workday. | EDUCATION

ducation resource is where learning

Please log in to your tenant:

Username: lmcneil

Password:

Advanced Workday Studio

Day 1

Workday Pro Overview

Workday Pro is an accreditation program for individuals with exceptional functional and technical expertise. Workday Pro is only available to customers.

Tracks

Workday Pro begins with specific tracks aligned to Workday functional areas.

Courses

Within each track are relevant courses, plus Touchpoints Kit training.

Written Test

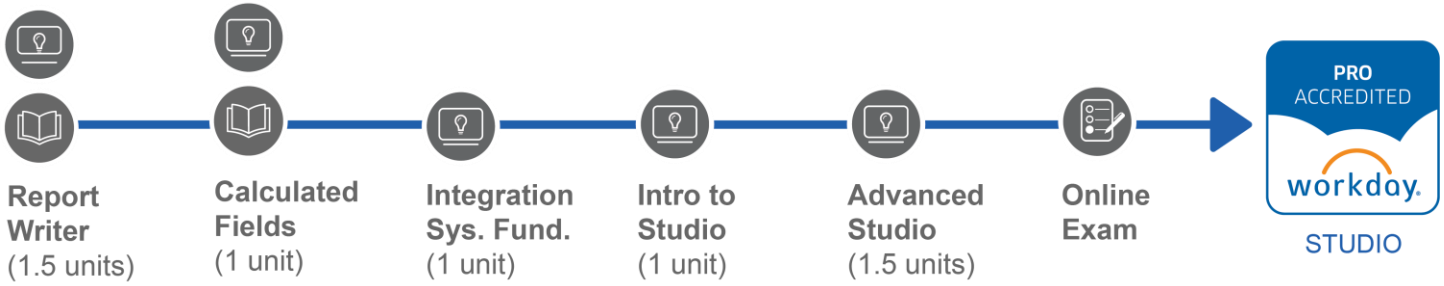
Participants must successfully complete a written test to be accredited.

Update Training

Update training is required twice a year to maintain the accreditation and will enable you to stay current with new feature releases.

Interested in becoming an accredited Workday Pro?

You've taken Advanced Studio. Here's what we recommend:



Learn More: community.workday.com/pro

Learn In-Person | Learn Independent | Learn Virtual | Online Exam | Learn On-Demand

Virtual Class Attendance

- Students are required to attend all class sessions to officially complete the course and receive transcript credit.
- The instructor will take attendance daily.
- Only registered students are allowed to attend class sessions.
  - Do not forward the WebEx link to colleagues.
  - Unregistered colleagues may not listen in or access your assigned training tenant.

What to Expect

Logistics

- Class hours
- Breaks & lunch options
- Participation

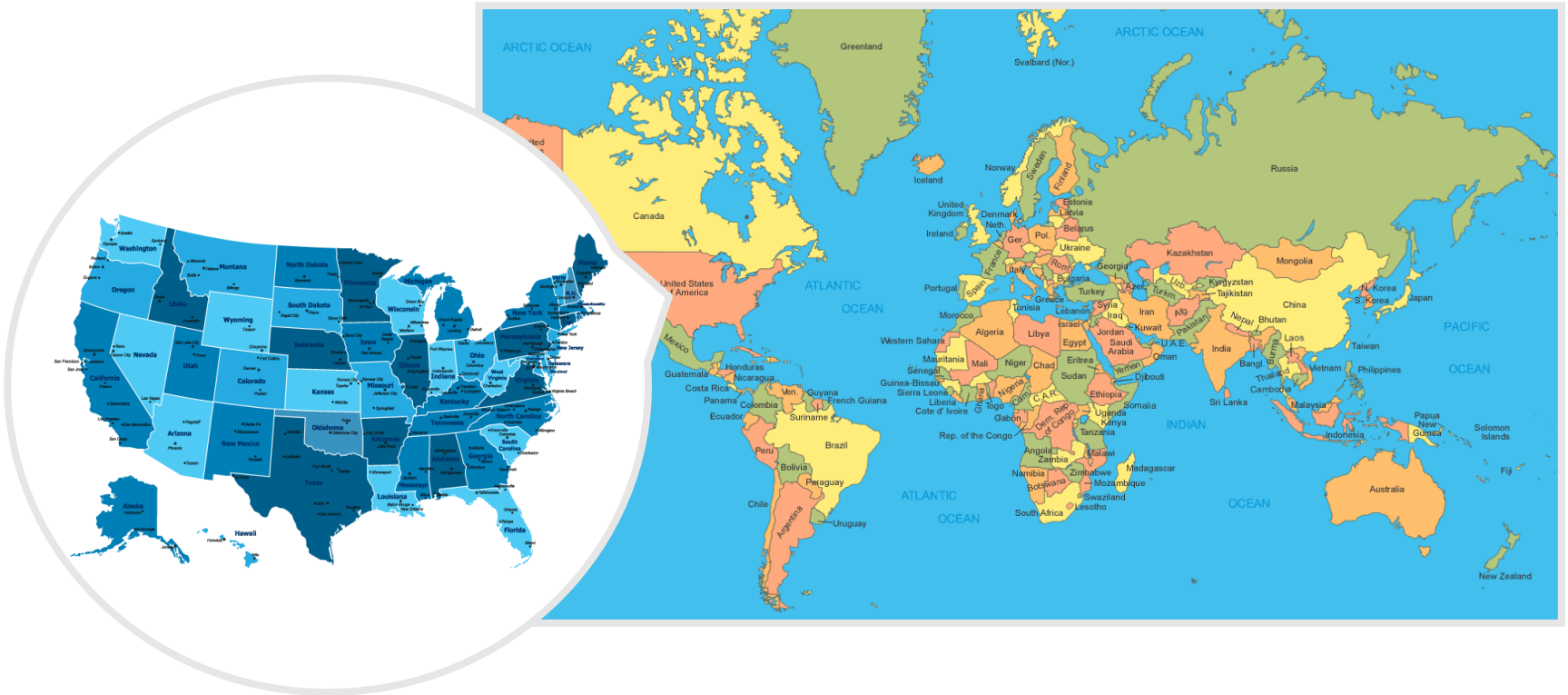
Structure

- Lecture
- Demonstrations
- Activities

Training Tenants

- Available 24 hours/day
- One tenant per student
- Logins provided by Instructor

Where in the World?



Agenda – Day 1

1. Review – Introduction to Workday Studio Training
2. RaaS with Launch Parameters
3. Mediation Context – Variables and Properties
4. Logging and the Workday Studio Debugger
5. Working with Collections

Agenda Day 2

6 Scalability

7 Workday-In Transport Services

8 Using Parameters in XSLT

9 PagedGet Component

10 Assemblies 101 – Homework

Agenda – day 3

10 Assemblies 101 – Review Homework

11 Route Component

12 Error Handling

13 Retrieval Service

Expectations

The class is designed for **developers** who have experience working with Workday Studio.

Note: It is expected that you can manage and navigate Studio without assistance.

This includes:

- Managing / switching workspaces
- Connecting to your tenant
- Importing / exporting .clars and existing projects
- Creating new assembly projects
- Setting properties of transports, components, and steps
- Configuring Delivery Service in the tenant
- Deploying and launching your assemblies

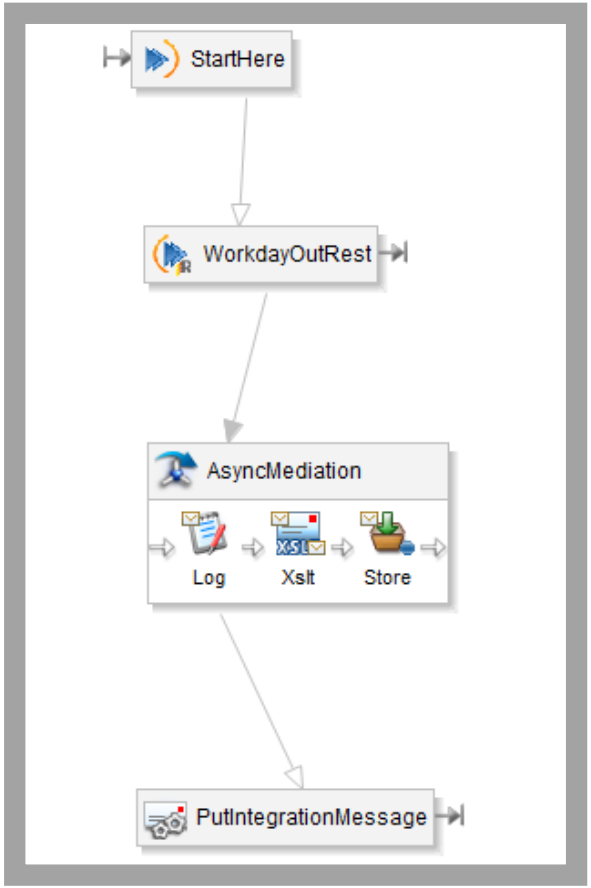
Review: Assemblies and Components from Introduction to Studio

workday. | EDUCATION

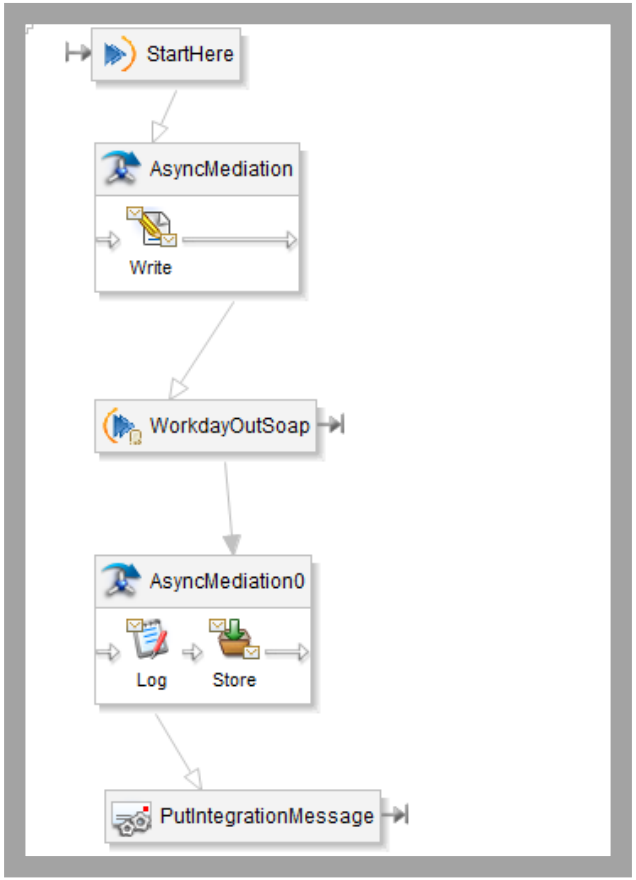
Learning Objective

- Review concepts and assembly components learned in Introduction to Studio.

Review MyHelloRaaS

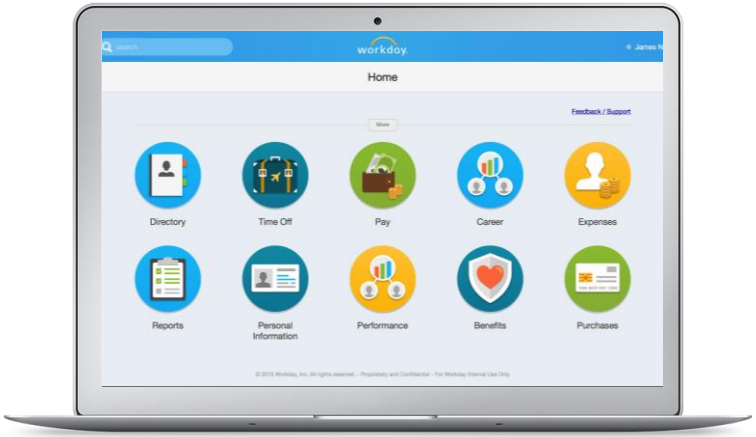


Review MyHelloWWS



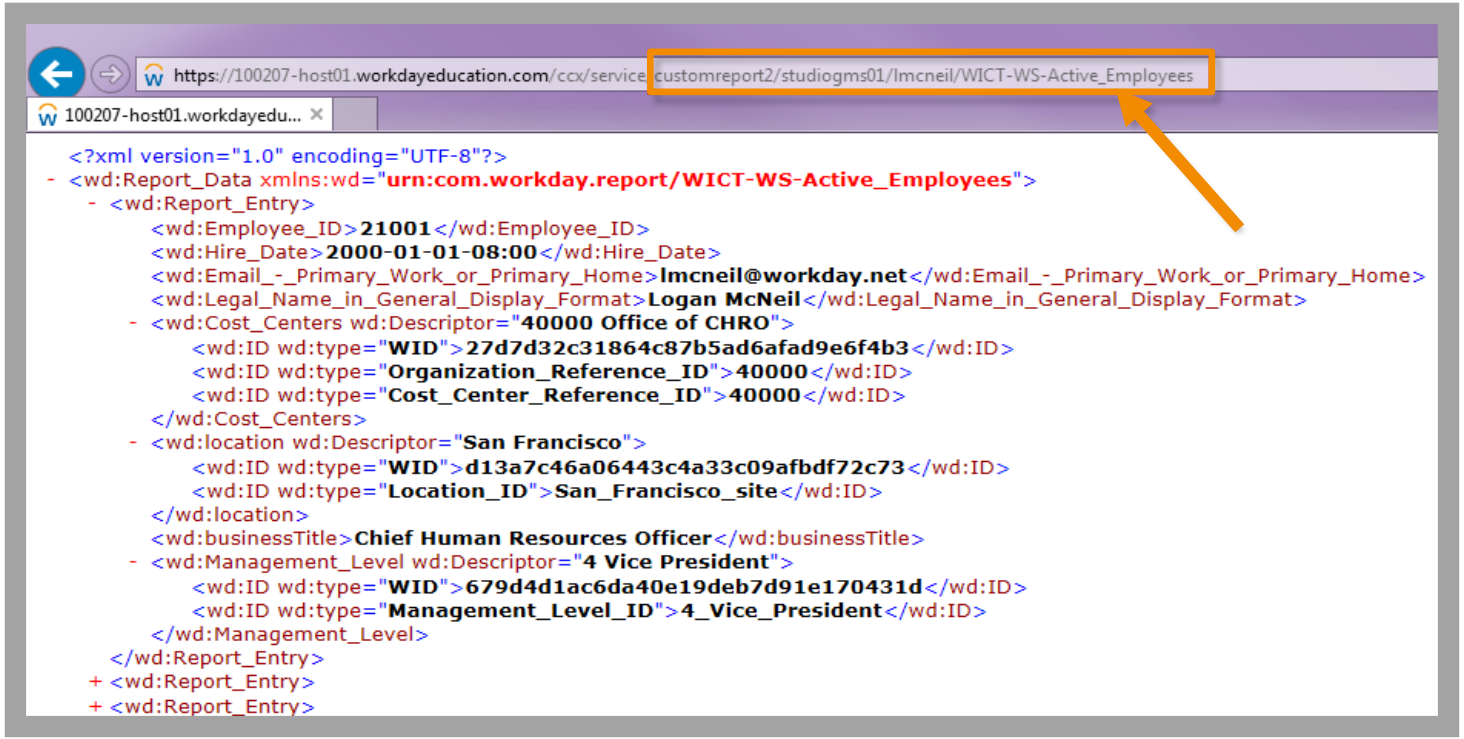
Demo – Review

Create RaaS Assembly

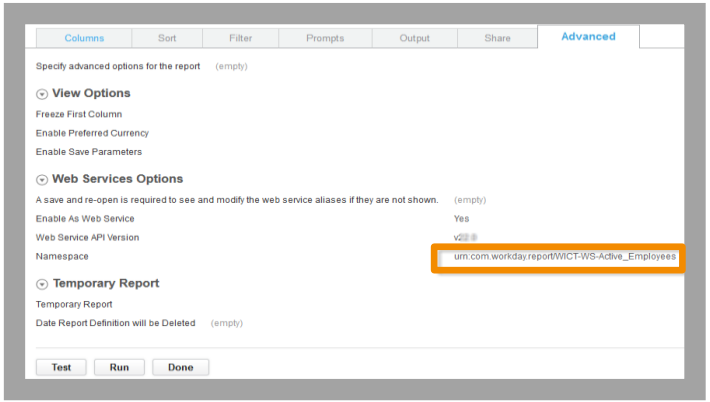


Obtain URL for Workday Out-Rest – Extra Path

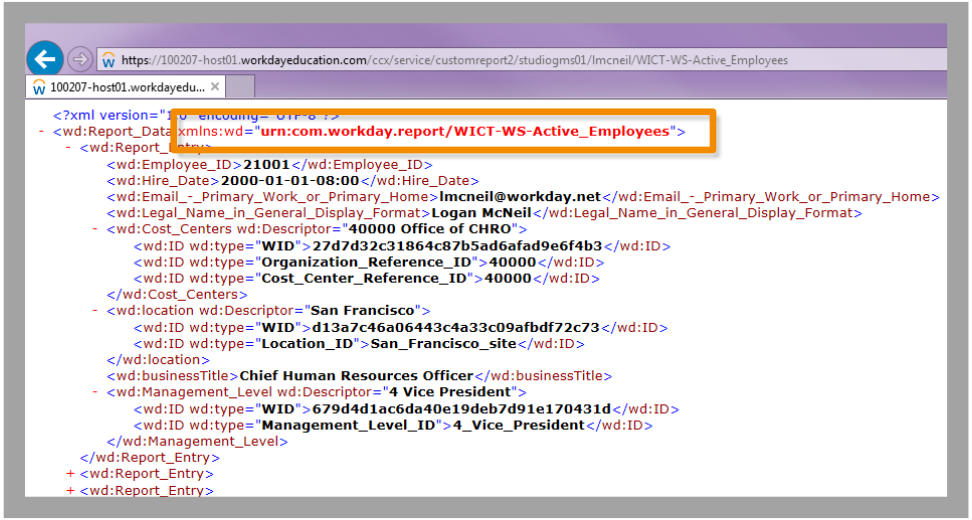
URL for EXTRA PATH: customreport2/<tenant>/<owner>/<report name>



Namespace of Report

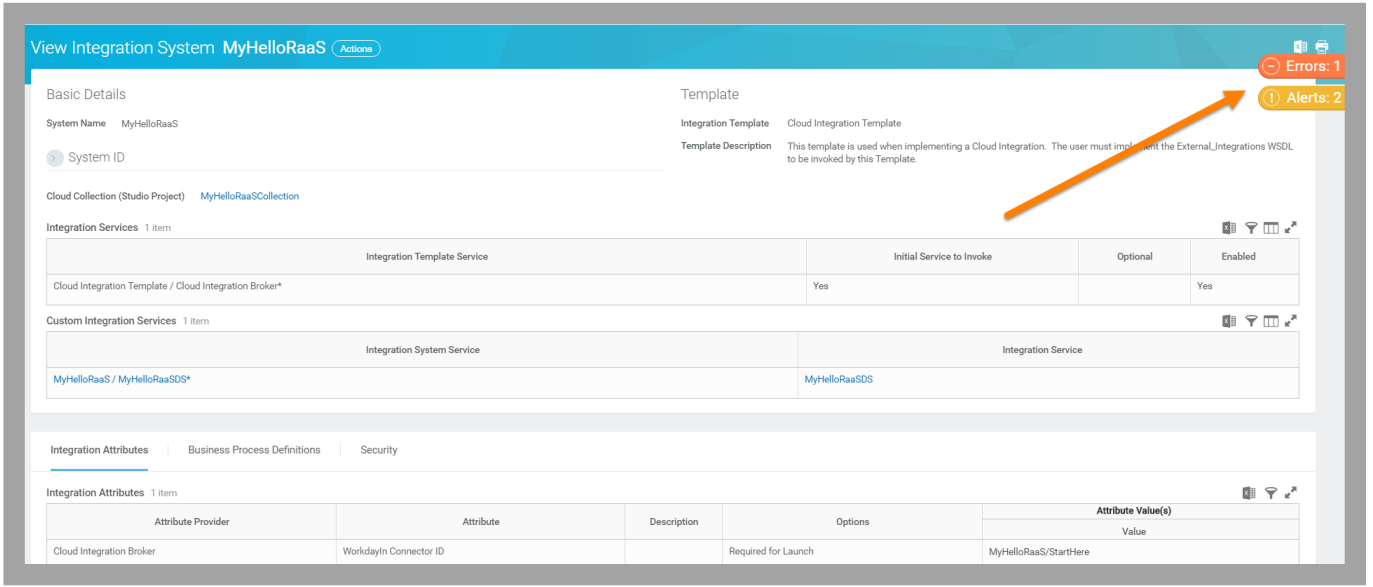


Note:
urn:com.workday.report/WICT-WS-Active_Employees

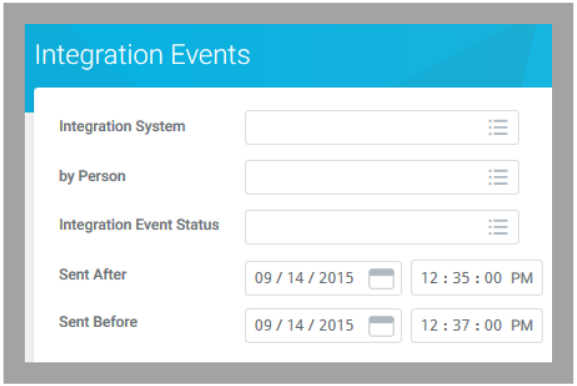


Delivery Service

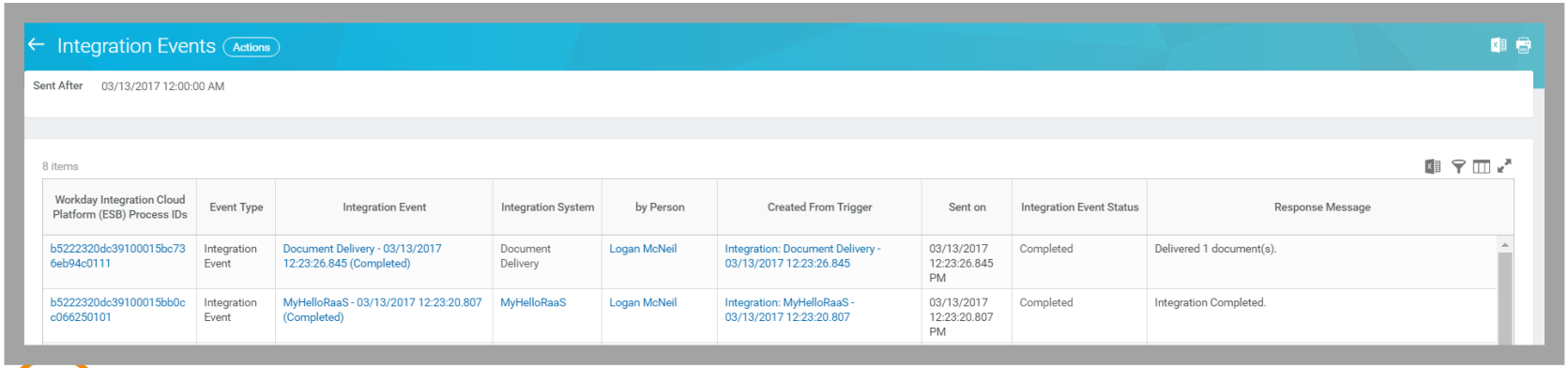
The delivery service is configured in the tenant. Notice the error message in the integration system.



Integration Events Report



You can view all the events associated with an integration by running the Integration Events report, then filtering results using a time range rather than an integration system name.



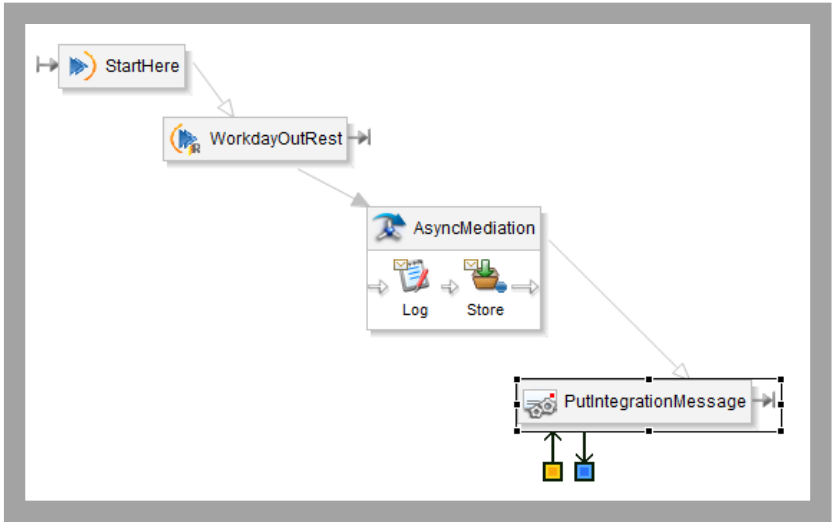
Workday Integration Cloud Platform (ESB) Process IDs	Event Type	Integration Event	Integration System	by Person	Created From Trigger	Sent on	Integration Event Status	Response Message
b5222320dc39100015bc736eb94c0111	Integration Event	Document Delivery - 03/13/2017 12:23:26.845 (Completed)	Document Delivery	Logan McNeil	Integration: Document Delivery - 03/13/2017 12:23:26.845	03/13/2017 12:23:26.845 PM	Completed	Delivered 1 document(s).
b5222320dc39100015bb0cc066250101	Integration Event	MyHelloRaaS - 03/13/2017 12:23:20.807 (Completed)	MyHelloRaaS	Logan McNeil	Integration: MyHelloRaaS - 03/13/2017 12:23:20.807	03/13/2017 12:23:20.807 PM	Completed	Integration Completed.

Activity 1

Create MyHelloRaaS Assembly

Estimated Time: 30 Minutes

- In the tenant, copy the Active Employees standard report as WICT-WS-Active Employees and web service-enable it.
- Generate the WorkdayXML and copy the extrapath for later use in the assembly Workday Out Rest. Create and test below assembly.



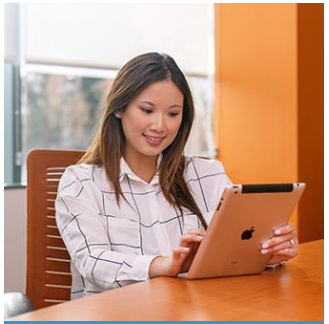
Introduction to MVEL: MVFLEX Expression Language

workday. | EDUCATION

Chapter Overview

- Introduction to MVEL
- Use report prompts with Assembly launch parameters to:
  - Implement the Simple Type launch parameter.
  - Implement the Reference Type launch parameter.

MVEL – MVFlex Expression Language

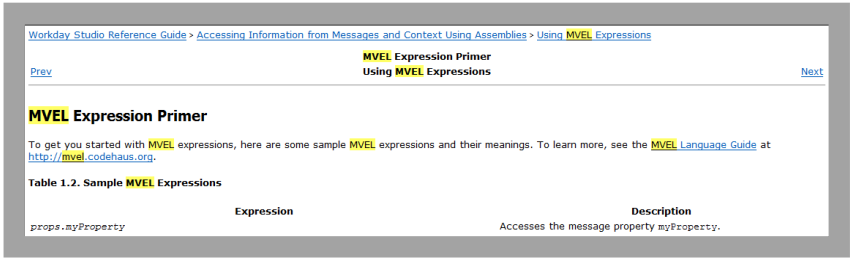


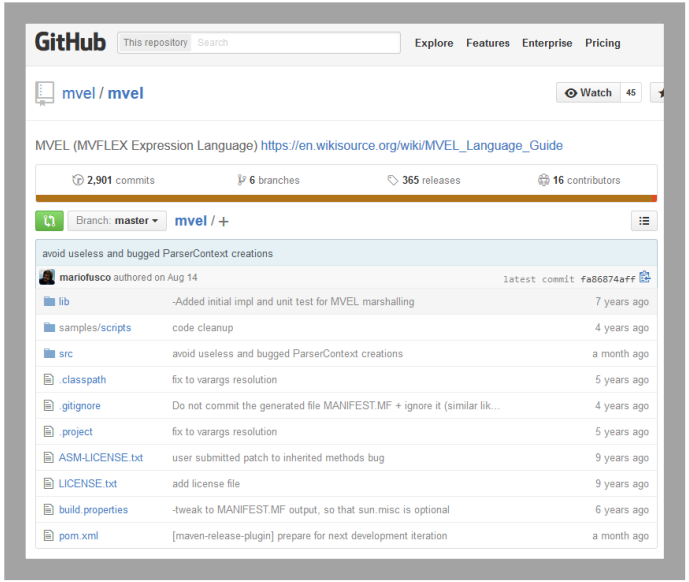
- The Workday Runtime Assembly Framework supports the use of MVEL expressions for inspecting the mediation context for a message part after processing by an assembly component.

