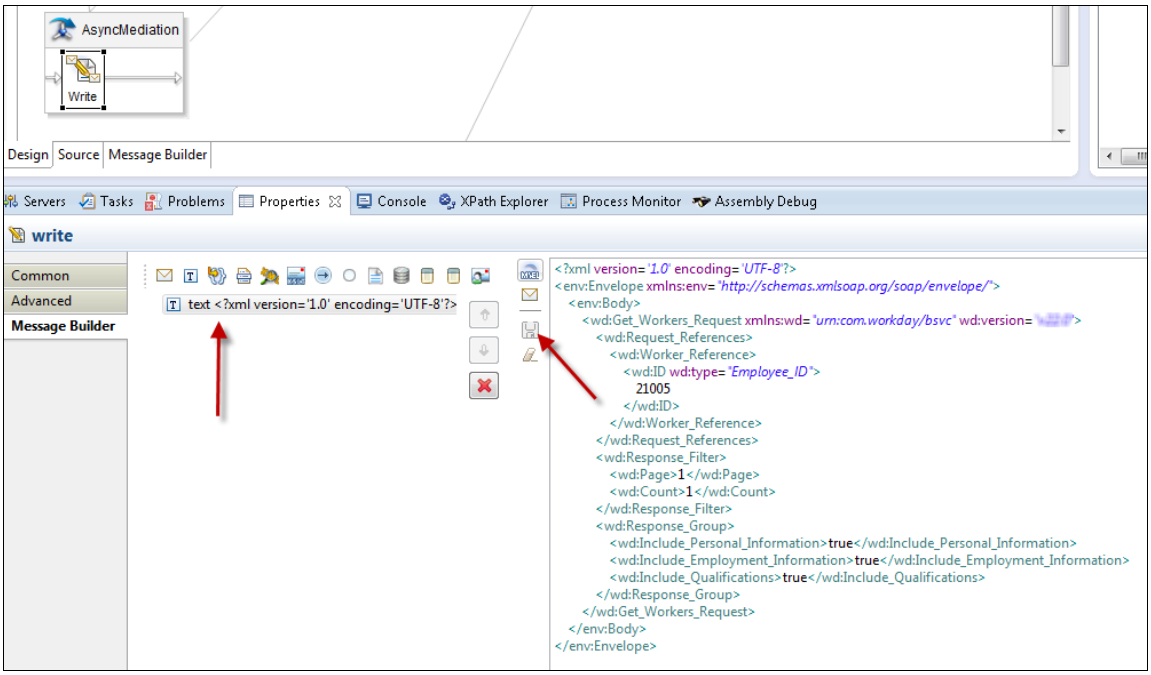
A synchronous, request-response transport that sends Workday application request messages to an HTTP URL. All out-transport components used in an assembly require an endpoint URL to be specified within their endpoint attribute. This defines the target destination for the message leaving the transport. However, the Workday-Out-Soap is configured by selecting an Application and Version. The URL is built based on your connection settings in Studio. You must create the Request Message that is sent to this application using a Write step.



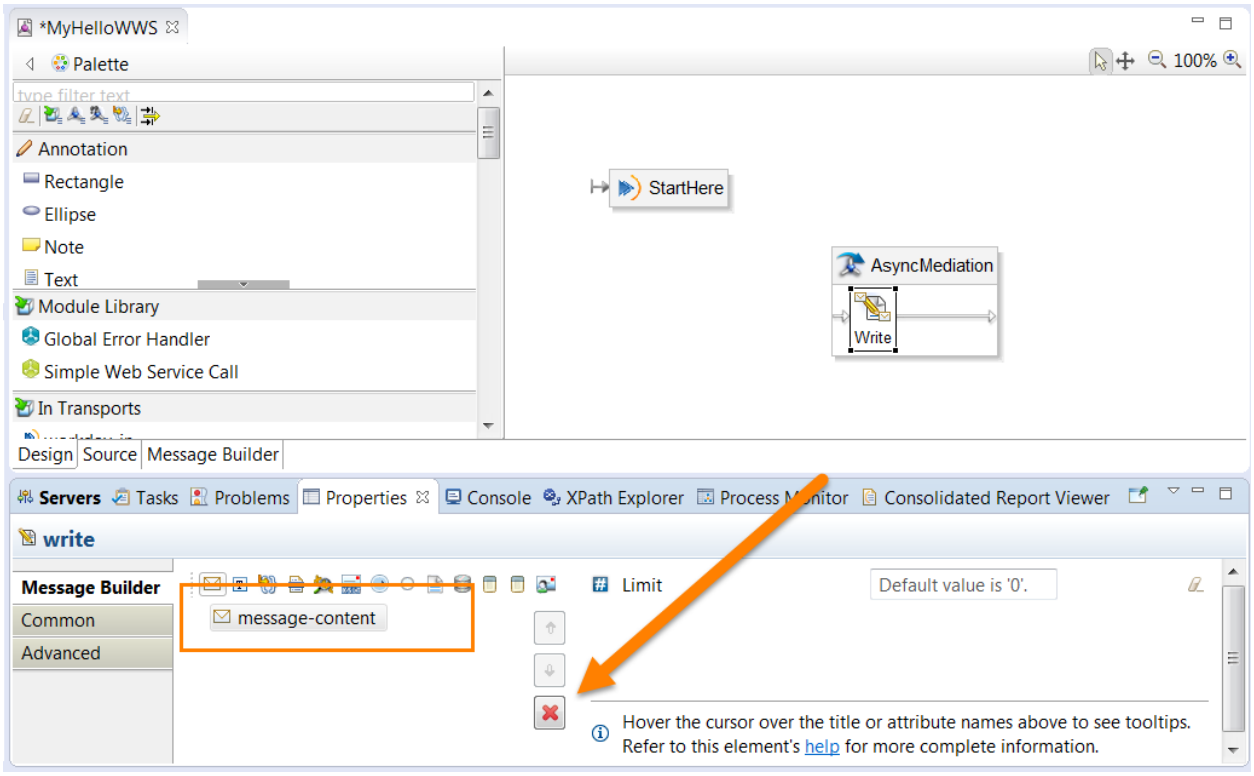


WRITE STEP

Like the Logging step, the Write step is placed into a Mediation and allows you to construct custom data and set it as the content of the mediation message. In the properties of the Write step, you may use text to create a message using the Message Builder. Rather than typing the text, you can use the Web Server Tester in the Schema Explorer to generate the Request XML and copy/paste it into the Write step properties (see Schema Explorer, Activity 5).

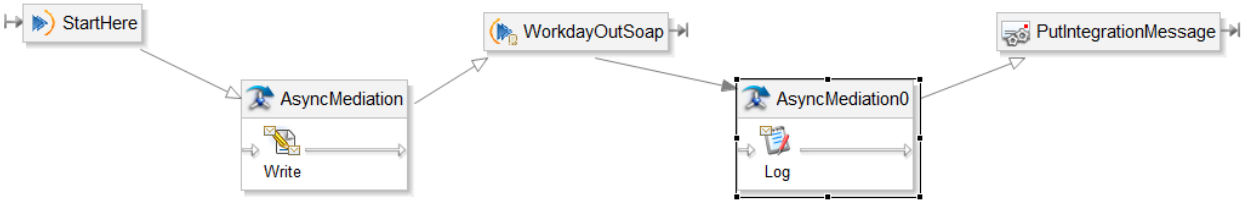


When a Write step is added to the assembly, the Message Builder by default contains the message-content element which will write the current contents of the mediation in addition to whatever text you may include. To ensure the content is not concatenated, make sure to remove the message-content element and replace with text.



ACTIVITY 5 –"MYHELLOWWS" ASSEMBLY

In this activity you will build your own assembly from scratch. It will be similar to the HelloWWS sample but will use a WorkdayOutSoap transport. You will use a write step to create the Request XML Rather than write from scratch, generate the Request XML using the SOAP Tester and paste into the Write step. Your final Assembly will look something like this:



1. Create a new Assembly Project in Studio called **MyHelloWWS**
2. Click **Next** and change the collection name to **MyHelloWWSCollection**
3. Click **Finish**

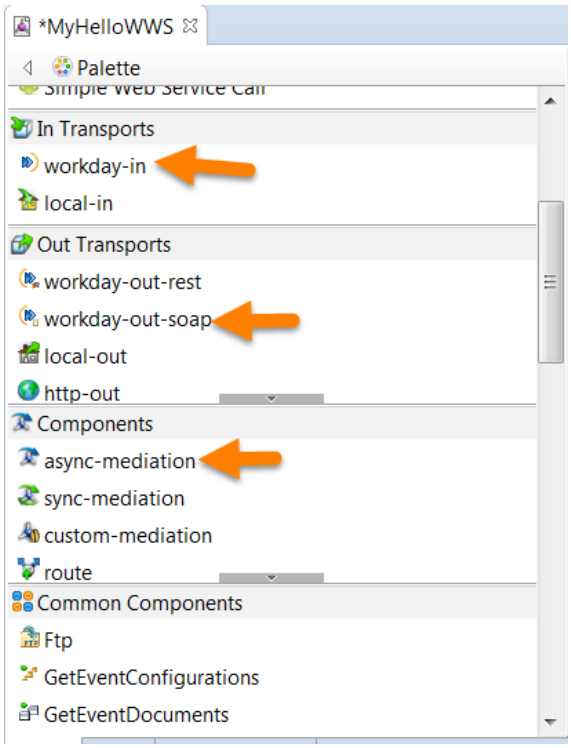
On the blank Assembly Diagram, Drag and Drop from the Palette:

a. **Workday In** transport

b. **AsyncMediation** with a **Write** Step

c. **WorkdayOutSoap**

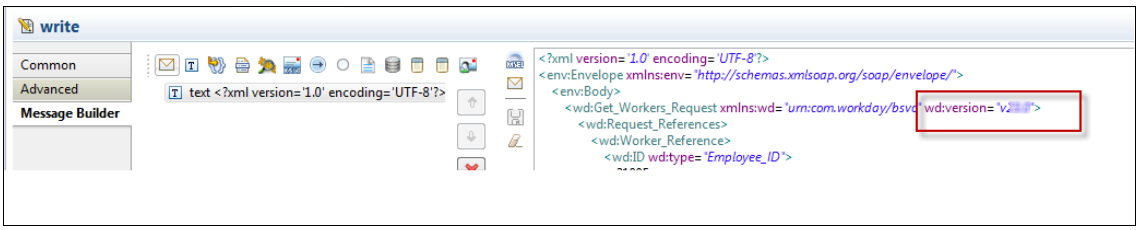
d. **PutIntegrationMessage** (PIM) - when the "*Select Parameter*" dialog is displayed, click **Finish**.



4. Wire the components together
5. Access the properties of the Write Step

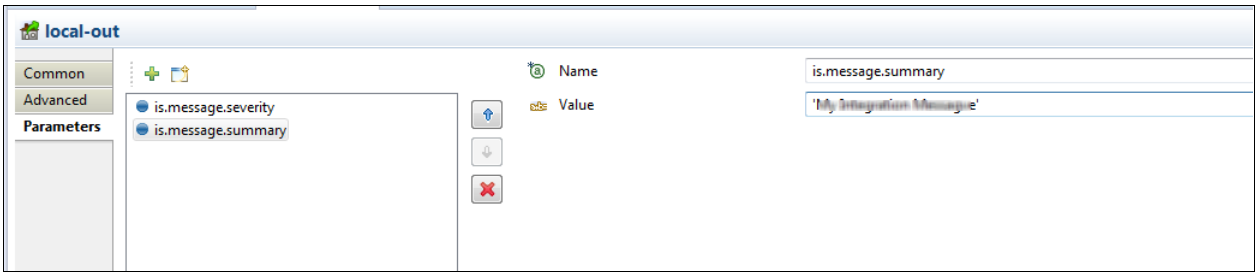
a. Use the Schema Explorer>Web Service Tester to generate a Request Message to the Get Workers Web Service Operation. Request one worker and generate the Request XML then copy it into a Message Builder Text value.

b. Be sure to use the current version of the web service in the SOAP Request (i.e. v29.0).



6. Access the properties of the Workday-Out-Soap and configure the **Human_Resources** Application and Version as **found in your request (i.e. v28.0)**

7. Access the Properties tab of the PIM and select Parameters to configure the *is.message.summary*.



8. Save the project, deploy and launch.

9. Do you see the response from Workday? _____

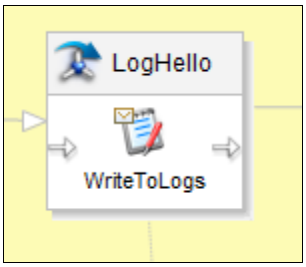
10. If not, what can you do (using what you have already learned) to view the response message? _____

11. Add a Log Step to the Assembly Diagram.

    a. Drag and drop an **AsyncMediation** onto the Assembly Diagram after the Workday Out Soap.

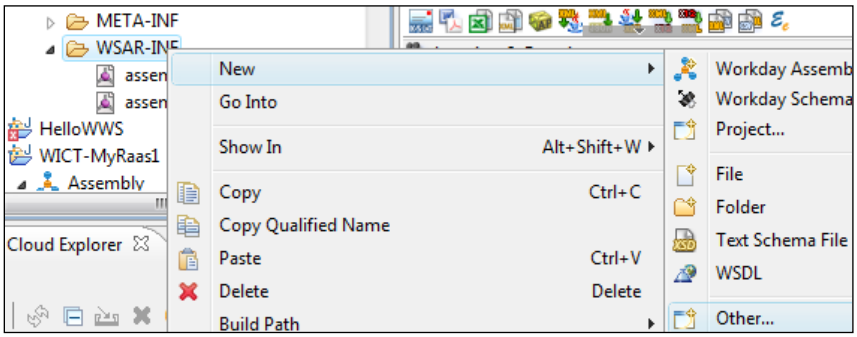
    b. Drag and Drop a **Log** step into the **AsyncMediation**.

12. Select the **Logging** step and access the properties to configure the *Message Builder*.

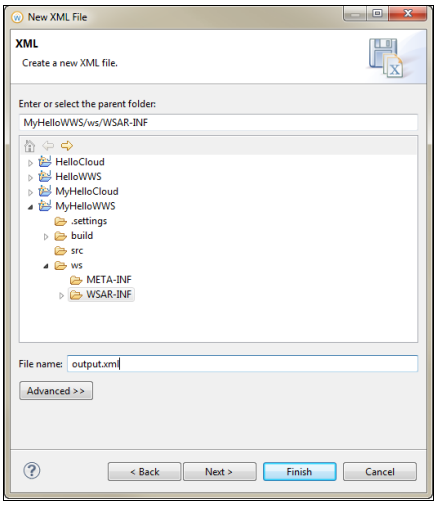


In the *Message Builder*, leave the "**message-content**"

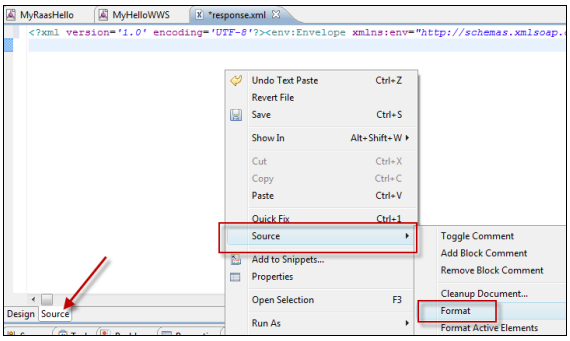
13. Save, deploy and launch to test.
14. Once successfully completed, right click on the Process Monitor in Studio to View the Log File.
15. Find the response xml and highlight the line then control +c to copy
16. Close the log file
17. In the project in the Project Explorer, expand the ws folder and right click on **WSAR-INF** folder.
18. Select *New >> Other...>>*



19. Select *XML >> XML file* then Next.
20. Name the file *output.xml* and click **Finish**.

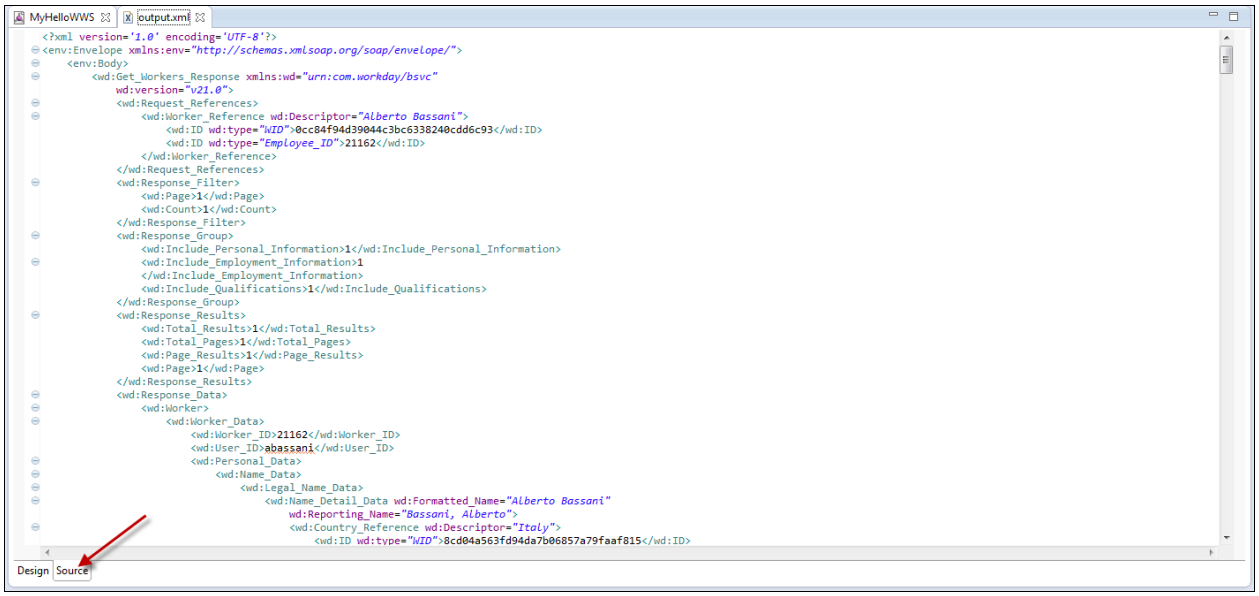


21. Select the **Source** tab and paste in the message you copied in the log file



22. To format the message, right click on the file and navigate to *Source >> Format*

23. Review the results and save



OVERVIEW OF ASSEMBLY COMPONENTS

IN-TRANSPORT COMPONENTS

In-transport components define the starting point of the assembly processing. An assembly must have at least one **workday-in** transport. If it has more than one, it supports multiple ways to kick off processing.

The assembly runtime also provides the **local-in** transport, which defines a re-usable assembly chain that other assembly chains can call, also known as a sub-assembly. This is analogous to declaring a sub-routine in a declarative programming language. A local-out transport is the mechanism by which you can invoke a local-in and is analogous to a sub-routine call.

OUT-TRANSPORT COMPONENTS

Out-transport components take the contents of the **MediationMessage** in **the MediationContext** and send it to another system, for example, an SFTP server. The assembly runtime provides several out-transport components to handle various protocols, for example, HTTP/S, SFTP, FTP/S, E-mail, and XMPP. It also provides two **workday-out** transports:

- **workday-out-soap** for sending SOAP messages to your Workday tenant.
- **workday-out-rest** for retrieving custom reports via Reports-as-a-Service (RaaS) Web services.

COMPONENTS

Basic Mediation Components are the main mechanism for modifying and transforming a message as it flows through an assembly. A mediation defines a sequential ordering of one or more individual processing steps that perform operations on a message as it is passed from one step to another within the mediation definition.

Mediations can either be synchronous or asynchronous. Synchronous mediations consist of two sub-elements, one for request steps and one for response steps. Asynchronous mediations define a single set of steps.

Advanced Assembly Components: Basic components provide a way to connect processing steps together into processing chains. The advanced components provide a way to dynamically control how the assembly runtime traverses the processing chain.

Advanced Mediation Components include:

- Route
- Splitter
- Aggregator

Common Components: Pre-packaged sub-assembly components delivered by Workday that you can use when developing Workday integrations.

SYNCHRONOUS VS. ASYNCHRONOUS MEDIATION COMPONENTS

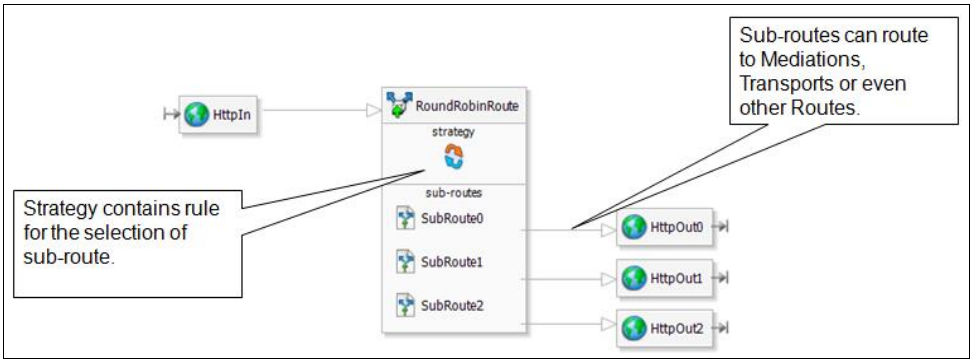
The synchronous mediation component support two lists of steps, request-steps and response-steps, where as asynchronous mediation component supports a single steps element. Steps perform data processing and generally manipulate the contents of the MediationContext. The routes-to attribute specifies which transport, mediation, or service will be called next.

- AsyncMediations only have steps that execute during the push (request) phase.
- SyncMediation have steps that execute during both the push (request) and pop (response) phases.

ROUTE COMPONENT

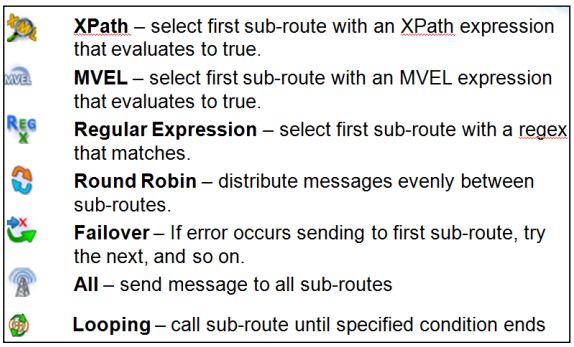
Basic components provide a way to connect processing steps together into processing chains. The advanced components provide a way to dynamically control how the assembly runtime traverses the processing chain.

For example, routes add support for iteration and conditional routing to the assembly runtime in order to process a list. A route consists of one or more sub-routes and a strategy. The strategy tells the route which of the sub-routes to choose during the push phase, also known as the request phase.



The assembly runtime provides several strategies that allow for content-based routing, where the content is the **MediationMessage**:

- **xpath**, which selects a sub-route on the basis of an XPath expression.
- **regex**, which selects a sub-route on the basis of a regular expression.
- **mvel**, which selects a sub-route on the basis of an MVEL expression.
- **round-robin**, which selects a different sub-route each time the route executes.



Some routes can also perform iteration or looping, depending on the following options:

- **all:** allows every sub-route execute in turn.
- **loop:** allows MVEL expressions to set properties in the **MediationContext** to enable iteration.
- **failover:** allows the assembly processor to try a sub-route, then a different sub-route, if it encounters an error.