Example: Parts[0] inspects the root message part, while parts[1] inspects the first attachment, and so on.

- MVEL provides a capability to auto-convert between objects. This can make manipulating variables, message parts, and even whole messages easier.

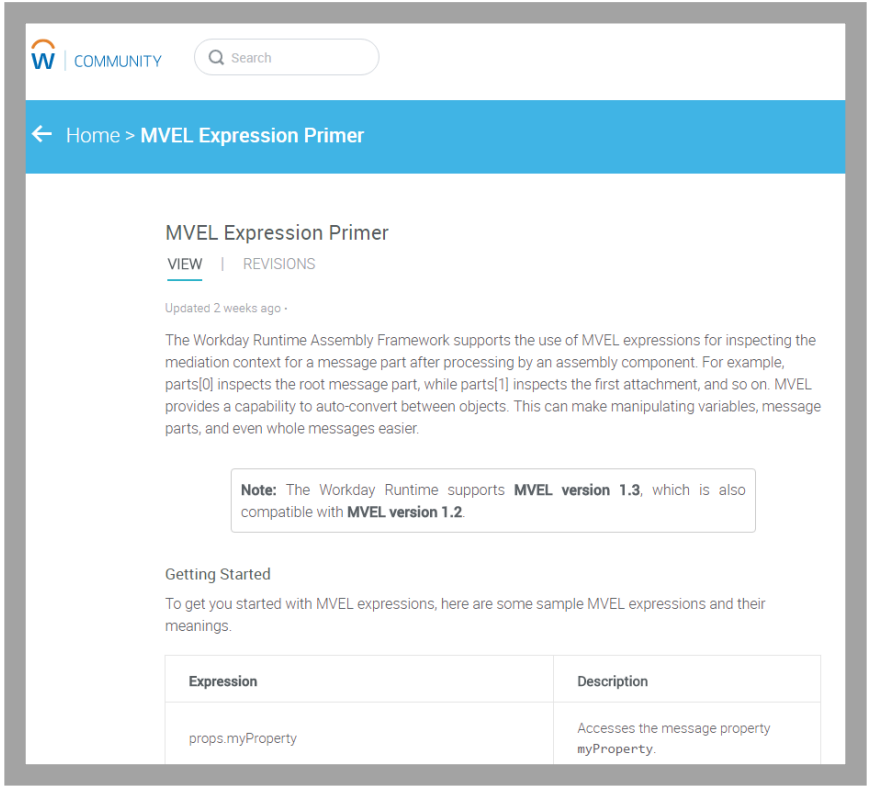
MVEL





- The Assembly Framework supports the use of MVEL expressions.
- See the MVEL Expression Primer in Workday Studio Help for usage examples.
- See the MVEL Language Guide at https://github.com/mvel/mvel for details on how to construct more advanced MVEL expressions.
- The documentation moves frequently, but is currently at http://mvel.documentnode.com/.

Community



Community has an MVEL Expression Primer.

https://community.workday.com/node/99466

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.
- Examples of the MVEL objects that Workday Studio and the assembly that runtime supports:

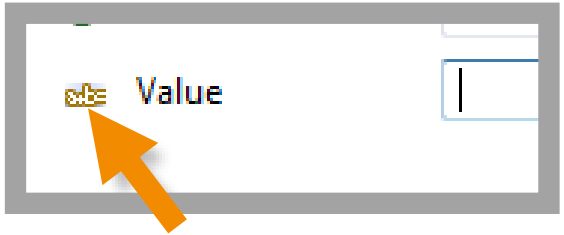
parts | props | vars | lp

MVEL

Object	Description
context	Provides access to the MediationContext object
intsys	Provides access to the integration system configuration for this assembly.
lp	Provides access to the integration launch parameters. The lp variable is only applicable to Workday internal server and integration Cloud developers.
message	Provides access to the MediationMessage interface.
parts	Provides array-based access to message parts using the MessageAdapter interface. The root part is part 0, the first attachment is part 1.
props	A utility adapter for gaining access to message properties using the AssemblyUtils.PropertyMapAdapter class.
vars	Provides access to the MessageContextVariables.

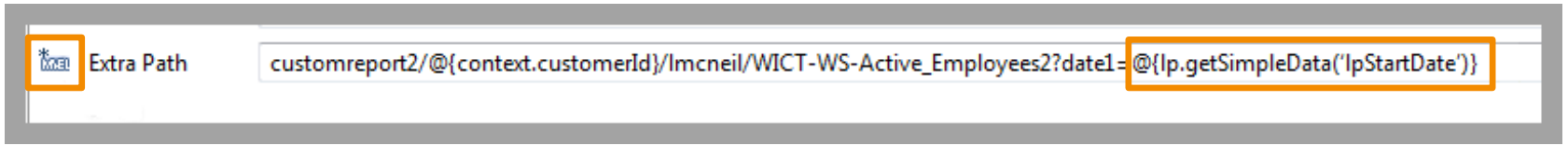
MVEL Expressions in Assembly

In the assembly steps and components, the a+b= icon represents a field that expects an MVEL expression. All literal text needs to be surrounded with single quotes (e.g., PIM parameters).



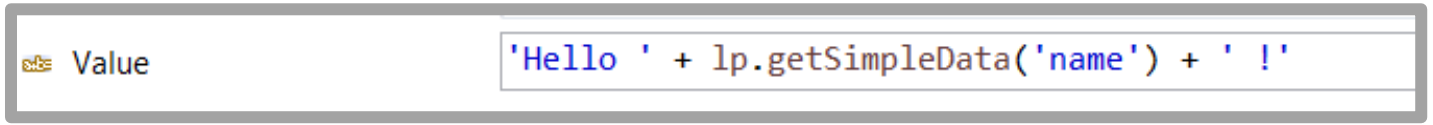
If there is not **a+b=** icon, that field does not expect an MVEL expression. In order to identify an expression to the assembly runtime, it must be surround by @{}.

For example:



MVEL Expressions in Assembly

- MVEL Text Highlighting
  - Strings will be in Blue
  - MVEL objects and methods will be in black
- MVEL boxes can now be multi-line
  - Make sure to try scrolling up if the line looks blank!

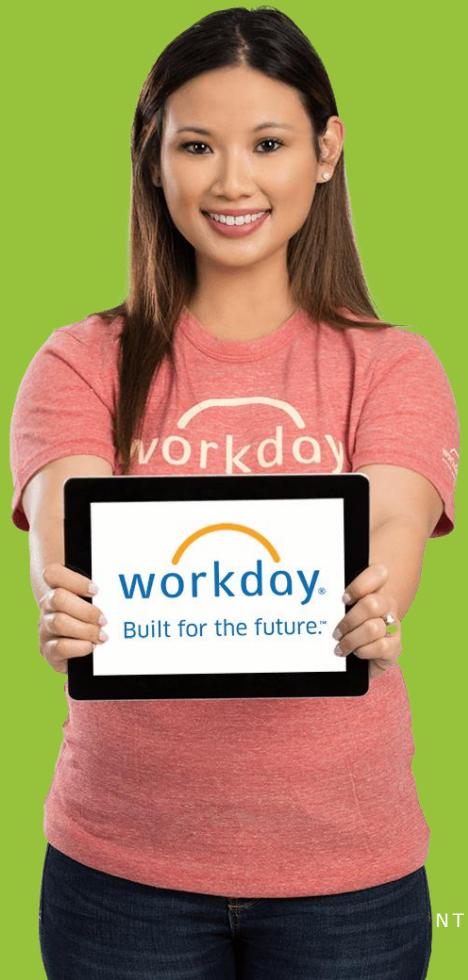


MVEL Expression – Best Practice

- MVEL is an incredibly powerful expression language which is tightly integrated into the Workday ESB's Assembly framework.
- It is possible to construct powerful and flexible expressions for manipulating Assembly context variables and associated Java objects, but, as always, with great power comes great responsibility!

Resource: https://community.workday.com/studio-mvel

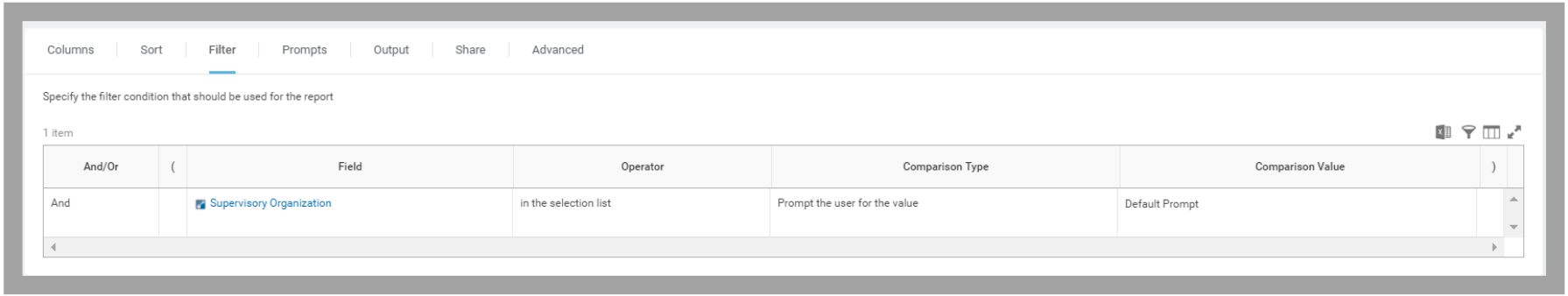
Workday In Transport: Launch Parameters



workday. | EDUCATION

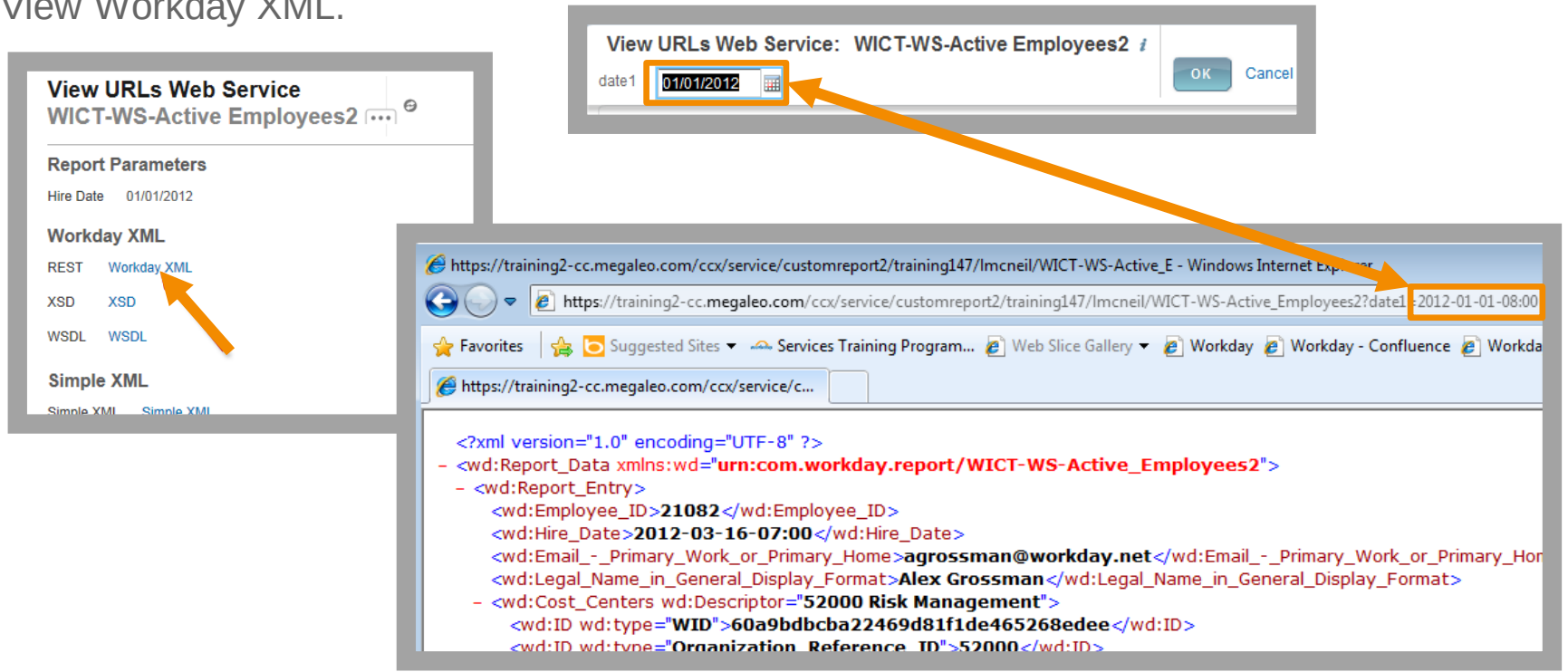
Activity: Copy Report and Add Filter with Prompt

1. Copy WICT-WS-Active_Employees and then put a '2' on the end of the name WICT-WS-Active Employees2.
2. Filter on Supervisory Organization prompt the user using Default Prompt.



URL Contains Parameter

View Workday XML.

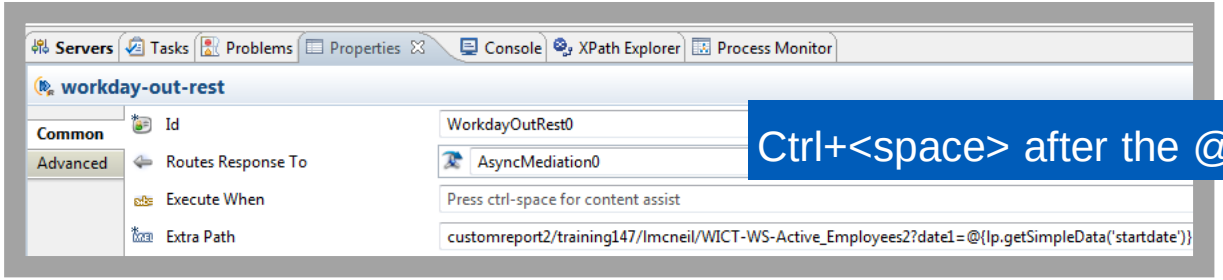


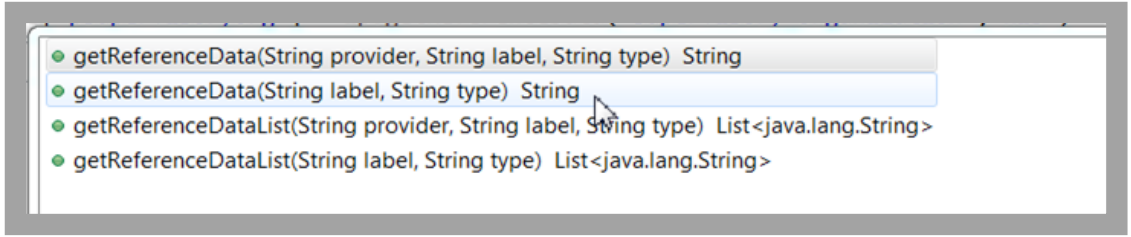
Extra Path URL Structure

- https://clint-harwood-1505133025-host01.workdayeducation.com/ccx/service/customreport2/studiogms01/lmcneil/WICT-WS-Active_Employees2?Remove_Exclude_From_Headcount=0&Supervisory_Organization!WID=b3a99d005c1143b4a672ebe66164cf13
- https://638-host01.workdayknowledge.com/ccx/service/ <extrapath>
  - customreport2/ = type of service
  - training101/ = tenant name
  - lmcneil/ = report owner
  - WICT-WS-Active_Employees2 = report name
  - ? starts parameters
  - & separates parameters
  - date1=2012-01-01-08:00 = a parameter

Content Assist

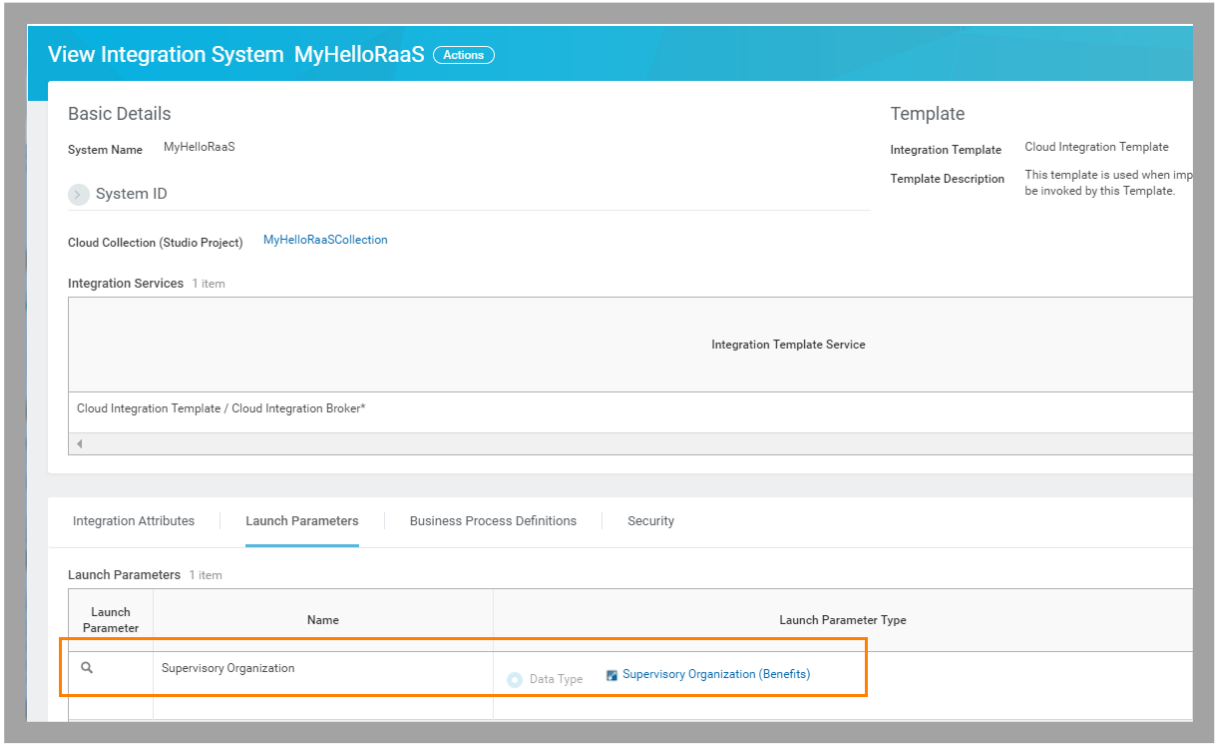
1. Press Ctrl+<space> after @{lp.
2. Opens list of methods on lp (launch parameters).
3. Don't forget to add the closing } on the end.





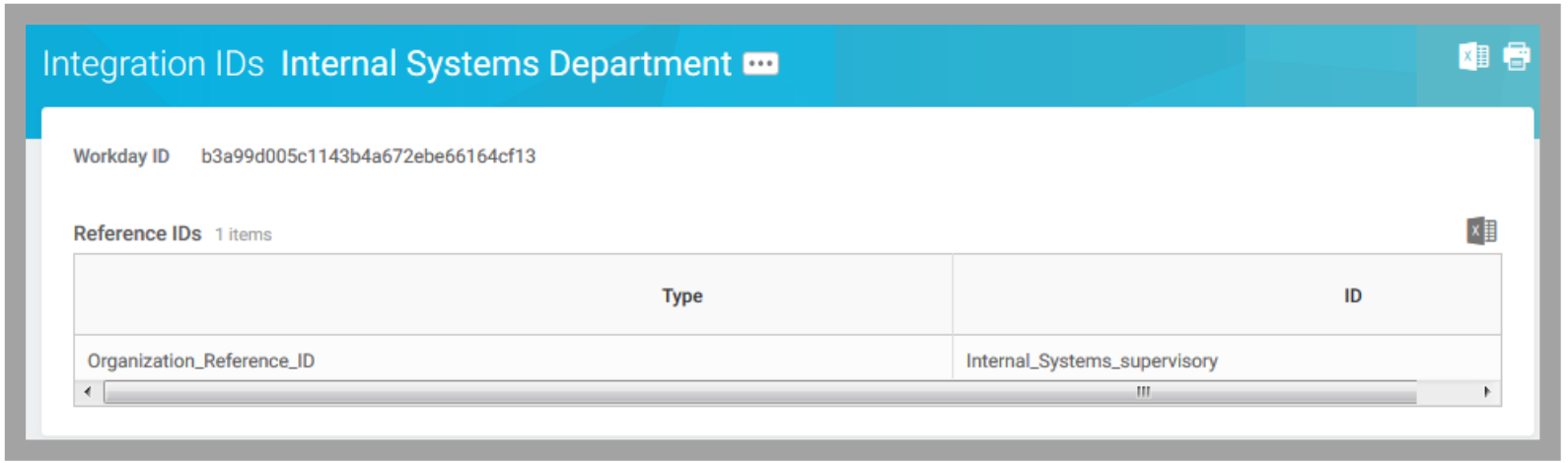
Integration System in Workday Tenant

Deploy and then verify in tenant the Required Date launch parameter.



Review: Workday Business Objects vs. Simple Type

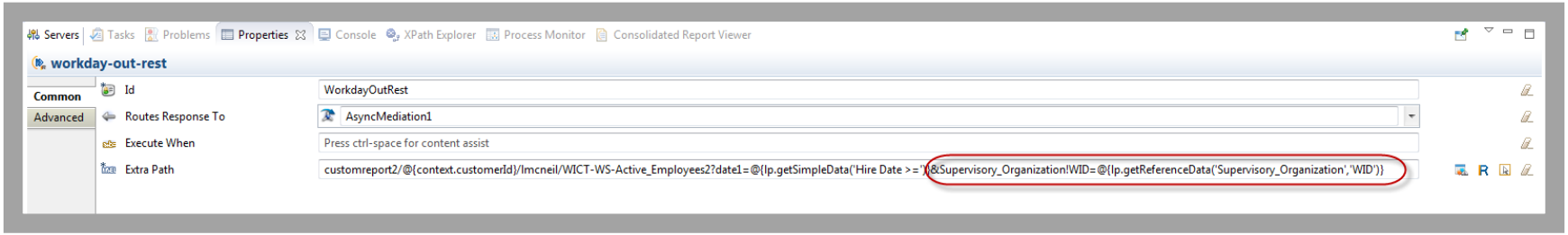
- Workday Business Object Instances have integration IDs.
- When the launch parameter is selected, it contains all valid integration IDs.



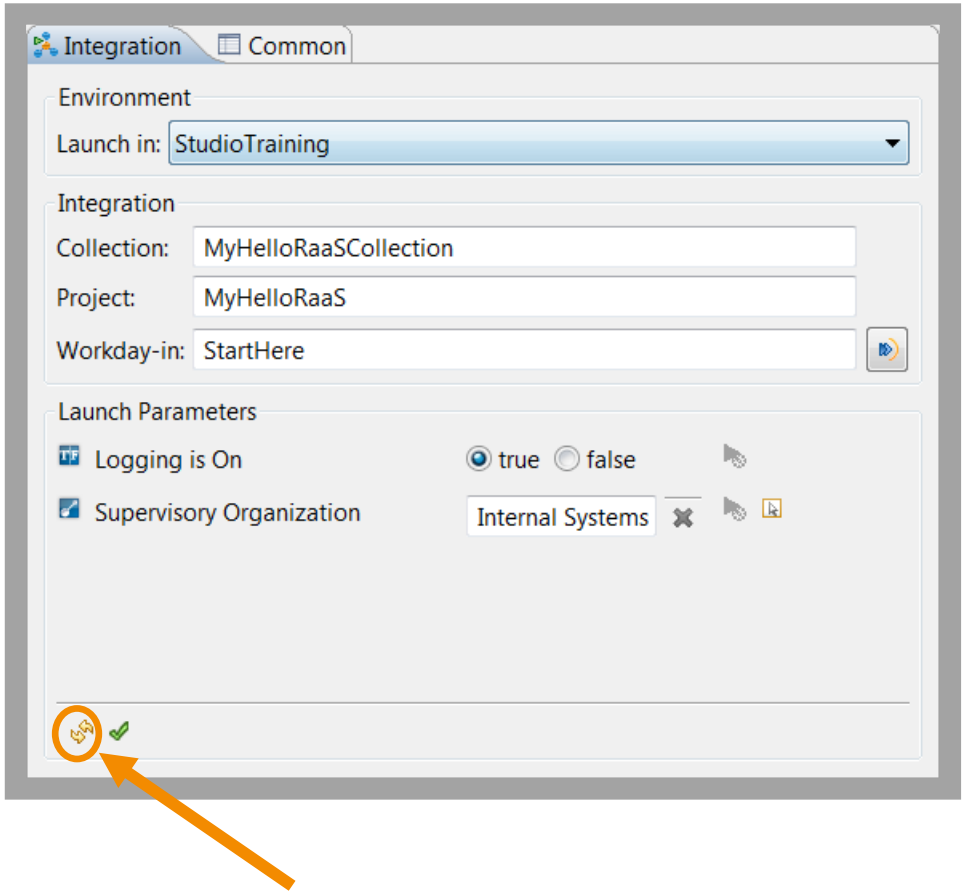
Launch Parameter Mvel in Extra Path – WID

Use the launch parameter in Extra Path:

…/&Supervisory_Organization!WID=@{lp.getReferenceData('lpSupervisoryOrg','WID')}



Show Launch Panel



You may need to click the “reload” button to display changes / new launch parameters.

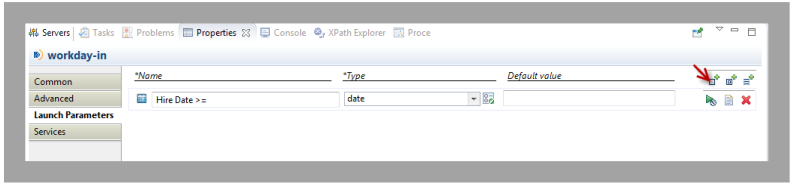
Make sure to use the prompts to select data.

Activity 2

Copy Report, Add Filter with Prompt

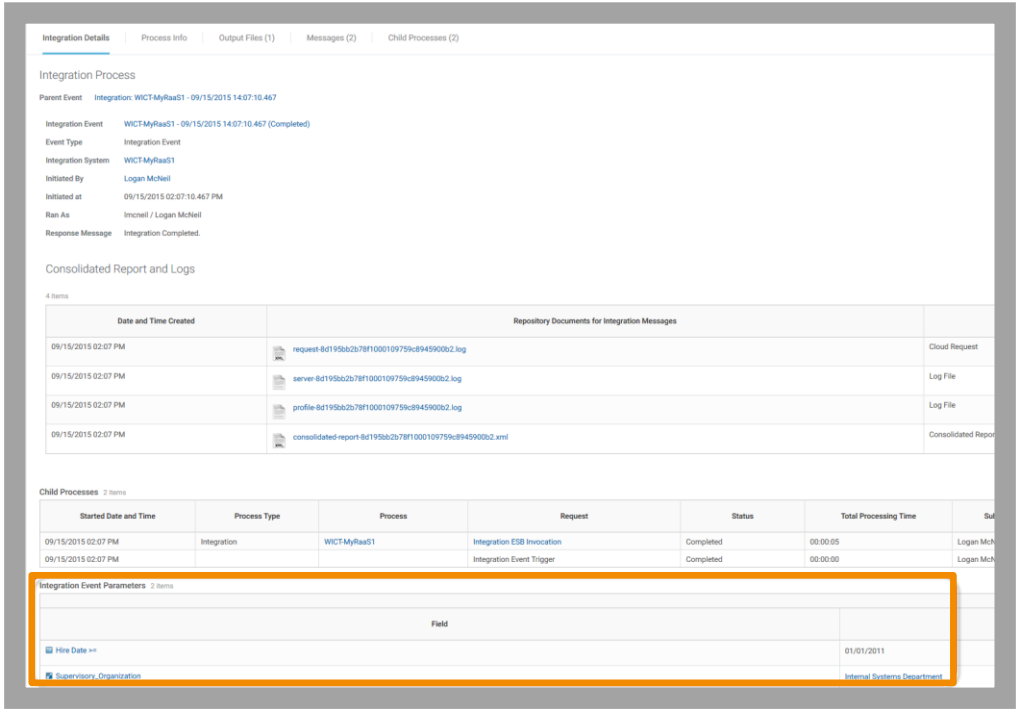
Estimated Time: 15 Minutes

1. Copy WICT-WS-Active_Employees, put a '2' on the end of the name.
2. Add Filter on Supervisory Organization, prompt the user, using Default Prompt.
3. Provide XML Alias Label Override for the prompt.
4. Add a launch parameter to the Workday In Transport.



Confirm Values After Launch

Looking at the Integration Event in the tenant, you can view the values that were entered in the launch parameters. This is a good place to begin troubleshooting if your output has no values.



Mediation Context: Properties & Variables

Learning Objectives

- Review Mediation Context key elements:
  - Message
  - Variable
  - Properties
- Identify Assembly steps used to access the Mediation Context to set/get properties and variables.
- Implement Eval step to set Properties in Mediation Context.
- Use MVEL statements to get/set Properties in the Mediation Context.

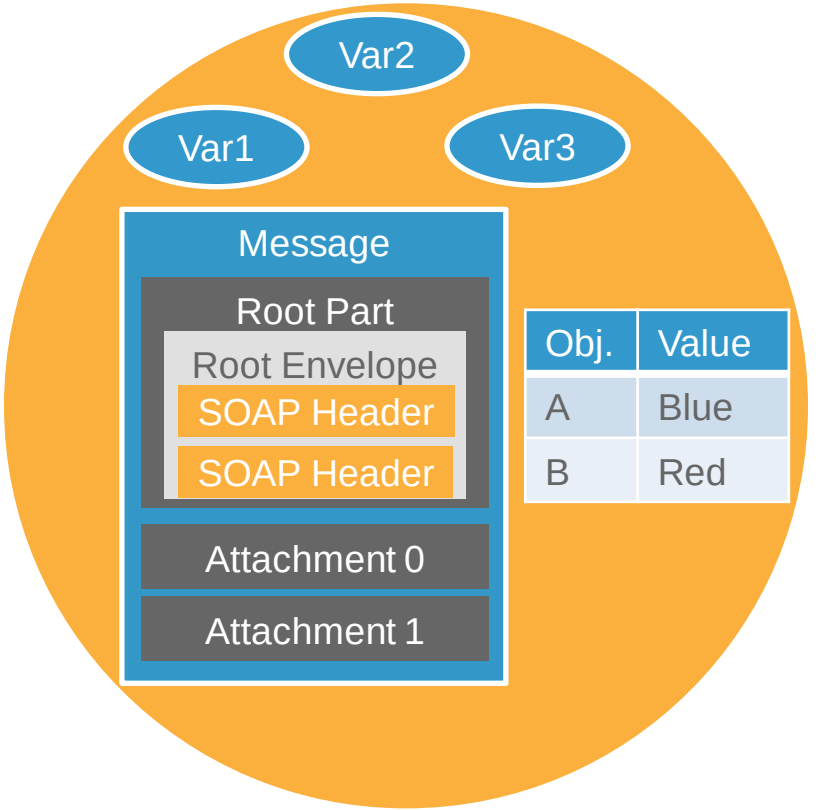
Mediation Context – Review

The Mediation Context contains three key items:

Properties: A general-purpose Map of name-object pairs that assembly components use for storing transient data outside of a message.

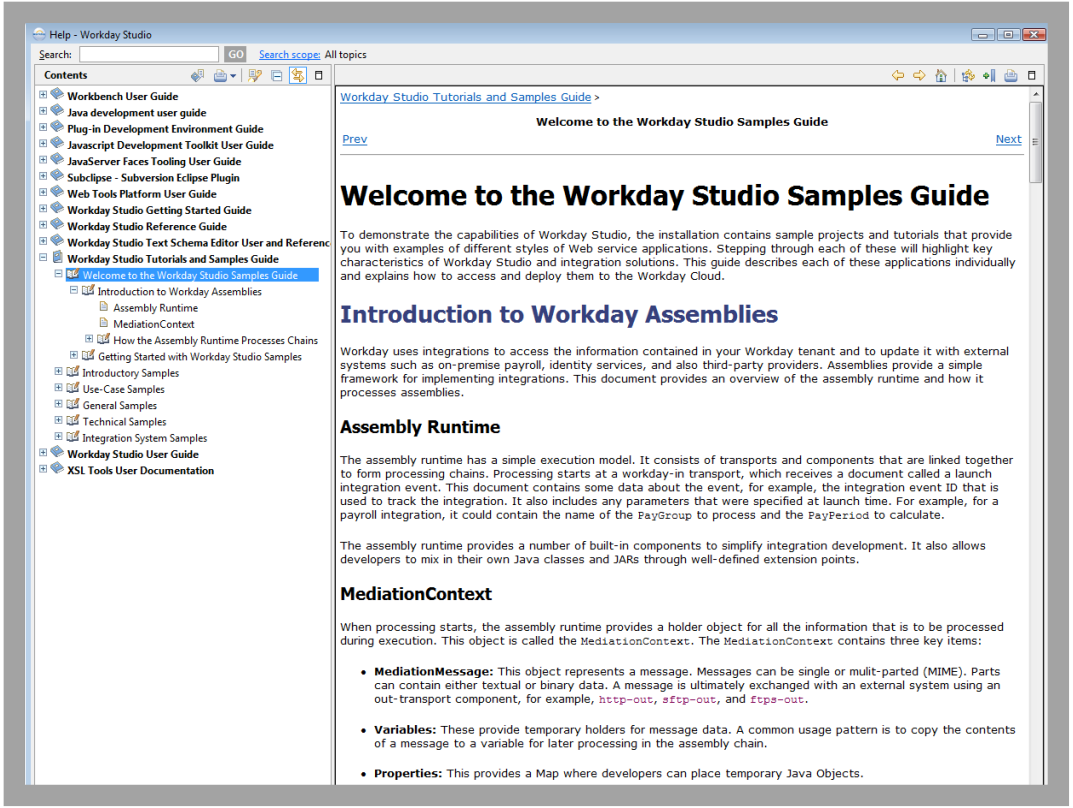
accessed later on in the interaction. They are transient, in that when the interaction is complete, they are immediately deleted.

with an external system using an out-transport component.



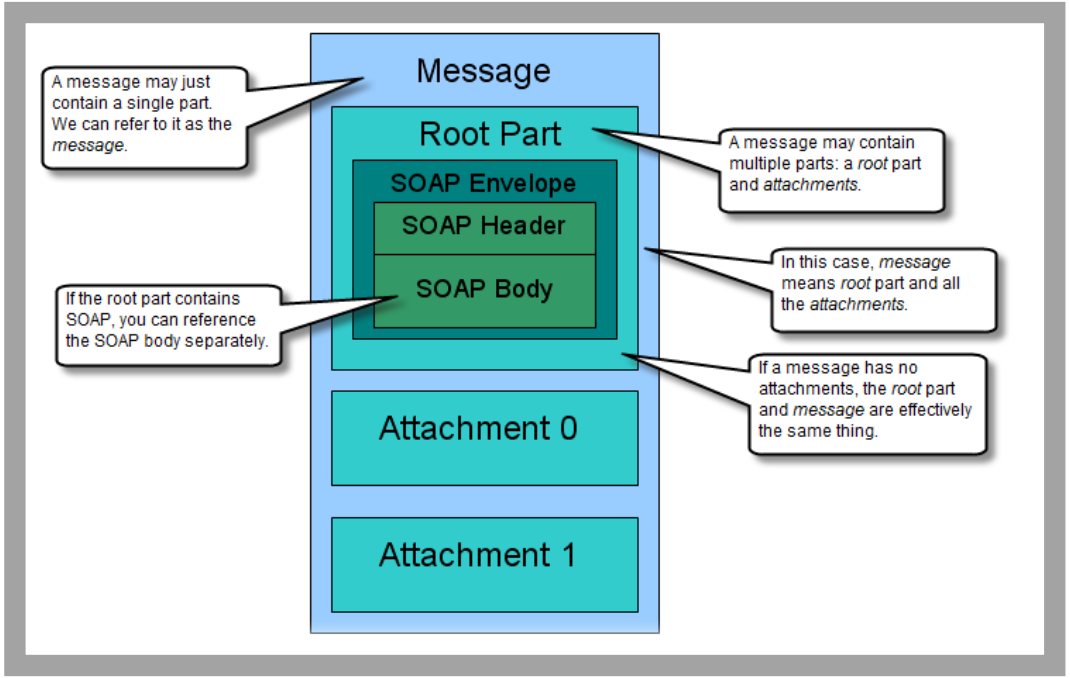
Mediation Context – Review

- When processing starts, the assembly runtime provides a holder object for all the information that is to be processed during execution.
- This object is called the MediationContext, often referred to as just 'Context'.



Mediation Message – Review

The root part of the message object is the “message”. The MVEL variable parts can be used to access the message object as an array. Parts[0]…where [0] is an array position that is always the “root.”

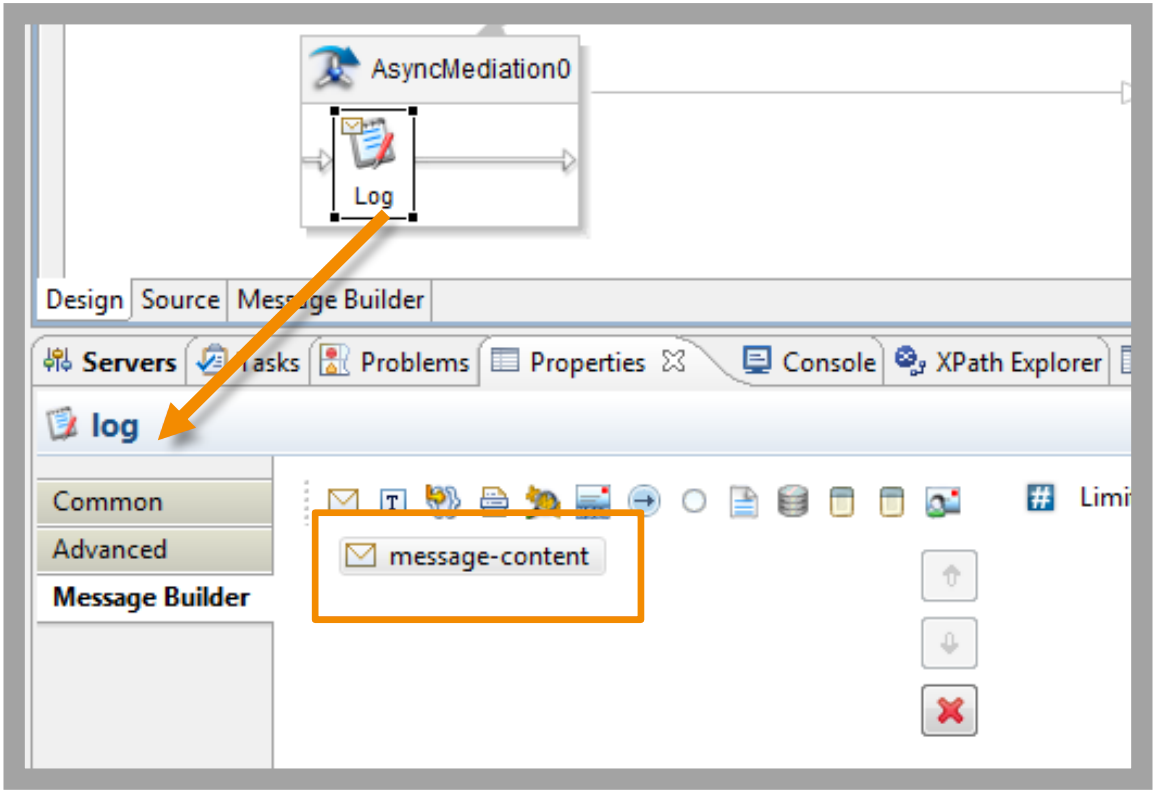


Note: This image displays a message that is a SOAP message. Not all messages are SOAP.

Workday expects a SOAP request and will send back a SOAP response.

The output to a restful call is Workday XML.

Logging the Mediation Message

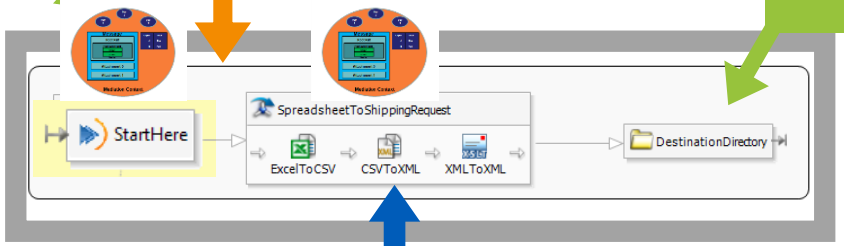


A Simple Model of an Assembly

The Mediation Context originates at a Workday-In *transport*.

The Context follows the path shown via connections.

Messages leave via *out-transport*.

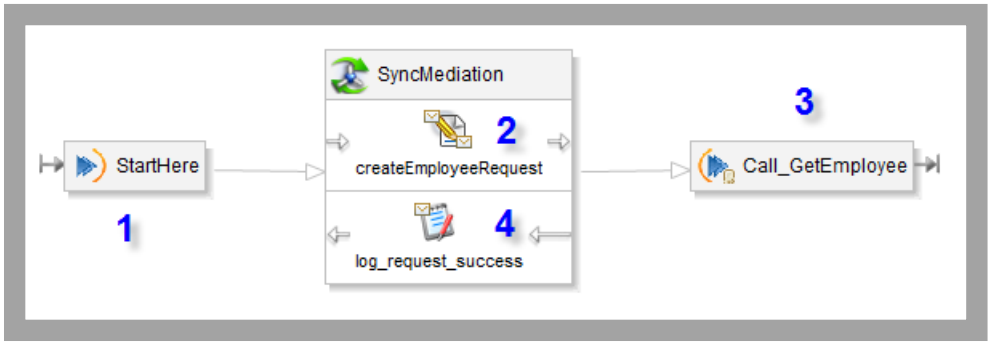


Each Mediation step is performed on the Mediation Context in sequence.

Mediations

Mediations are a grouping concept. They allow you to group sequences of related steps. Steps perform data processing and generally manipulate the contents of the Mediation Context.

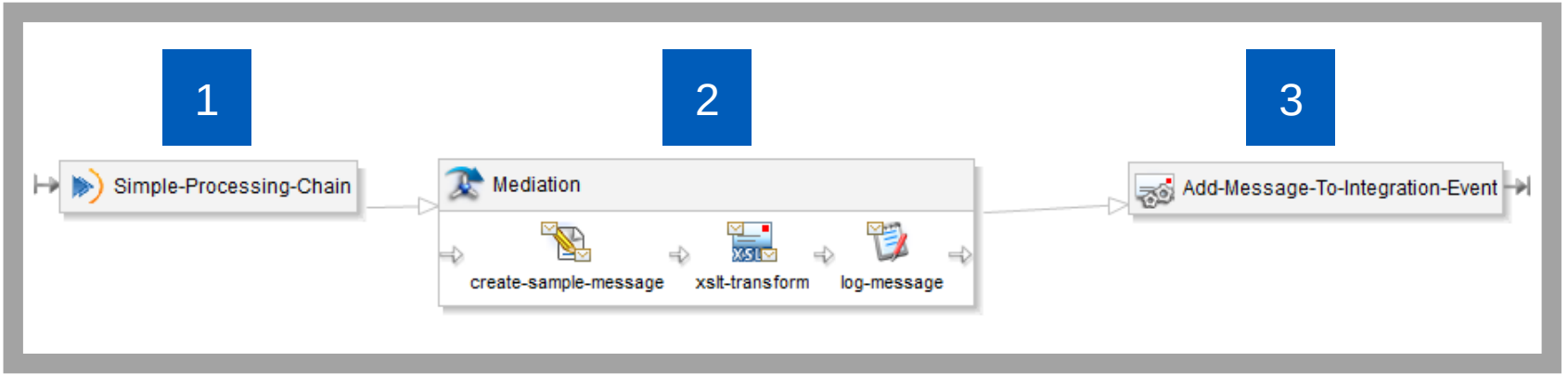
- An async-mediation includes only request steps that execute during the push phase.
- A sync-mediation includes both request steps that execute during the push phase and response steps that execute during the pop phase.



Example: A simple assembly-processing chain.

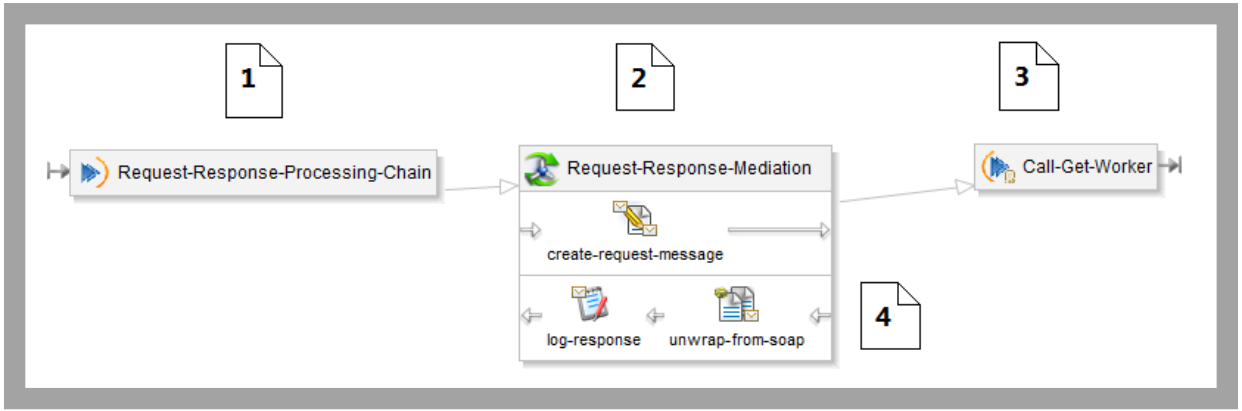
Follow the Course of the Message – Asynchronous Mediation

- The message always returns back to the entry point, or in this case, the Workday In Transport.
- The assembly runtime pushes each component that it encounters into a logical stack of components and executes it. When processing reaches the end of the chain, the stack begins to unwind.



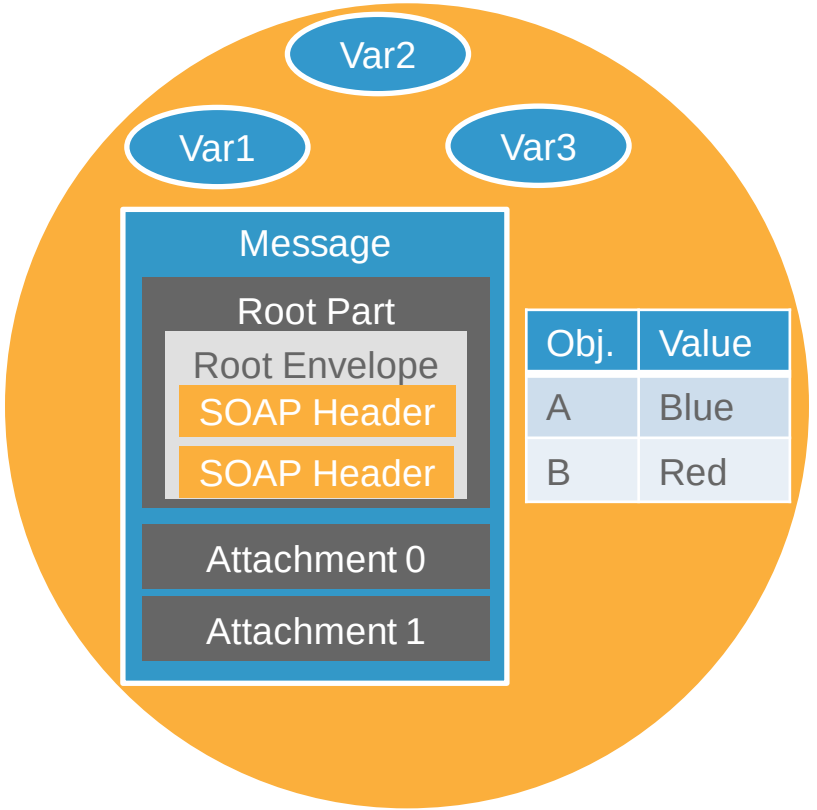
Follow the Course of the Message – Synchronous Mediation

- The message always returns back to the entry point, or in this case, the Workday In Transport.
- In this assembly, a Synchronous Mediation response is able to “catch the message as it works its way back as the stack unwinds.



Mediation Context Properties & Variables

- Variables: Variables provide assembly developers with a temporary storage location for messages, where they can be accessed later on in the interaction.
  - vars['myVar']
- Properties: This is a general-purpose Map of name-object pairs that assembly components use for storing transient data outside of a message.
  - props['myProperty']
- Simplicity is key. Do not make your assembly more complicated than it has to be by storing data in properties or variables unnecessarily.



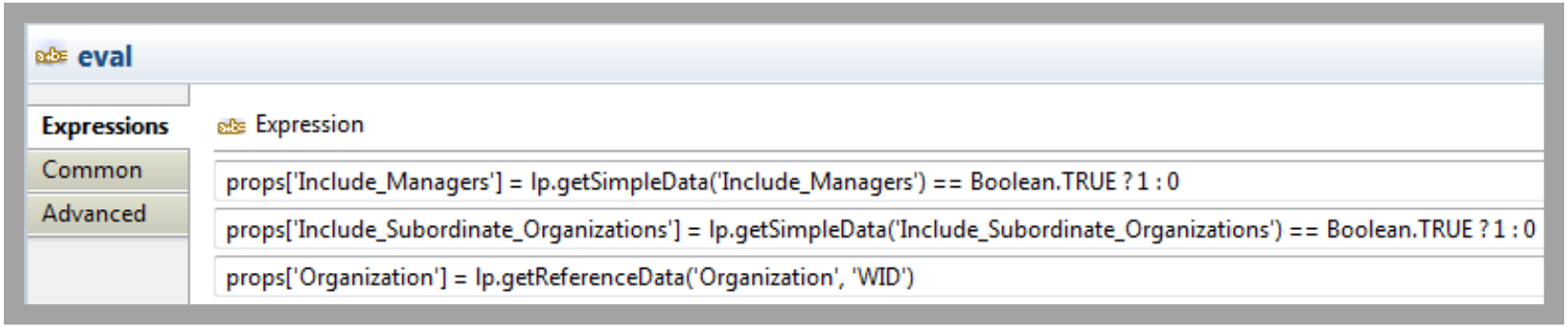
Recommendation: Naming Conventions

- A large number of properties are added to the mediation context by the ESB and some of the shared components.
- Prefixing the properties you declare within your integration name will make them easier to find when debugging the assembly. It will also add a conceptual scope to the properties.

Note: Always ensure that your property names are meaningful.

Setting Properties & Variables in Assembly

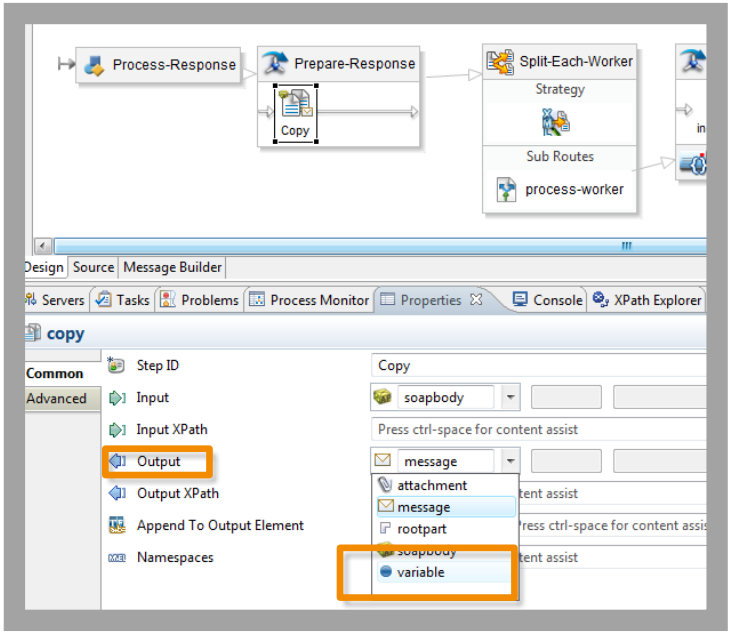
An Eval step allows you to set properties, variables, and to evaluate expressions directly.



Copy Message into Variable (and Variable into Message)

Copy step:

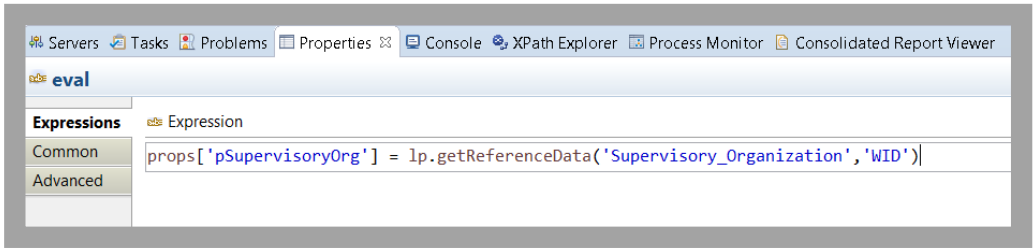
- Can be used to copy information from messages to variables in the message context and copy information from variables back into messages.
- Can be used to extract the contents from one of the message parts and insert them into a part within an entirely new message that is generated as the output from the copy step.



Add an Eval Step to Assign Value to Property

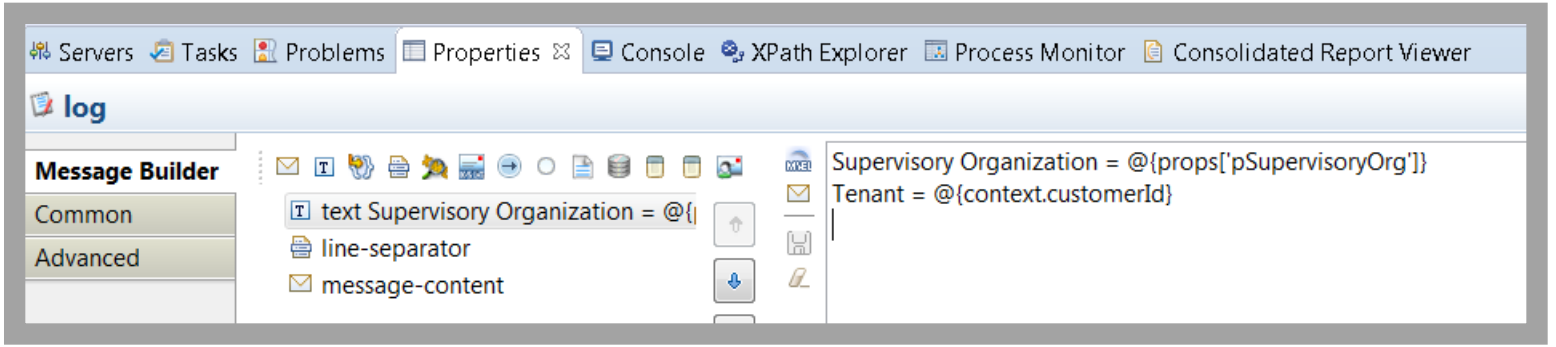
- Add an Asynchronous Mediation with an Eval step before the Workday Out Rest transport.
- Create a property for both launch parameters (Hire Date and Supervisory Organization).
  - props['pSupervisoryOrg'] = lp.getReferenceData('Supervisory_Organization','WID')

Note: If you copy/paste the syntax from the PDF, please be mindful of your naming conventions so that they match. Also, you may run into an issue with the single quotes pasting incorrectly. To correct, just use backspace and retype the single quotes after you paste.