SCALABILITY

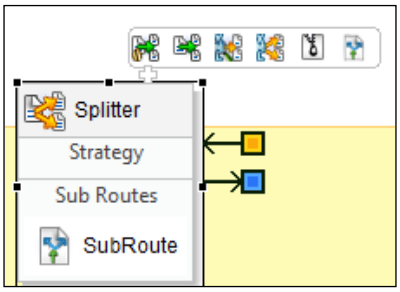
Assemblies provide a number of features to enable the creation of scalable integrations. All Integrations must be developed in a scalable way and must assume that the data extract they process can be arbitrarily large. Whenever employee information is processed you must assume that the number of employees can grow to an arbitrary number. The basic approach to ensuring that integrations are scalable is to use splitters/aggregators to split the document into individual (Employee) records or to use pages as returned by the new public web services.

SPLITTER COMPONENT

The **Splitter** is a mediation component that can process bulk messages containing more than one data record. This component is similar to the **Route** component in that it supports strategies and outputs called sub-routes.

You add strategies and sub-routes to the **Splitter** component by selecting the appropriate icon from the context toolbar when you hover over the **Splitter** component.

- Custom-splitter
- Standard-splitter
- Unzip-splitter
- Xml-stream-splitter
- Xpath-splitter

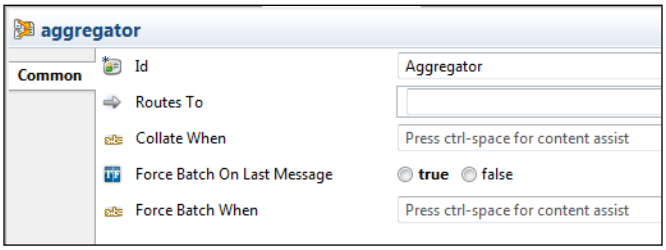


AGGREGATOR COMPONENT

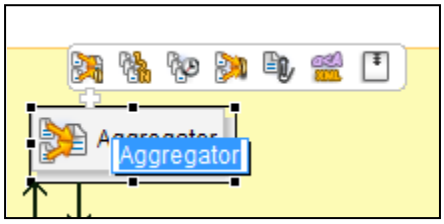
The **Aggregator** component concatenates mediation messages into a single batch message and routes it to the configured mediation destination.

Configurable options include:

- force-batch-when
- collate-when



For each **Aggregator** component, you can configure a custom, size, or time-based batch strategy and a collator strategy that defines how messages are actually aggregated.

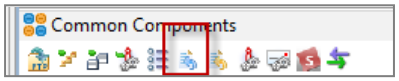


COMMON COMPONENTS

Workday Studio provides several sub-assembly components that you can use when developing Workday integrations. Studio displays sub-assembly components on the Common Component**s** tab of the Assembly Editor palette. When you add a sub-assembly from the Common Component**s** category to the assembly diagram, Studio inserts a local-out element in the assembly XML source with the appropriate endpoint for the sub-assembly in the Workday runtime. You have already used a Common Component in class, the **PutIntegrationMessage**.

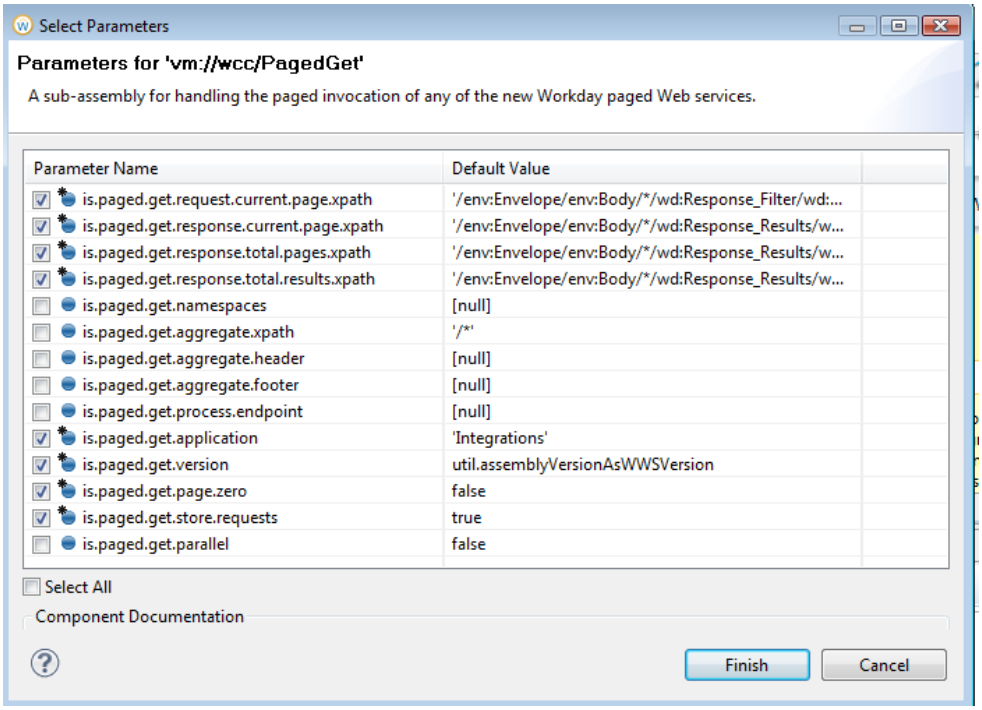
PagedGet Common Component

The **PagedGet** common component handles the paged invocation of any of the Workday Web services and allows data to be retrieved and processed page by page.



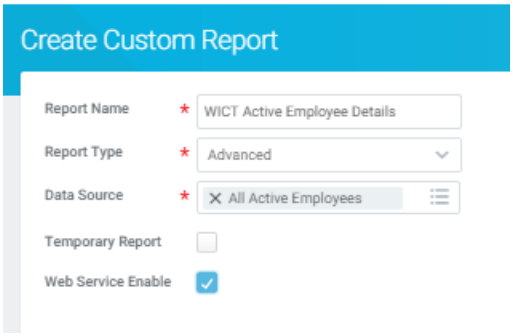
It implements the paging logic required by Workday Studio paged-get Web service operations. For example, you can configure the sub-assembly to send each page of response data to another sub-assembly for processing, or you can aggregate all pages of response data.

The **PagedGet** common component has both "in" and "out" parameters. Below shows a screenshot of the "out" parameters.

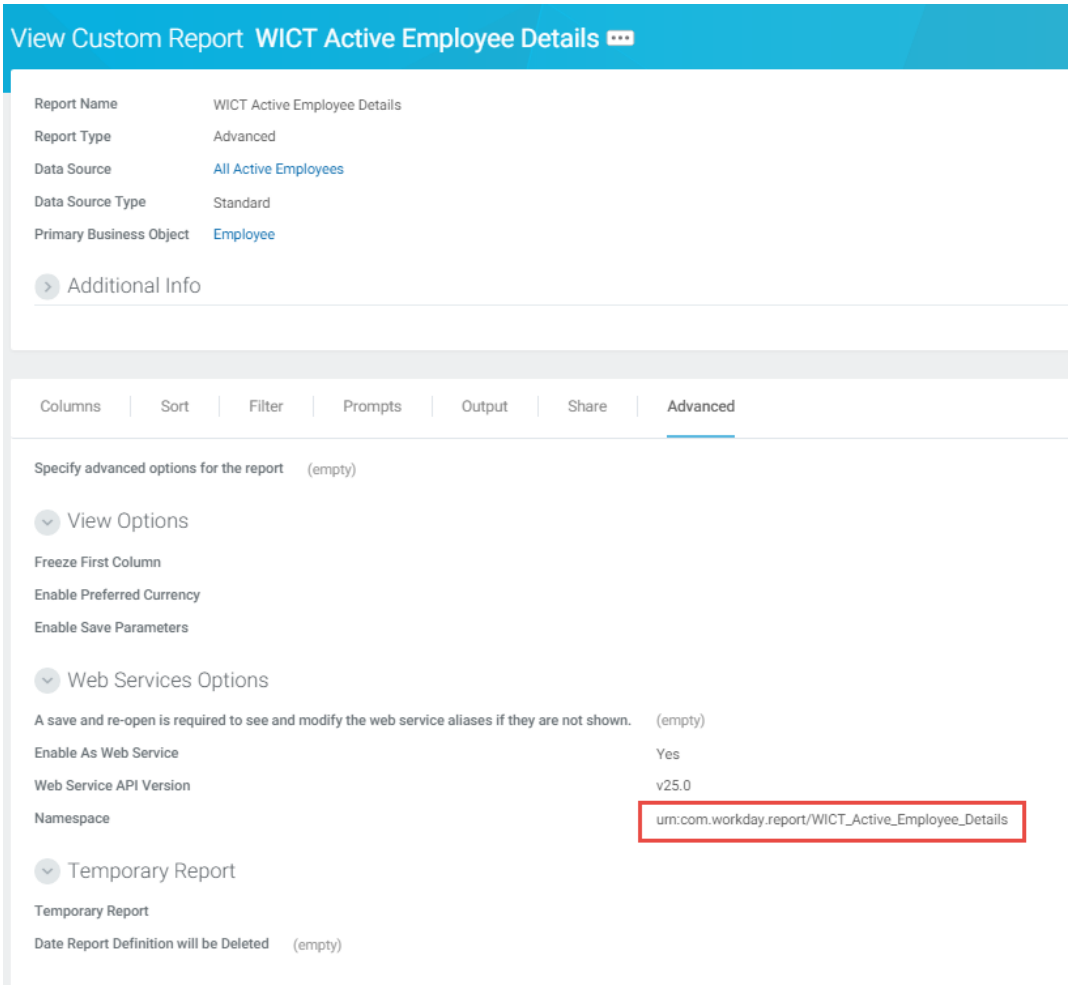


REPORTING AS A SERVICE (RAAS)

Workday enables you to expose advanced custom reports as web services that can be called from Studio integration.



When you enable a custom report as a web service, Workday generates a unique RaaS namespace for the report, using the following format: *urn:com.workday.report/Report_Name* where *Report_Name* is the name of the report.



The RaaS namespace remains fixed even if the report owner changes due to transfers of ownership or the report name changes. This prevents integrations that rely on the report's output from being unable to retrieve data if the report owner changes.

You can edit the namespace for a custom report. However, you should consider the following:

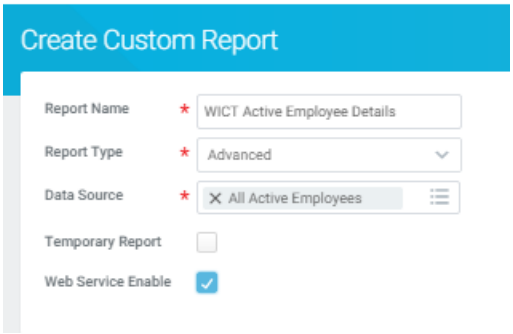
- Workday does not verify that a changed RaaS namespace is unique.
- If you have any integrations that use the report, you will have to update them to use the new RaaS namespace.

Reports that are enabled as a web service must have labels defined for all columns, prompts and related business objects.

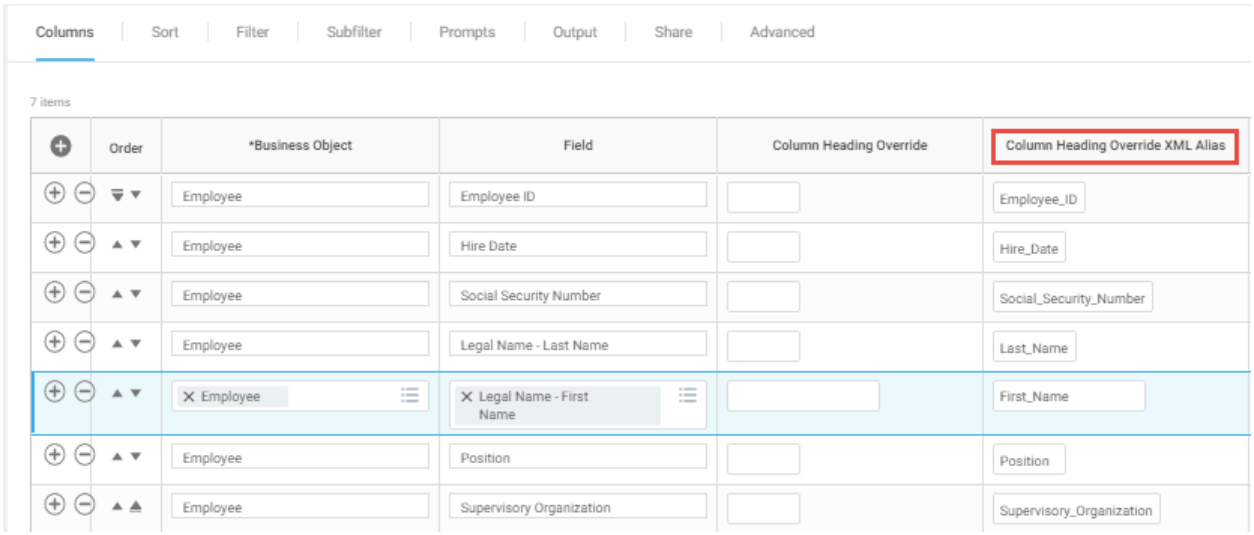
ACTIVITY 6 – CREATE WEB SERVICE ENABLED REPORT

In this activity, you will build a web service enabled custom report called ***WICT Active Employee Details*** that will be called from a Studio Integration.

1. Search for "Create Custom Report"
2. Select the Create Custom Report task and create the following report called **WICT Active Employee Details**

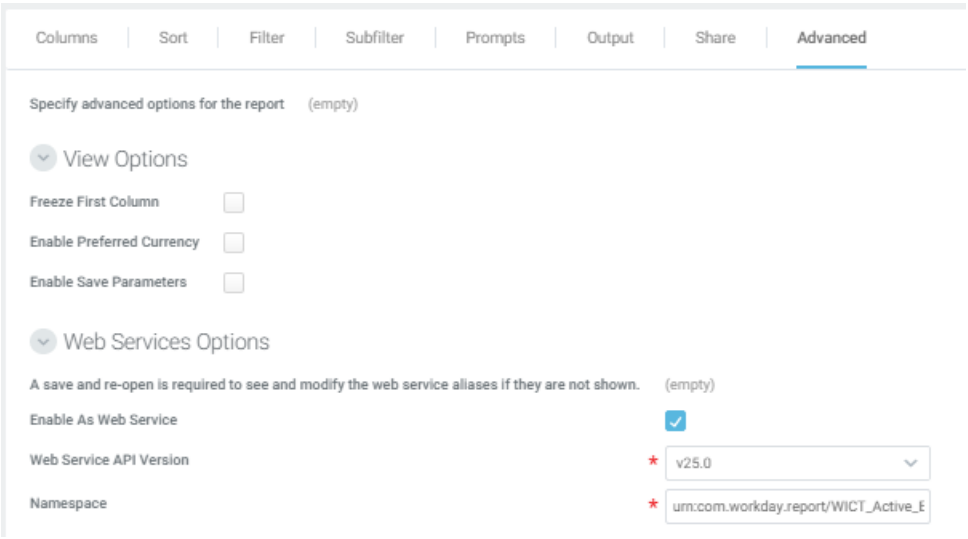


3. Check the **Web Service Enable** box.
4. Click the **OK** button to initiate the Report Writer editing page
5. Add the required fields (and corresponding Column Heading Override in screenshot):
    a. Employee ID
    b. Hire Date
    c. Social Security Number
    d. Legal Name – Last Name
    e. Legal Name – First Name
    f. Position
    g. Supervisory Organization
6. **Colum Heading Override XML Alias** (on Columns tab), verify the following headers. You will need to modify Last_Name and First_Name. The headings cannot contain spaces.



7. Navigate to the **Advanced** Tab

8. Observe the *Namespace* and *Web Service API Version*



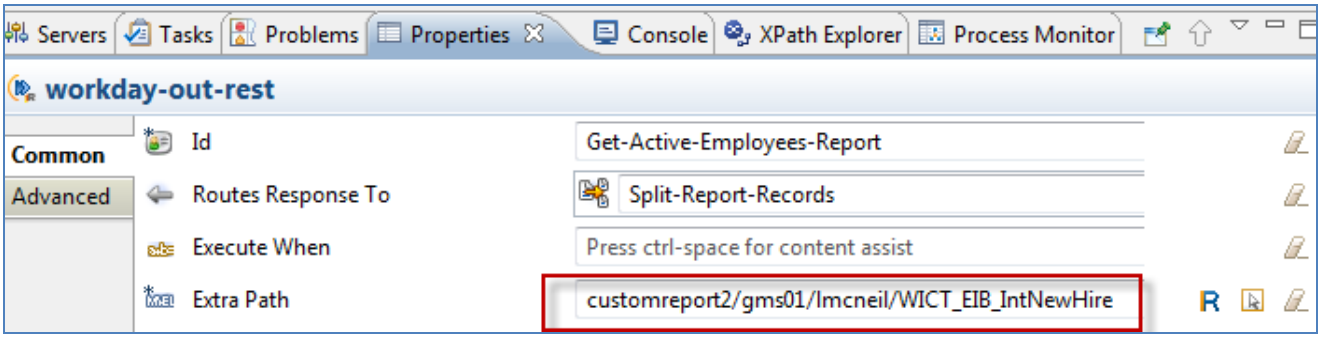
9. Click **Ok** to save the report and then **Run** to view the results (286 instances)

WORKDAY-OUT-REST TRANSPORT

The Workday-Out-Rest is a synchronous, request-response transport that sends Workday application request messages to an HTTP URL. In this transport, you configure an ExtraPath that is the Workday-system path to which the REST request is sent.

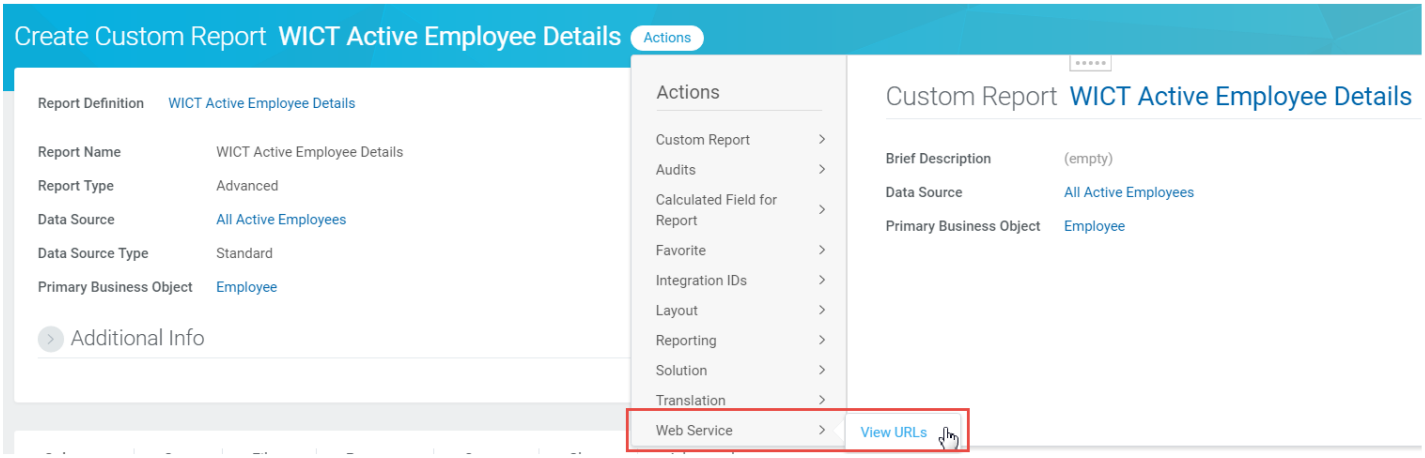
This next example, the endpoint invokes a custom report WICT EIB IntNewHire owned by lmcneil on the gms01 tenant in the default Workday XML format:

customreport2/gms01/lmcneil/WICT_EIB_IntNewHire

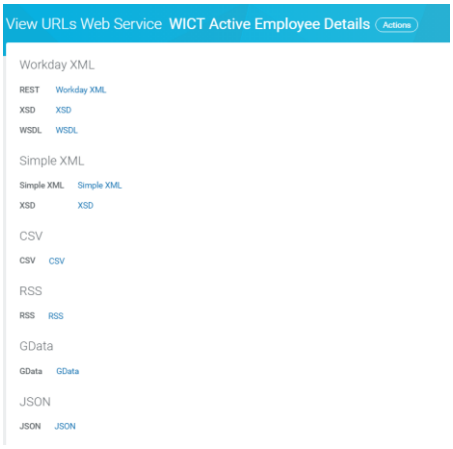


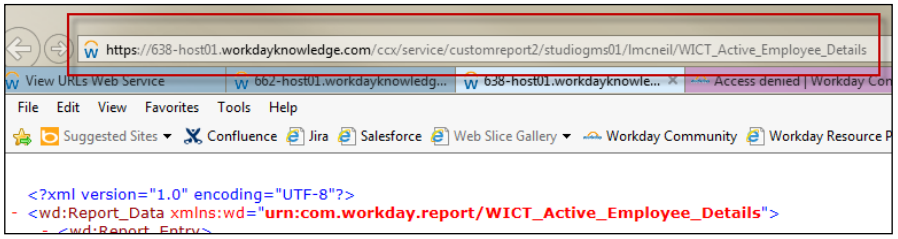
NOTE: It is against Workday Best Practices to use a delivered report in an integration system.

To find the ExtraPath of the report, navigate to the report in the tenant and use the related actions menu to view the Web Service URLs.

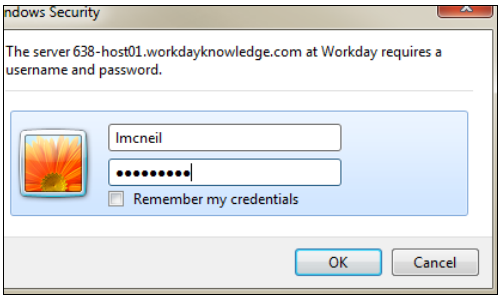


Then click on the Workday XML link to view the report in a browser window. You can copy the extrapath out of the browser URL.





Note: The first time you click on this link in a browser session, you will need to enter a user/password. Make sure pop-ups are not disabled.



ALTERNATIVE OUTPUTS

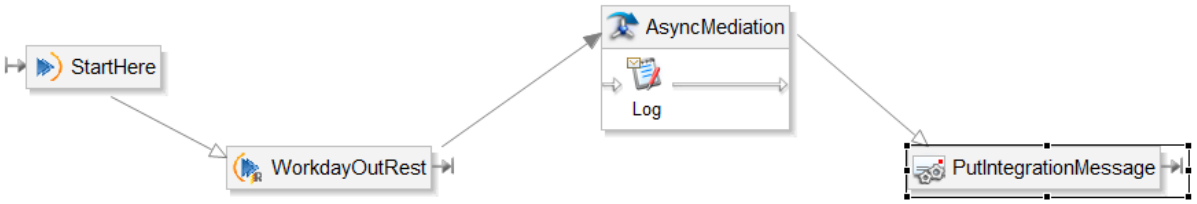
In the extra path of the Workday Out rest component, the default RestURL retrieves the Workday XML. Alternatively, you can retrieve the Alternate Formats available by using the &format on the end of the extra path.