Modify the Log Step to Include Reference to Properties

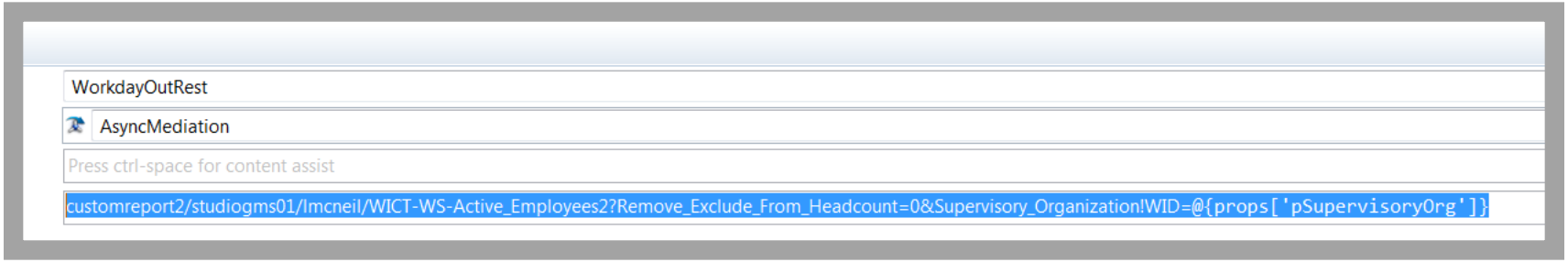
- Use MVEL in the existing Log step to log the values in the properties, including the Tenant Name and properties you just stored.
- **context.CustomerId** is a property set by the ESB in the Mediation Context that returns the current tenant name. It can be used in place of hardcoding a tenant value.



Include Reference to Properties in the “Extra Path”

Modify the Extra Path of the WorkdayOut Rest to reference the properties for the Hire Date and Supervisory Organization prompt values.

Also, replace the hardcoded value for the tenant name (if not using report service).



customreport2/@{context.customerId}/lmcneil/WICT_WS_Active_Employees2?Supervisory_Organization!WID=@{props['pSupervisoryOrg']}

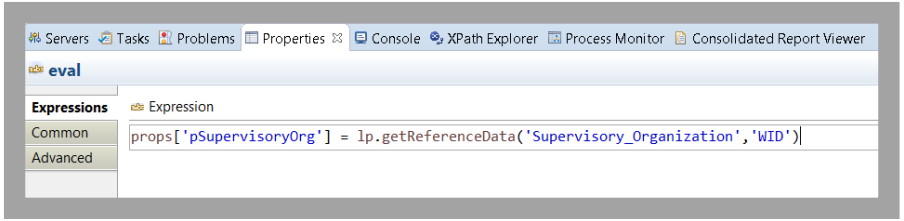
Activity 3

Modify MyHelloRaaS to use an Eval statement

Estimated Time: 10 Minutes

Modify the MyHelloRaaS Assembly in Studio.

- Add an Asynchronous Mediation with an Eval step before the Workday Out Rest transport.



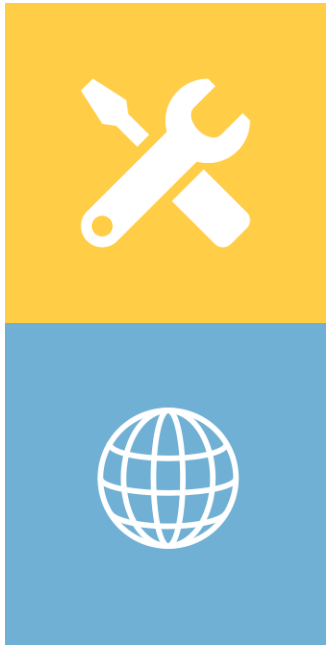
Logging

workday. | EDUCATION

Learning Objectives

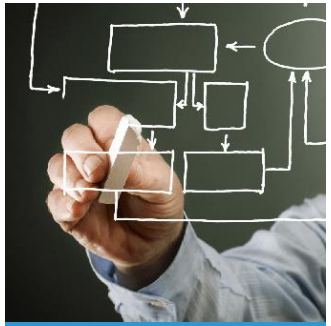
- Discuss recommended Logging methods
- Identify common logging errors

How to See your Mediation Message



- PIM
- Log and Cloud-log Steps
- Debugger
  - Best Method

Common Logging Pitfalls

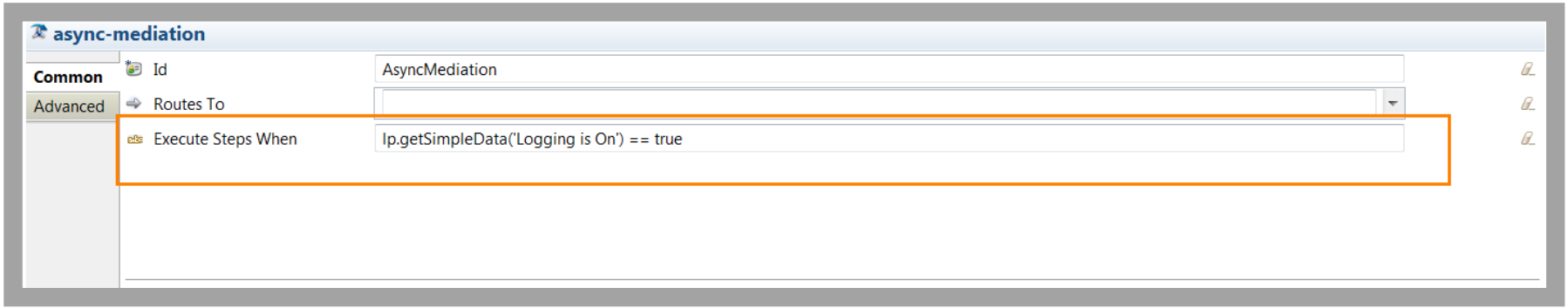


- Too Many Log Steps
- Log steps left on in Production code

- Slows things down
- Can cause memory issues

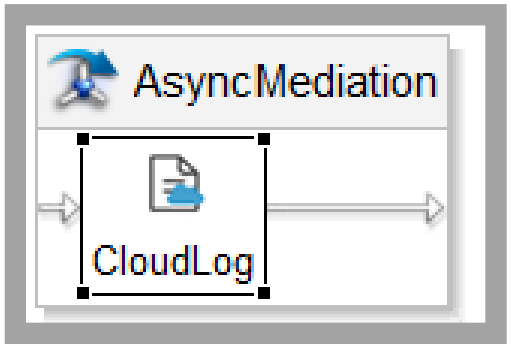
Recommended Methods

- Remove the Log steps before SIT/UAT
- Disable the log steps using “Execute Steps When”
  - Launch Parameter
  - Attribute (covered later)



Cloud Log Step

- Cloud Log step
- Builds an html log in 4J style inside of a variable
- At the end of the integration store the variable to the integration event
- Up to 16MB of messages

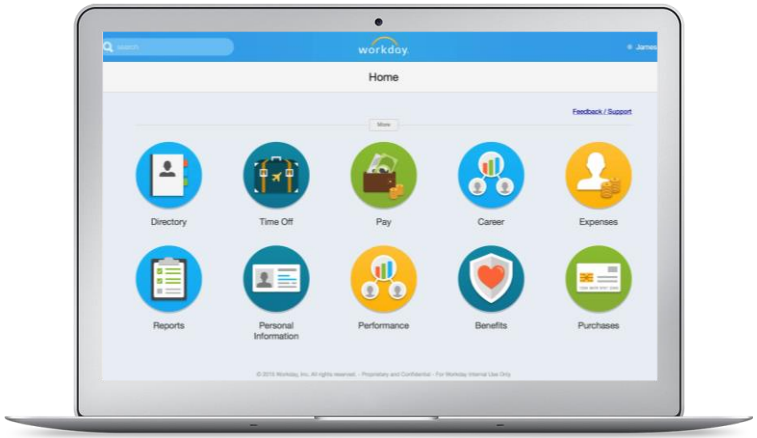


Activity 4

Logging

Estimated Time: 10 Minutes

Set up a Launch Parameter to Control Logging



Using the Assembly Debugger

workday. | EDUCATION

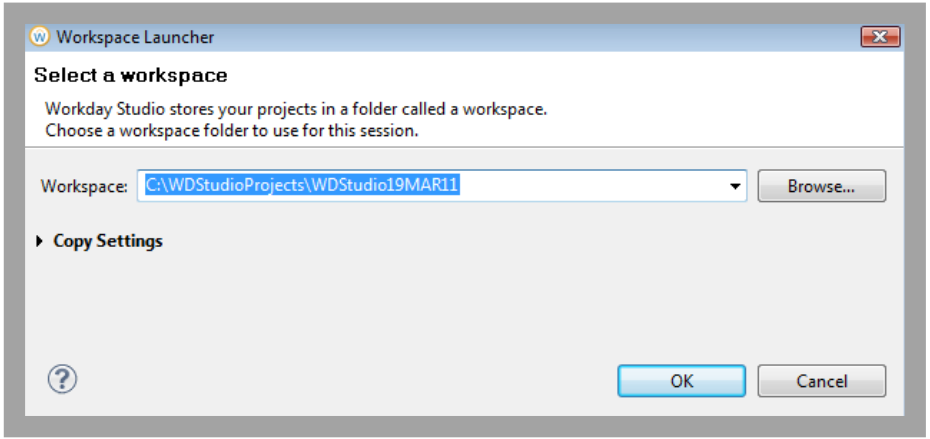
Workday Studio Assembly Debugger

Review the Debugger

- Set breakpoints
- Run until breakpoint
- Step
- Display mediation context message
- Display mediation context variables
- Display mediation context properties
- Evaluate MVEL expression

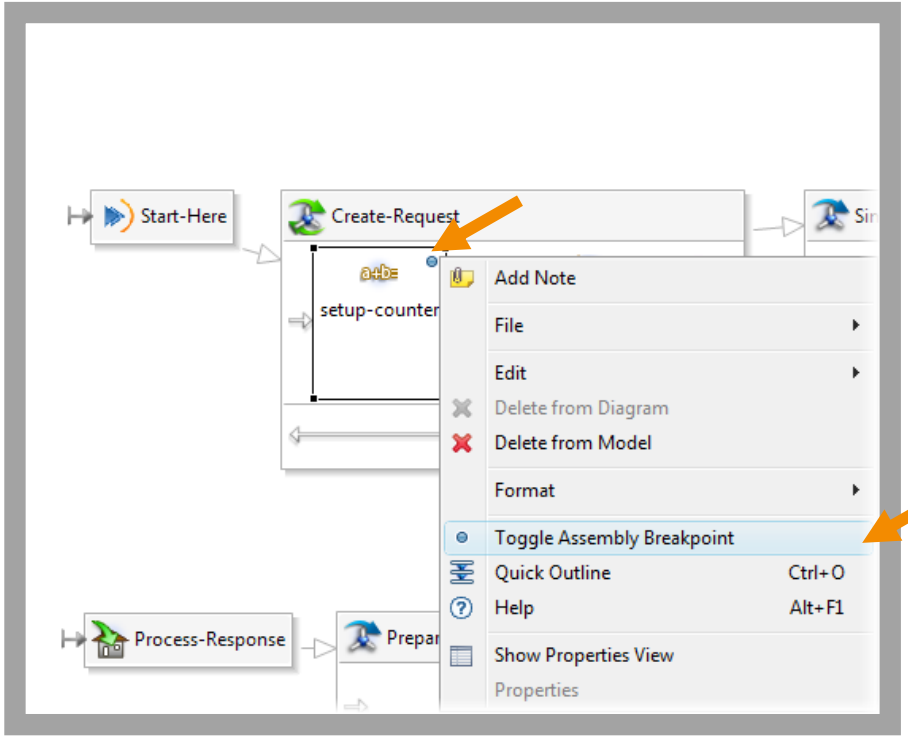
Workspace File Path – No_Spaces

- You cannot have spaces in your Workspace file path. To view your current Workspace, navigate to **File > Switch Workspace**.
- If you have spaces, you will need to click on **Browse…** to select a new location and create a new Workspace. Once created, Studio will restart.
- Connection details are linked to the Workspace and may need to be configured.

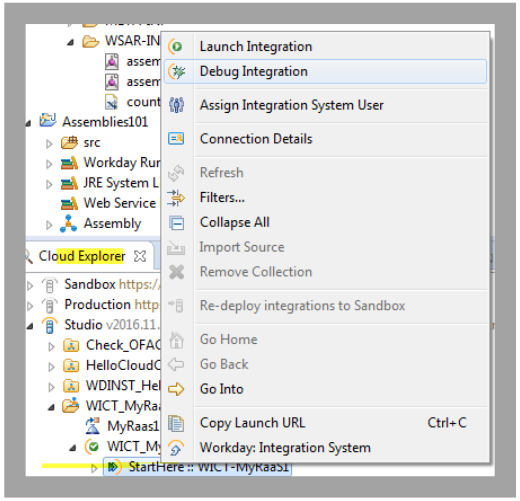


Debugger: Set Breakpoints

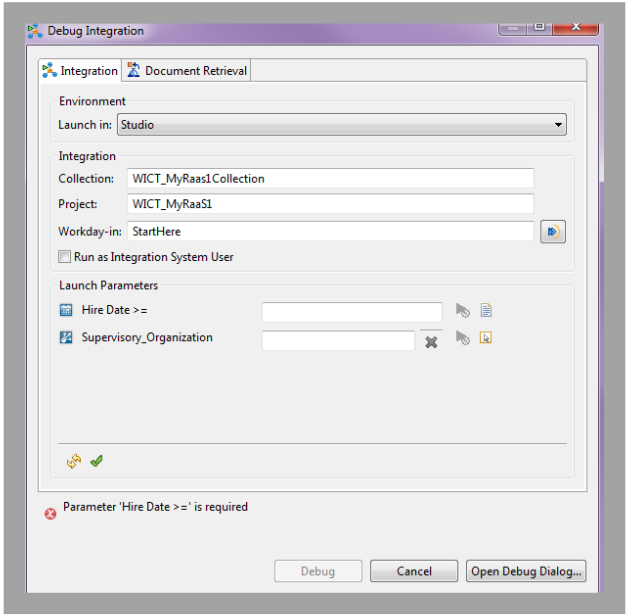
Most components or steps provide Toggle Assembly Breakpoint.



Launch Debug Mode



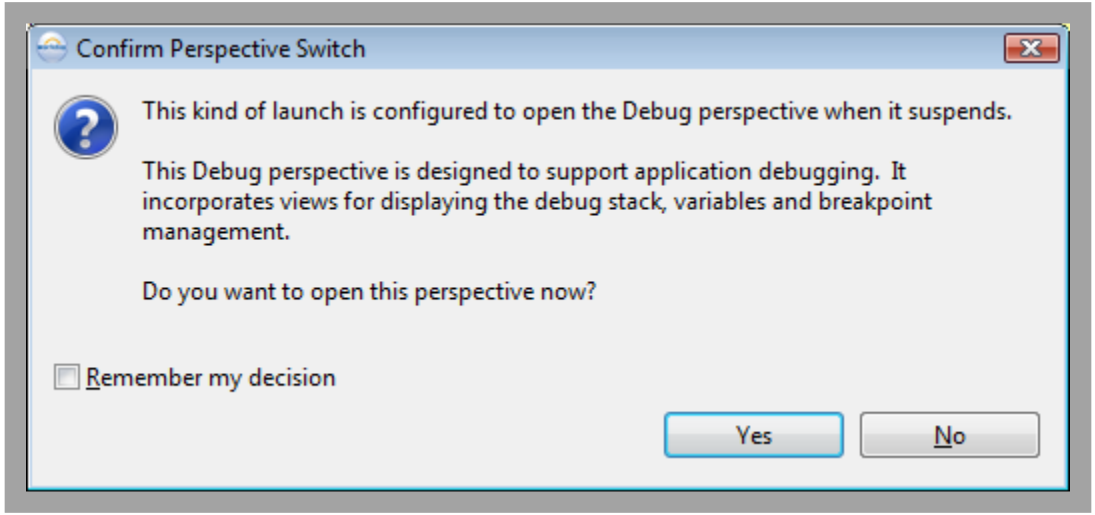
Make sure to enter any runtime prompts that are required.



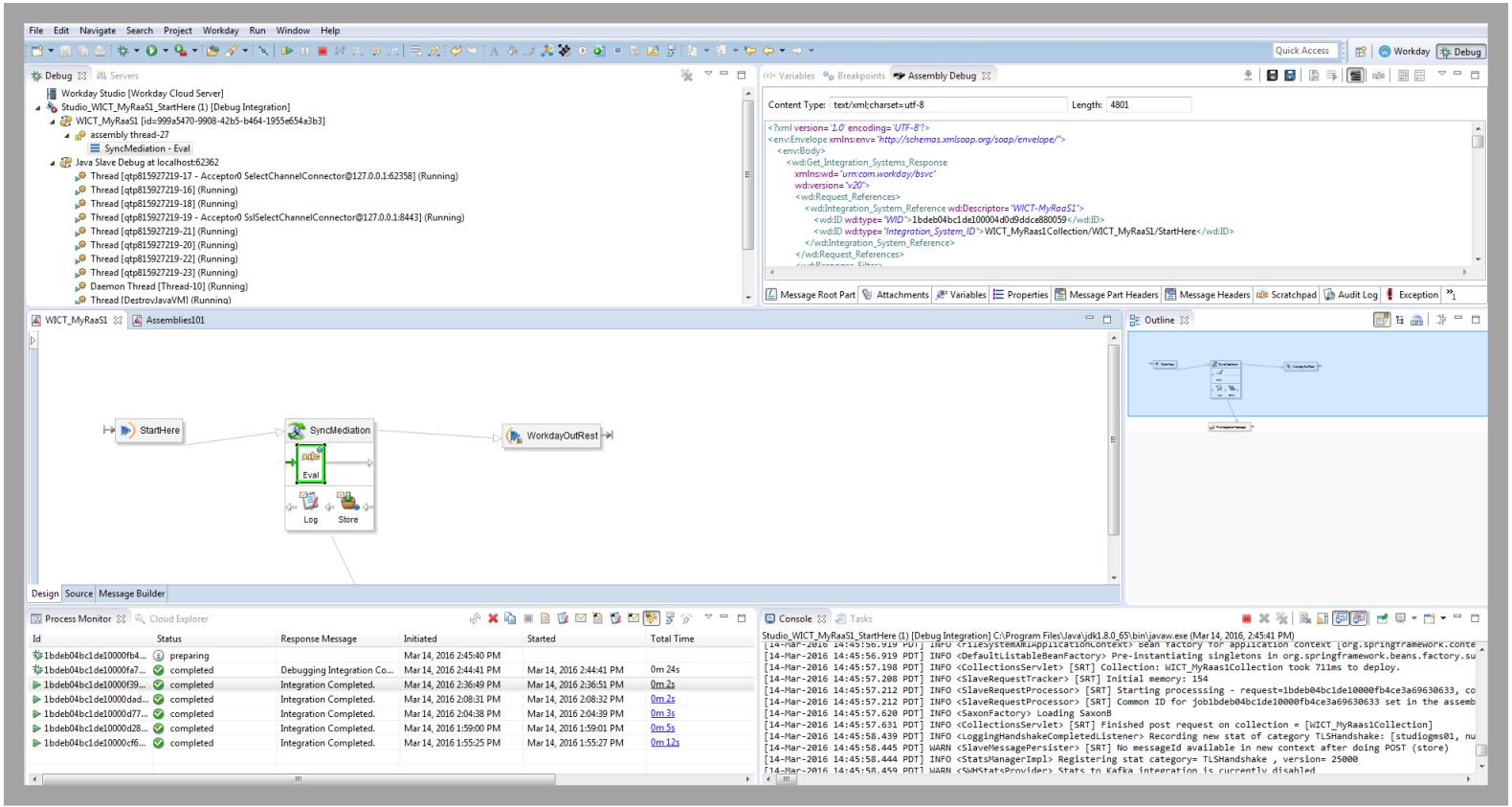
Make sure to enter any runtime prompts that are required.

Debug Integration, Switch Perspective

When prompted, click **Yes** to switch to the Debug perspective.

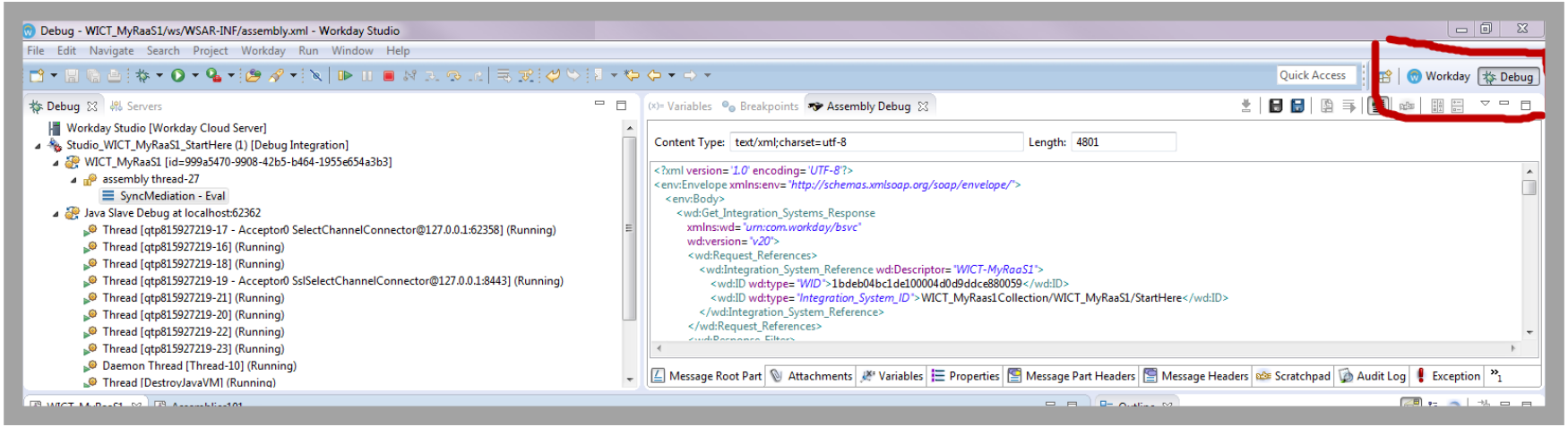


Debugger Perspective



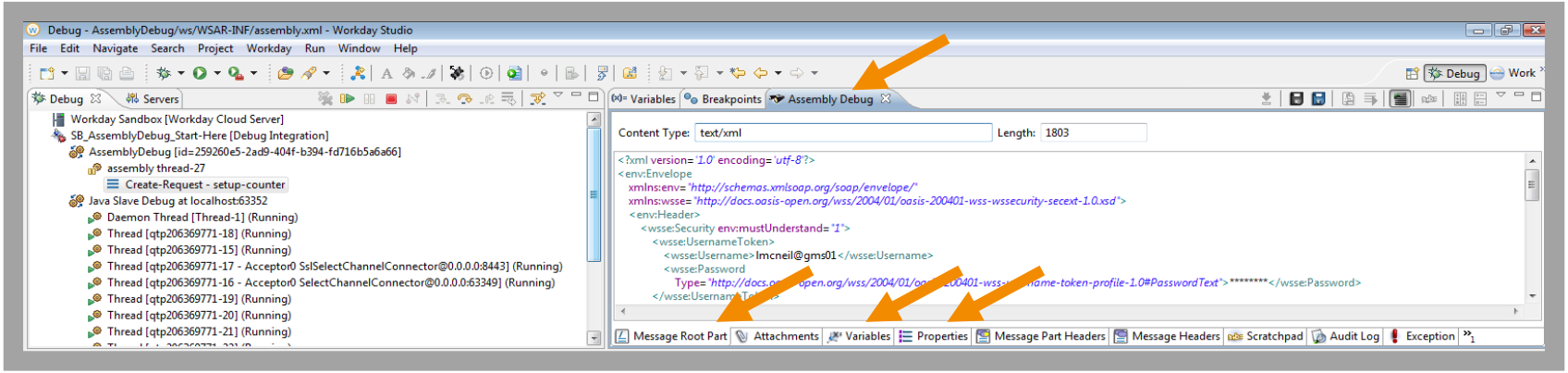
Debugger: Perspective

Click on Workday or Debug Perspective to switch between the two.



Debugger: View Mediation Context

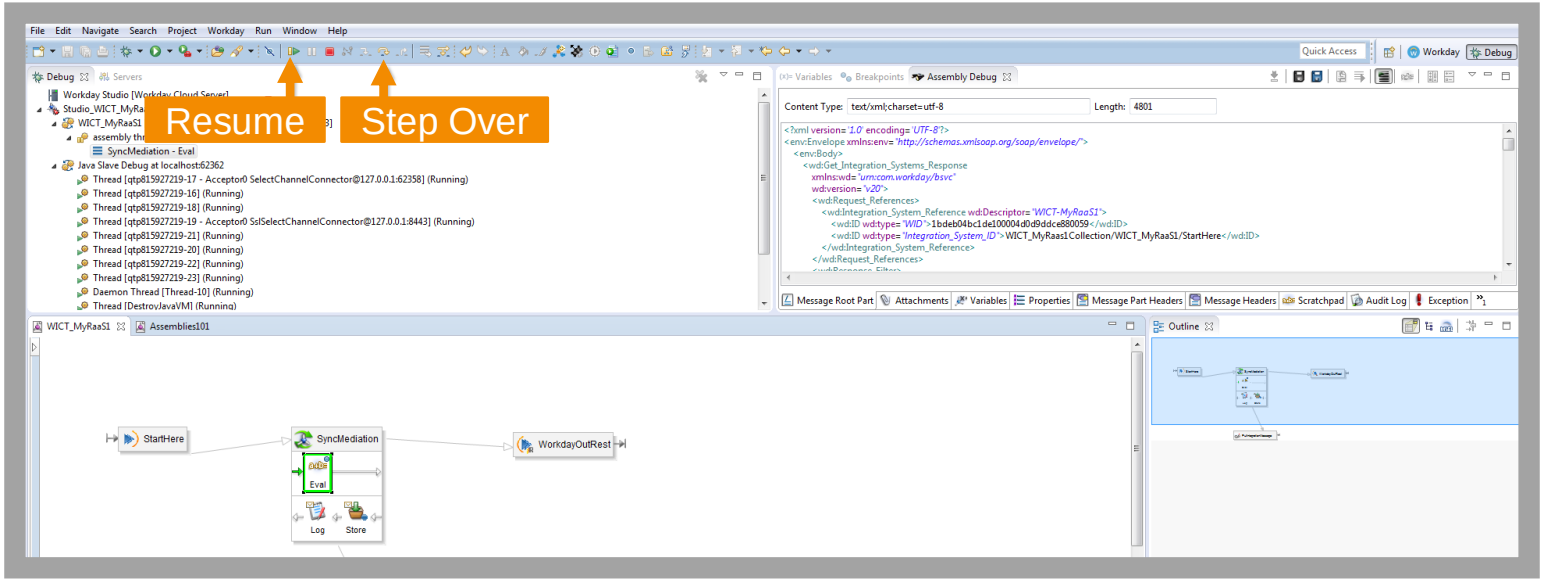
The Assembly Debug view is your “window” into the Mediation Context. You can view the Mediation Message (root part), Variables, Properties, and much more.



Resume (Run to Breakpoint) or Step Over

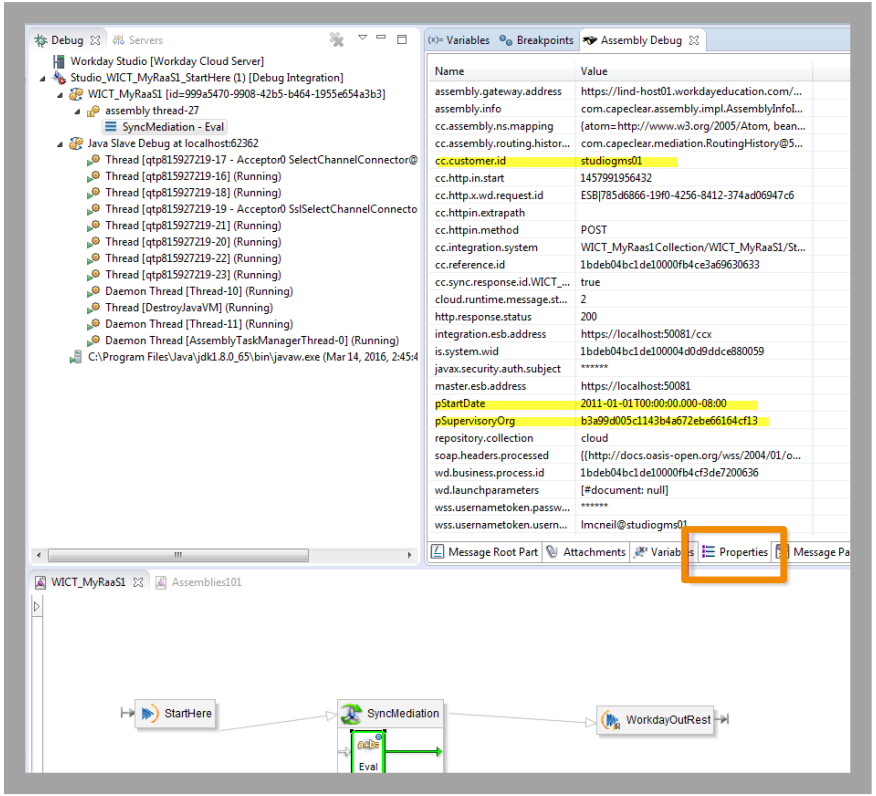
If you set multiple breakpoints, you may use Resume to run to the next breakpoint.

You may also use Step Over to stop and review before and after each step.



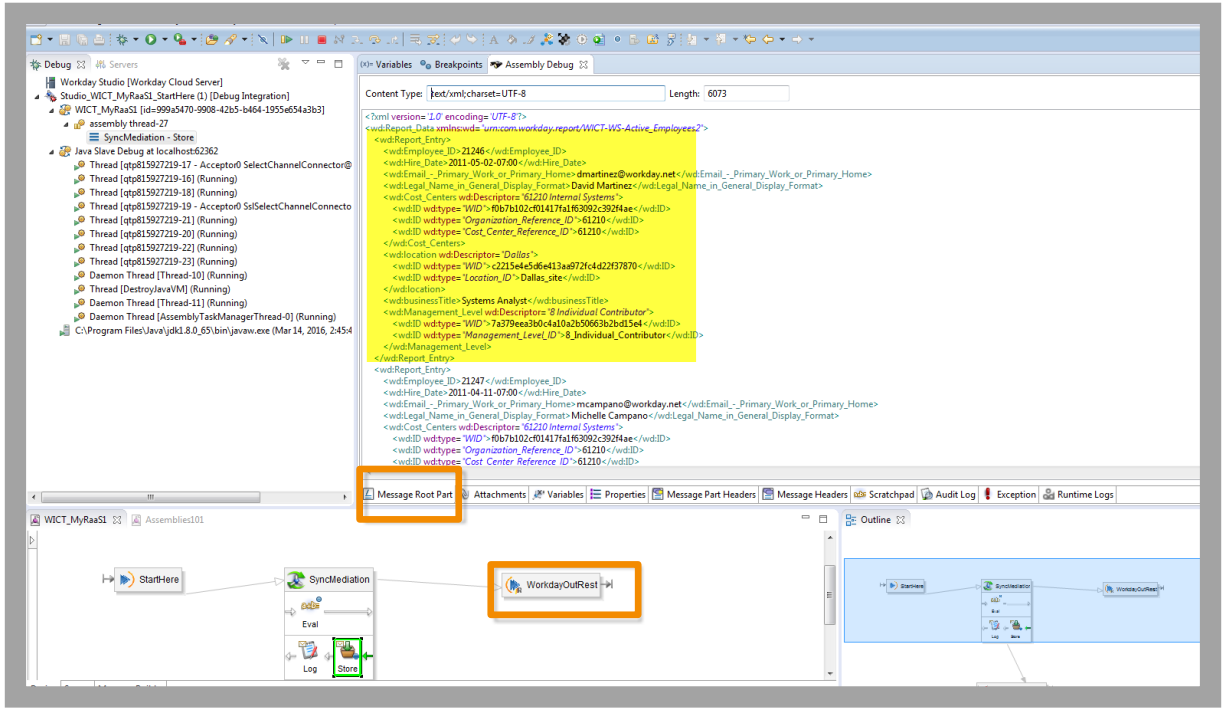
Assembly Debug View – Properties

The Properties tab will display properties set by the ESB and properties as they are set in your assembly.



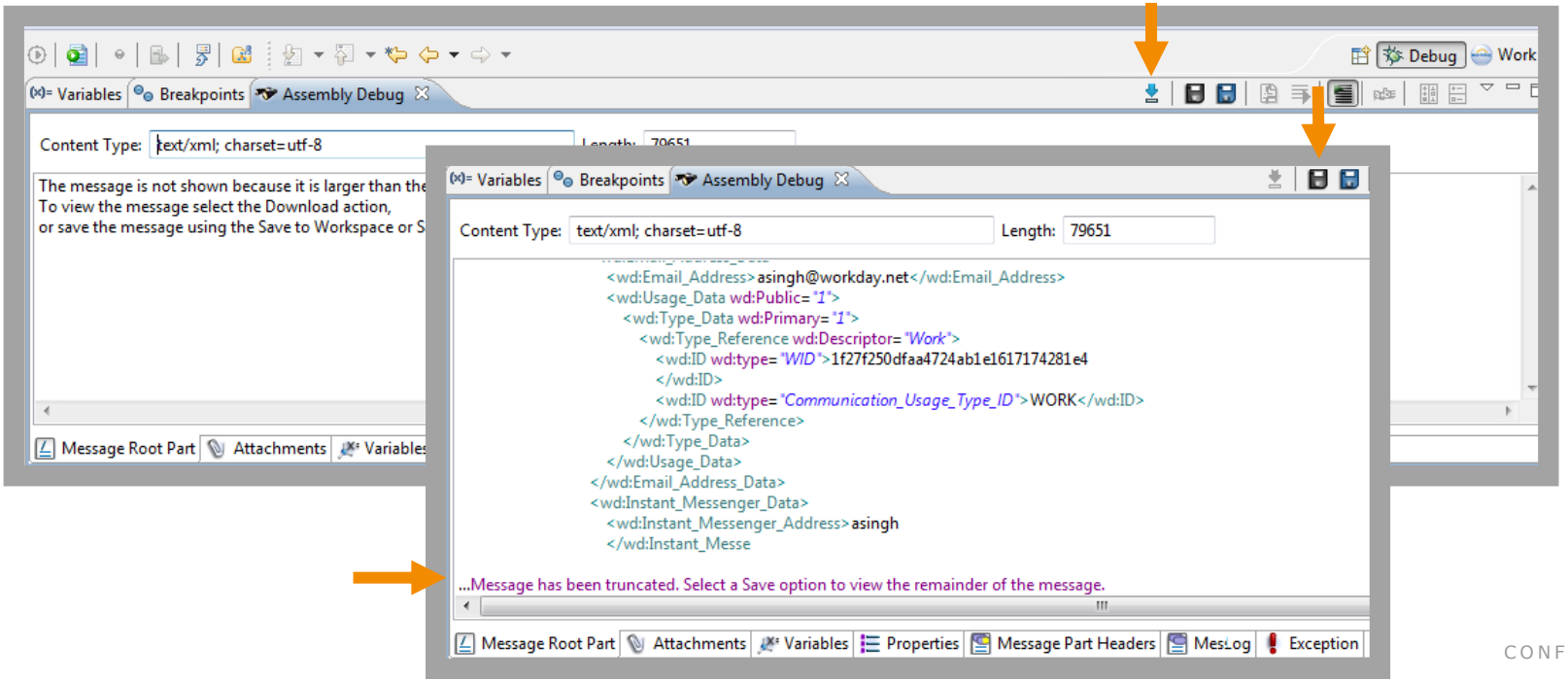
Assembly Debug View: Message Root Part

The Message Root Part display the Mediation Message. This is the message that is currently being processed by the assembly.

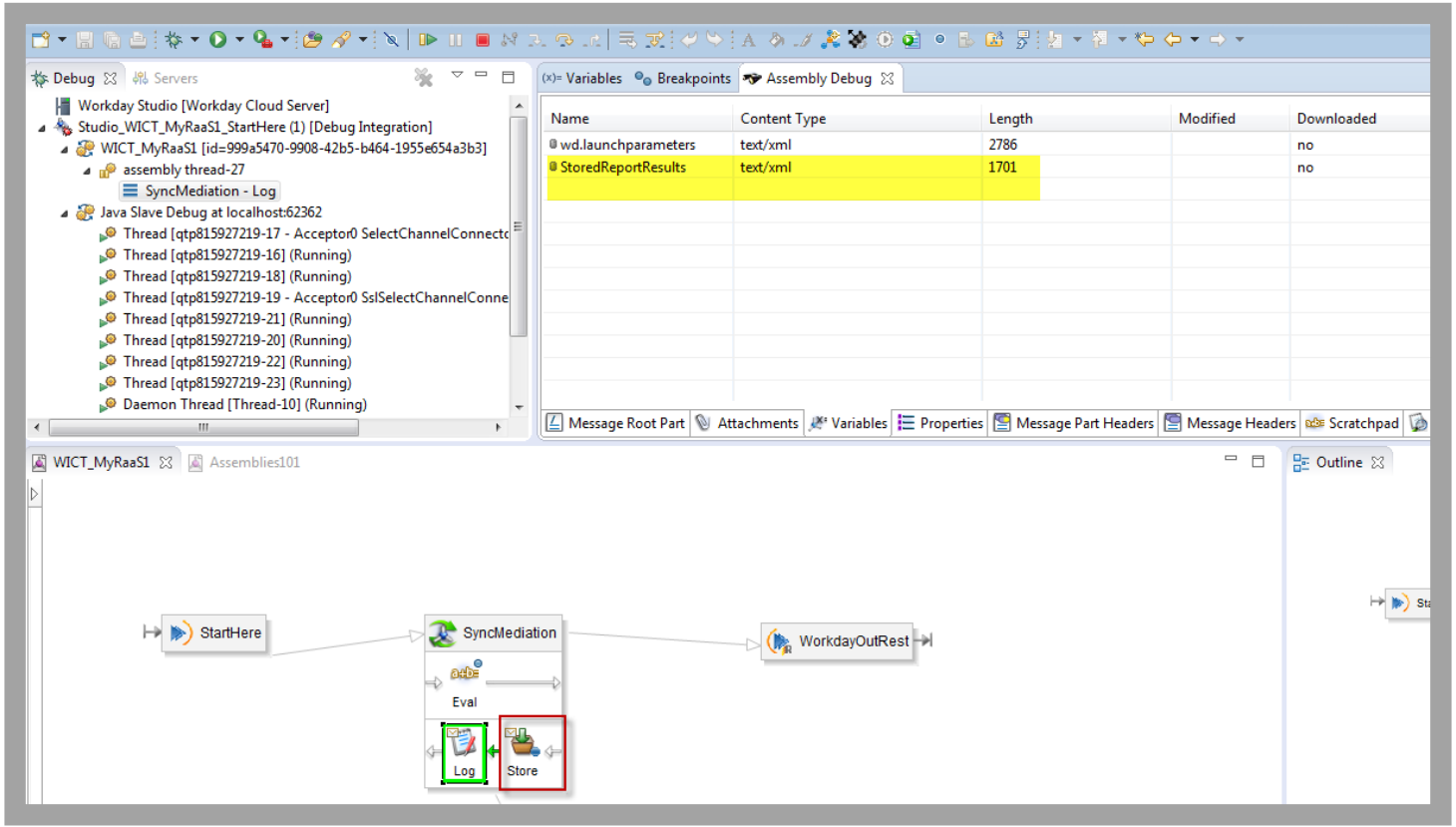


Viewing and Saving Messages

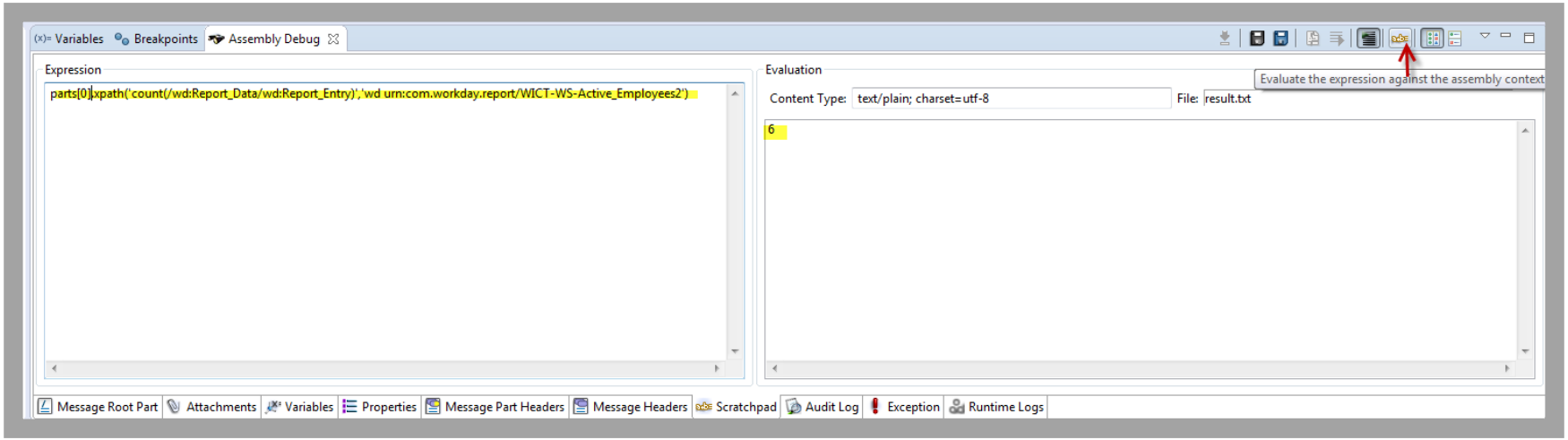
If you see this message, click the Download the Item icon (which will display up until a “truncated” message), or save to Workspace, or a file system (if you want the entire message).



Assembly Debug View: Variable



Assembly Debug View: Scratchpad

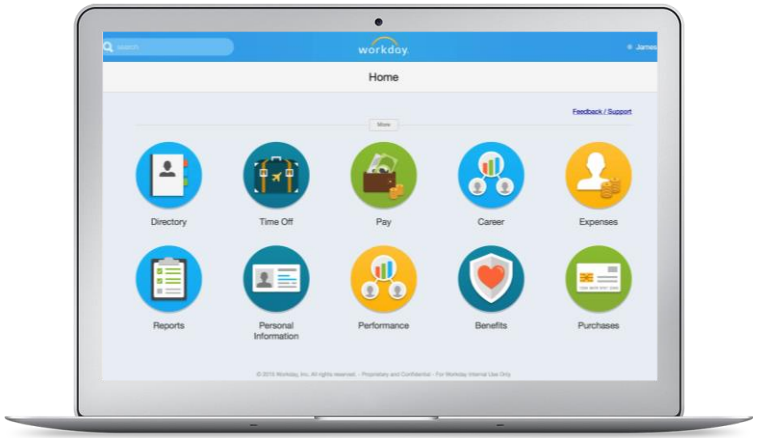


Activity 5

Debugger

Estimated Time: 10 Minutes

Use the Debugger to step through MyHelloRaaS assembly.



Working with Collections

workday. | EDUCATION

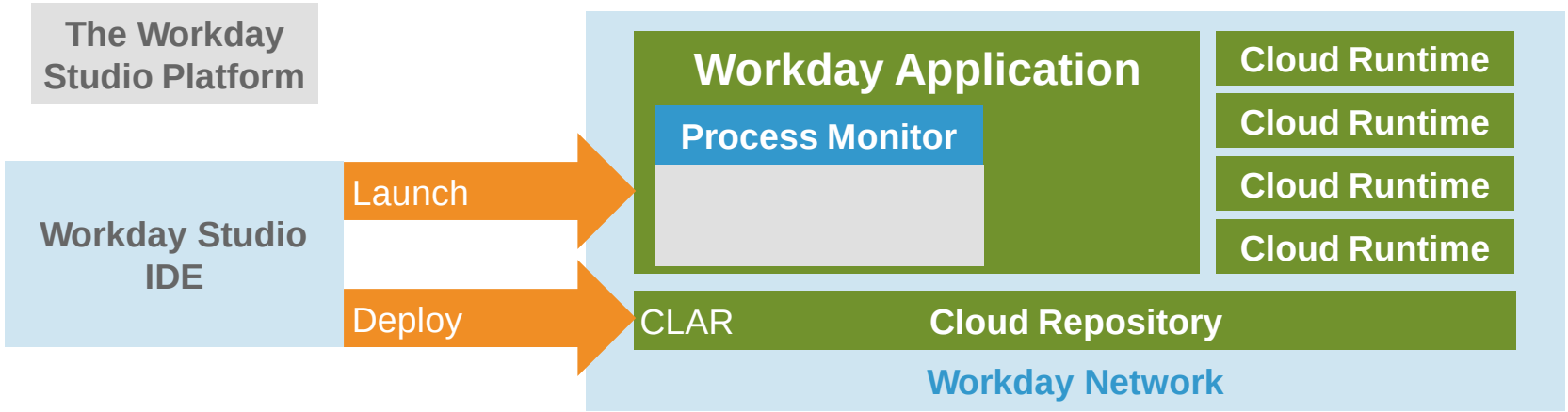
Learning Objectives

- Review Collection Concept
  - Multiple Projects in a Collection
  - Workday In Transport
  - Local In Transport
- Introduce Manage Collections

Review – Cloud Runtime

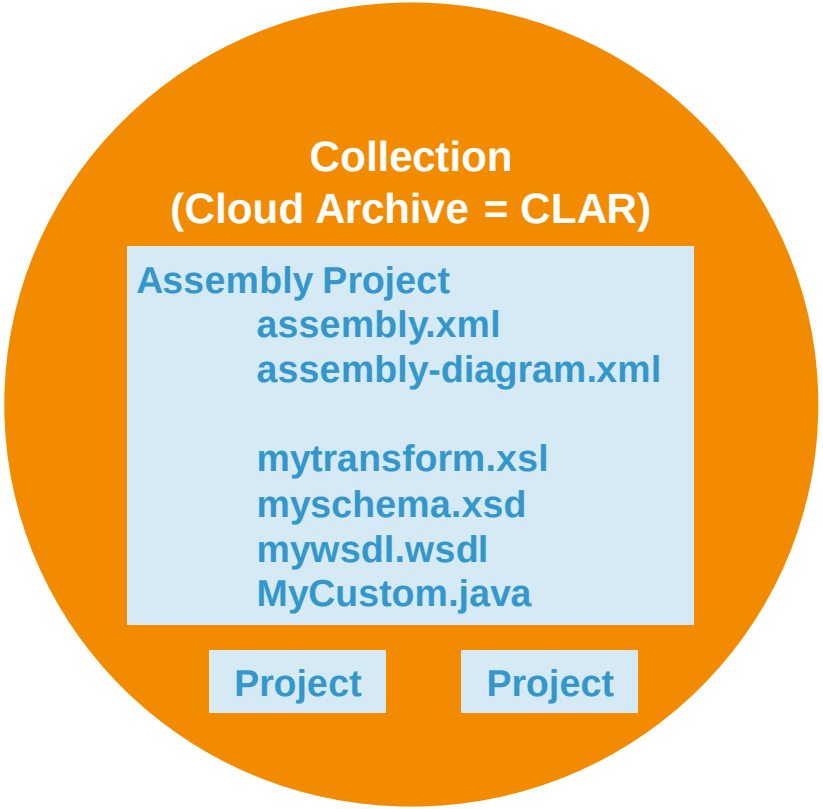
When the Workday Studio platform launches an integration, it allocates a *Cloud Runtime* to it.

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.



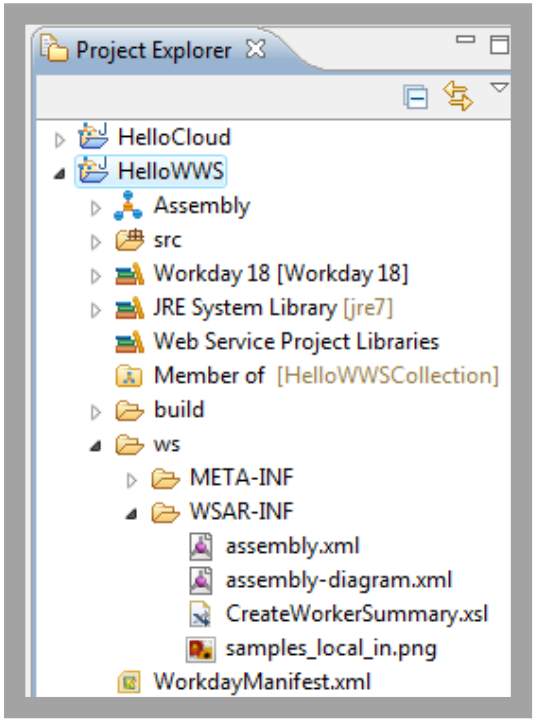
What's in a Collection?

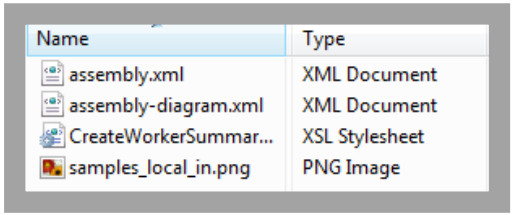
- Integrations are applications designed to integrate and run within Workday's web service environment.
- Studio organizes integrations into groups called collections and deploys the integrations to Workday's *Cloud Repository*. Workday's network hosts the Cloud Repository.

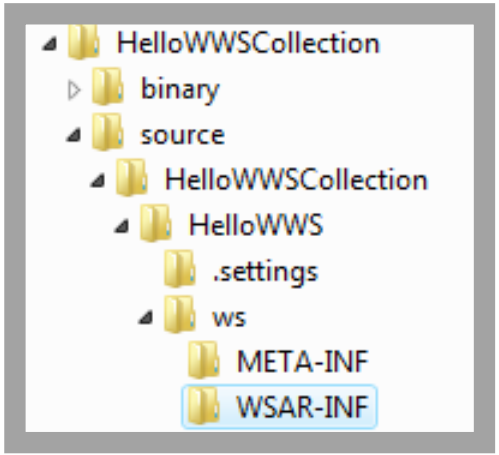


What Is in a CLAR File?

A benefit of using Workday Studio is that it takes the files created in the project and packages it into a structured deployment file called a .CLAR.







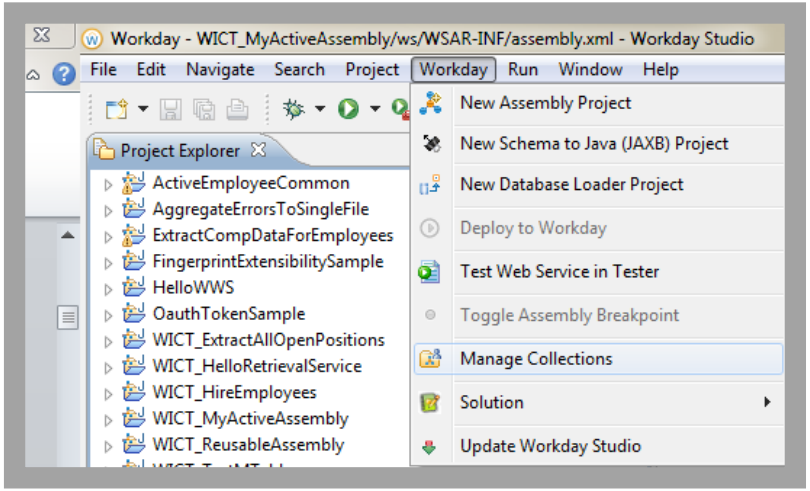
Structure of the Clar

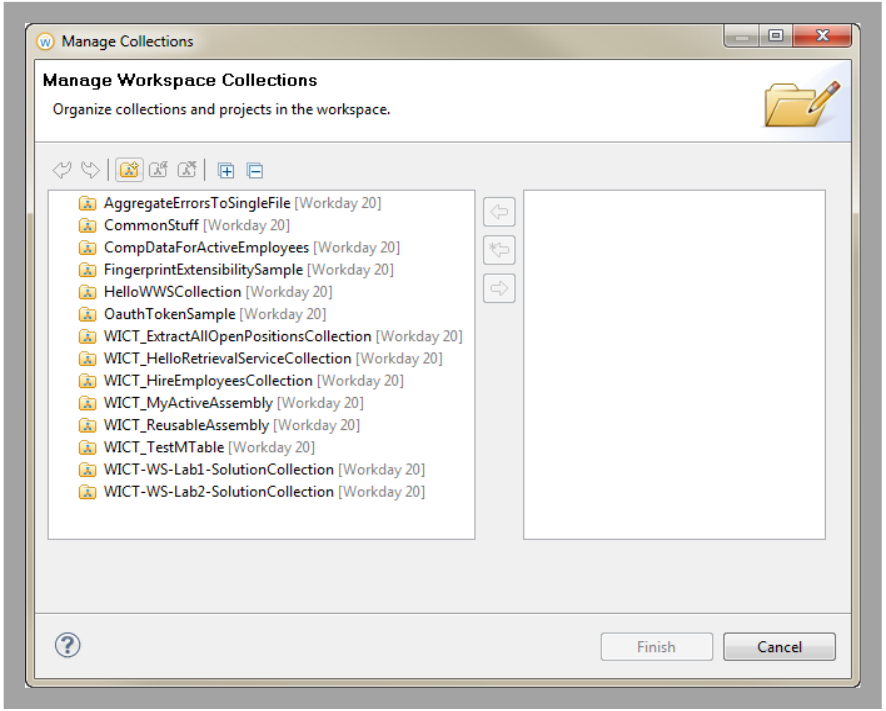
- Binary
  - Precompiled jars for execution
- Source:
  - Collection-name: This folder contains one or more projects.
    - Project-name(s): Each project contains the following sub-folders:
      - ✓ META-INF, which contains the MANIFEST.MF file.
      - ✓ WSAR-INF, which contains the assembly.xml and assembly-diagram.xml files and any additional files

Manage Collections

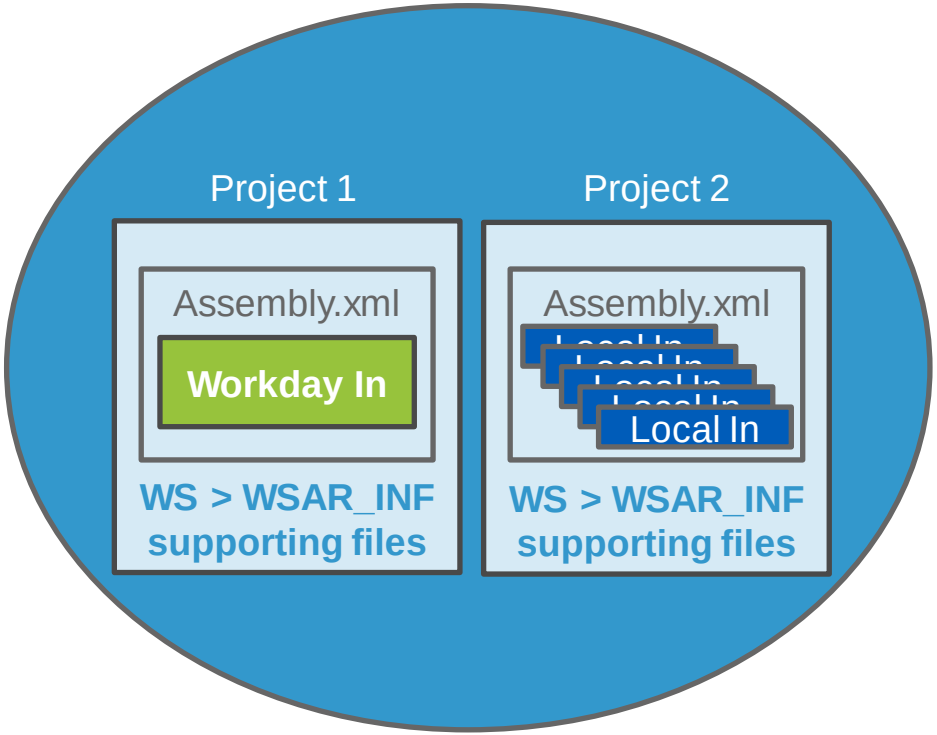
You can create, rename, remove, and organize the collections in your workspace.