Example:

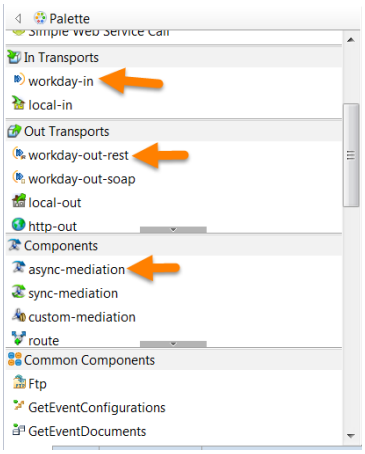
customreport2/studiogms01/lmcneil/WICT_Active_Employee_Details?format=simplexml
customreport2/studiogms01/lmcneil/WICT_Active_Employee_Details?format=csv

ACTIVITY 7 –"MYHELLORAAS" ASSEMBLY

In this activity you will build your own assembly from scratch. You will use the Workday-Out-Rest transport to call the report you wrote in the last activity. Log the report output to a Log Step and place a summary message on the integration event. Your final assembly will look like this:



1. Create a new Assembly Project in Studio called **MyHelloRaaS**
2. Click **Next** and change the collection name to **MyHelloRaasCollection**
3. Click **Finish**
4. On the blank Assembly Diagram, Drag and Drop from the Palette:



   a. **Workday In** transport

   b. **WorkdayOutRest** transport

   c. **AsyncMediation** with a **Log** Step (configured to log the response message)

   d. **PutIntegrationMessage** (PIM) - when the "*Select Parameter*" dialog is displayed, click **Finish**.

5. Wire the components together
6. Access the properties of the Workday-Out-Rest and configure the extra path for your report.generically the URL is:

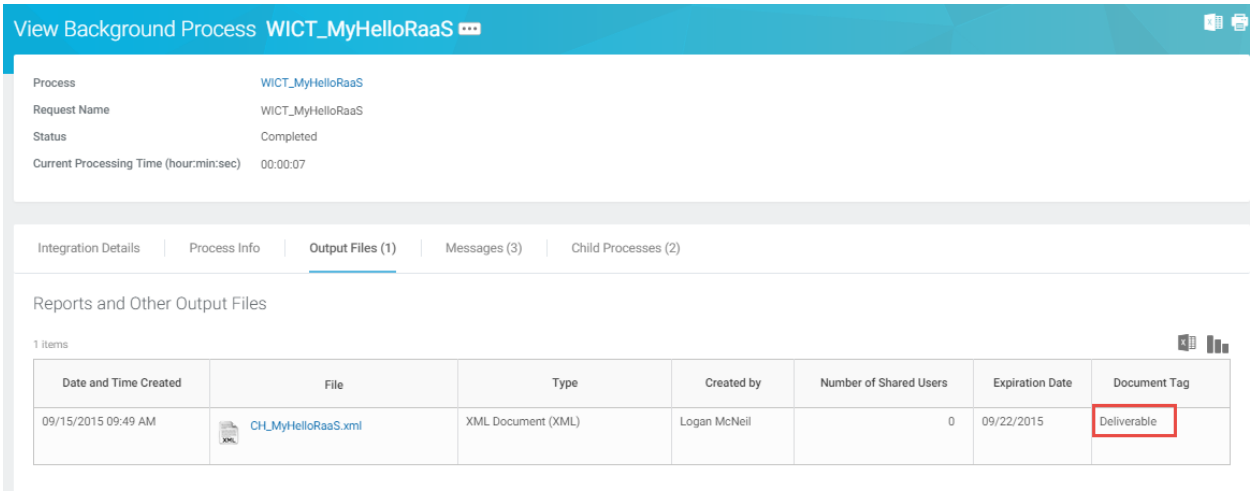
   customreport2/<tenant>/<owner>/<report name>

   Recommended: View the Workday XML (REST) by accessing the report in the tenant and using the relation menu to navigate to *Web Service >> View URLs*.
   The first time you access the REST URL in your browser session, you will need to enter Logan's username and password. **Make sure pop-ups are not disabled.**

7. Configure the other components as necessary.
8. Save the project, deploy and launch.

STORE AND DELIVER

Our next example demonstrates an assembly that creates a simple document, stores it in the document repository, otherwise known as the Workday Blobitory or W: Drive, then adds the document link to the Integration Event in Workday.



STORE STEP

The **store** step stores an atom document or blob of data in the specified collection and workspace within the document message store, also known as the Blobitory or Workday W: Drive. An atom document entry is the default output format.

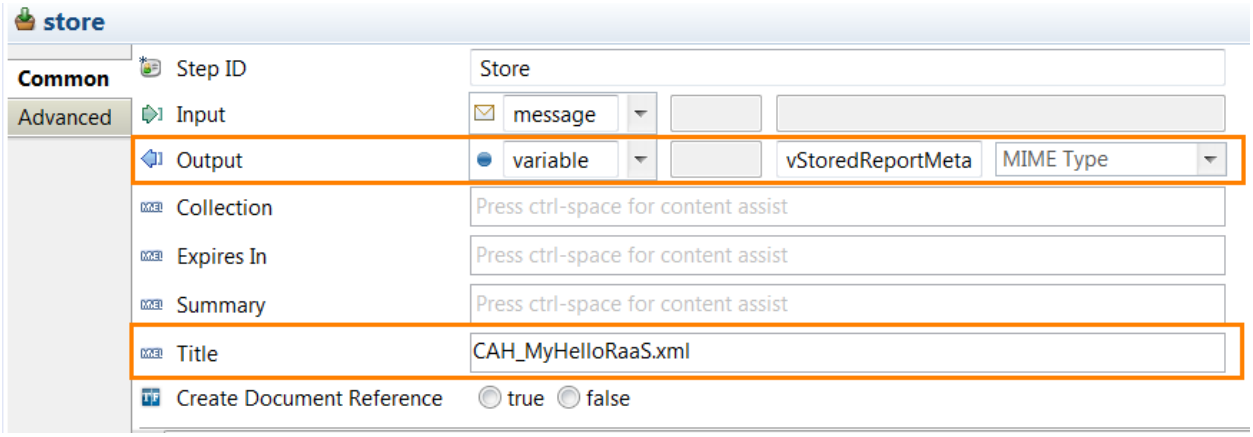
The **Store** step properties of interest include:

- **Output**: Specifies that the step should direct its output to a message variable.
- **Output-Variable**: Specifies the name of the message variable. In this case, it is 'vStoredREportMeta'. This variable contains metadata about the atom document and not the file contents
- **Title**: The title or filename of the document. If not specified, this value defaults to the filename of the external presentation of the message or variable being stored.
    - When using the store step in conjunction with a PutIntegrationMessage component, do not define an is.document.file.name parameter for the PutIntegrationMessage component. Instead, specify a title value for the store step.
- **Create Document Reference**: A Boolean value, which specifies whether the store step automatically calls Workday to create a document reference. The default is true. When set to true, the store step calls the WCC putIntegrationMessage sub-assembly to create a document reference. The parameter settings when calling the sub-assembly are as follows:
        - is.message.severity: INFO

- is.message.summary: value of documentReferenceDescription
- is.document.deliverable: false

If you need to call a different set of parameters, set createDocumentReference to false, then call the putIntegrationMessage sub-assembly explicitly, setting the parameters that you require.

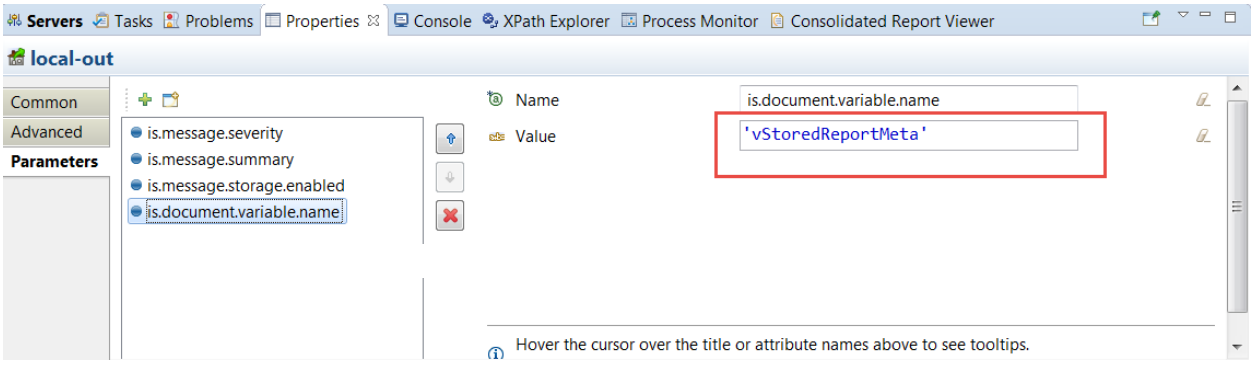
The variable is stored in the Mediation Context and is accessible to any component downstream.



PUT INTEGRATION MESSAGE

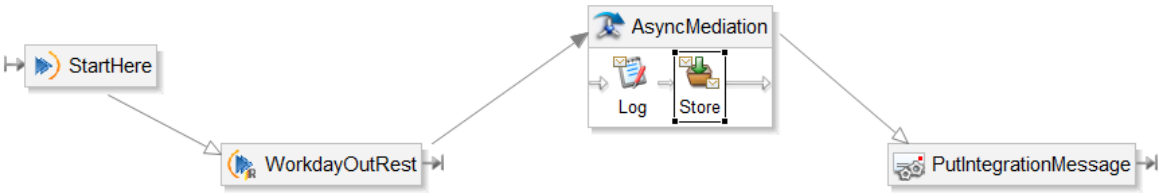
Output File

PutIntegrationMessage is used to update the Integration Event for the process in Workday. The **PutIntegrationMessage** component is configured to add the document link to the Integration Event by setting the *is.document.variable.name* parameter to the name of the variable.



ACTIVITY 8 –"MYHELLORAAS" - STORE AND ....

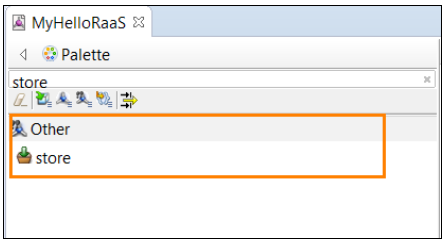
In this activity you will modify the MyHelloRaas Assembly and add Store step once the response is received from Workday. You will modify the PIM to include the result file on the integration event and W:Drive. Your final integration will look something like this:

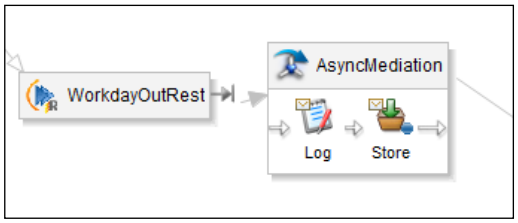


STORE THE OUTPUT OF THE REPORT IN A VARIABLE AND ATTACH AS AN OUTPUT FILE TO THE INTEGRATION EVENT.

1. Open the Assembly Diagram in **MyHelloRaaS**

2. On the Assembly Diagram, Drag and Drop a **Store** step from the Palette and drop it into the existing **AsyncMediation** after the log step.

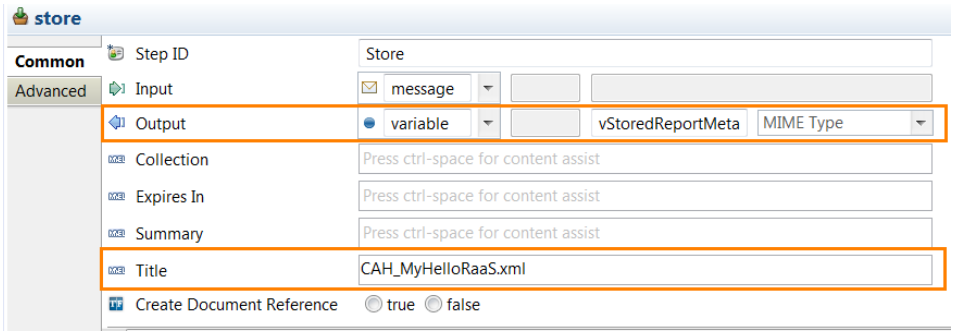




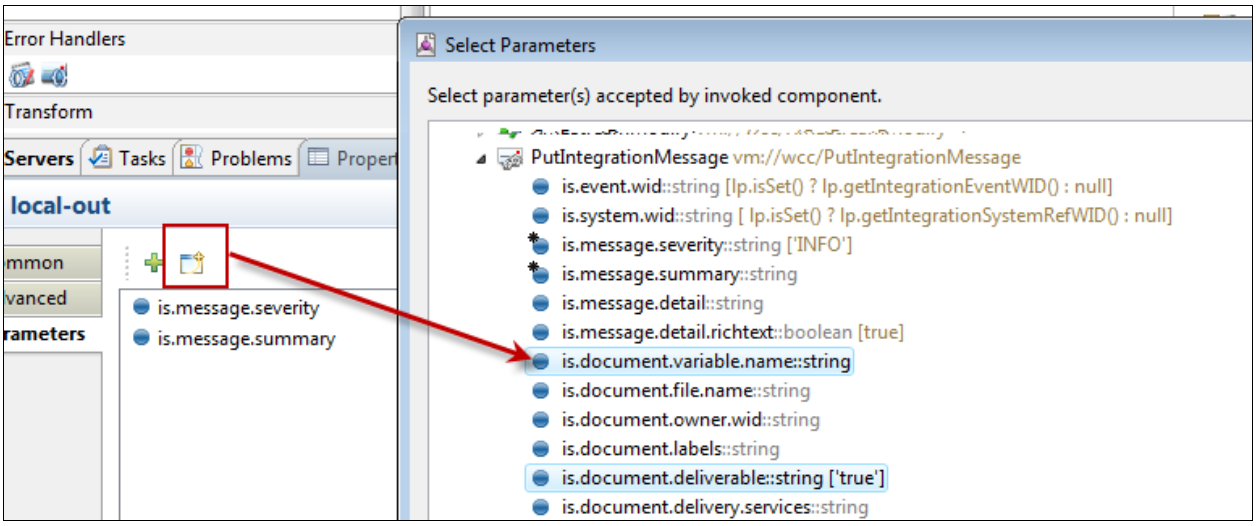
3. Access the properties of the **Store** step and configure the properties to *Output* a *Variable* called v**StoredReportMeta**.

4. Add a Title of <your initials>_**MyRaaSOutputFile.xml**

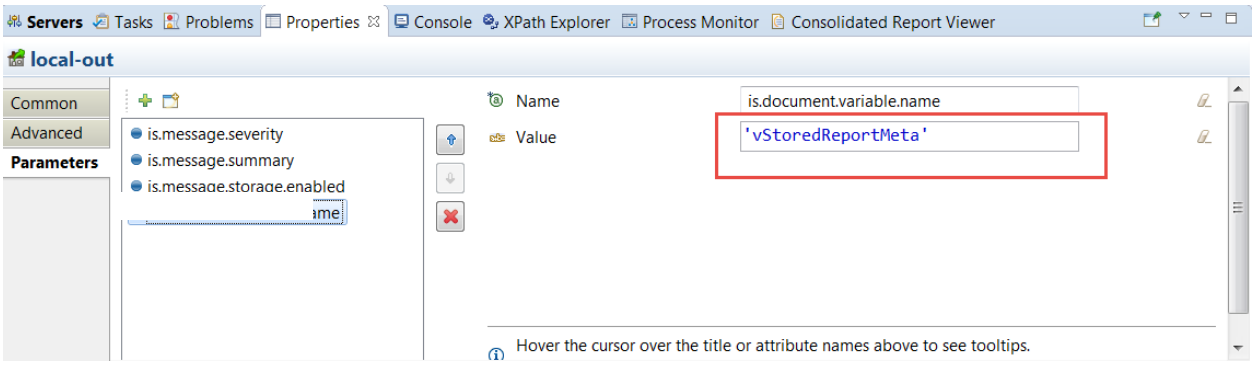
5. Check the False radio button for the *Create Document Reference*



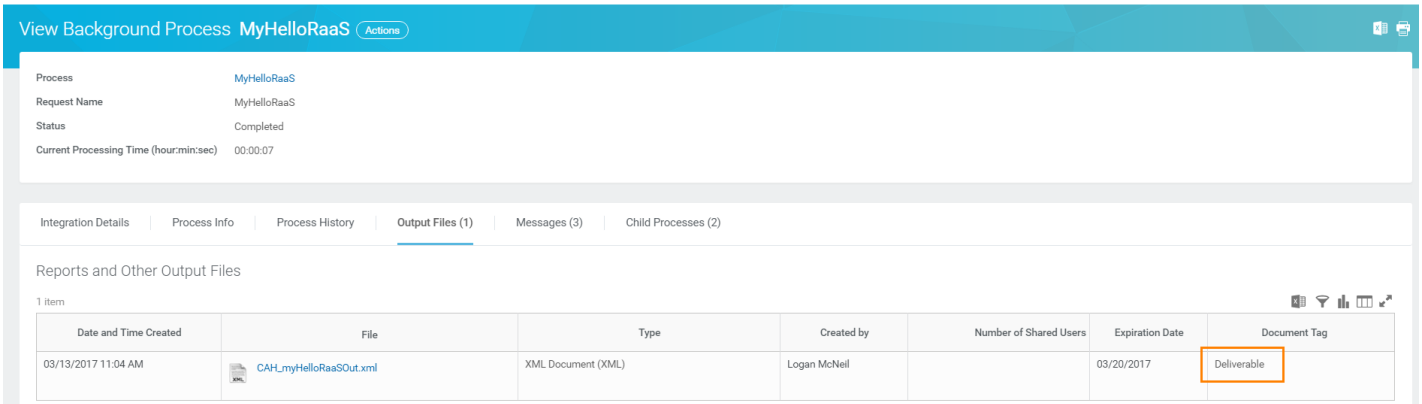
6. Access the **PutIntegrationMessage** properties Parameters tab. Click the Select Parameter tool button. Select *is.document.variable.name* then click **OK**.



7. Access the **PutIntegrationMessage** and configure the *is.document.variable.name parameter*. *Don't forget the 'single quotes'.



8. Save the project, deploy and launch. View the integration event Output tab to verify file.



DELIVERY SERVICE

Using the Workday Delivery service, you can configure integrations that allow documents to be delivered over a range of different transports such as AS2, FTP, FTPS, and SFTP. The main advantages of using this service over other drag-and-drop transports within an assembly are that you can manage the configuration of those transports within the Workday application, and also that you can have the documents delivered as part of a larger business process with, for example, an explicit approval step, prior to delivery.

INTEGRATION BUSINESS PROCESS

Workday supports integration-specific business processes using the Integration Process Event business process type. This business process type allows you to:

- Separate the execution of the integration itself from the act of retrieving or delivering files.
- Chain together several integration systems, allowing a subsequent integration system to consume the integration file(s) produced by 1 or more earlier integration systems.

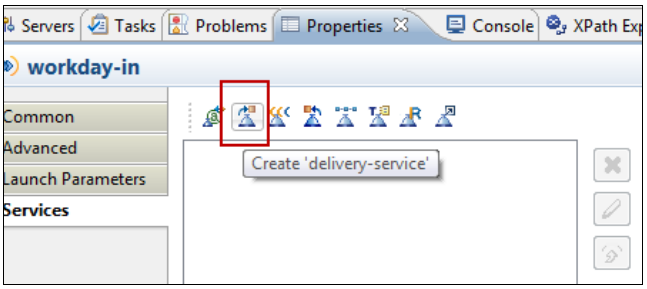
The Integration Process Event default definition is usually delivered with your tenant. The default Integration Process Event consists of 2 steps, *Initiation* and *Service* (*Fire Integration*).

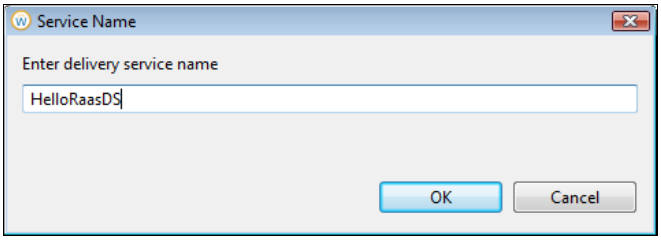
DELIVERY SERVICE

Workday separated the relationship between the Delivery Service definition on the integration and the Delivery Step within a Business Process (BP). It is the BP Delivery Step that defines how a document is delivered to an external endpoint. Functional users configure the Delivery Step in the Workday application, that is, details of what to deliver and where.

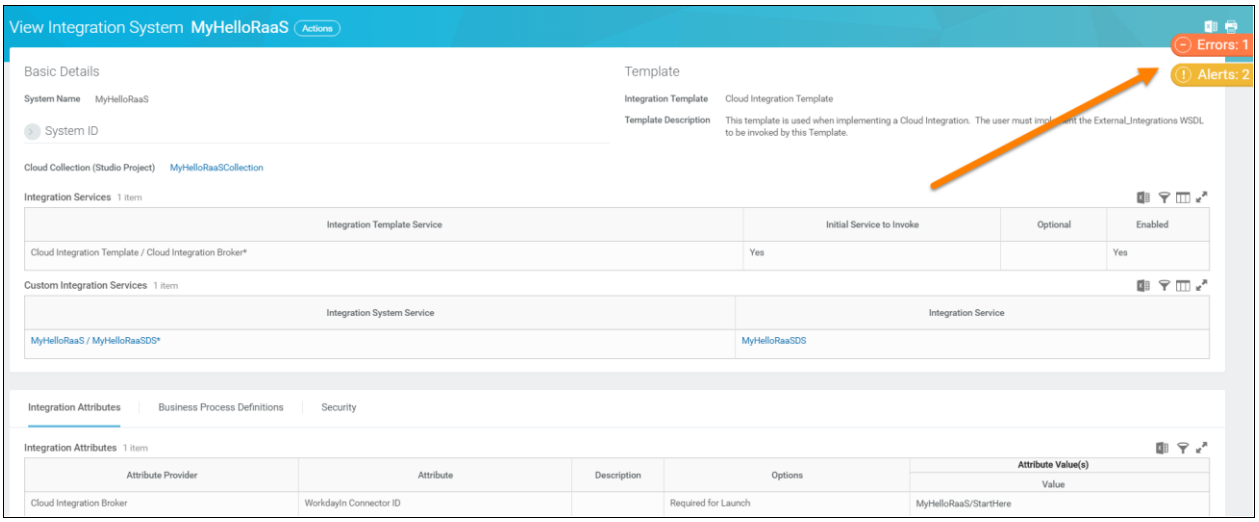
When an integration creates documents, each Delivery Service, in itself, acts as a document-tagging system. You can define multiple Delivery Services in your **workday-in** definitions, each for different purposes, and allow each Delivery Service to apply a different tag to the document. By default the tag is the Delivery Service name. When functional users configure the BP Delivery Step in Workday, they can filter out the appropriate documents by directing documents with a specific tag to a specific location. You can also have multiple BP Delivery Steps for the same integration within its BP Definition.

The Delivery Service is defined on the Workday In transport properties.

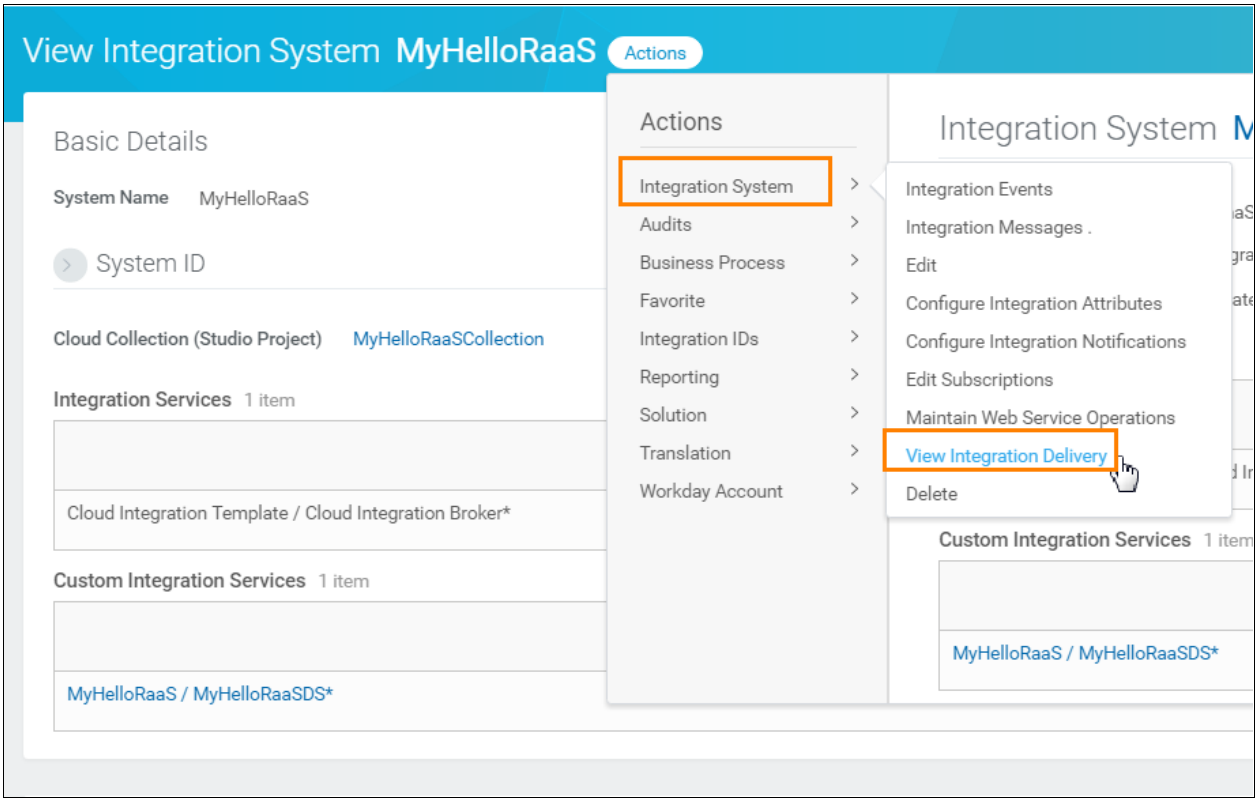




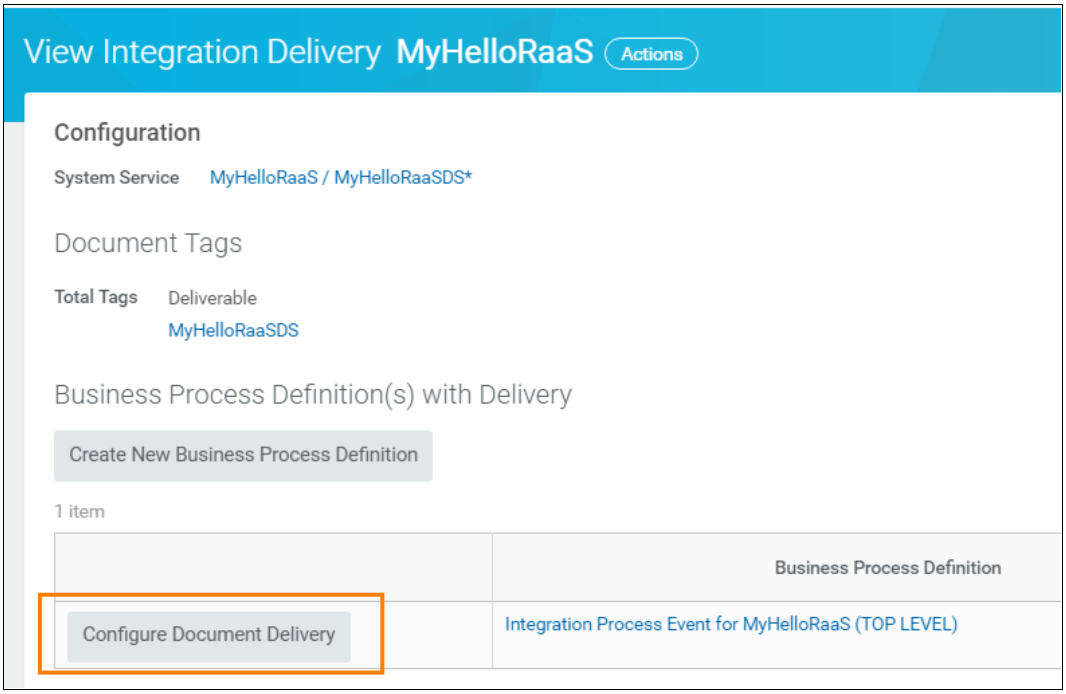
Once deployed to Workday, the Delivery Service is configurable through the tenant. Search for and retrieve your integration system. You will notice the "your attention is required" message.



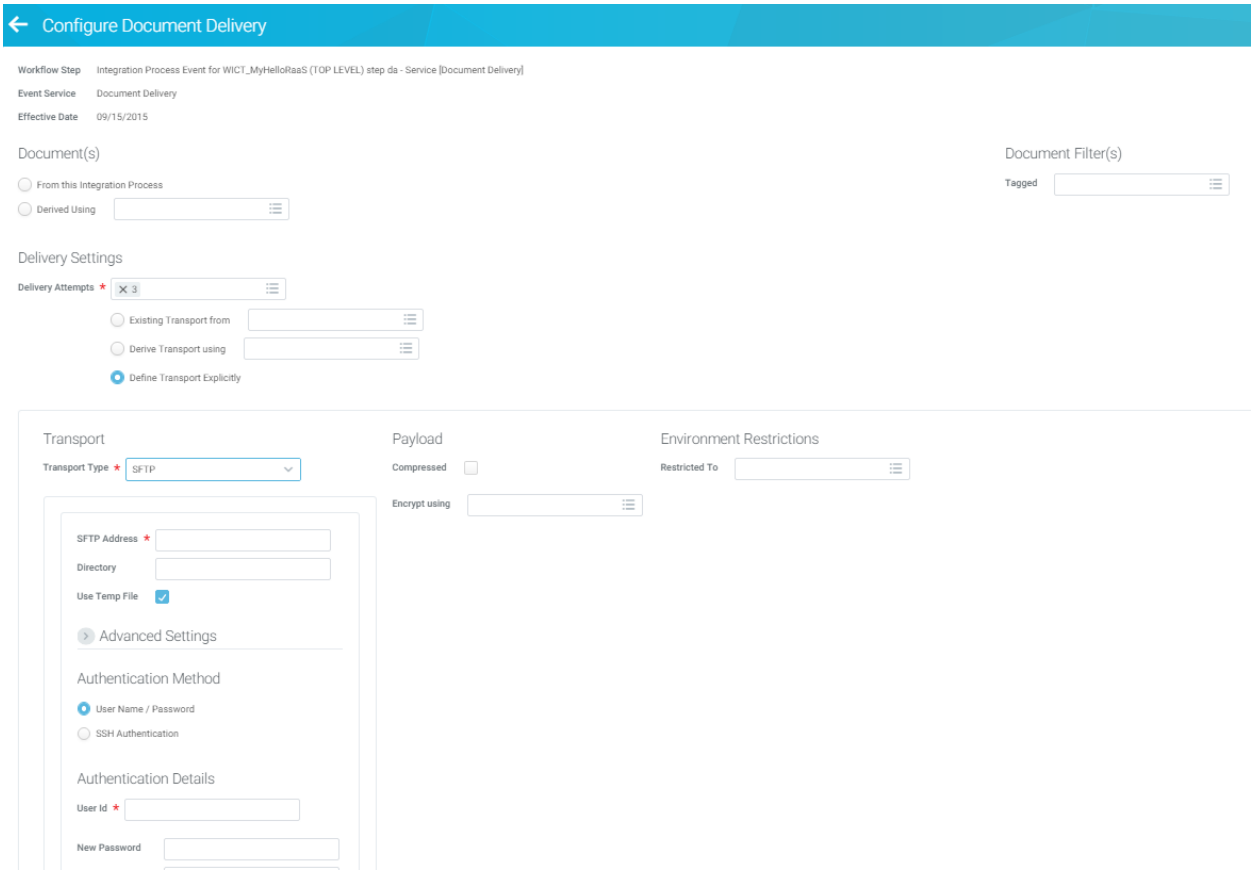
You will need to configure the Integration System Delivery service.



When the View Integration Delivery page is displayed, you have a button to **Configure Document Delivery**. Notice this is part of a Business Process called *Integration Process Event for <your integration system> (Top Level)*.

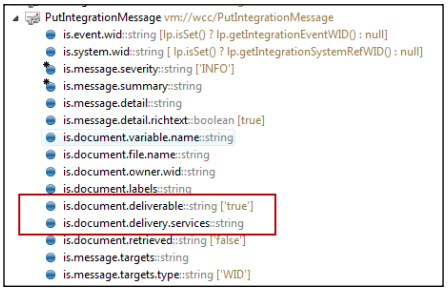


The document is created in this integration process. If you are using multiple delivery services, you will want to "Tag" the delivery step. The name of the delivery service(s) created on the integration system will be available in the Document Filter(s) Tagged field. Configure the transport which represents the endpoint where the file(s) will be delivered.



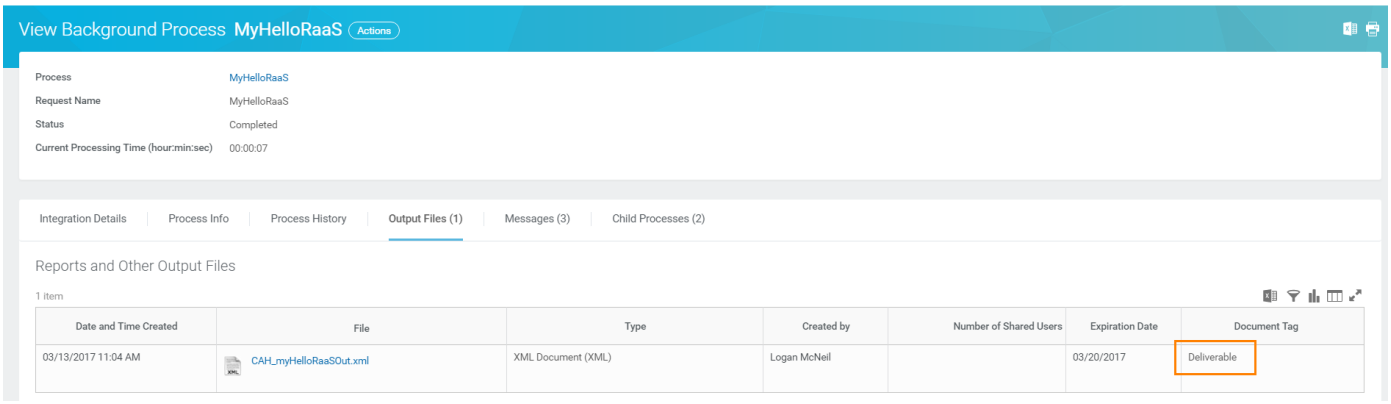
PUTINTEGRATIONMESSAGE

There are two parameters that can be configured for the Delivery Service on the Put Integration Message.



- ***is. document.deliverable***

  If set to false, this parameter ensures that the document added to the integration event is never automatically delivered to a customer or external provider. An example would be an audit log of the integration system. This value defaults to true.

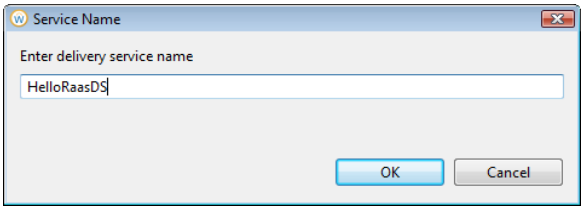


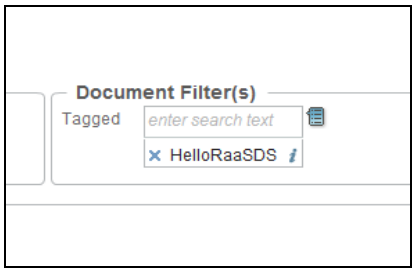
DELIVERY SERVICE TAGGING

When an integration creates documents, each Delivery Service, in itself, acts as a document-tagging system. Each Delivery Service is configured separately.

You can define multiple Delivery Services in your **Workday-In** transport. This enables you to have a Delivery Service to apply a different tag to different documents.

By default the tag is the Delivery Service name. If your integration produces multiple output files, you can use the PIM parameter, ***is.document.delivery.services***, to identify which delivery service should transport the file.

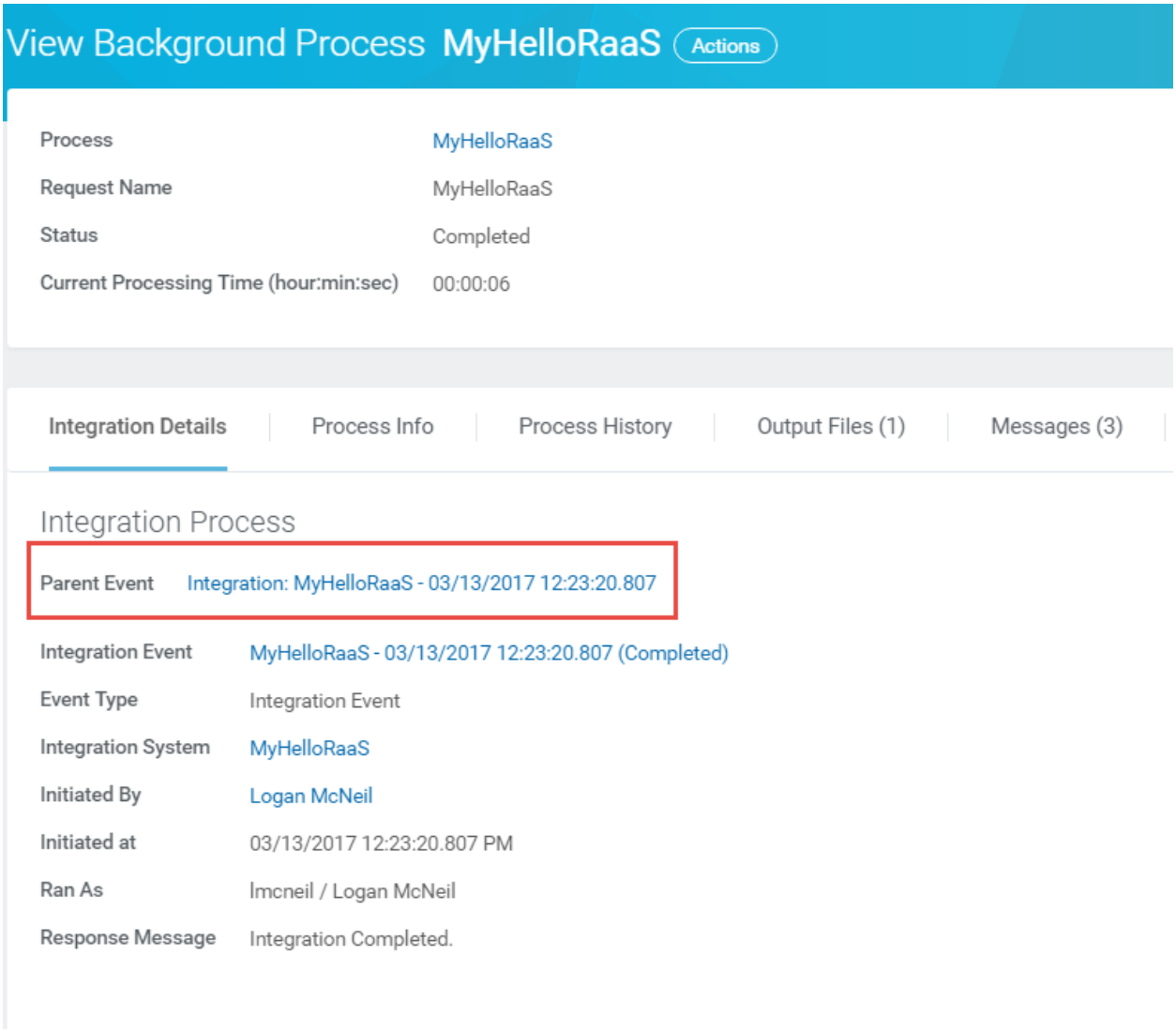




If nothing specified, deliverable documents will be delivered using all delivery services associated with the integration system.

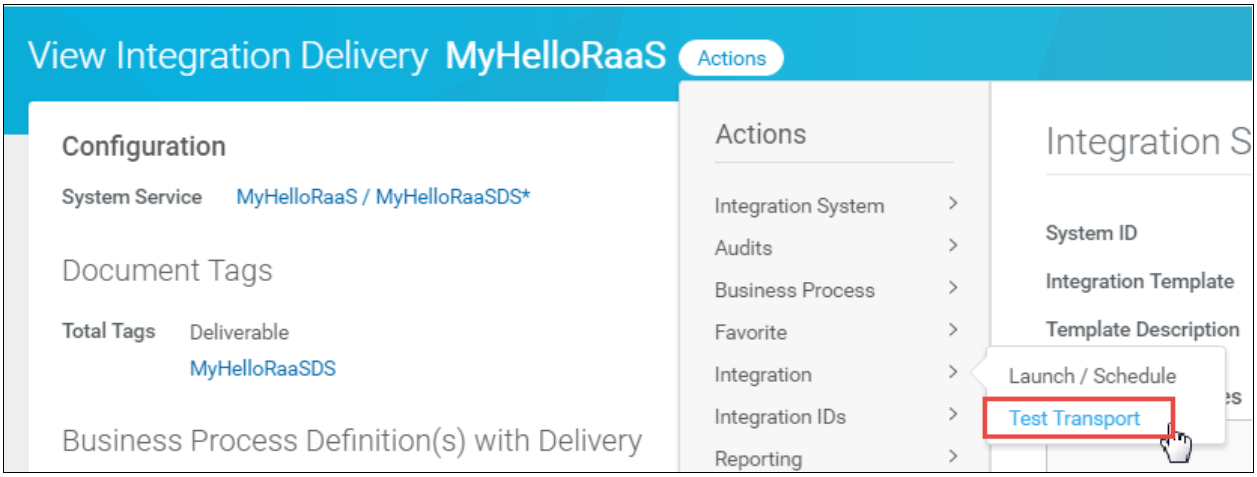
VERIFYING DELIVERY

Because the execution of the integration system is separate from the delivery, the integration Event will not indicate whether the file was delivery was successful. The integration is just one step in the overall business process. You can view the "Parent Event" to see the status of all of the steps defined in the business processes associates with that integration system.

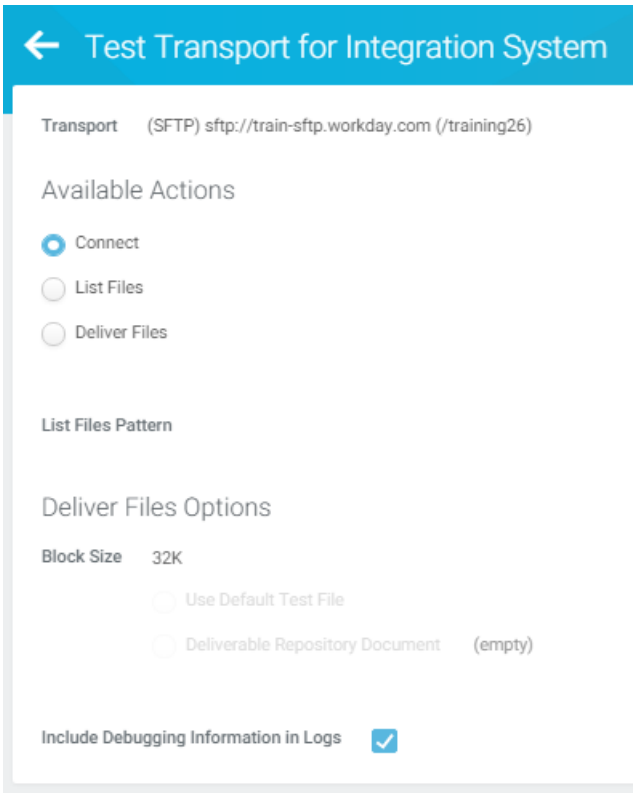


TEST TRANSPORT

You can verify your file was delivered using the Test Transport action of the integration system.



Use the Test Transport task to test the configured Delivery Service or Retrieval Service for an integration system to verify and troubleshoot connectivity to the external endpoint. When testing your integration system's configured transport, you can either use a 10KB test file, or select a file from a previous integration event.



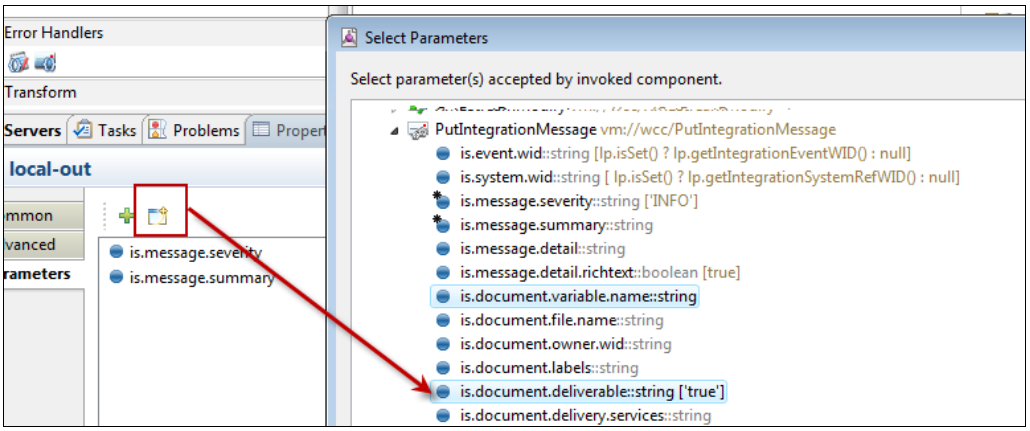
ACTIVITY 9 –"MYHELLORAAS" -....AND DELIVER

In this activity you will modify the MyHelloRaas Assembly and add a Delivery Service to deliver the "deliverable" document to the SFTP Server. The instructor will provide the SFTP endpoint access information.

CREATE AND CONFIGURE THE DELIVERY SERVICE

1. Open the Assembly Diagram in **MyHelloRaaS**

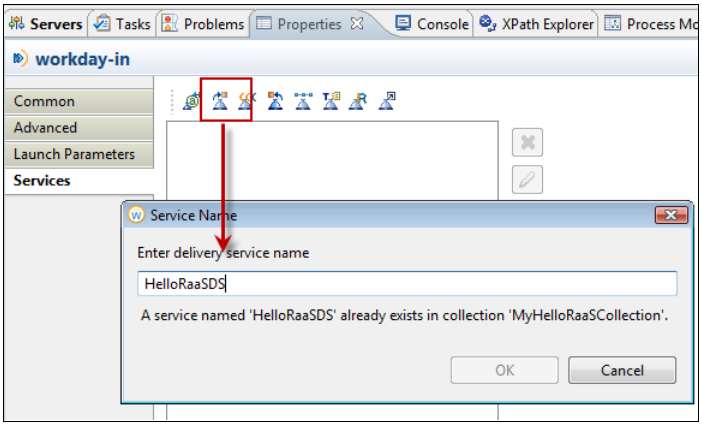
2. Access the **PutIntegrationMessage** properties Parameters tab. Click the Select Parameter tool button. Select *is.document.deliverable* then click **OK**.