Workday > Manage Collections





My .CLAR (Collection)



Sub-Assemblies

workday. | EDUCATION

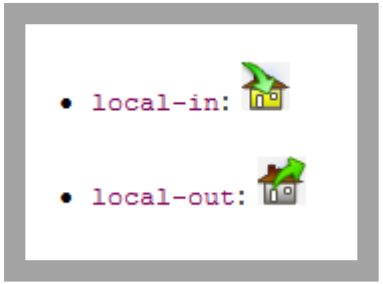
Learning Objectives

- What is a Subassembly
  - Workspace Components
  - Common Components
- Why use a Subassembly
- How Subassemblies work
  - Local In Transport
    - In Parameters (Properties)
    - Out Parameters (Properties)
  - Local Out Transport
    - Set In Parameter Values
- Implement an integration system using a subassembly

Sub-Assemblies

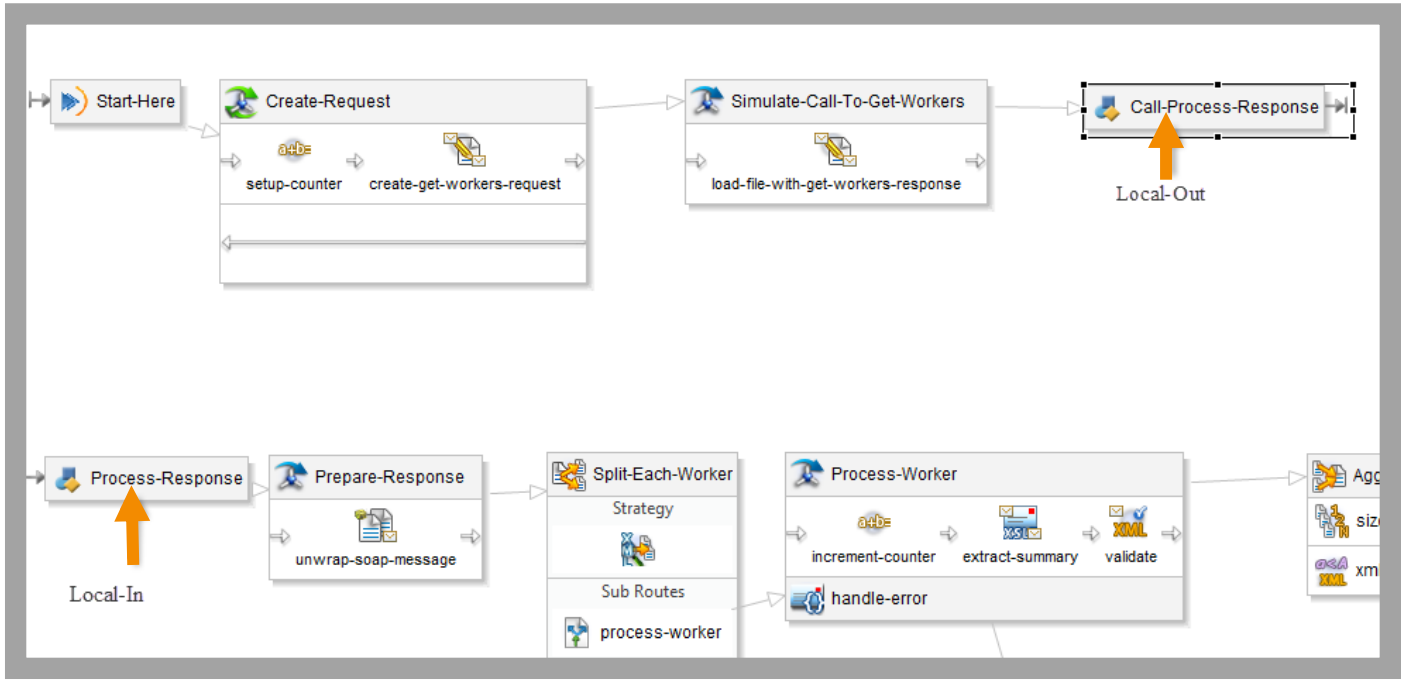
Workday Studio provides a way to develop modular sub-assemblies that assemblies can share and re-use, enhancing developer productivity. A sub-assembly is an assembly that contains a local-in transport as the first assembly component in the processing chain.

- The **local-in** defines an entry point to an assembly processing chain, which can be called from the same assembly or other assemblies using the **local-out** transport.
- As a developer, you should develop sub-assemblies if the processing that the assembly performs is likely to be reused by other applications.



Sub Assembly in Same Assembly as Workday Integration

- SubAssemblies can exist in their own project or on the same assembly as the integration that calls it.
- This is a way to logically separate processing.

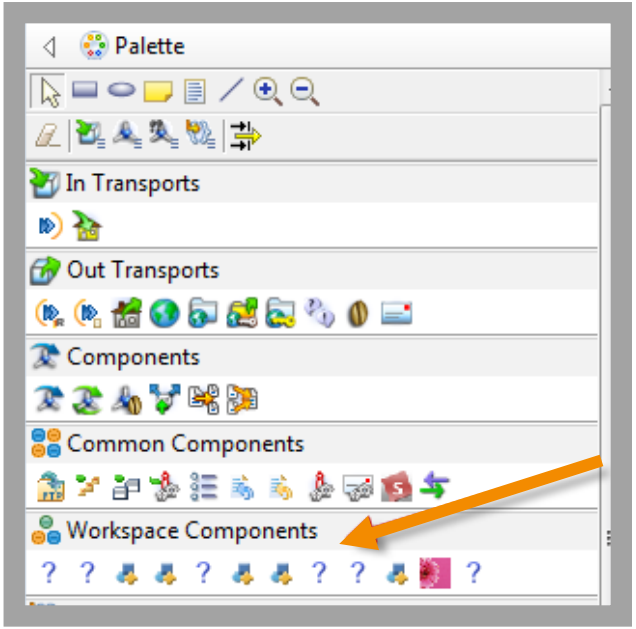


Workspace Components

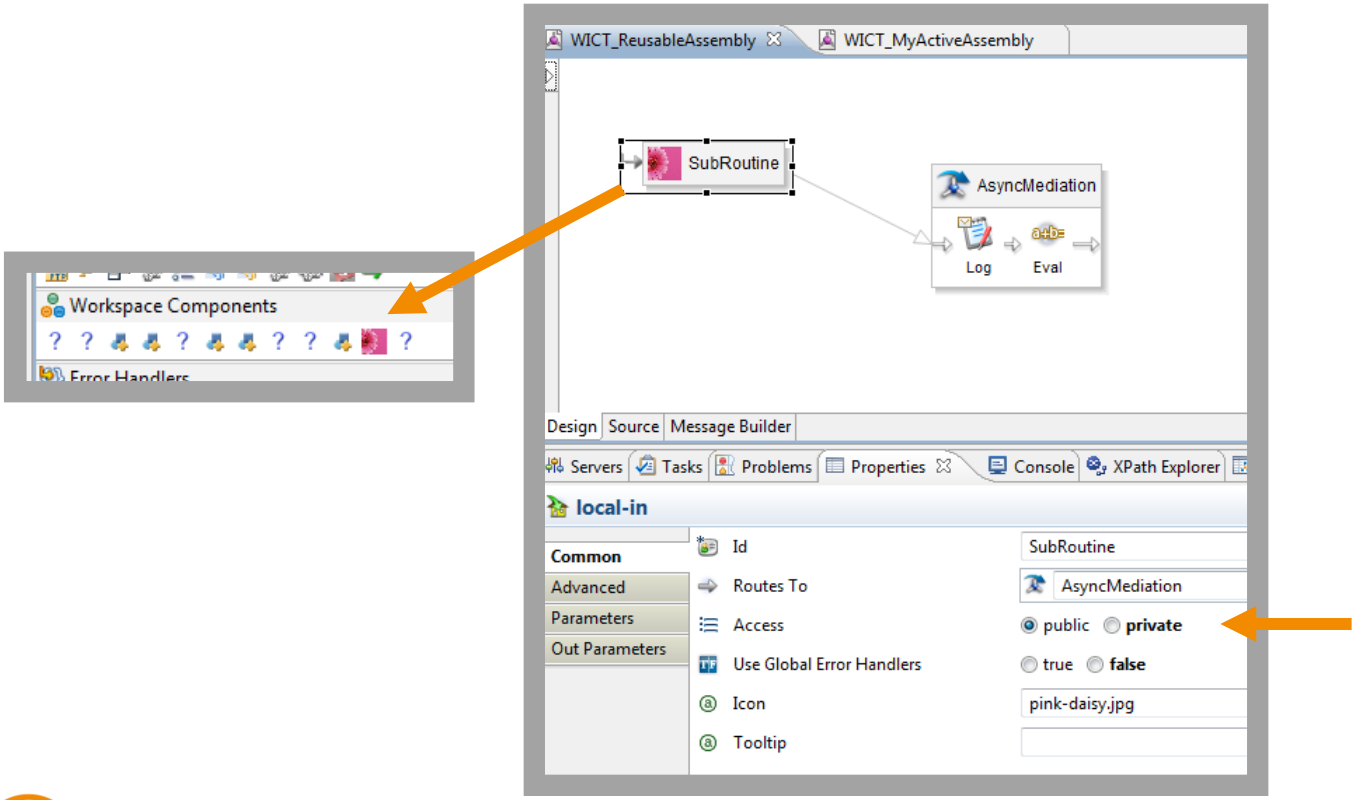
By placing only “Local-Ins” on an assembly, you create a sub-assembly.

- It can not be deployed on its own to Workday because it does not contain a Workday In transport.
- The sub-assemblies that you create in the current workspace will show up on the palette under “Workspace Components.”

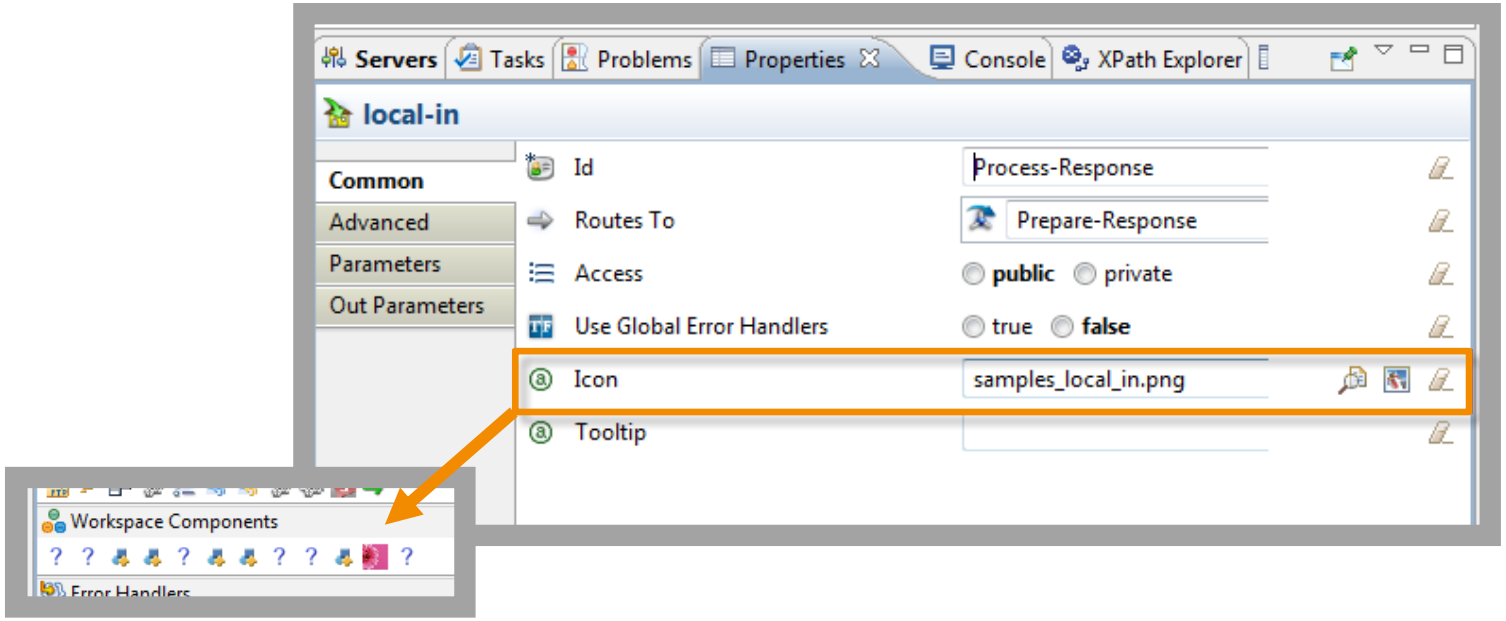
Note: Workday-delivered sub-assemblies show up on the Palette under “Common Components.”



Local In

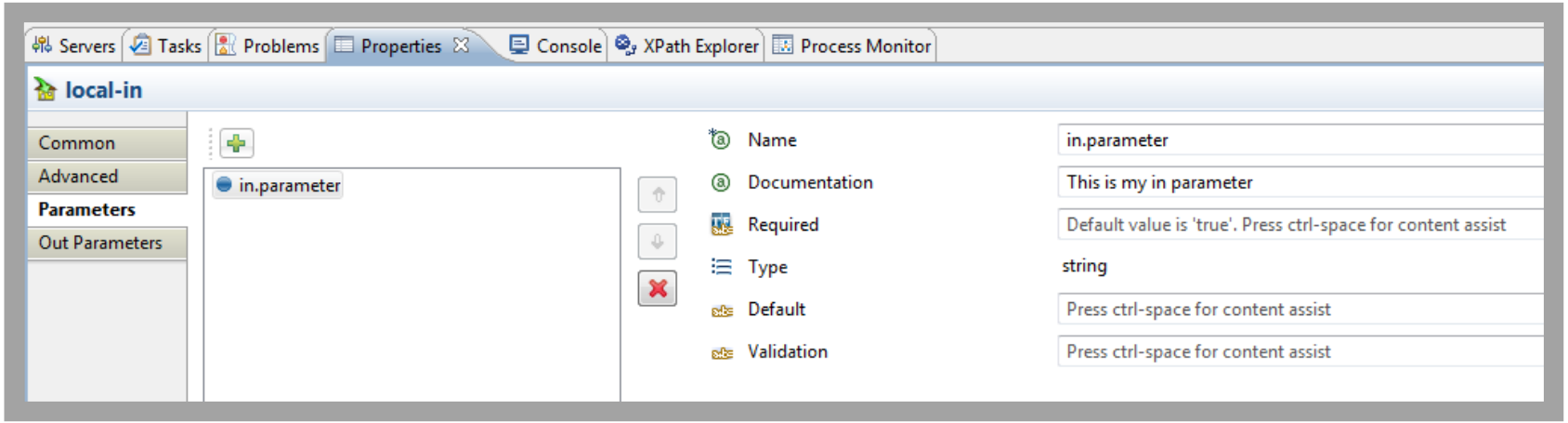


Icon



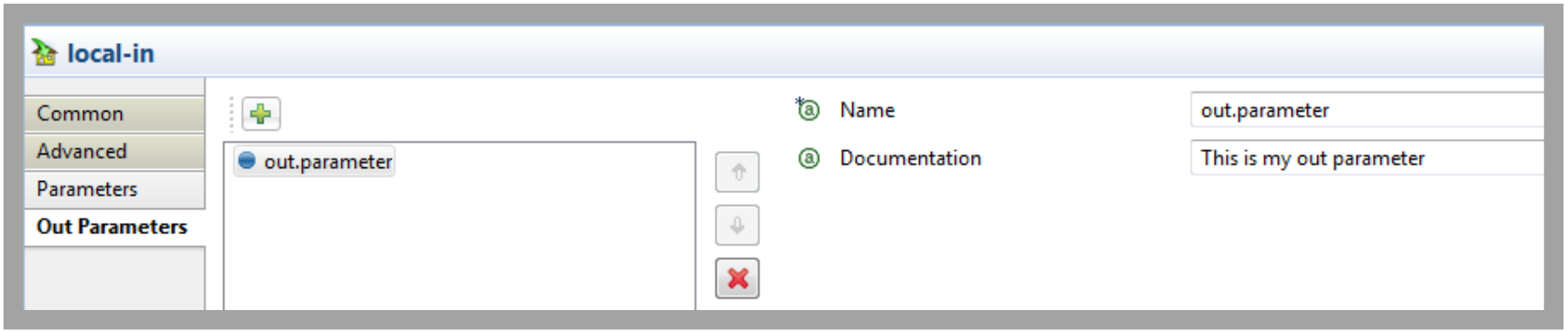
In Parameters (Properties)

- Parameters required for input can be defined as “Parameters.”
- These are configured on the local out and stored in the Mediation Context as properties.
- When the sub-assembly completes all values in the In Parameters are deleted from the Mediation Context.



Out Parameters (Properties)

- Out Parameters are properties set in the Mediation Context by the sub-assembly.
- When the sub-assembly completes, all values in Out Parameters persist and are still available in the Mediation Context.

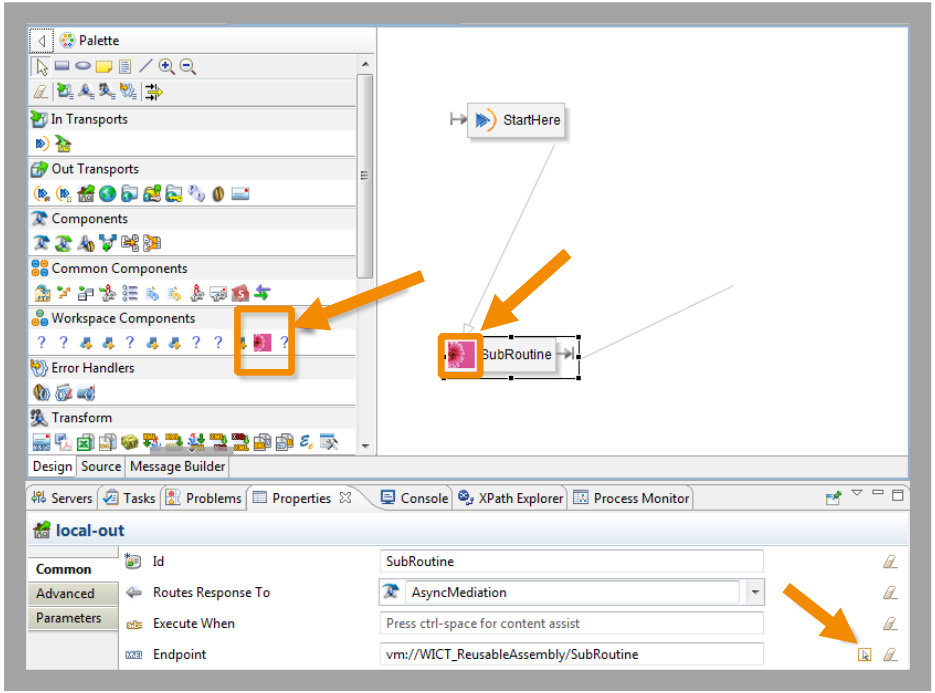


Calling a Sub-Assembly - Local Out

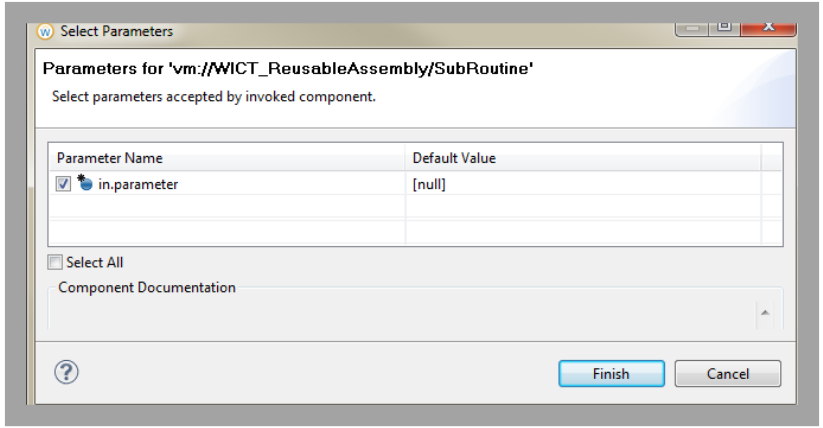
Two ways to add local-out in an assembly:

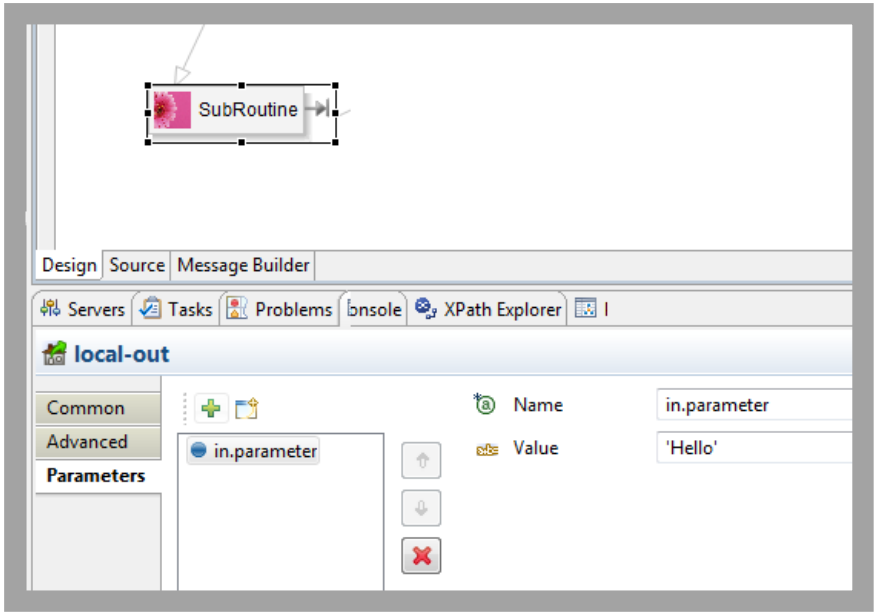
1. The easiest way is to drag the icon from the Workspace Components.
2. Drag the generic local out from Out Transports and configure the endpoint.

Additionally, make sure to save the Local In before you add the Local Out to an assembly.



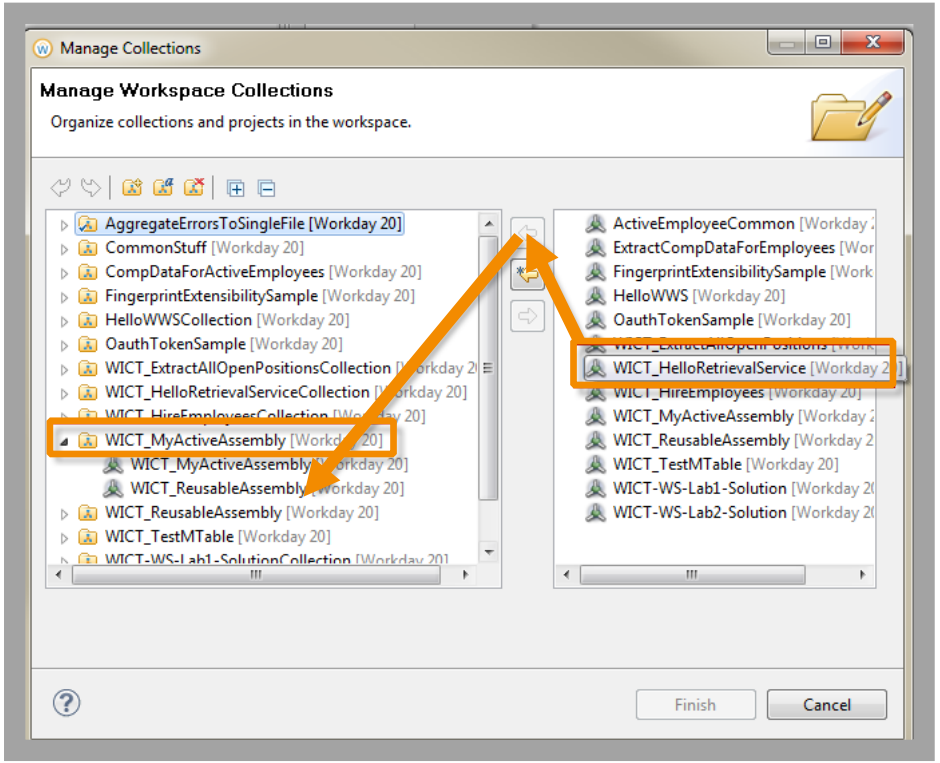
Local-Out Parameters (Properties)





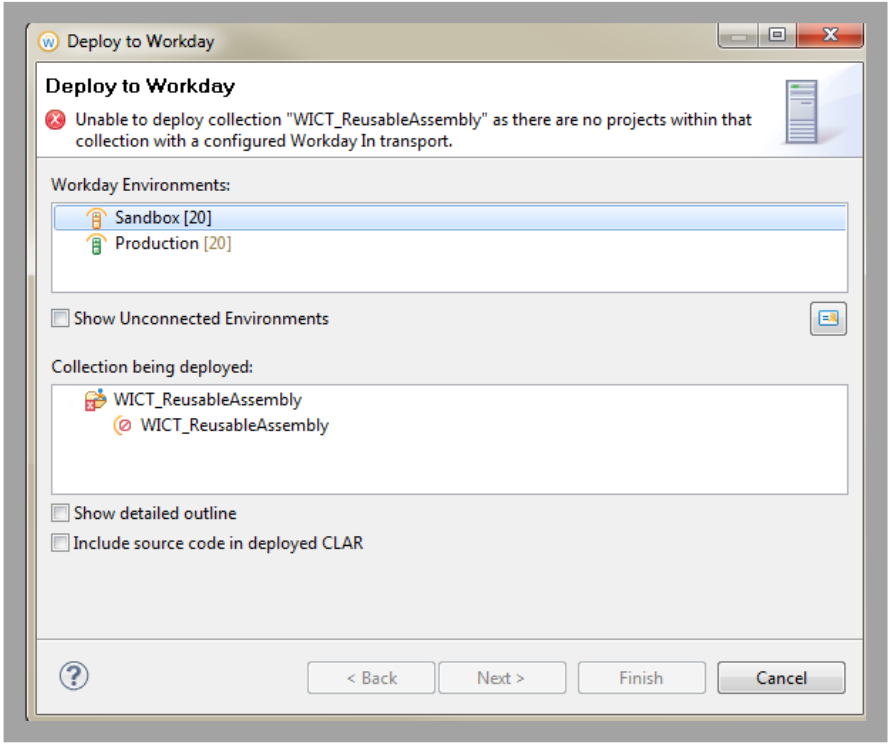
Managing Collection

- A Studio project can belong to multiple collections. Go to **Workday > Manage Collections** to view the collections in the workspace.
- You can then select workspace projects to add to the collection. You can put your re-usable components in their own projects, then include those projects in multiple collections.
- All components you want to access at run time must belong to the same collection.