3. The value of the *is.document.deliverable* should default to 'true'.

4. Go to the **Services** tab on the properties of the **Workday In** Transport on the assembly diagram.

5. Select the Create 'Delivery-Service' tool button and name it *HelloRaaSDS* then click **OK**.

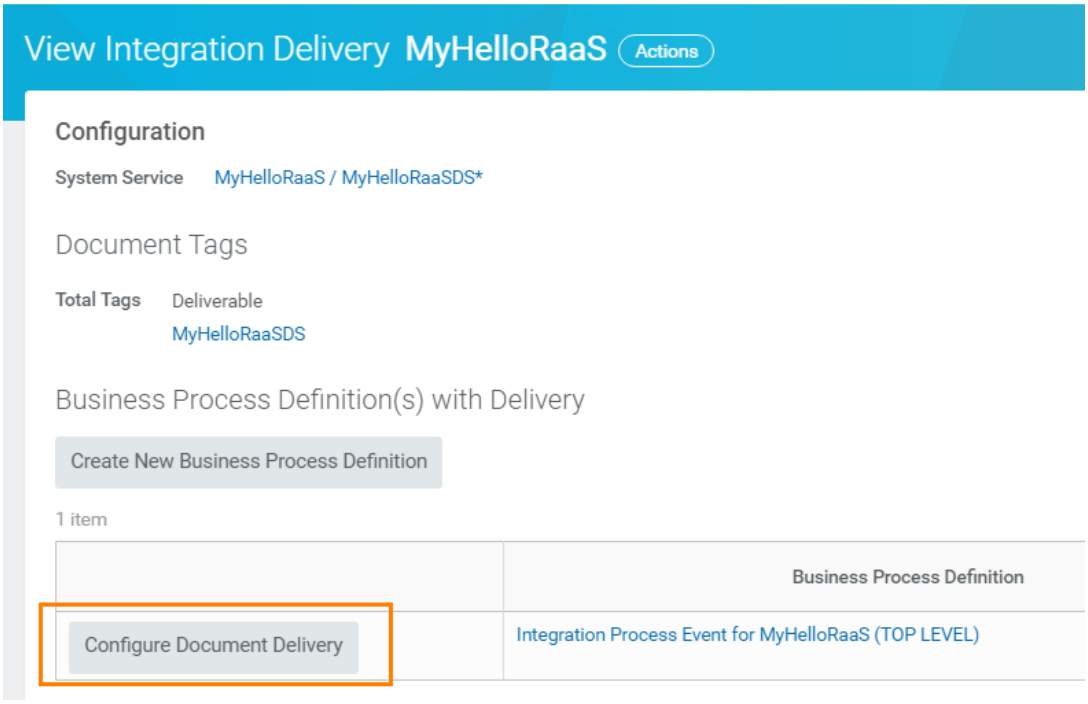


6. Save and Deploy the project to Workday.

7. Search for and retrieve the **MyHelloRaaS** integration system in the tenant.

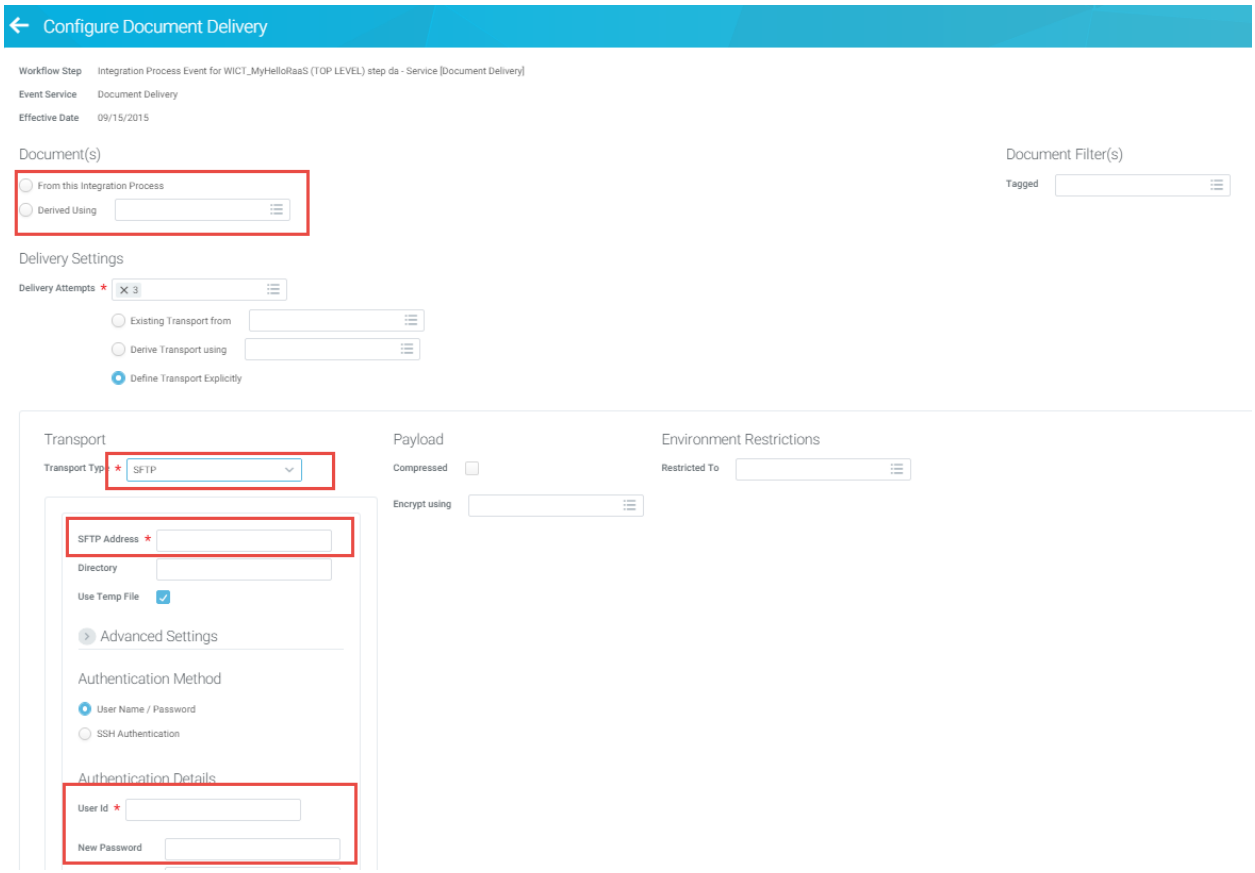
8. On the related action menu, navigate to *Integration System >> View Integration Delivery*

9. Select the **Configure Document Delivery** button. Accept today as the Effective Date and click **OK**.



10. In the Configure Document Delivery page, define the following:

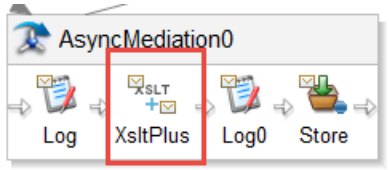
    a. **Document(s)** - From this Integration Process
    b. **Transport**
        i. **Transport Type** - SFTP
        ii. **SFTP Address** - sftp://wd2-sales-sftp.workday.net:3022
        iii. **Directory** - <Provided by Instructor>
        iv. **User ID** - <Provided by Instructor>
        v. **Password** - <Provided by Instructor>



11. Click **OK** to save then **Done**
12. Launch the integration from Studio and verify the results

XSLT IN STUDIO

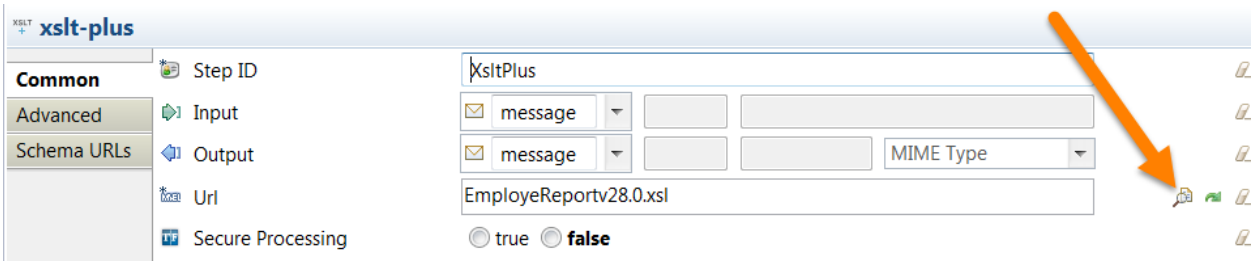
There are times when Workday does not present output in the desired format for simple integration tasks. One example is upper-case and lower-case letter. Workday stores people's names in mixed-case, but another system might require the last name appear as all upper-case letters. Another example might be Social Security Number. SSN is typically formatted as a 9 digit numeric value (if unmasked). Another system could require that the data be passed but, only the last 4 digits. In this case an Extensible Stylesheet Language Transformation (XSLT) could be implemented in Workday Studio.



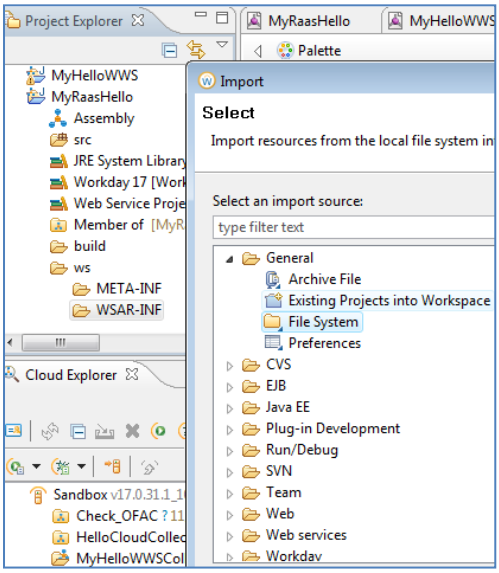
XSLT+ STEP

This step can be used to transform an XML document using an XSL Transformation (XSLT) stylesheet. The XSLT+ step is added to a mediation, and supports XSLT version 3.0 features.

You can either import and attach the XSLT file, or create it in the properties of the XSLT Step.



To import an existing XSLT file, navigate in your project to the **WS >> WSAR-INF** folder and right click to select **Import**. Under *General*, select *File System* and click **Next**. Browse to select the directory that contains the existing file. Once that directory is select, all files will be displayed. Select the file(s) that you want to import into your project.



To create a new XSLT file, navigate in your project to the **WS >> WSAR-INF** folder and right click to select **New >> Other**. Under the **XML** folder select *XSL* and click **Next**. Enter a file name with an extension of *.xsl*. If you are starting from scratch, click **Next** to select a template. If you plan to copy/paste, click **Finish**.

XSLT REMINDERS

In the XSLT file, the Namespace and column Override Headers are referenced. The values must match exactly or the file will return a blank.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:wd="urn:com.workday.report/WICT_Active_Employee_Details">

    <xsl:output method="xml"/>


    <xsl:template match="/">
    <Summary>
       <xsl:for-each select="/wd:Report_Data/wd:Report_Entry">
        <Worker>
            <EMPLID><xsl:value-of select="wd:Employee_ID"/></EMPLID>
            <HIREDT><xsl:value-of select="wd:Hire_Date"/></HIREDT>
            <!-- substring the SSN -->
            <SSN><xsl:value-of select="substring(wd:Social_Security_Number,6,4)"/></SSN>
            <LAST><xsl:value-of select="wd:Last_Name"/></LAST>
            <FIRST><xsl:value-of select="wd:First_Name"/></FIRST>
            <POSITION><xsl:value-of select="wd:Position/@wd:Descriptor"/></POSITION>
            <SUPERVISORYORG><xsl:value-of select="wd:Supervisory_Organization/@wd:Descriptor"/></SUPERVISORYORG>
        </Worker>
       </xsl:for-each>
     </Summary>
    </xsl:template>

</xsl:stylesheet>
```

The XSLT is based on the Workday XML Output of the report.

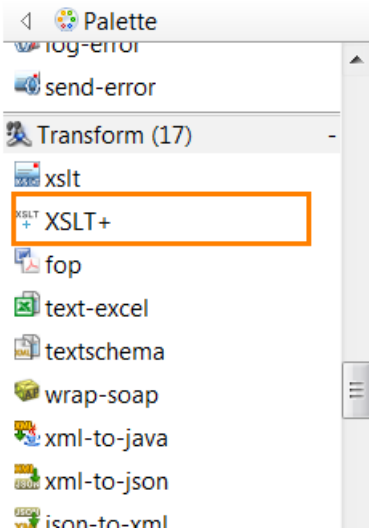
```
<?xml version="1.0" encoding="UTF-8"?>
- <wd:Report_Data xmlns:wd="urn:com.workday.report/WICT_Active_Employee_Details">
  - <wd:Report_Entry>
      <wd:Employee_ID>21001</wd:Employee_ID>
      <wd:Hire_Date>2000-01-01-08:00</wd:Hire_Date>
      <wd:Social_Security_Number>545212822</wd:Social_Security_Number>
      <wd:Last_Name>McNeil</wd:Last_Name>
      <wd:First_Name>Logan</wd:First_Name>
    - <wd:Position wd:Descriptor="Chief Human Resources Officer">
        <wd:ID wd:type="WID">ae5982654df8420dacc5f530e096215b</wd:ID>
      </wd:Position>
    - <wd:Supervisory_Organization wd:Descriptor="Executive Management Group">
        <wd:ID wd:type="WID">9378b36dabd94cc391ef5392166aa00e</wd:ID>
        <wd:ID wd:type="Organization_Reference_ID">Executive_Management_supervisory</wd:ID>
      </wd:Supervisory_Organization>
    </wd:Report_Entry>
  + <wd:Report_Entry>
  + <wd:Report_Entry>
  + <wd:Report_Entry>
  + <wd:Report_Entry>
  + <wd:Report_Entry>
  + <wd:Report_Entry>
  + <wd:Report_Entry>
  + <wd:Report_Entry>
  + <wd:Report_Entry>
```

ACTIVITY 10 –"MYHELLORAAS" AND XSLT

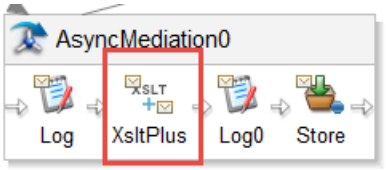
In this activity you will modify the MyHelloRaas Assembly and add an XSLT step to transform the results of the report once received back from Workday.

1. Open the Assembly Diagram in **MyHelloRaaS**

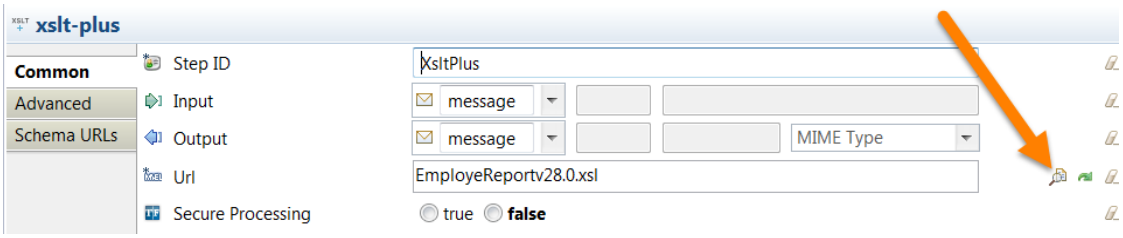
2. On the Assembly Diagram, Drag and Drop an **XSLT+** step from the Palette and drop it into the existing **AsyncMediation** after the **Log** step before the **Store** step.



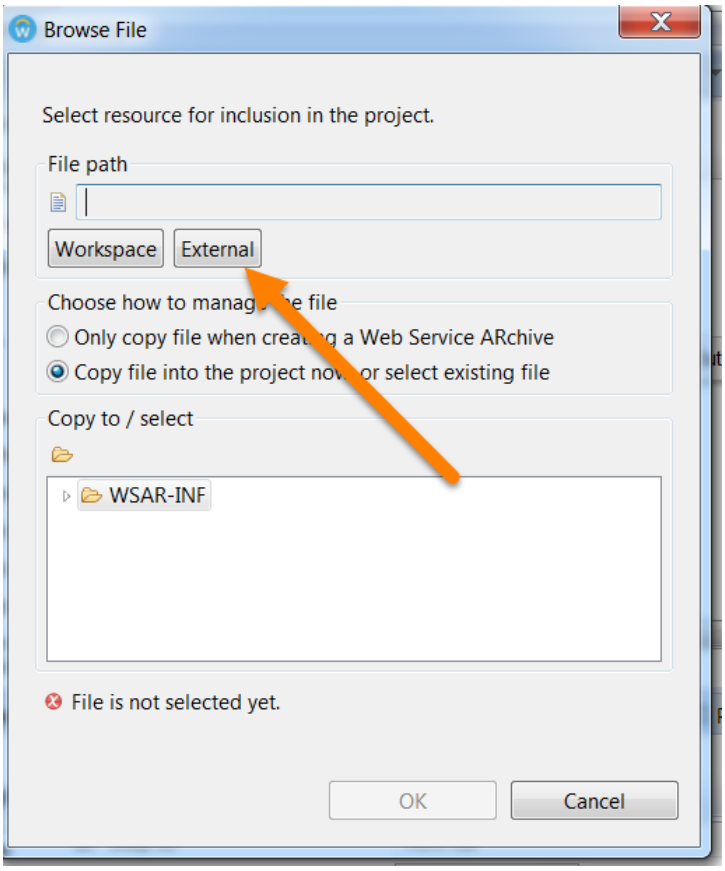
3. Access the properties of the **XSLT+** step and configure the URL for the XSLT file.



   a. Import the provided XSLT file by clicking on the **Browse for File** icon.



   b. Click **External**



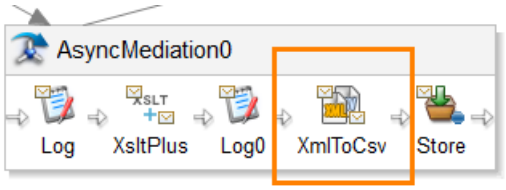
c. Browse to the location of the provided XSLT, select it and click OK.

4. In the same **AsyncMediation**, add another **Logging** step after the XSLT step and log the result message of the XSLT transformation.

5. Save the project, deploy and launch.

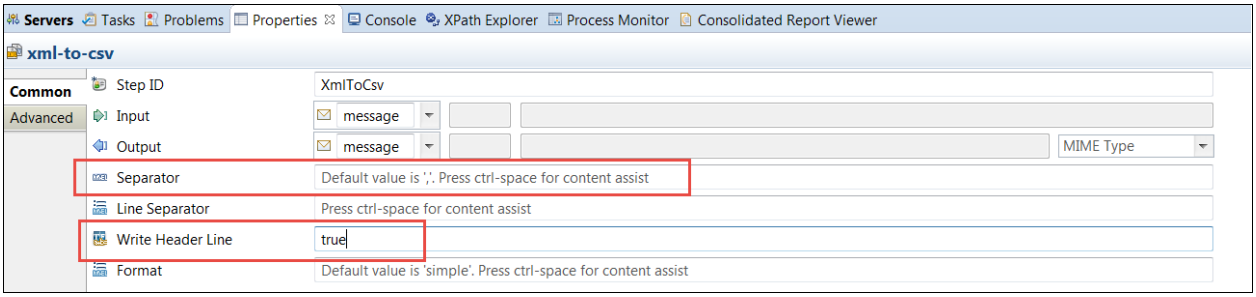
CREATING A DELIMITED FILE

The **xml-to-csv step** enables you to easily create delimited files from an xml mediation message. Frequently output specifications will require a comma separated variable (CSV) file and it is important to know how to best create one with Studio.



The preferred method for creating a CSV is the xml-to-csv step. This step which would be added into a simple mediation component, after the integration is done processing but before it stores the output. Using the Separator box will allow you to control the character which separates the columns, typically a comma, but Pipe (|) and tab (TAB) characters are also common.

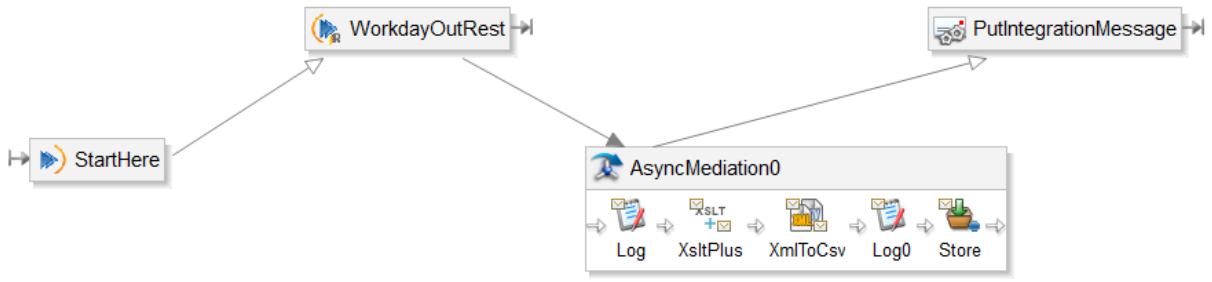
If you wish to have column headers, set the Write Header Line to 'true'.



We do commonly find solutions in the field which manually create a CSV using XSLT. This solution, which does work at small sizes does not scale well, and is onerous to maintain.

ACTIVITY 11 –USING "MYHELLORAAS" TO CREATE A DELIMITED FILE

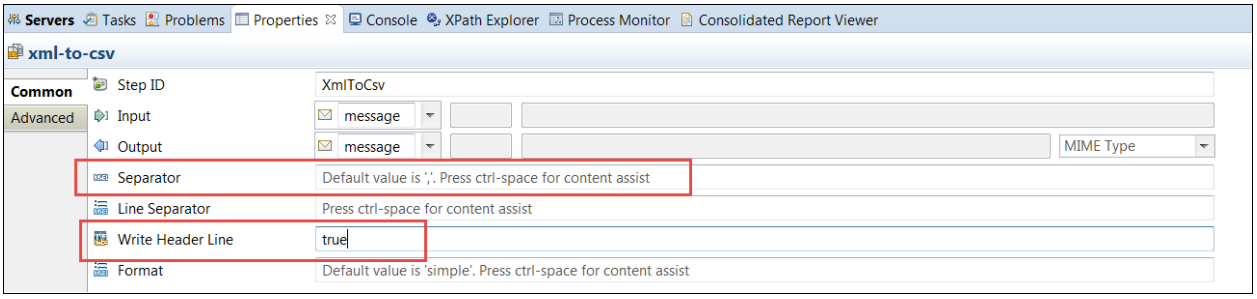
In this activity you will modify the MyHelloRaas Assembly and add an xml-to-csv step to convert the xml mediation message to a CSV output file.. Your final assembly will look something like this:



Open the Assembly Diagram in **MyHelloRaaS**

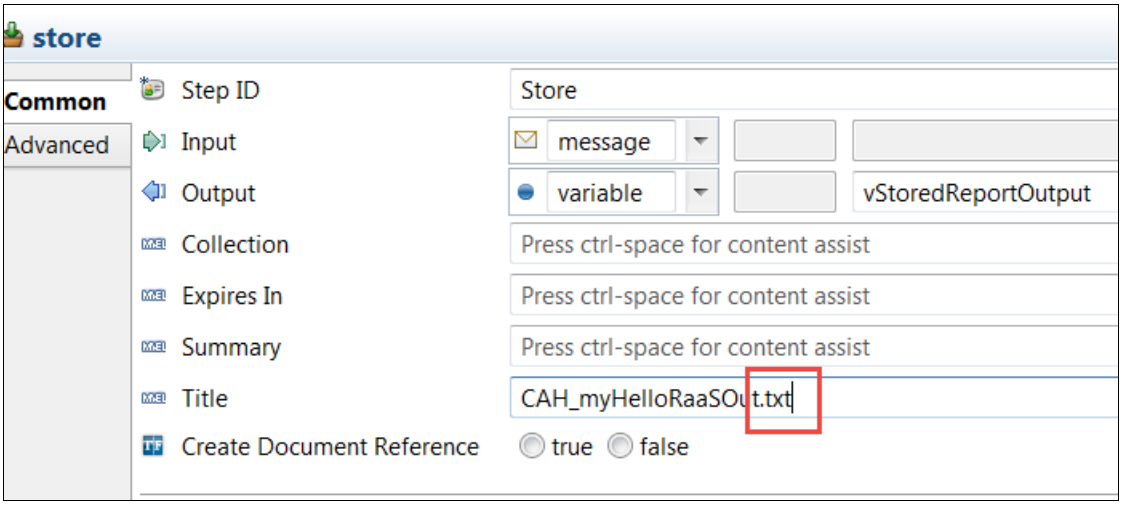
1. On the Assembly Diagram, Drag and Drop an **xml-to-csv** from the Palette and drop it into the existing **AsyncMediation** after the **XSLT+** step before the **Store** step.

2. Configure the **xml-to-csv** step by setting the **Write Header Line** box to **true**, and leaving the Separator box at the default



3. Add a log step after the **xml-to-csv** step to easily see the output in the log.

4. Change the file extension in the **store** step from **.xml** to **.txt**



5. Save the project, deploy and launch.

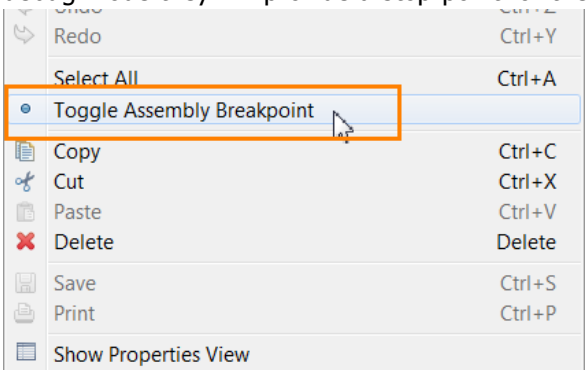
6. Review the output file

THE WORKDAY STUDIO DEBUGGER

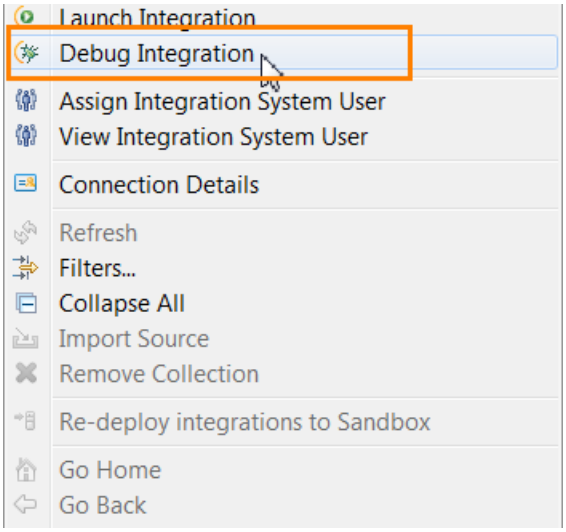
Workday Studio has a tool that will allow you to better understand programmatic flow and the state of the Mediation Context at any given point.

The Workday Studio Debugger will allow you to suspend execution of a process at any component and then step through the code, one item at a time, while being able to investigate the values of the Mediation Message as well as any variables or properties while execution is paused.

To indicate to the processor where to pause the run, toggle on one or more assembly breakpoints. These breakpoints will not affect a normal launch of the integration, but when the assembly is run in debug mode they will provide a stop-point for the debugger.

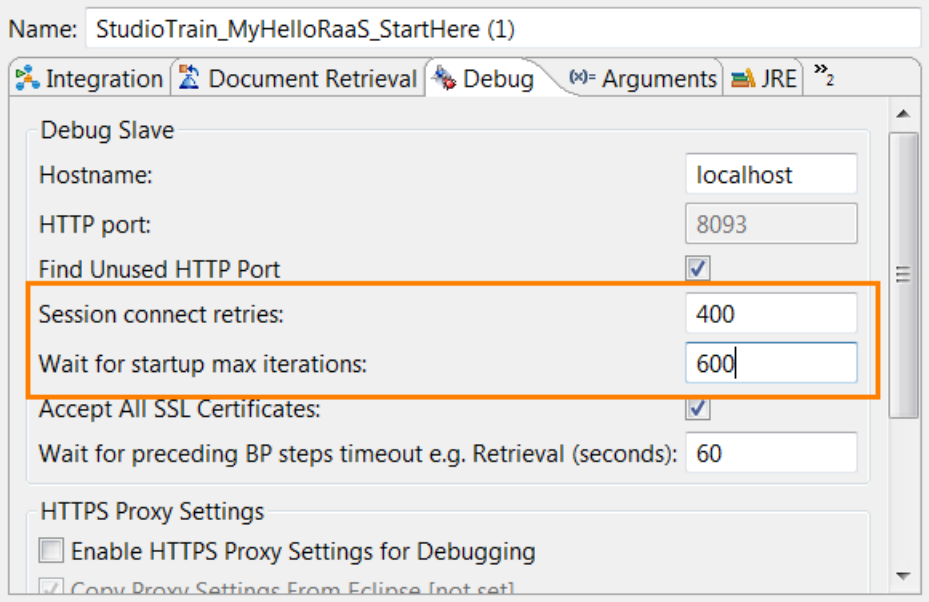


To launch an assembly in debug mode we will right click on the appropriate Workday-in transport on the Cloud Explorer and select "Debug Integration." The Debugger is different from a normal launch in that even though we begin the launch in the cloud, the Debugger runs locally using the local source code.

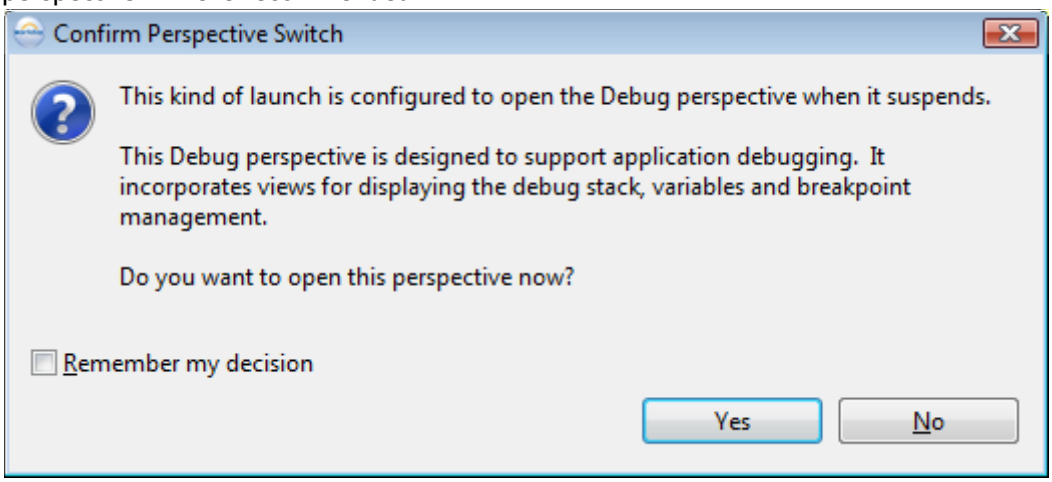


You may need to adjust two settings on the Debug Launch dialog. To reach them click Open Debug Dialog then click on the Debug tab. This will allow your local machine more time to establish communications with the tenant in the cloud and build a stable debug session. Generally you want to set

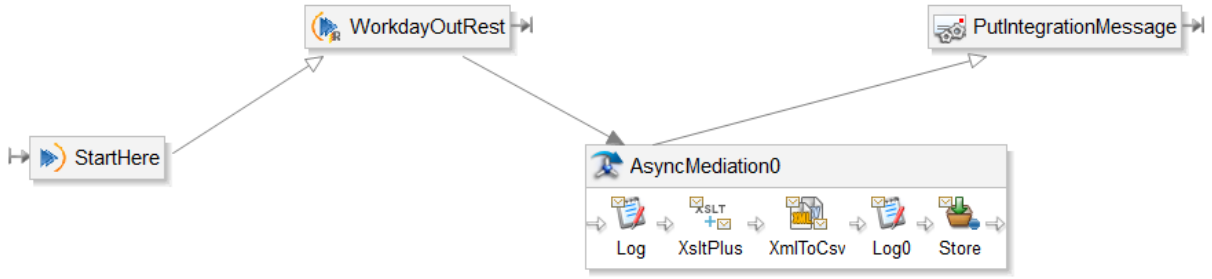
the Session Connect Retries to 400 and the Wait for Startup Max Iterations to 600.



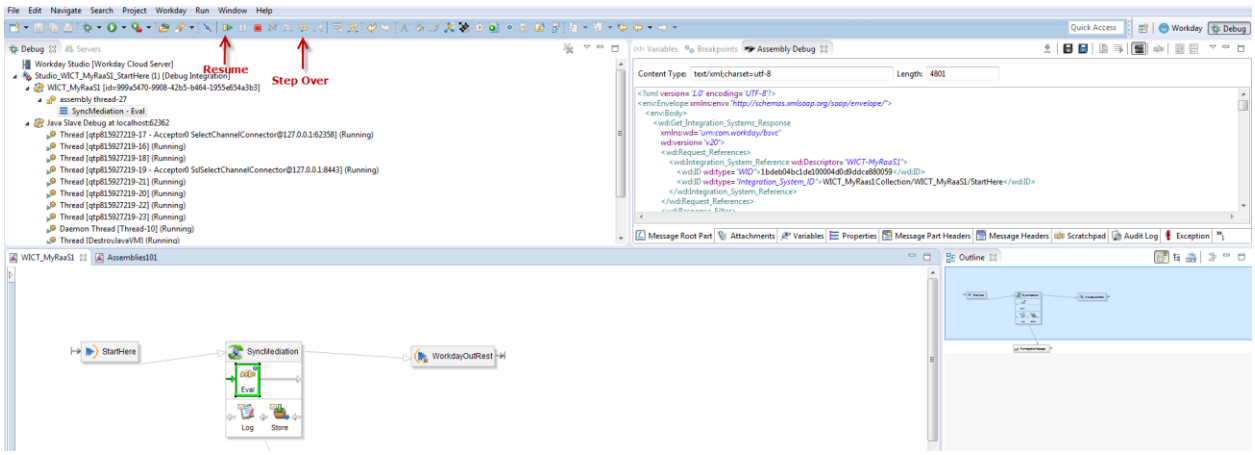
Once the Debugger reaches the first assembly breakpoint it will prompt to move to the Debug perspective. This is recommended.



The Debugger will highlight in green the step that it has suspended execution on. The Debugger will stop on the step but not execute the step until instructed to do so.

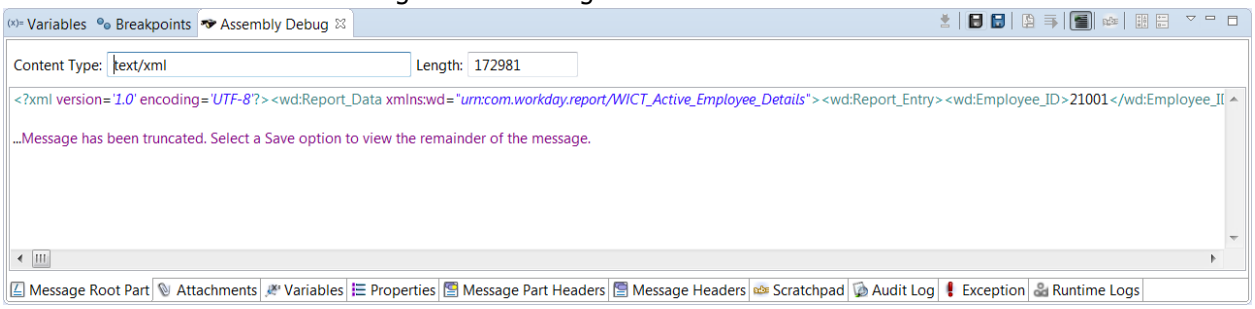


From here you may navigate forward through the Assembly using the Step Over and Resume buttons. Step Over will either execute the current, highlighted step or move to the next. Resume will restart the run and continue until the next breakpoint is reached.

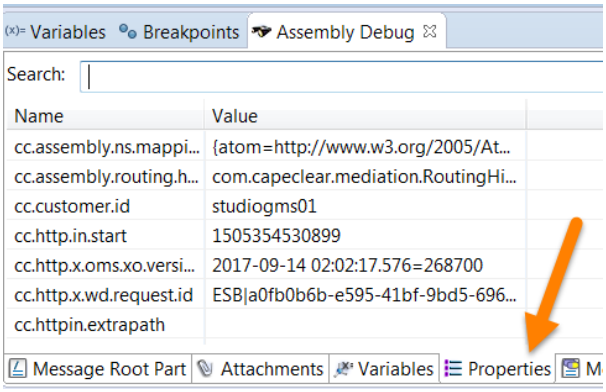


At any time with execution paused you can look at any value you wish using the Assembly Debug Tab.

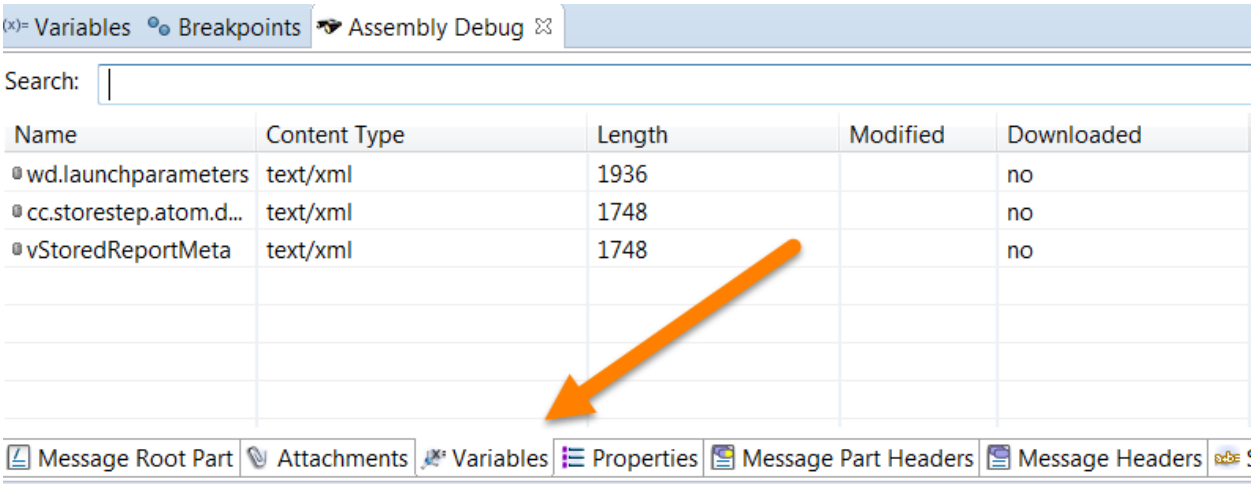
You can see the Mediation Message in the Message Root Part Section.



Properties and Variables can be seen on their own tabs. The Properties Tab looks like this, giving direct access to the property and the value:



The Variables tab only gives a view of the Variable names, but does allow you to save the Variable off locally to review its contents.

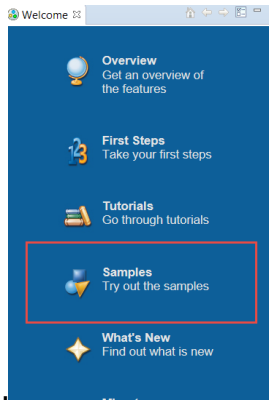


ACTIVITY 12 –USING THE WORKDAY STUDIO DEBUGGER

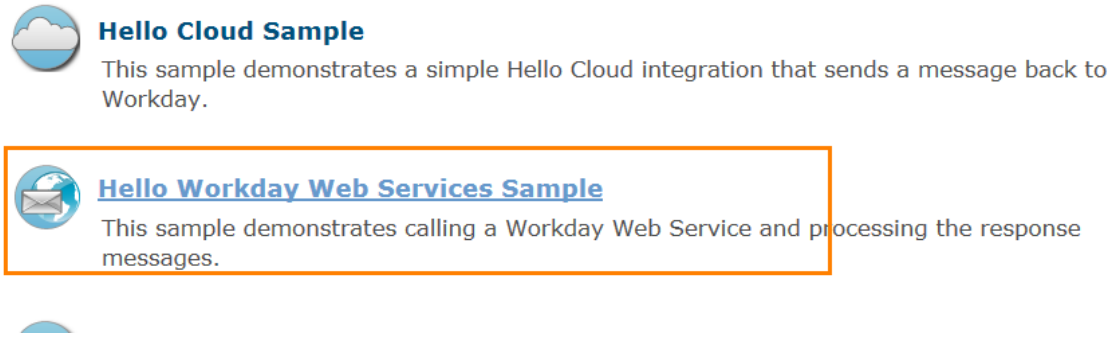
In this activity you will use the Workday Studio Debugger to investigate the HelloWWS assembly

Task 1: Get the HelloWWS Sample

1. Click Help>Welcome and Restore if necessary.



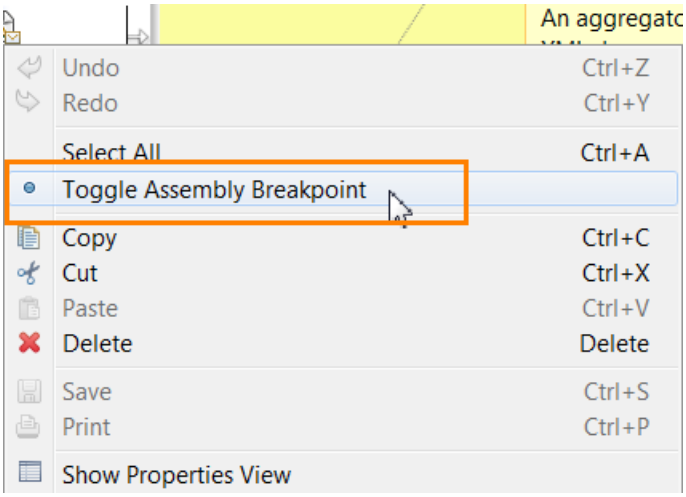
2. Click on the Samples Item.
3. Then Select the Hello Workday Web Services Sample



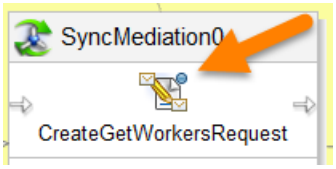
4. It will open a window downloading the HelloWWS project to your workspace. Click Finish
5. Deploy HelloWWS to your Training Tenant.

Task 2: Run HelloWWS in Debug Mode

1. Right click on the CreateGetWorkersRequest write step and Toggle on a Breakpoint



Notice the blue dot appearing in the upper right of the icon