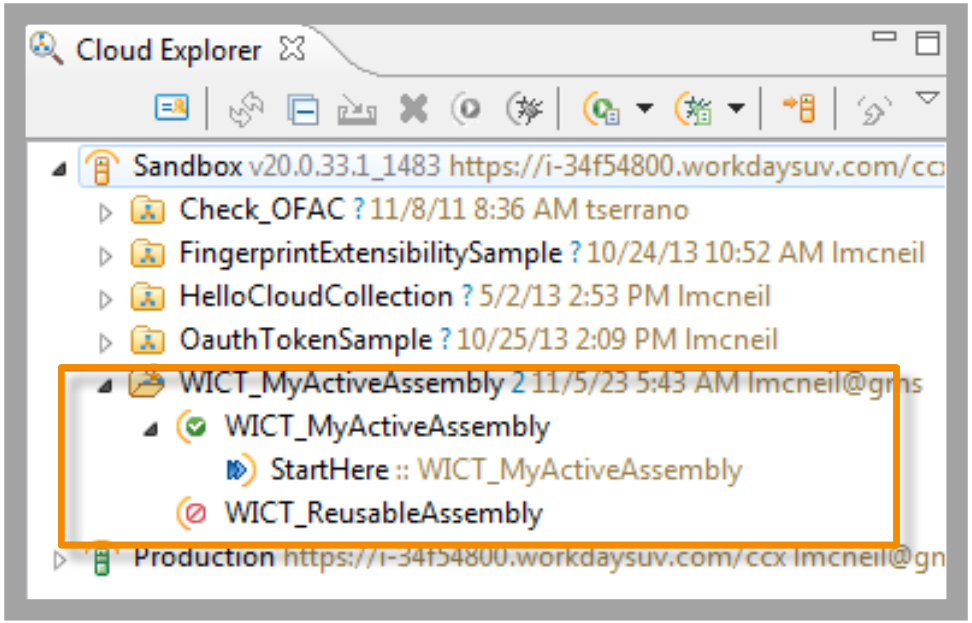
Deploy to Workday

You cannot deploy a collection that does not have a Workday-In transport.



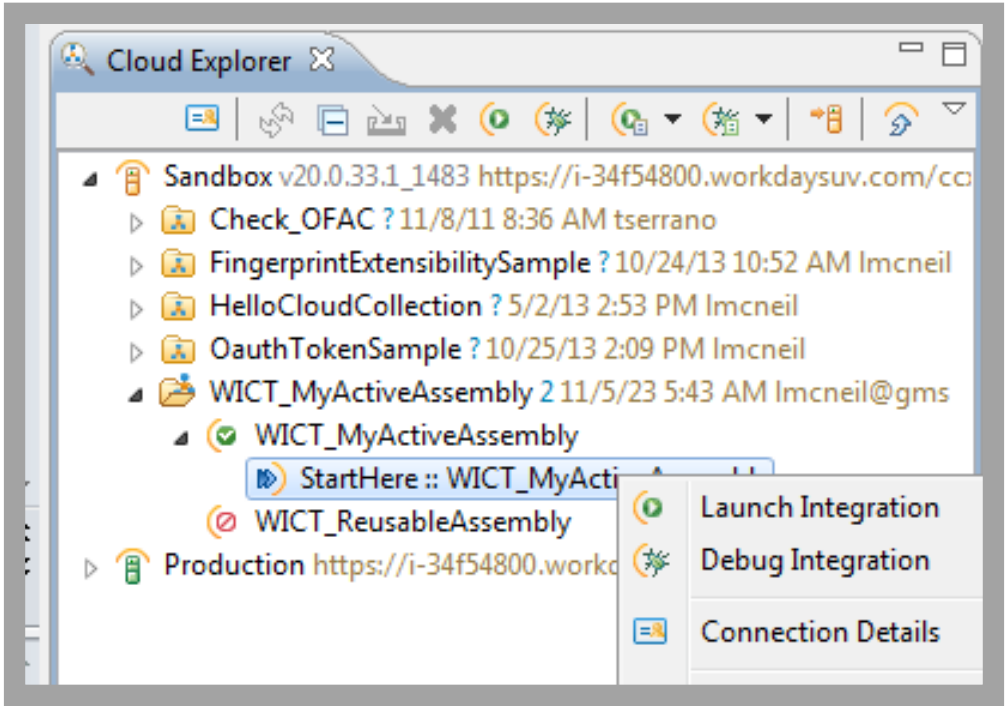
Deploy to Workday

Once the integration project is deployed, both the assembly and sub-assembly are visible in the Cloud Explorer view.



Launch the Integration

Launch the integration by right-clicking on the Workday-In transport.

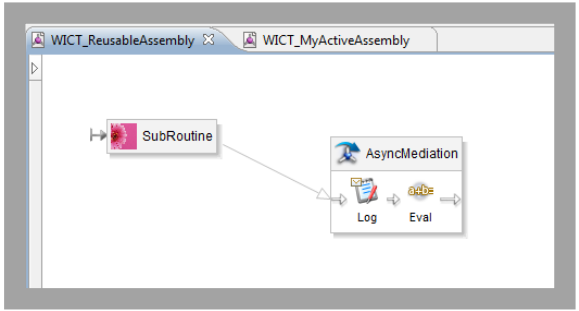


Activity 6

Sub-Assembly

Estimated Time: 20 Minutes

- Create an Active and a Reusable sub-assembly.
- Call the Reusable Assembly from the Active one.
- Don't forget to manage the collections before deploying.



workday. | EDUCATION

ducation resource is where learning

Please log in to your tenant:

Username:

Password:

Advanced Workday Studio

Day 2

Scalability

Learning Objectives

- Identify Scalability needs
- Processing Messages
  - Where does the data come from?
    - Outbound (Data From Workday)
      - ✓ RaaS vs. Soap API
      - ✓ PagedGet Common Component Usage
    - Inbound
      - ✓ External File
      - ✓ How large of a file?
      - ✓ What format?
      - ✓ How many records to add/update?
- Assembly components that implement Scalable integrations
  - Splitter
  - Aggregator
- Implement Splitter and Aggregator in Assembly

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.

Scalability Checklist

- Never apply the following assembly components to a large document.
  - Validate using XML Schema
  - Log Message; use the Store step instead if you need to store the entire message
  - Java Script
  - Text to Excel step

Never apply XPath expressions to large documents.

- Never use the XPath Splitter – the XML Stream Splitter must be used instead.
- XPath expression also appears in the XPath routing strategy and the Validate using XPath step. Never use these with large documents.

Never apply MVEL expressions to large documents.

- Find a way to extract the data you need from another smaller message.

Enable Streaming for Assembly components that have streaming switched off by default.

ETV and XTT components DO stream!

Streaming

- Reading a file sequentially and processing it one part at a time
  - Good strategy to handle large lists of data; i.e. Employee Data
- Reduced Memory Requirements
- Increased performance
- Some components stream, but you may have to enable it
- Some XSLT programs can stream (XSLT 3.0)
  - XSLT streaming is very fast but will take effort to set up

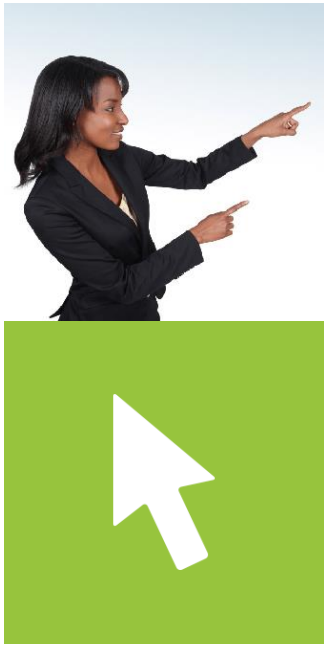
XSLT

- XSLT+ Component
- ETV and XTT now available
- Same as used in Document Transforms
- Faster than Splitter>XSLT>Aggregator pattern, but harder to set up
- Windowed Streaming
  - Stream in a single record, process normally, go on
  - Make sure one record will fit in memory
- Do not use Saxon namespace extension functions or instructions

Process Multiple Records

- The basic approach to ensuring that integrations are scalable is to use splitters/aggregators to split the document into individual records or to use pages as returned by public web services.
- The **splitter** is a mediation component that can process bulk messages containing more than one data record.

Using Splitters Together with Aggregators



- While splitters and aggregators can be used independently of each other, they have been designed to work together. When splitters and aggregators are used together, they form processing pairs.
- An aggregator automatically detects when an upstream splitter generates its last message. This causes the aggregator to automatically batch up any remaining messages using the pluggable collater, regardless of the batch strategy that is plugged into the aggregator. This ensures that no messages are left lying around after a splitter has finished splitting messages.

Common Methods for Getting Data

Inbound

- External File

Outbound

- RaaS
- Soap API

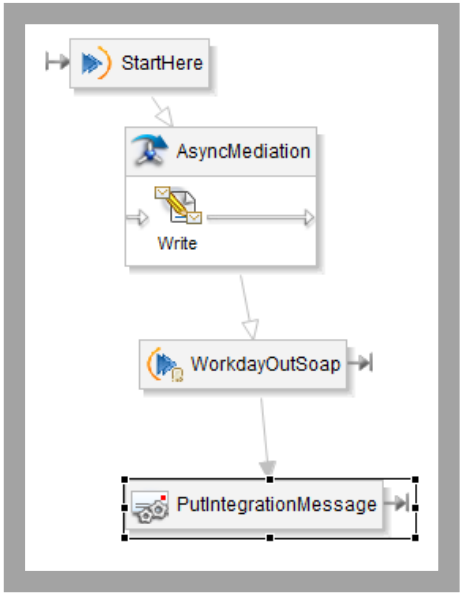
Getting Data – RaaS

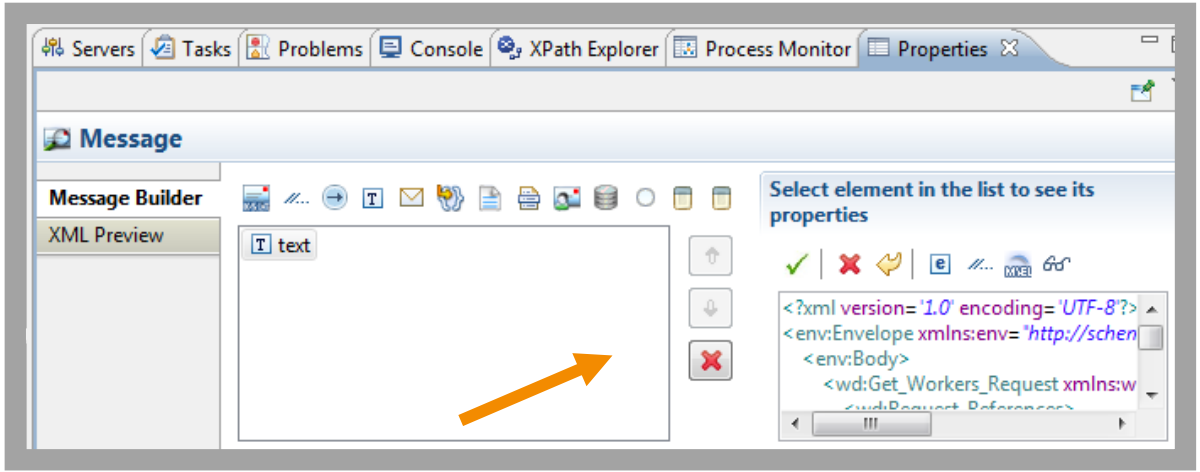
What is maximum size of xml output from a web service-enabled report?

Workday Web Services are better for large data

Getting Data – Soap API (Web Service Request)

When calling a web service, (not a custom report), you must “write” your own XML request. You can use the WWS tester to generate the request and then copy / paste into the Message Builder of the Write step.





Soap API: Request and Response

Request

```
<?xml version='1.0' encoding='UTF-8'?>
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
   <env:Body>
      <wd:Get_Workers_Request xmlns:wd="urn:com.workday/bsvc" wd:version="v23.1">
         <wd:Response_Filter>
            <wd:Page>1</wd:Page>
            <wd:Count>10</wd:Count>
         </wd:Response_Filter>
         <wd:Response_Group>
            <wd:Include_Personal_Information>true</wd:Include_Personal_Information>
            <wd:Include_Employment_Information>true</wd:Include_Employment_Information>
            <wd:Include_Qualifications>true</wd:Include_Qualifications>
         </wd:Response_Group>
      </wd:Get_Workers_Request>
   </env:Body>
</env:Envelope>
```

The request determines how many objects in response.

<wd:Count>10</wd:Count>

- What is maximum per page?
- What is minimum per page?
- What is the default if Count in not specified in request?
- What if I have more than the maximum to process in the tenant?

Soap API: Request and Response

Request

```
<?xml version='1.0' encoding='UTF-8'?>
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
   <env:Body>
      <wd:Get_Workers_Request xmlns:wd="urn:com.workday/bsvc" wd:version="v23.1">
         <wd:Response_Filter>
            <wd:Page>1</wd:Page>
            <wd:Count>10</wd:Count>
         </wd:Response_Filter>
         <wd:Response_Group>
            <wd:Include_Personal_Information>true</wd:Include_Personal_Information>
            <wd:Include_Employment_Information>true</wd:Include_Employment_Information>
            <wd:Include_Qualifications>true</wd:Include_Qualifications>
         </wd:Response_Group>
      </wd:Get_Workers_Request>
   </env:Body>
</env:Envelope>
```

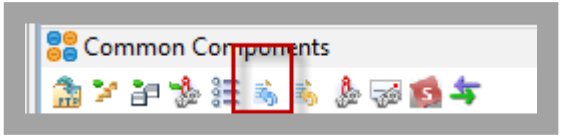
- What if you have more than 999 objects that you need to be returned in the response? For example, 1500 objects?

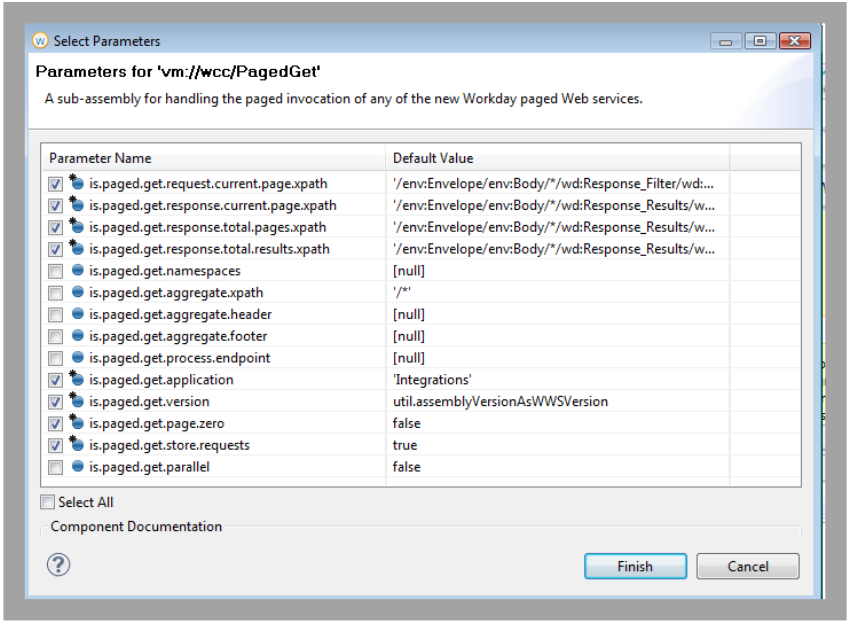
  You would have multiple calls to Workday for each page of results.

- How many calls can Workday Soap Out be configured to make?
- How would you deal with multiple requests / response pages to a single service?

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.
- Has both in and out parameters.
- Will explore in detail in a later section





Getting Data – Inbound

When an inbound file contains multiple records, it can be split using the splitter in order to process the data.

```
<root>
	<employee_entry>
		<employee_ID>21142</employee_ID>
		<email_address>testemail1@bogus.com</email_address>
	</employee_entry>
<employee_entry>
		<employee_ID>12345</employee_ID>
		<email_address>testemail2@bogus.com</email_address>
	</employee_entry>
<employee_entry>
		<employee_ID>21019</employee_ID>
		<email_address>testemail3@bogus.com</email_address>
	</employee_entry>
<employee_entry>
		<employee_ID>67890</employee_ID>
		<email_address>testemail4@bogus.com</email_address>
	</employee_entry>
</root>
```

Using the Splitter / Aggregator

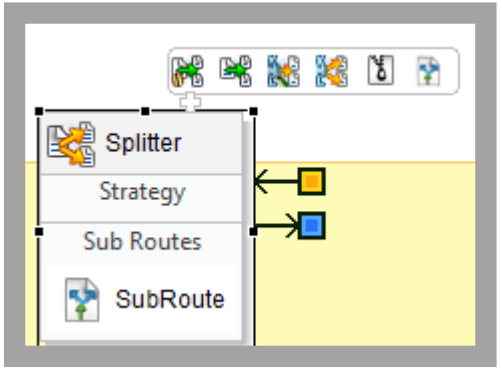
workday. | EDUCATION

Splitter

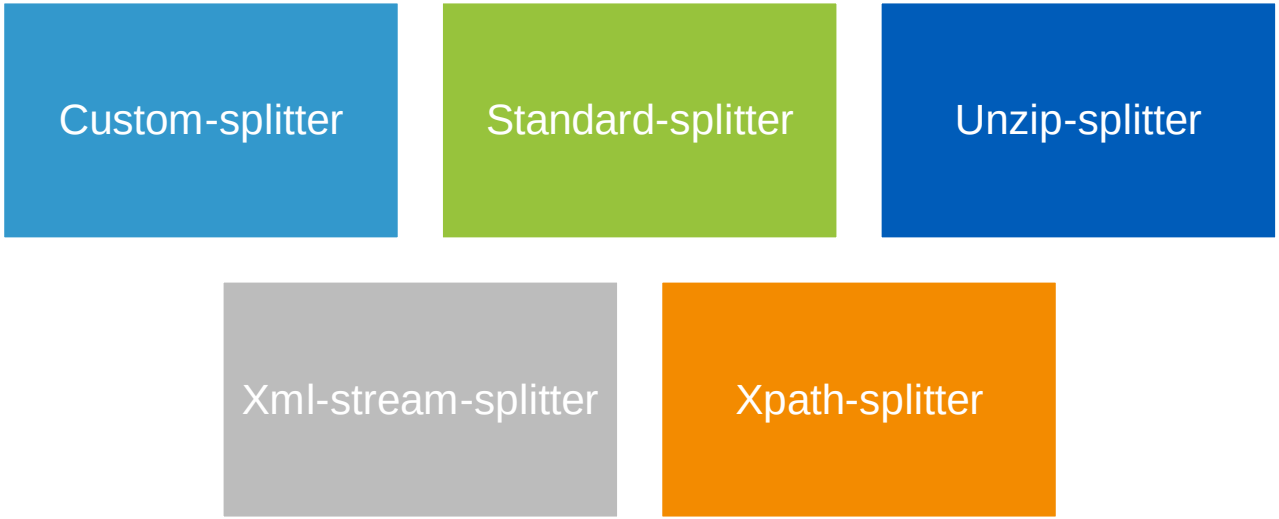
- Once routed to the splitter, each record is sent as a new mediation message to the data destination, which is the splitter's first sub-route.
- You can define the logic that extracts the data from bulk messages using a custom-splitter strategy, or in many cases, you can use a standard-splitter strategy.

Examples: When data records comprise a fixed number of lines or when they can be described in terms of a regular expression.

- 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.

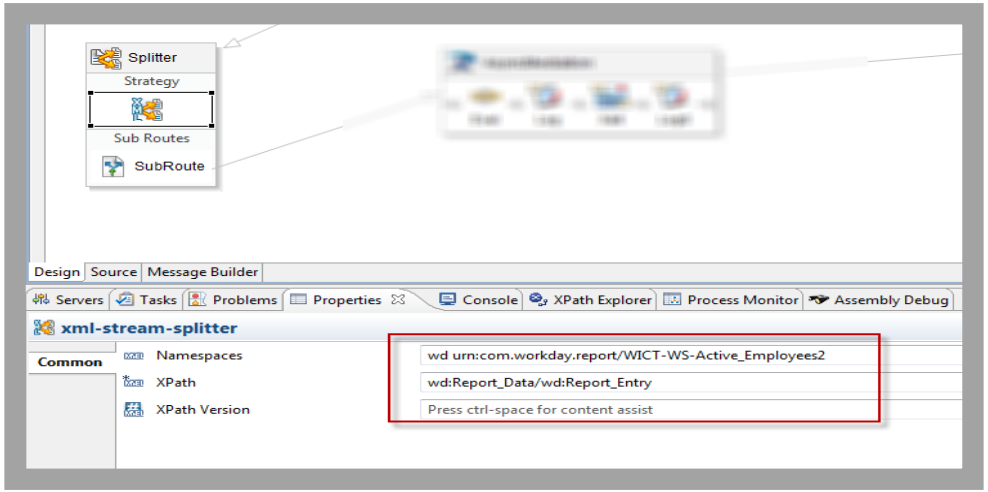


Splitter Strategies



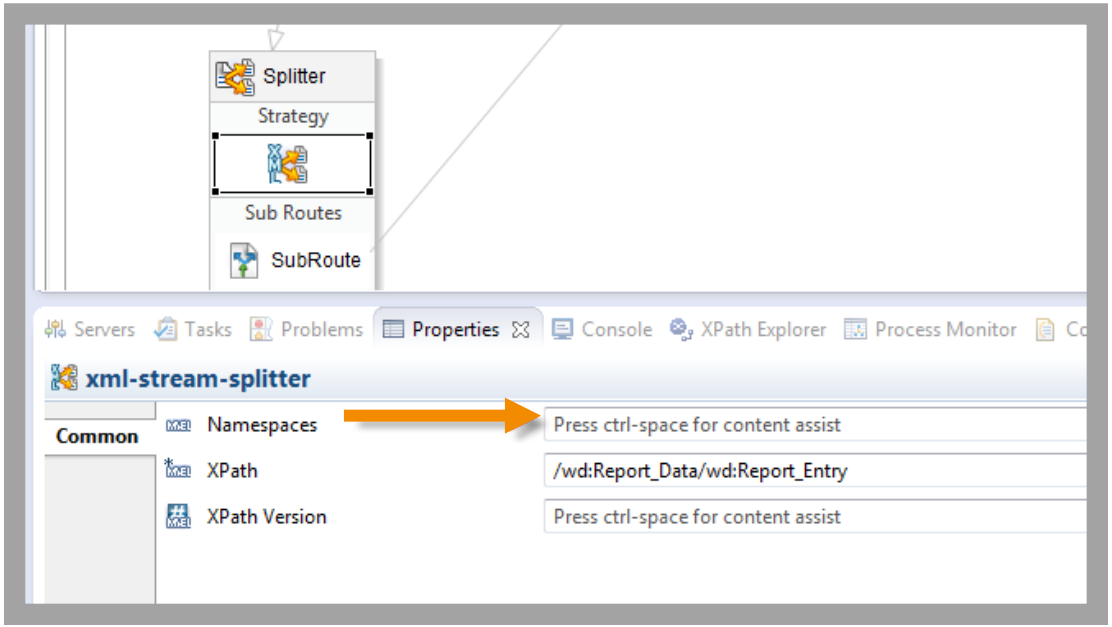
Splitter Strategy – XML Stream Splitter

- When using the xpath-splitter or xml-stream-splitter, you can optionally specify a namespace. If not specified, the default namespace (for the **wd** alias) of the assembly is **wd urn:com.workday/bsvc**.
- However, for custom reports, the Workday application uses the same wd: prefix to map to the Namespace **urn:com.workday.report/report-name**.



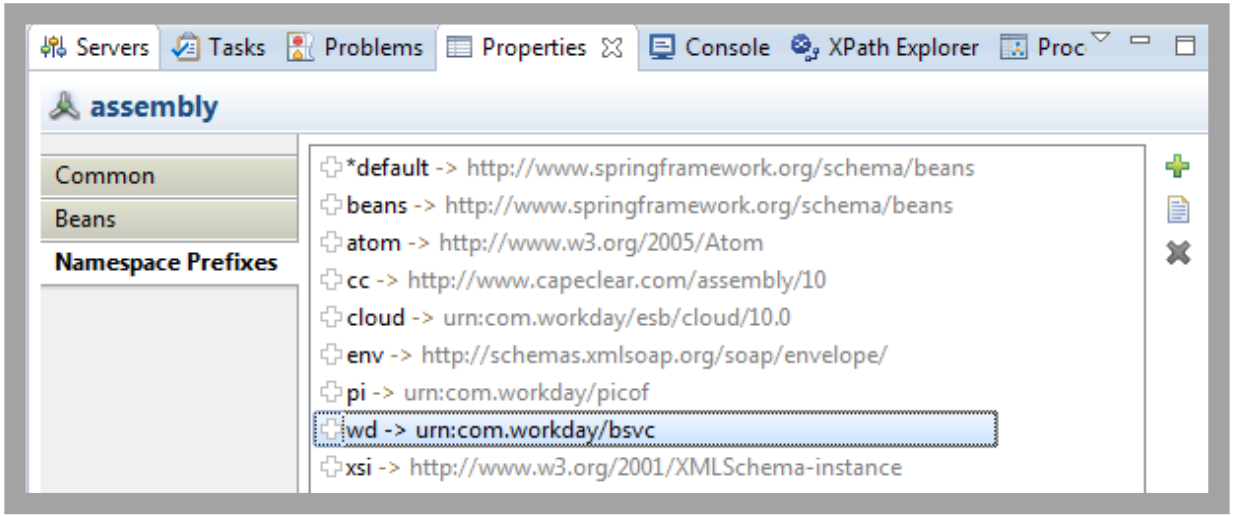
XML Splitter Strategy – Namespace

If you don't specify a Namespace, you can use a generic reference to select the correct element in the message.

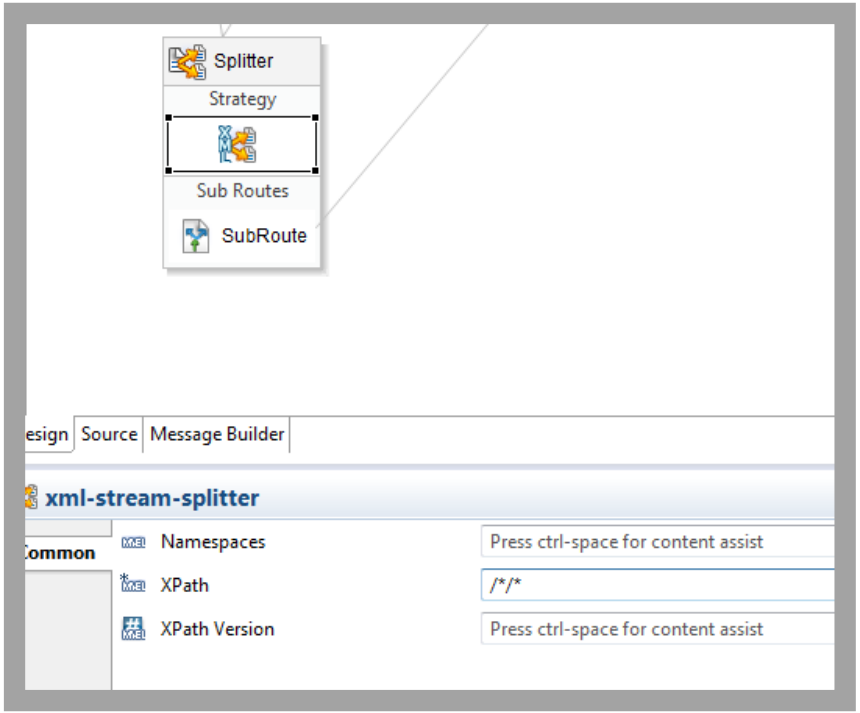


Assembly Namespace

- If you require a different namespace prefix (i.e., one that is commonly used within your organization), you can add / edit the namespace prefix for the new declaration. Navigate to **Properties > Namespace Prefixes** for the assembly component.
- For wd, the default namespace is **urn:com.workday/bsvc**.