2. Right click on the LogWorkerName Log step and Toggle on a Breakpoint.

3. Right click on the LogResults Log step and Toggle on a Breakpoint.

4. Open the Collection and Project in the Cloud Explorer.

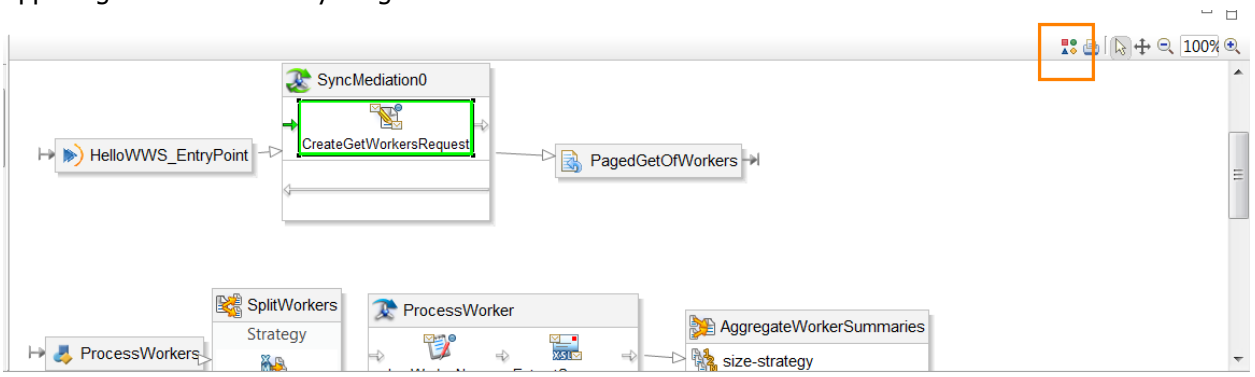
5. Right Click on the Workday In in the Cloud Explorer and Select "Debug Integration."

6. Click on the Console tab and watch the console log scroll until the Assembly reaches the first breakpoint .

7. The Assembly will put up a window that asks if you want to change perspectives. Click Yes.

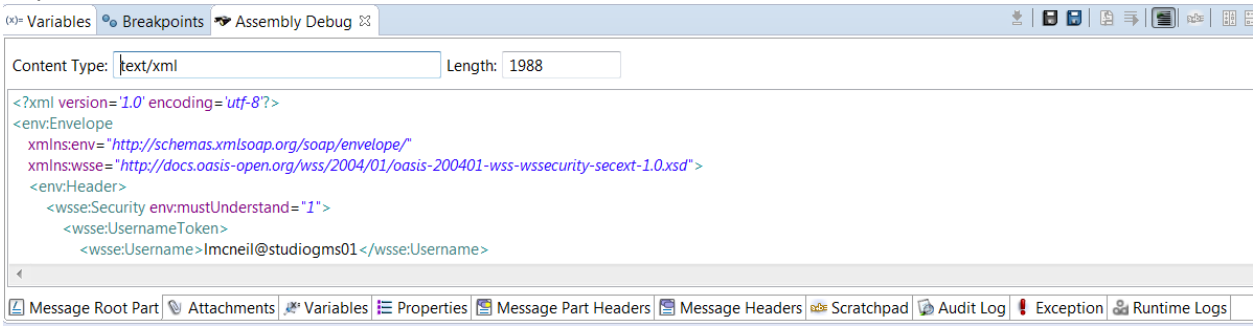
8. Notice the CreateGetWorkersRequest write step is highlighted in green. You may wish to toggle off annotations to make things easier to see. This is done using the toggle in the

upper right of the Assembly Diagram.



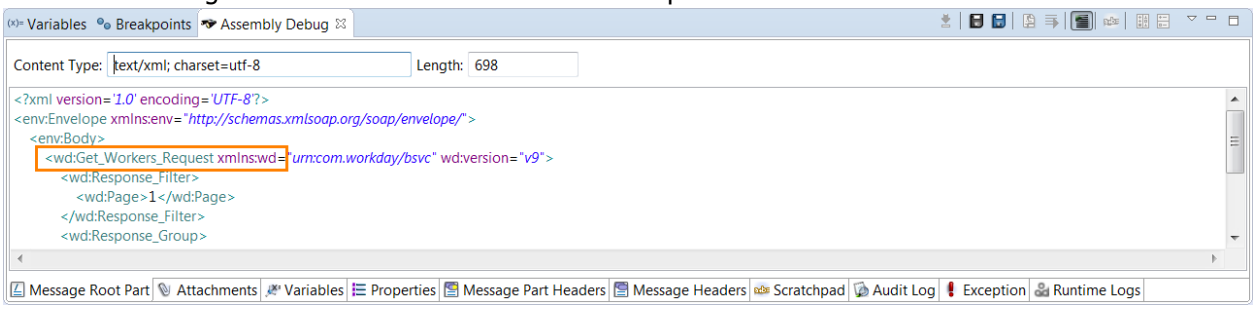
9. Go to the Assembly Debug tab in the upper right.

10. Look at the Message Root Part This now contains the response to the Workday In transport's requests.



11. Click Step Over (F6).

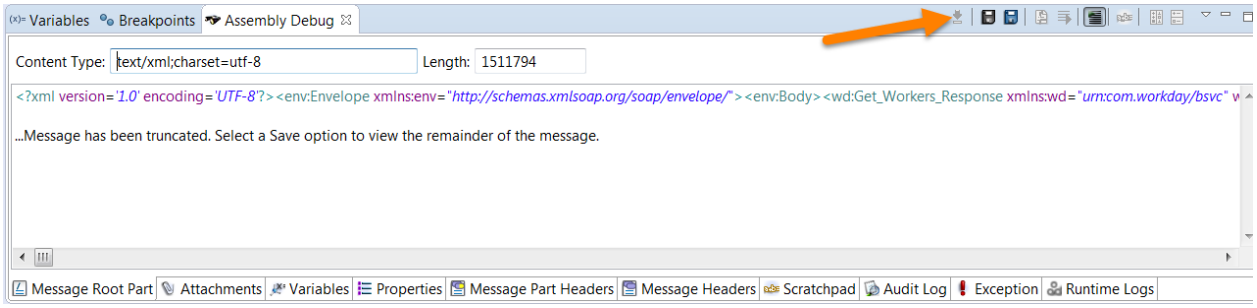
12. Notice the Message Root Part now contains the XML Request Document



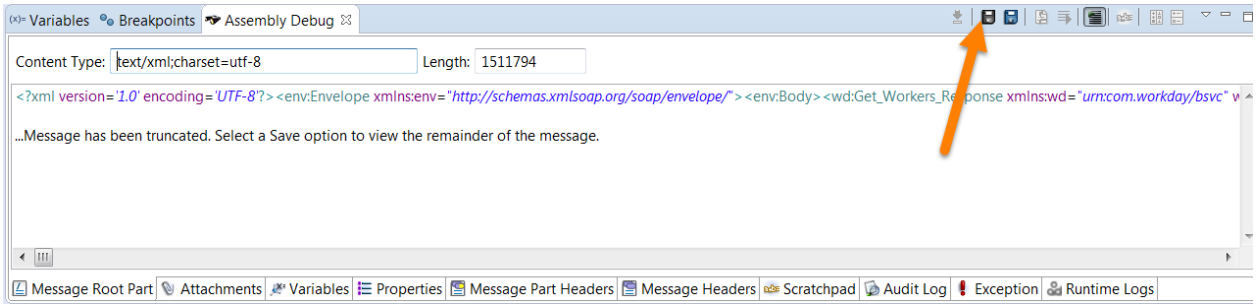
13. Click Step Over (F6) two times.

14. Notice that the green highlight square moves showing you the focus of the processing.

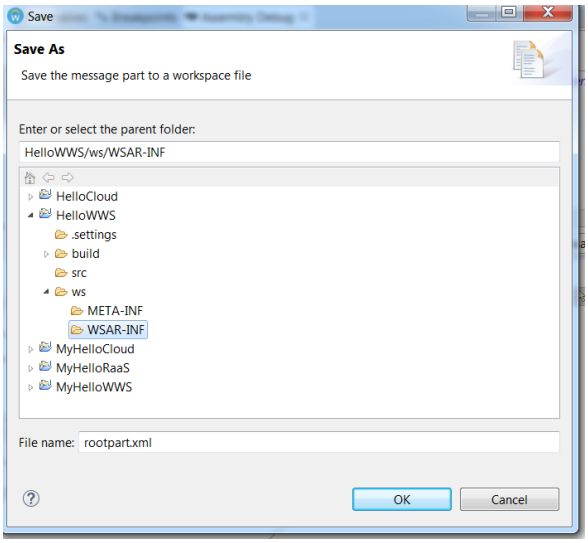
15. Notice the Message Root Part now contains one page of the XML Response Document. You may need to click 'Download the Item' to see it.



16. Save a copy of the mediation message to your assembly using the save icon on the Message Root Part Tab.

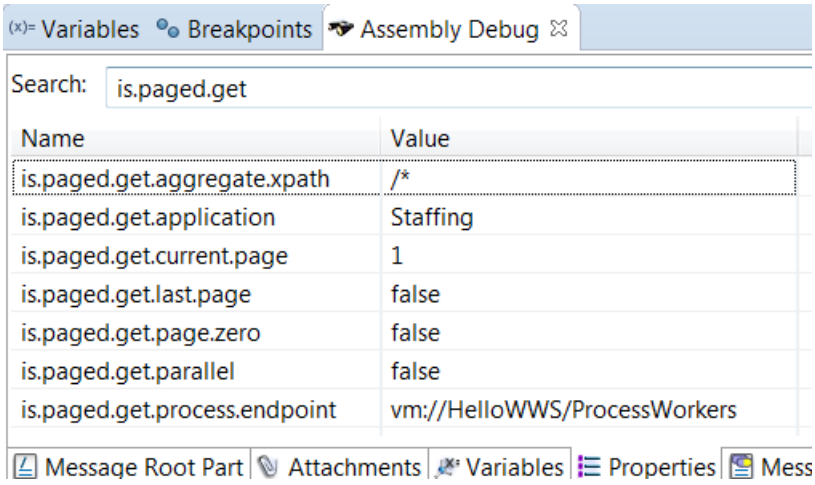


17. Save the File to the HelloWWS>ws>WSAR-INF folder



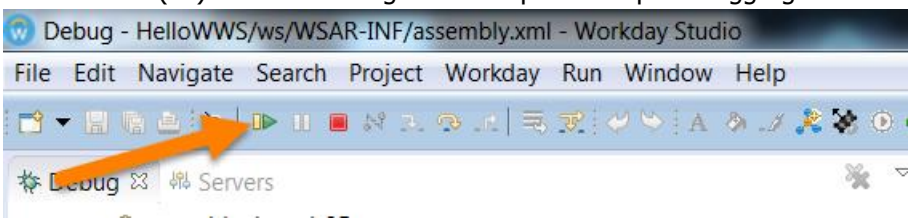
NOTE: In a live-data scenario please be careful to never allow personal or confidential information into your source code folders. Consider building a separate, non-deployable project to house these sorts of files.

18. Look at the Properties Tab. Notice the Properties all starting with "is.paged.get…" these are properties created by the Paged Get component.

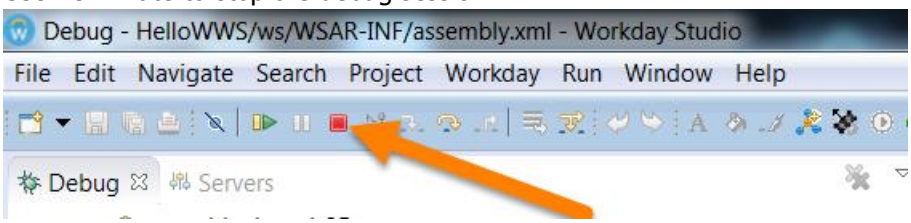


19. Use Step Over (F6) to move through a full loop of the Splitter Aggregator. Note how the Mediation message is changing.

20. Use Resume (F8) to move through a full loop of the Splitter Aggregator



21. Use Terminate to stop the debug session



STUDIO BEST PRACTICES

ASSEMBLY DIAGRAMS

Care should be taken when creating an assembly to arrange the components so that it is easy for another developer to understand how the assembly behaves.

- Use Swimlanes to easily organize your components into logical units and manage overall flow
- Arrange the assembly components so that messages flow from left-to-right and top-to-bottom.
- Break assembly components up into logical and functional concerns sub-assemblies (using local-in and local-out transports).

  This will aid readability of the main assembly flow and permits isolated and logical unit testing.
- Group related components using horizontal lines to separate functional areas of an assembly. Add notes as appropriate to individual steps to explaining important/complex processing.
- Add notes to individual components explaining anything of interest. This is particularly useful on routing steps are components where an "Execute When" condition has been added.
- Don't have arrows that cross each other or other components. Arrows do not have to be straight lines - bend them around other components if necessary rather than have them cross over them.
- Add more HTTP out or Workday out transports if that makes it easier to layout the components. If there are multiple calls to a single web service, it is better to have multiple out transports, then to complicate the assembly diagram by trying to route all messages to the one transport.

PROCESSING AND FILE-SIZE LIMITS

Workday terminates any Studio-built integration that:

- Takes more than two hours to process.
- Generates more than 3 GB of files during the integration run.
- Generates any single file larger than 250 MB during the integration run.
- Uses more than 6 GB of memory during processing.

There are a few other limits that you should be aware of:

- XPath operations are limited to messages sizes of 1 MB.
- Custom reports are limited to 2 GB.

- An integration's delivery and retrieval services cannot deliver or retrieve more than 1000 integration documents per integration run.

GENERAL GUIDELINES – PERFORMANCE

COMMUNITY- WORKDAY WEB SERVICES BEST PRACTICES: HTTPS://COMMUNITY.WORKDAY.COM/DOC/INT/JAS1382993320287

The most common issue encountered thus far is the two hour processing time limit on Studio integrations. There are a few common contributing factors:

- excessive web service calls
- excessive report invocations
- using inbound web service operations
- large volumes of data

REPORTS

Even more so than web services, custom reports have a very high per-invocation overhead. It is not unusual to have a report take a minimum of four seconds to execute and return data even for a trivial amount of information. For example, the execution of a report that returns information on a single worker may take four seconds to run. The same report returning information on 1000 workers may only take 30 seconds to run.

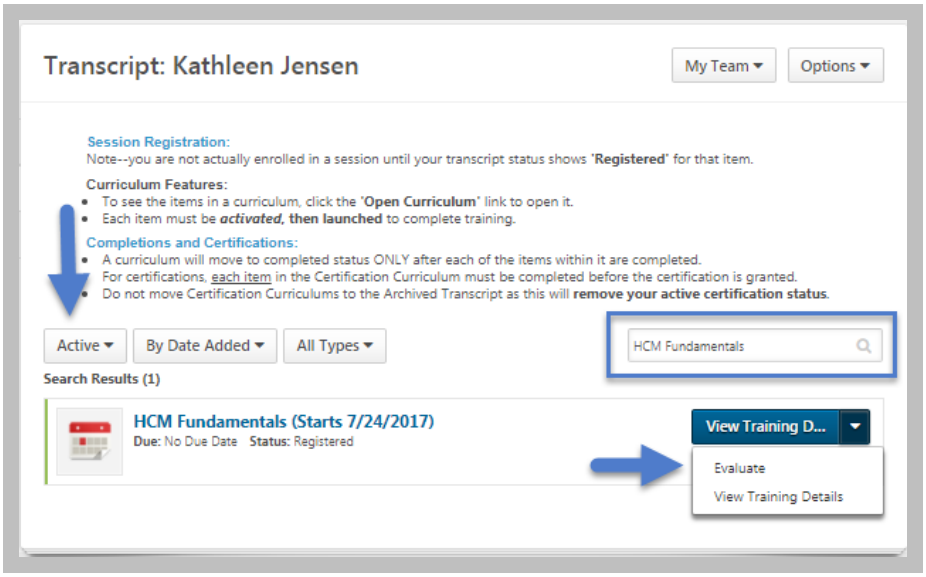
Below are some guidelines for avoiding long running integrations.

- Be aware that inbound web service operations currently tend to be slower than outbound operations.
- If you need to enrich data from a file before calling an put operation, for example by calling get_workers, then you should use the PagedGet common component to call the web service and retrieve all the data you will need at once, using paging.
- Always use paging when calling public web services.

APPENDIX A – CLASS EVALUATIONS

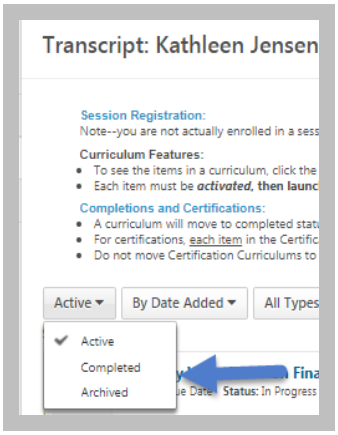
AVAILABLE AT THE START OF THE LAST DAY OF CLASS

1. Log in to the Learning Center: https://workday.csod.com
2. Select **View Transcript**.
3. Locate the training session in your **Active** tab. (Use the search field to quickly find your training session.)
4. Click the **View Training Details** pull-down menu and select **Evaluate**.



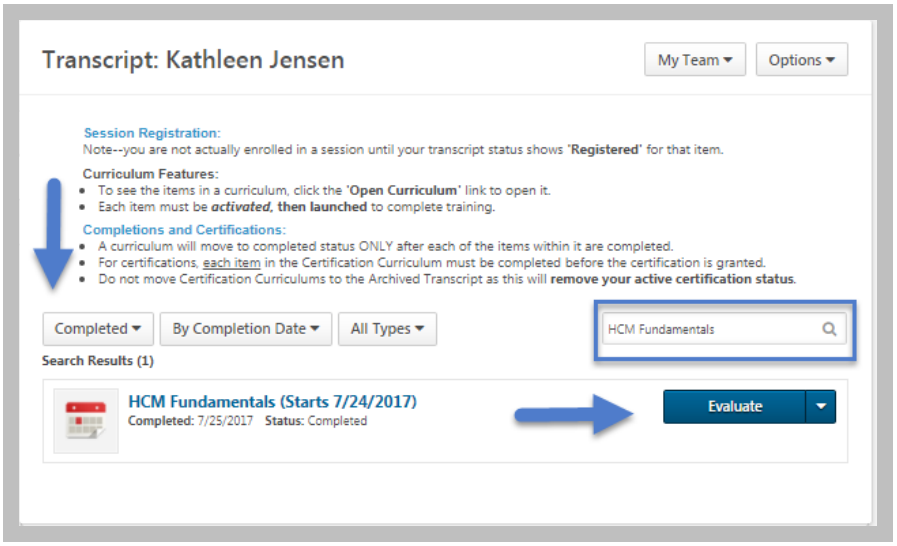
AVAILABLE AFTER CLASS ENDS AND ROSTER SUBMITTED

1. Log in to the Learning Center: https://workday.csod.com
2. Select **View Transcript**.
3. Select the **Active** tab to toggle to your **Completed** training.



4. Locate and select the completed training session. (Use the search field to quickly find your training session.)

5. Click **Evaluate**.



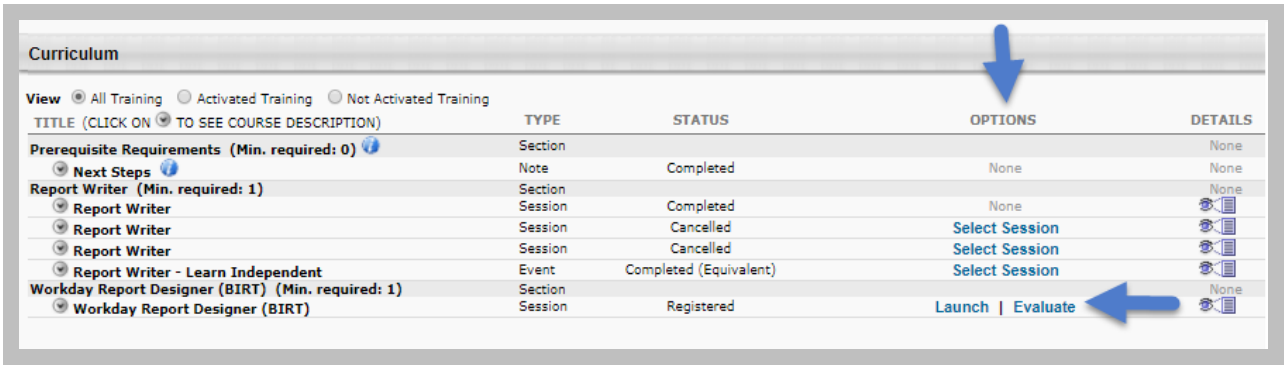
CLASS EVALUATION (SESSION WITHIN A CURRICULUM): AVAILABLE AT THE START OF THE LAST DAY OF CLASS

1. Log in to the Learning Center: https://workday.csod.com

2. Select **View Transcript**.

3. Locate the training session within the curriculum in your Active tab. (Use the search field to quickly find your training session and select the Curriculum Training Tile link to open the curriculum.)

4. Select **Evaluate** under the Options column.

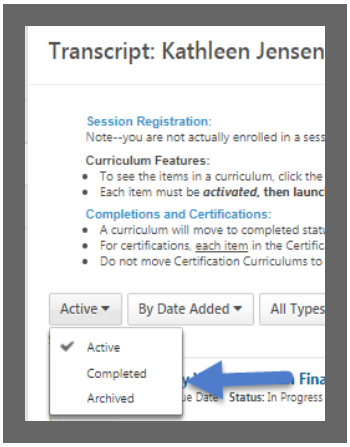


CLASS EVALUATION (WITHIN A CURRICULUM): AVAILABLE AFTER CLASS ENDS AND ROSTER SUBMITTED

1. Log in to the Learning Center: https://workday.csod.com

2. Select **View Transcript**.

3. Select the **Active** tab to toggle to your **Completed** training.



Note: If the curriculum is still Active, meaning the curriculum requirements have not been met, the curriculum will remain on the Active tab.

4. Locate and select the completed training curriculum. Select the Training Title link to open the curriculum and locate the session. (Use the search field to quickly find your training session.)

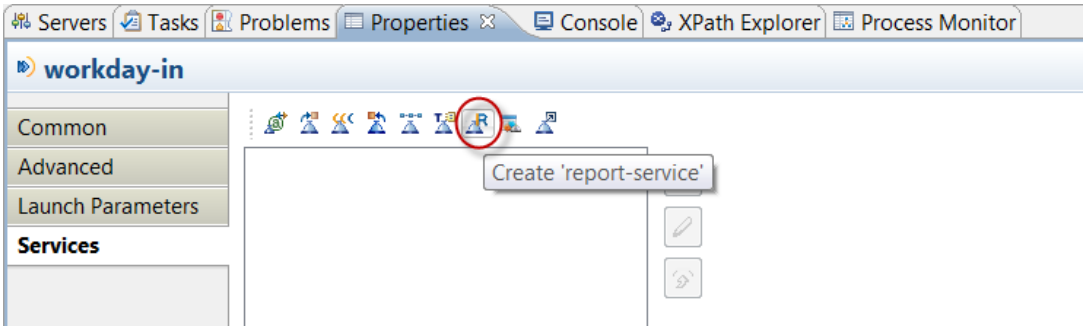
5. Click **Evaluate**.

APPENDIX B: DEFINING & CONFIGURING THE REPORT SERVICE FOR AN INTEGRATION

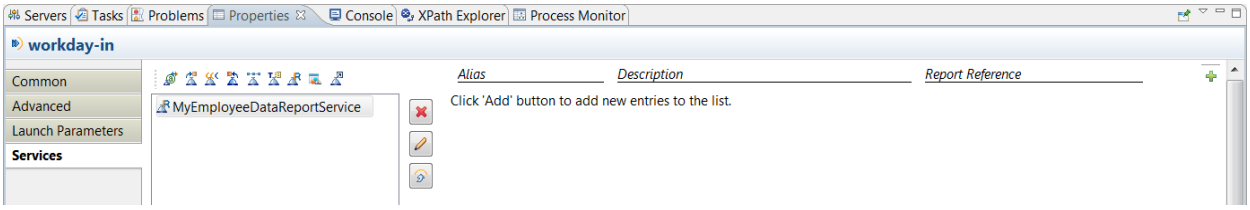
TASK 1: DEFINE THE REPORT SERVICE

1. Right-click the workday-in component, select **Show Properties View**, then select the **Services** tab.

2. Click **Create 'report-service'**.



3. Enter a service name, for example, *MyEmployeeDataReportService*, then click **OK**. The new report service is displayed in the left pane of the **Properties** view for the **workday-in**.

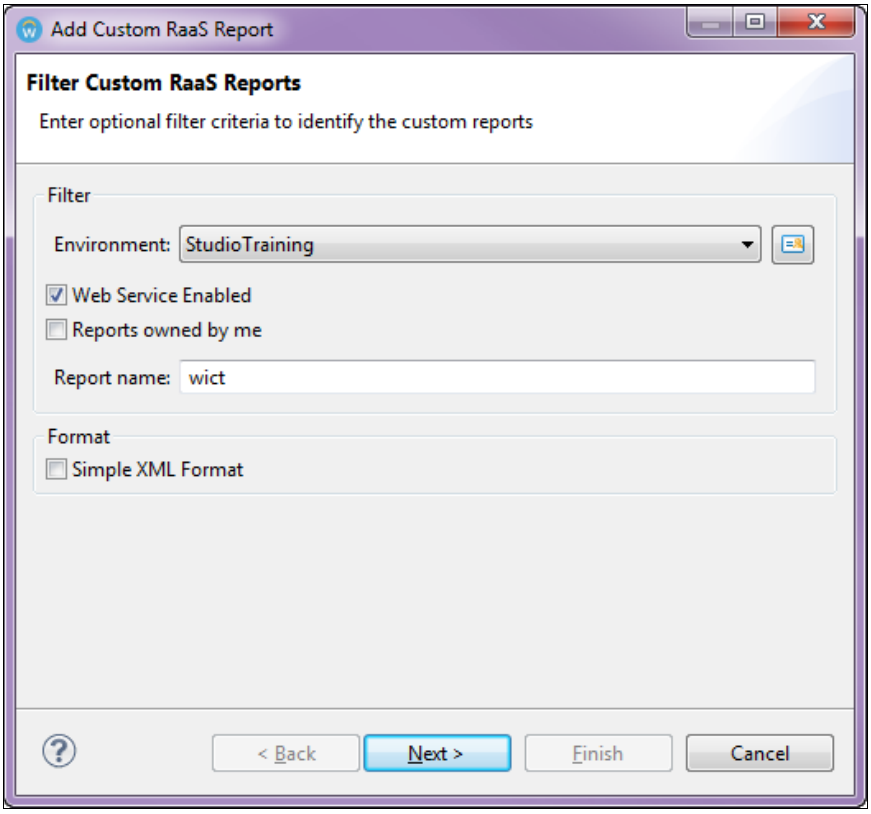


Each report service can have one-to-many report mappings, which you define as aliases on the right pane. To define an alias for a custom report, click the **Add New Entry** button to add a new entry to the list. For each alias that you add, specify the following details:

Alias: the alias name. This should be a meaningful name that refers to the custom report, such as the report name. Avoid using spaces in the alias. For example, if the report name is Active Employee Details, use Active_Employee_Details. This is just a symbolic name, an arbitrary string that is unique within the workday-in component's report service.

Description: a text description, for example: a Workday custom report of All Active Employees

4. Click the Select a RaaS Report button. The Add Custom Raas Report wizard opens.



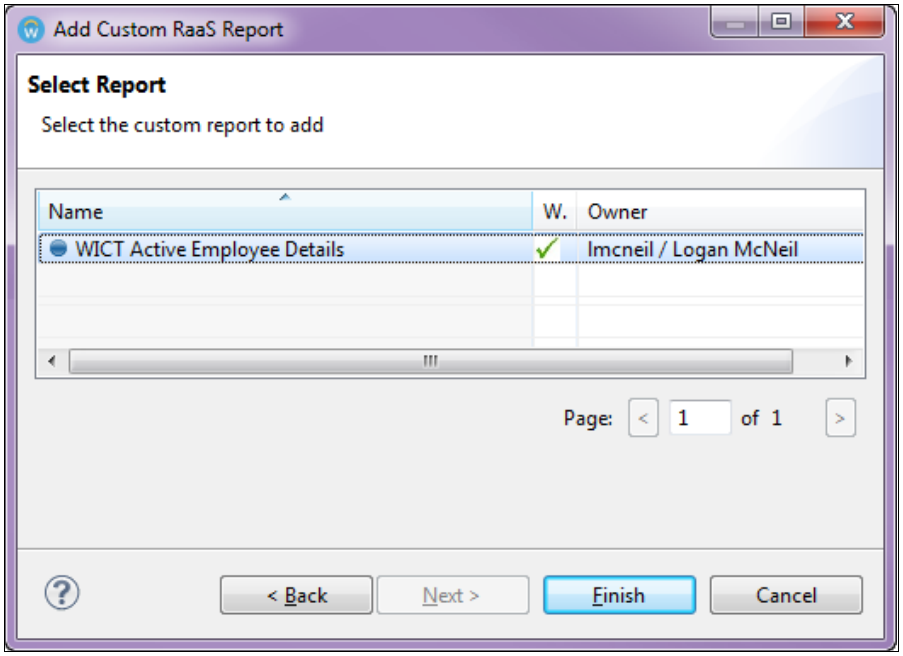
5. Select the Workday **Environment**, for example, Sandbox, then specify one or all of the following options to filter the list of reports returned:

    a. **Web Service Enabled:** the configuration of the report service requires that a Web-service enabled (RaaS) report is associated with each defined report name, so you should select this option to return only Web-service enabled reports.

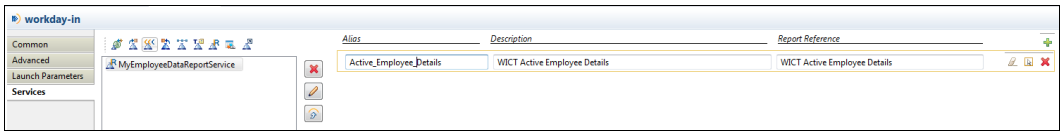
    b. **Reports owned by me:** select this option to return only reports that you own in Workday, that is, custom reports that you created, or that another user transferred to you in Workday using the **Custom Report> Transfer Ownership** related action.

    c. **Report name**: you can further refine the search by specifying the report name, or a partial name, to return a list of reports that match the search string.

    d. Click **Next**. On the **Select Reports** screen, select the report that corresponds to the alias that you want to add.



e. Click **Finish**. The report name is displayed in the **Report Reference** property field.



After you define report aliases within a report service definition on a workday-in, you can reference a report alias in the same assembly within a workday-out-rest component.

When you deploy the assembly, the Assembly Runtime uses this report service definition to create a report service object related to the integration system. When you launch the workday-in integration, the integration launch message includes a data structure that maps the alias to the extra path information for the report.

TASK 2: REFERENCING REPORTS USING THE WORKDAY-OUT-REST COMPONENT

On the **Properties** view for the workday-out-rest component, you can reference reports in either of the following ways:

Option 1: Using the **Add Custom RaaS Report** wizard to return a list of reports from Workday, if you have not yet added the report alias in a report service definition.

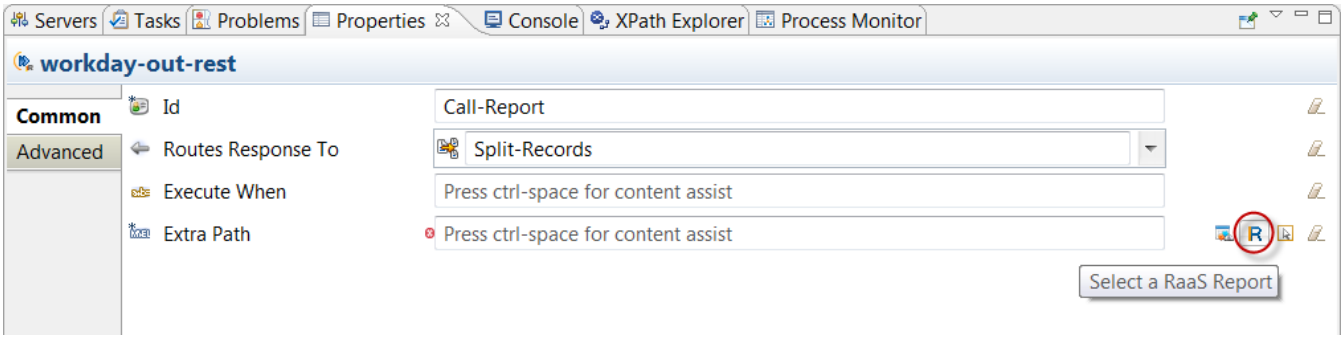
Option 2: Using the **Select Report Alias** dialog box, if you have already added the report alias in a report service definition

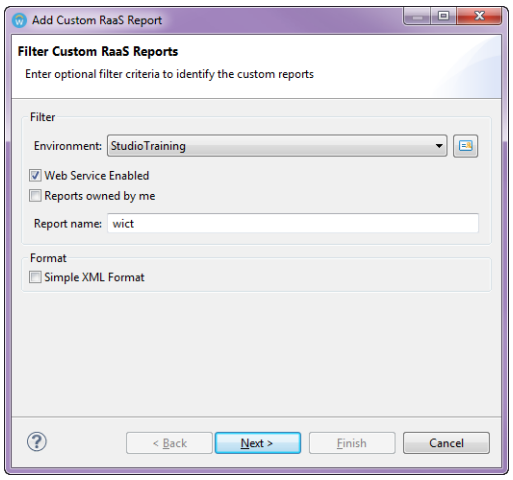
.

OPTION 1: USING THE ADD CUSTOM RAAS REPORT WIZARD TO RETURN A LIST OF REPORTS FROM WORKDAY

This wizard allows you to add report aliases to the report service definition after you select the report. It also populates the extra-path property value for the workday-out-rest component, which retrieves the report as an XML message using the REST API. To open the wizard, follow these steps:

6. In the **Properties** view for the workday-out-rest, click **Select a RaaS Report** beside the **Extra Path** property field.





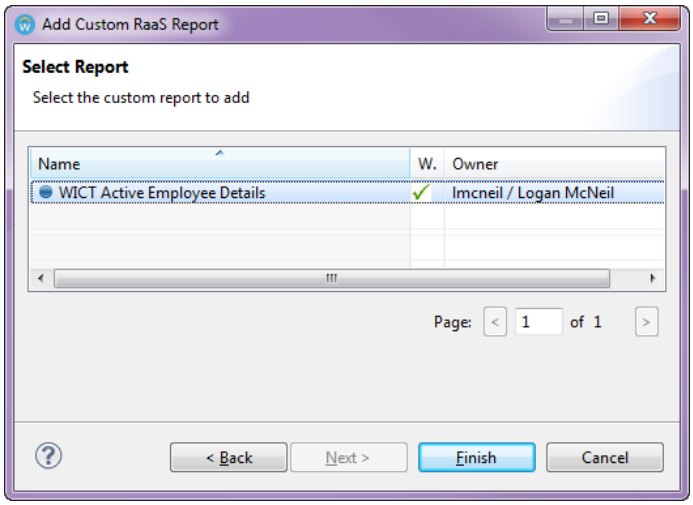
7. Select the Workday **Environment**, for example, Sandbox, then specify one or all of the following options to filter the list of reports returned:

    a. **Web Service Enabled:** the configuration of the report service requires that a Web-service enabled (RaaS) report is associated with each defined report name, so you should select this option to return only Web-service enabled reports.

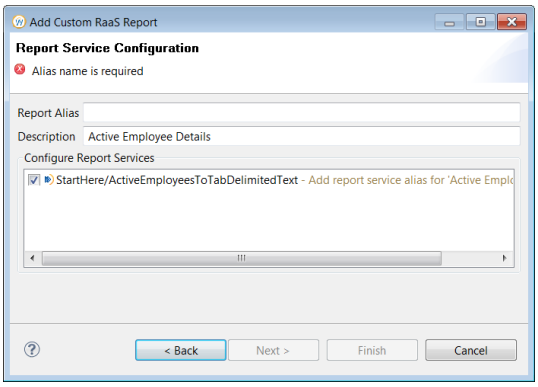
    b. **Reports owned by me:** select this option to return only reports that you own in Workday, that is, custom reports that you created, or that another user transferred to you in Workday using the **Custom Report> Transfer Ownership** related action.

    c. **Report name**: you can further refine the search by specifying the report name, or a partial name, to return a list of reports that match the search string.

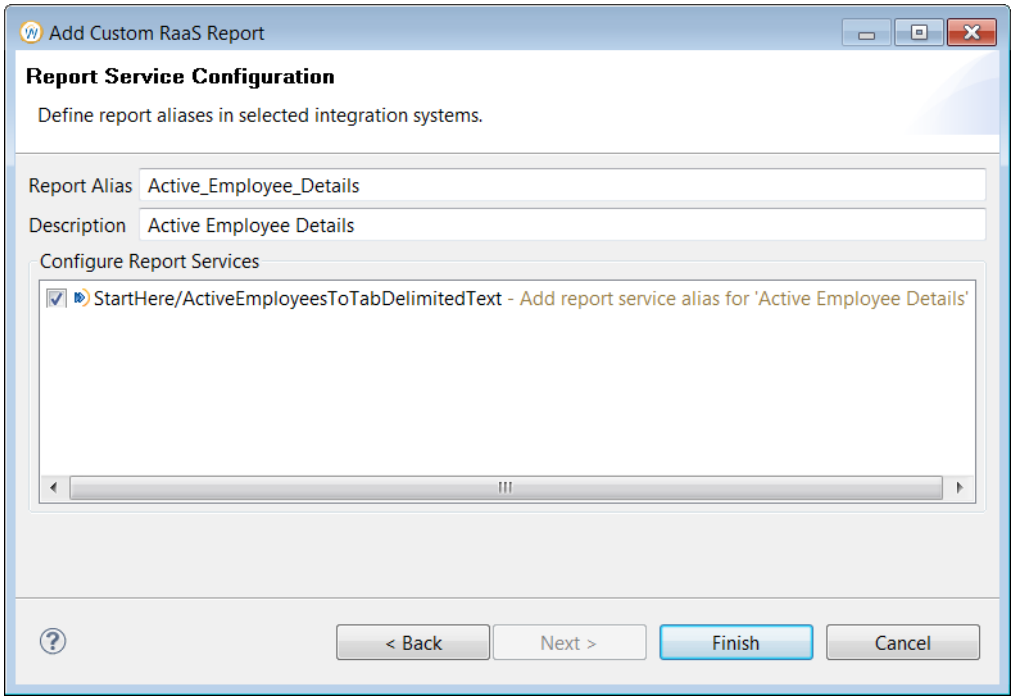
8. Select the custom reports to add.



9. Click **Next**. The **Report Service Configuration** screen is displayed



10. In the **Report Alias** field, enter the alias name. This should be a meaningful name that refers to the custom report, such as the report name. Avoid using spaces in the alias. For example, if the report name is Active Employee Details, use Active_Employee_Details. This is just a symbolic name, an arbitrary string that is unique within the workday-in component's report service.

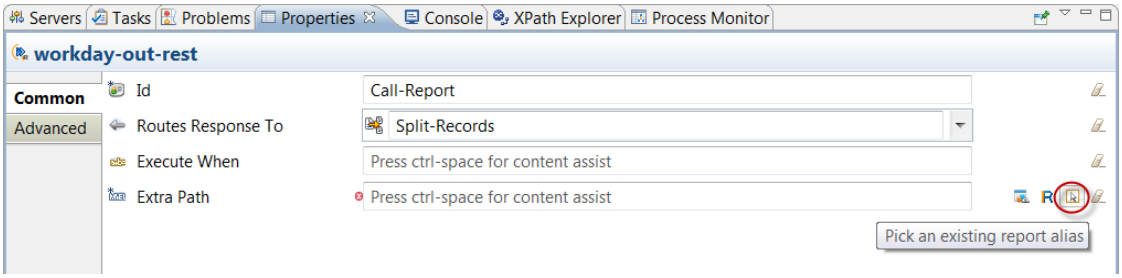


11. Click **Finish**.

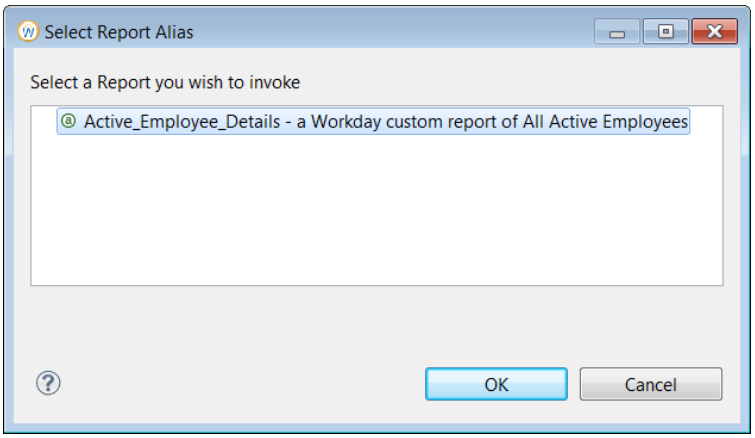
OPTION 2: USING THE SELECT REPORT ALIAS DIALOG BOX

After you define report aliases within a report service definition on a workday-in, you can reference a report alias in the same assembly within a workday-out-rest component. The workday-out-rest retrieves the report as an XML message using the REST API.

12. In the Properties view for the workday-out-rest, click Pick an existing report alias beside the Extra Path property field.



13. Select the alias for the report that you want to invoke, and then click **OK**.

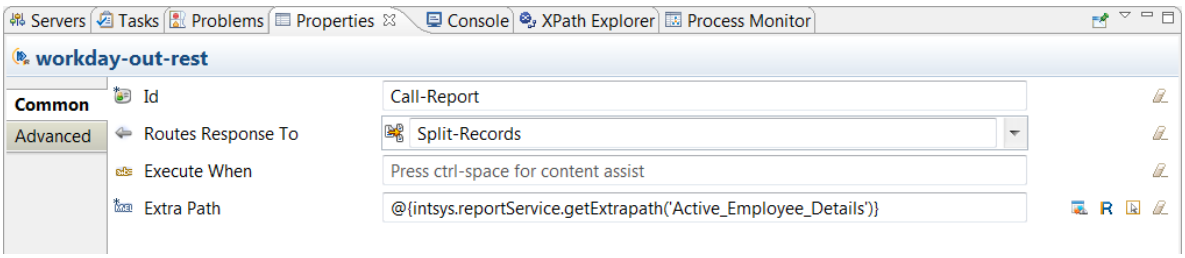


TASK 3: CONFIGURING THE EXTRA PATH

After you complete the steps described in Option 1 or Option 2, Studio inserts an MVEL expression in the Extra Path property field.

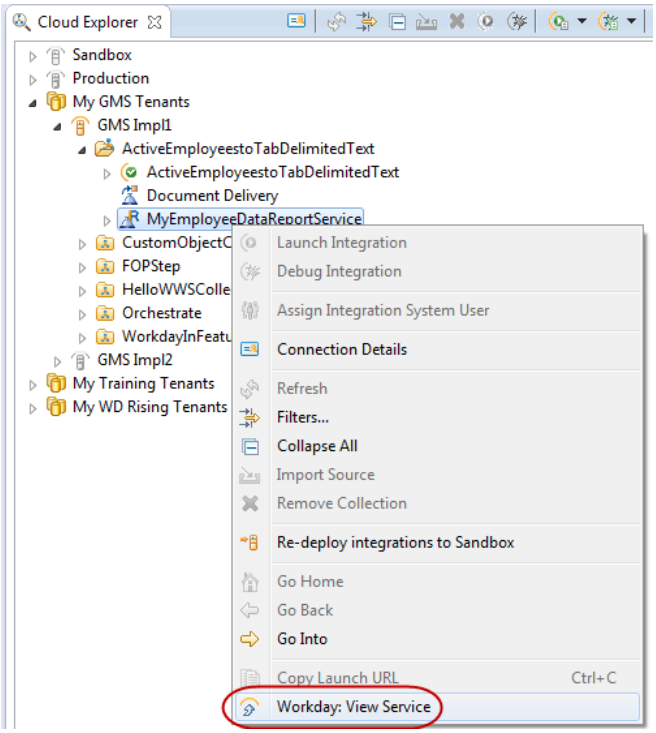
@{intsys.reportService.getExtrapath('*report-alias-name*')}

- *report-alias-name* is an alias that you defined for a report service in the assembly.
- For example:@{intsys.reportService.getExtrapath('Active_Employee_Details')}

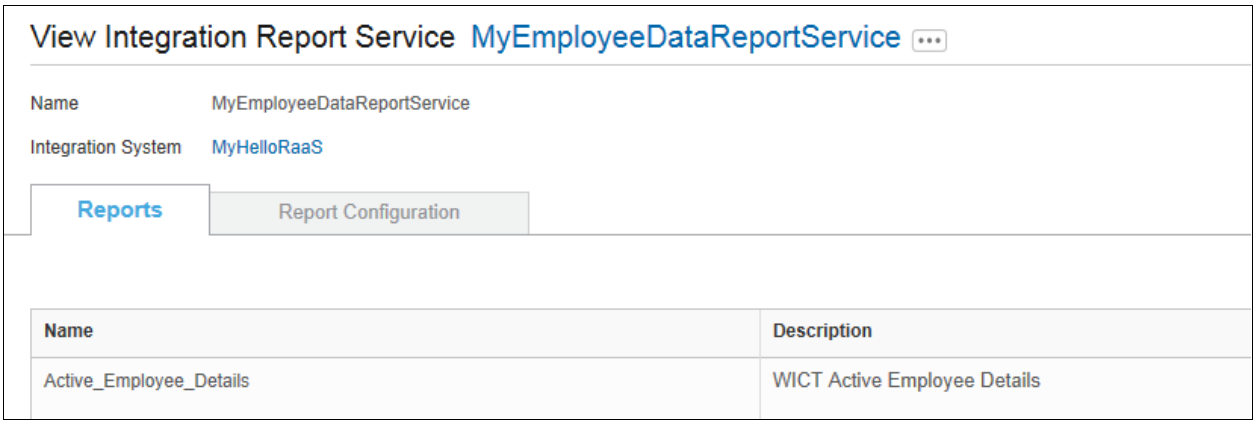


TASK 4: CONFIGURE INTEGRATION REPORTS FOR INTEGRATION REPORT SERVICE

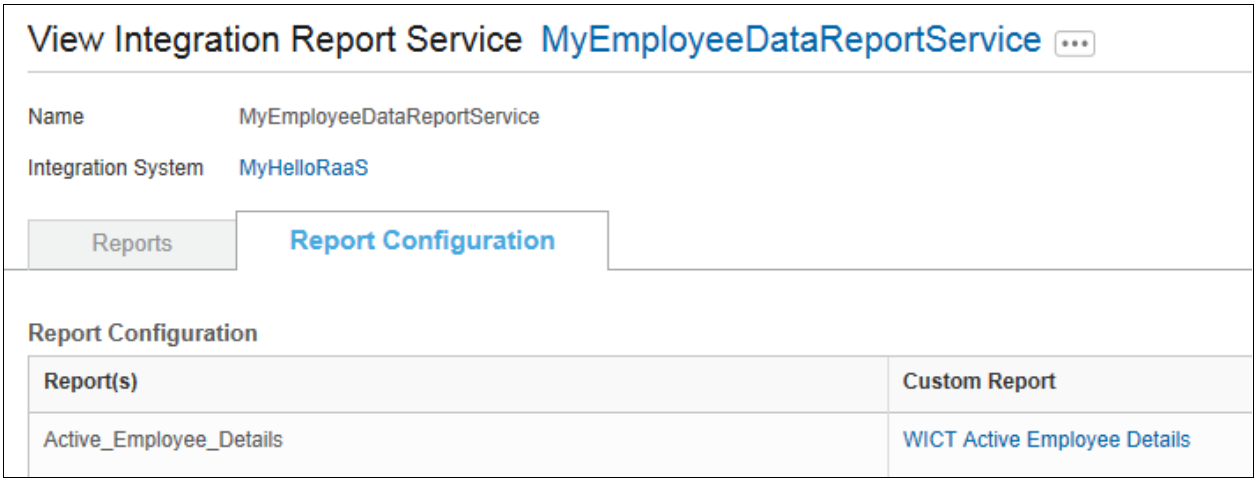
14. In the **Cloud Explorer**, right-click the report service node below the integration's workday-in, then select the **Workday: View Service** menu.



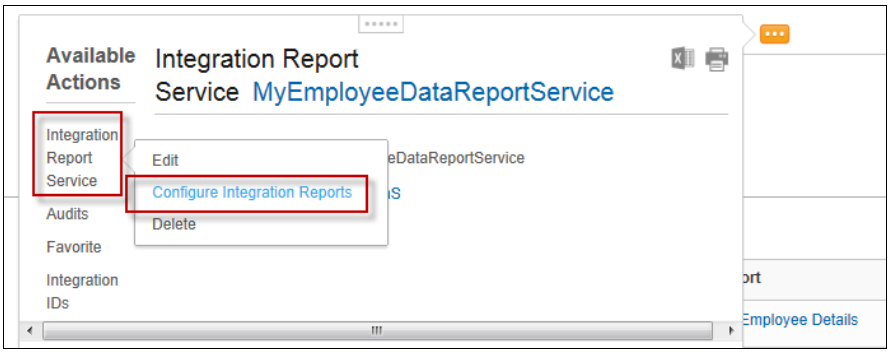
15. Access the Integration System in the tenant.



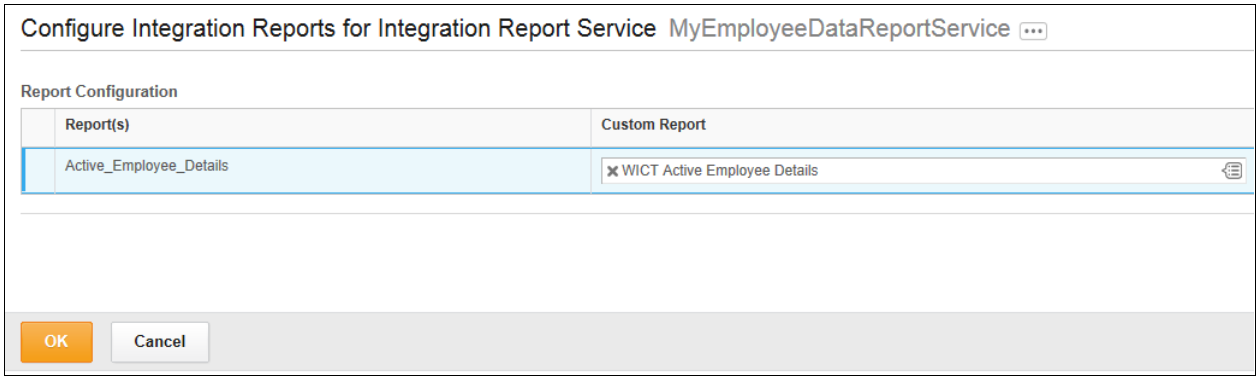
16. Click the Report Configuration tab.



17. You can configure the list of integration reports for the report service by selecting its **Actions** menu, then select **Integration Report Service> Configure Integration Reports**.



18. The **Configure Integration Reports for Integration Report Service** page is displayed, which displays the custom report that the report alias references.



19. You can reference a different custom report, if required, by selecting the report from the **Custom Report** prompt. To do so, click the prompt icon, then select the custom report entry.