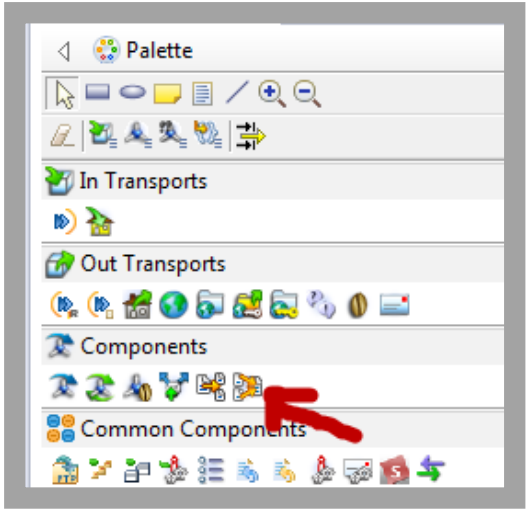
Alternative Configuration /*/*



- The abbreviated syntax is more compact and allows XPaths to be written and read easily using intuitive and, in many cases, familiar characters and constructs.
- The full syntax is more verbose, but allows for more options to be specified, and is more descriptive if read carefully.
- This method, while sometimes convenient, is not scalable and can sometimes make debugging significantly more difficult.

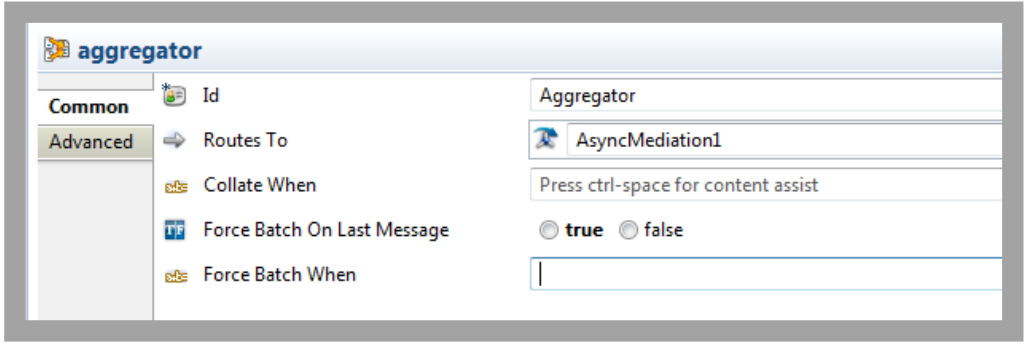
Aggregator Component

- Concatenates mediation messages into a single batch message and routes it to the configured mediation destination.
- You can configure a custom, size, or time-based batch strategy and a collater strategy that defines how messages are actually aggregated.



Aggregator

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



Aggregator Configurable Options

force-batch-when:

- Allows you to specify an MVEL expression that controls when an aggregator adds messages to a batch. This expression is evaluated each time the aggregator is visited. When true, the messages are aggregated into a batch. If not specified, the batch-strategy determines when to add messages to a single aggregated message batch.

collate-when:

- Allows you to specify an MVEL expression, which is evaluated for each received message and, if true, is added to the collator. If not, it is not collated. This means that the process determines at runtime whether an individual message gets collated.

force-batch-on-last-message:

- Overrides the batch strategy and forces batching when an upstream splitter component indicates, after splitting, that the current message is the last message.

Aggregator: Batch Policy

The Workday Integration Server determines whether an aggregator batches up the current set of collated messages based on a batch strategy.

Example: For the size-batch strategy, you can force batching to be a certain size.

The Batch Policies include:

Size-Batch-strategy:

- The batch strategy defines when to close the current batch and start a new one. This strategy determines how many messages the batch message should contain (-1 is unlimited).

Time-Batch-strategy:

- The batch strategy defines when to close the current batch and start a new one based on a specified time period.

To add a strategy, click the corresponding icon on the context toolbar that appears above the component when you hover over it using the mouse.

Aggregator: Collater

- Collators enable you define how to batch the individual messages.
- Collators include:

message-content-collater:

- Defines how to aggregate non-XML messages.

xml-message-content-collater:

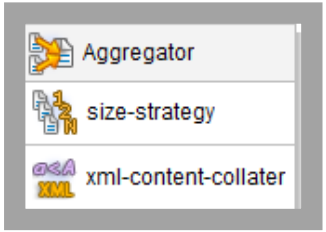
- Defines how to aggregate XML document content in an XML message stream.

zip-file-collater:

- Defines how to aggregate multiple files into a single ZIP file.

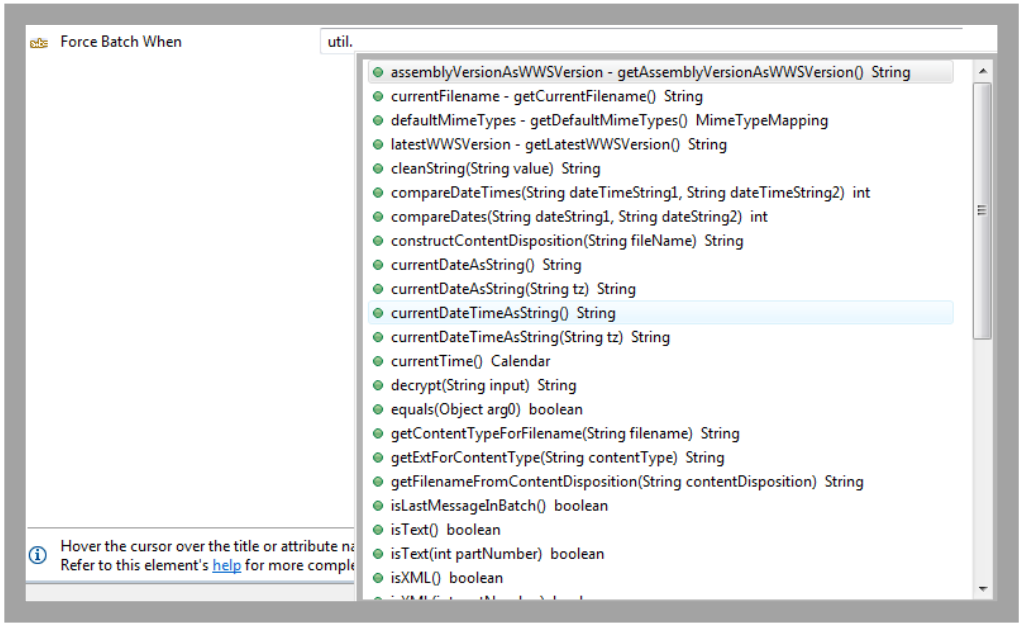
mtable-collater:

- Defines how to aggregate MTable rows in a new MTable instance, where an MTable is an in-memory table of data.



Util

The Util variable is bound to an instance of the MVELUtilHelper utility class, which provides handy functions for use within MVEL expressions such as time and date manipulation and XML testing.



Activity 7

Using Splitter / Aggregator and XSLT

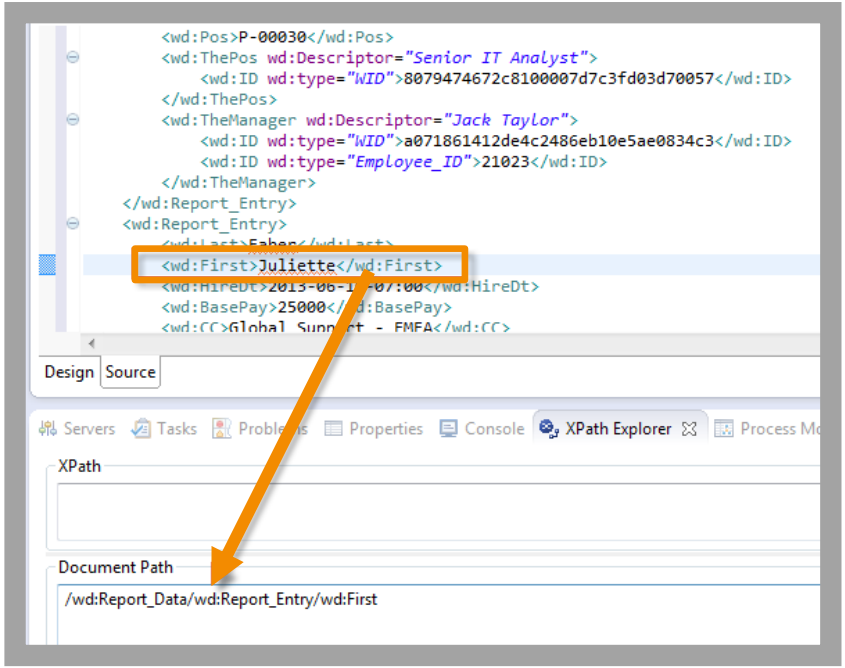
Estimated Time: 20 Minutes

You will modify the MyHelloRaaS assembly to take the output of the report and split the response. The split processing should be placed in a subassembly on the same diagram. As you process the report entry, apply a transformation to each record to make a summary. Once all records are processed, aggregate the summaries together in a final output file. The summary should include the Employee ID, Hire Date, Name, Location, and Gender. **You will need to add Gender to the report in Workday.**

```
<?xml version="1.0" encoding="UTF-8"?>
- <Summary>
  - <wd:WorkerSummary xmlns:wd="urn:com.workday.report/WICT_WS_Active_Employees2">
      <wd:ReferenceID>21246</wd:ReferenceID>
      <wd:HireDate>2011-05-02</wd:HireDate>
      <wd:Name>David Martinez</wd:Name>
      <wd:Location>Dallas</wd:Location>
      <wd:Gender>Male</wd:Gender>
    </wd:WorkerSummary>
  + <wd:WorkerSummary xmlns:wd="urn:com.workday.report/WICT_WS_Active_Employees2">
  + <wd:WorkerSummary xmlns:wd="urn:com.workday.report/WICT_WS_Active_Employees2">
  + <wd:WorkerSummary xmlns:wd="urn:com.workday.report/WICT_WS_Active_Employees2">
  + <wd:WorkerSummary xmlns:wd="urn:com.workday.report/WICT_WS_Active_Employees2">
</Summary>
```

Review: What Is XPath?

- **XPath**, the XML Path Language, is a query language for selecting elements from an XML document.
- You can load an XML file into Studio and use the **XPath Explorer** view to determine the XPATh for an element.

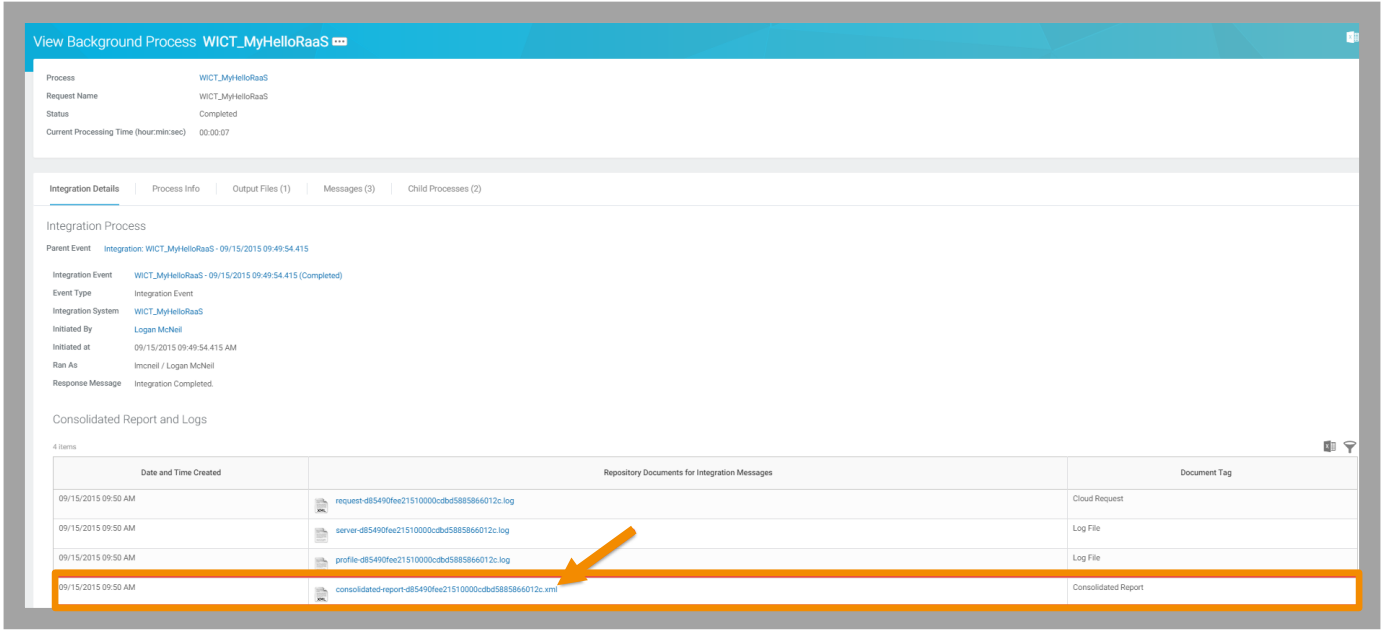


Consolidated Report Viewer

workday. | EDUCATION

Consolidated Report Viewer – Motivations

Consolidated Reports (CR) contain performance information that has been difficult to access in its xml format.



Consolidated Report Viewer

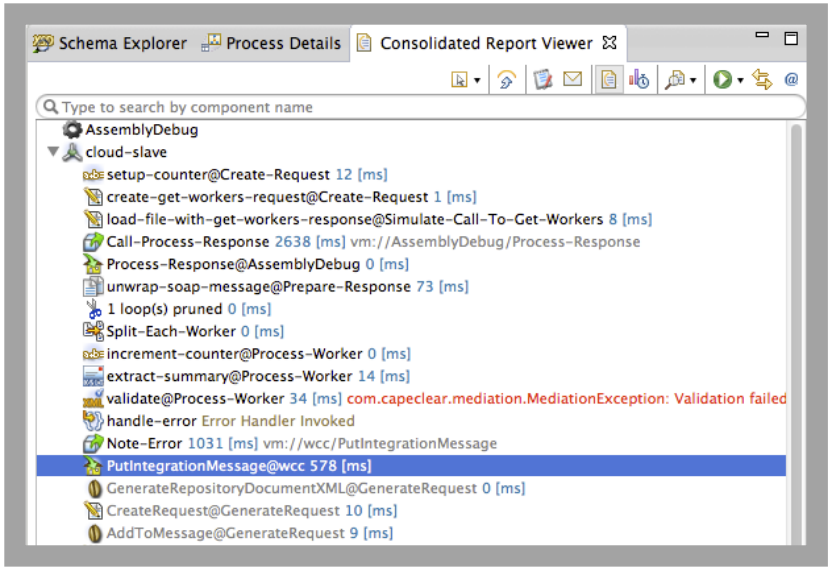
View consolidated report files in a Studio-based graphical interface that enables immediate viewing of performance information, re-playing of integrations, and linking to an assembly.

Benefits:

- Directly access performance information such as number of calls per web service or step, average, and total times.
- Determine which steps require the most memory.
- Re-play integration on assembly itself (with performance-timing data displayed) to visualize how integration is executed).

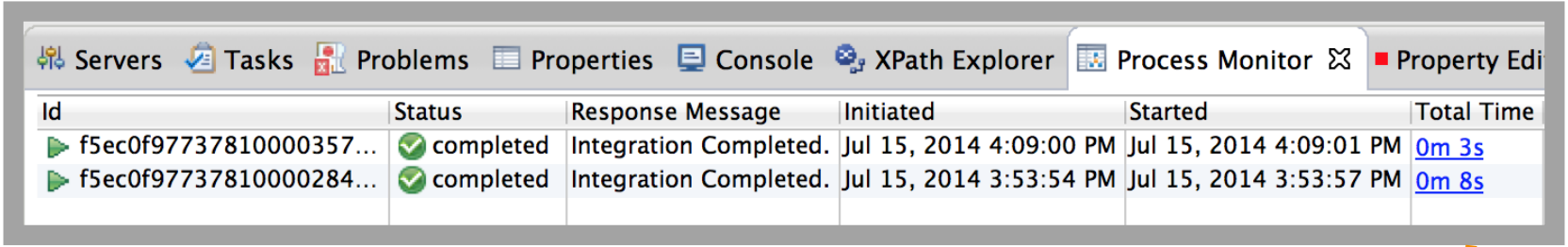
Consolidated Report Viewer – Overview

- The viewer is available from Process Monitor: right-click on **View Consolidated Report**.
  - View log file and request message.
  - Filter by type of step.
- You can also open consolidated reports not linked to by launching an integration from Studio (e.g., downloaded from application).



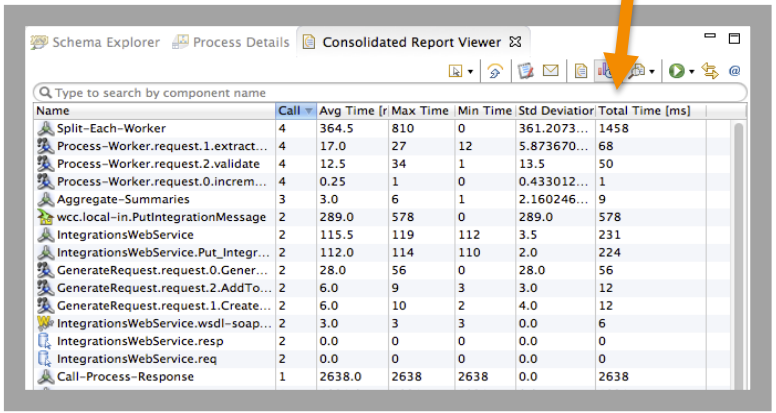
Consolidated Report Viewer – Details on Performance

- Process Monitor displays the total time for an integration.



Id	Status	Response Message	Initiated	Started	Total Time
f5ec0f97737810000357...	completed	Integration Completed.	Jul 15, 2014 4:09:00 PM	Jul 15, 2014 4:09:01 PM	0m 3s
f5ec0f97737810000284...	completed	Integration Completed.	Jul 15, 2014 3:53:54 PM	Jul 15, 2014 3:53:57 PM	0m 8s

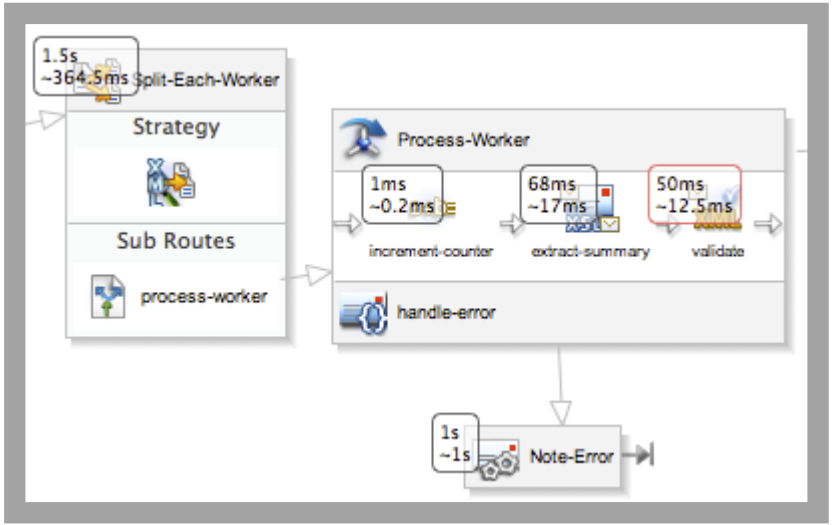
- The total time links to web service statistics.
  - Configurable columns (in preferences)
  - Sortable by calls, time, etc.



Name	Call	Avg Time [r	Max Time	Min Time	Std Deviation	Total Time [ms]
Split-Each-Worker	4	364.5	810	0	361.2073...	1458
Process-Worker.request.1.extract...	4	17.0	27	12	5.873670...	68
Process-Worker.request.2.validate	4	12.5	34	1	13.5	50
Process-Worker.request.0.increm...	4	0.25	1	0	0.433012...	1
Aggregate-Summaries	3	3.0	6	1	2.160246...	9
wcc.local-in.PutIntegrationMessage	2	289.0	578	0	289.0	578
IntegrationsWebService	2	115.5	119	112	3.5	231
IntegrationsWebService.Put_Integr...	2	112.0	114	110	2.0	224
GenerateRequest.request.0.Gener...	2	28.0	56	0	28.0	56
GenerateRequest.request.2.AddTo...	2	6.0	9	3	3.0	12
GenerateRequest.request.1.Create...	2	6.0	10	2	4.0	12
IntegrationsWebService.wsdl-soap...	2	3.0	3	3	0.0	6
IntegrationsWebService.resp	2	0.0	0	0	0.0	0
IntegrationsWebService.req	2	0.0	0	0	0.0	0
Call-Process-Response	1	2638.0	2638	2638	0.0	2638

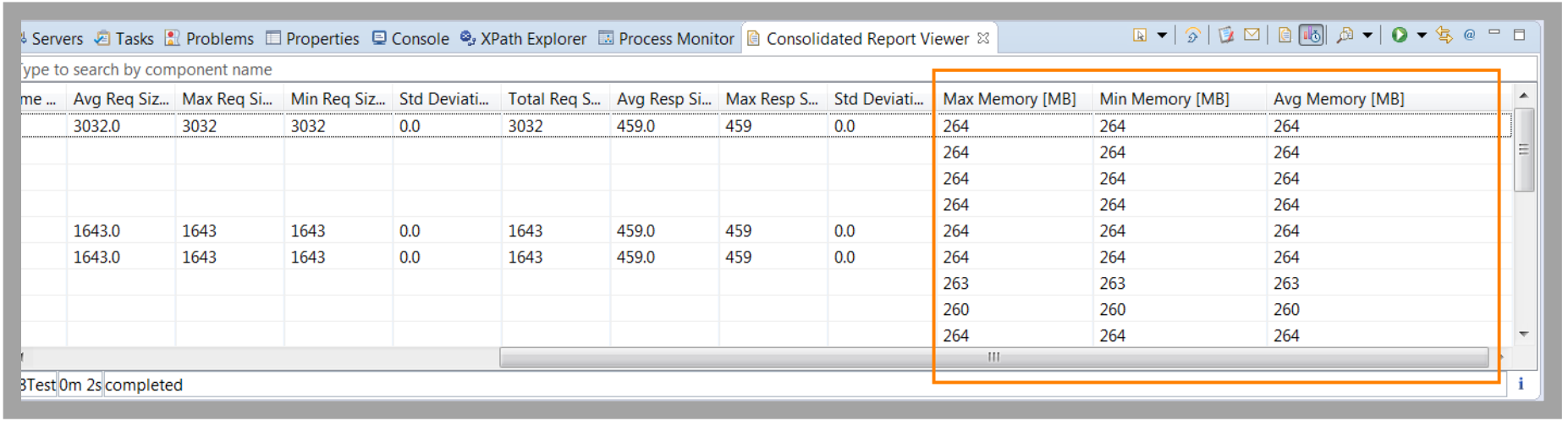
Consolidated Report Viewer – Other Features

- Consolidated report links to an assembly – highlighting a line in CR highlights corresponding component in assembly.
- The assembly re-player steps through integration and highlights each step along the way.
- The toggle button to show performance info on assembly itself.



Memory Usage in Consolidated Report Viewer

- WD 28 Has improved the Memory Stats found in the CRV



me ...	Avg Req Siz...	Max Req Si...	Min Req Siz...	Std Deviati...	Total Req S...	Avg Resp Si...	Max Resp S...	Std Deviati...	Max Memory [MB]	Min Memory [MB]	Avg Memory [MB]
	3032.0	3032	3032	0.0	3032	459.0	459	0.0	264	264	264
									264	264	264
									264	264	264
									264	264	264
	1643.0	1643	1643	0.0	1643	459.0	459	0.0	264	264	264
	1643.0	1643	1643	0.0	1643	459.0	459	0.0	264	264	264
									263	263	263
									260	260	260
									264	264	264

Consolidated Report Viewer – Considerations

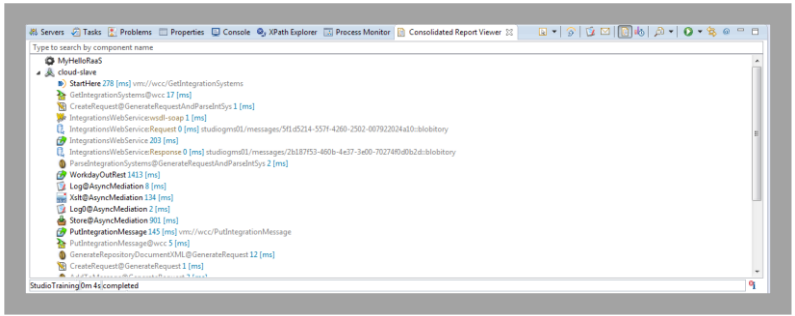
1. The viewer attempts to reconcile the consolidate report and assembly, but if the assembly has been edited since being deployed/launched, the viewer may be unable to properly link the report of an old assembly for re-playing.
2. Loops may be pruned (e.g., re-player and CR won't have individual details if you execute a loop many times).
3. The steps are not broken out in hierarchy – working on enhancement.

Activity:Consolidated Report Viewer

Consolidated Report Viewer

Estimated Time: 10 Minutes

In this activity, you will view the consolidated report for MyHello RaaS in the consolidated report viewer.



Workday-In Transport Services:

Integration Attributes, Integration Maps, and Sequence Generator

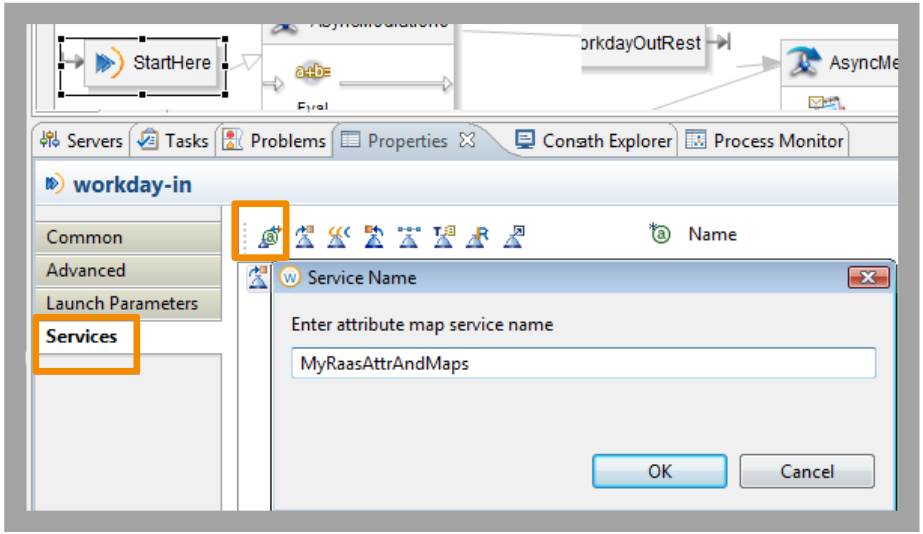
Learning Objectives

- Identify Workday-In Services
- Create and Configure:
  - Integration System Attributes
  - Integration System Maps
  - Sequence Generators
- Access Attributes, Maps, and Sequence Generator values in assembly using MVEL expression.

Integration System Attributes & Maps

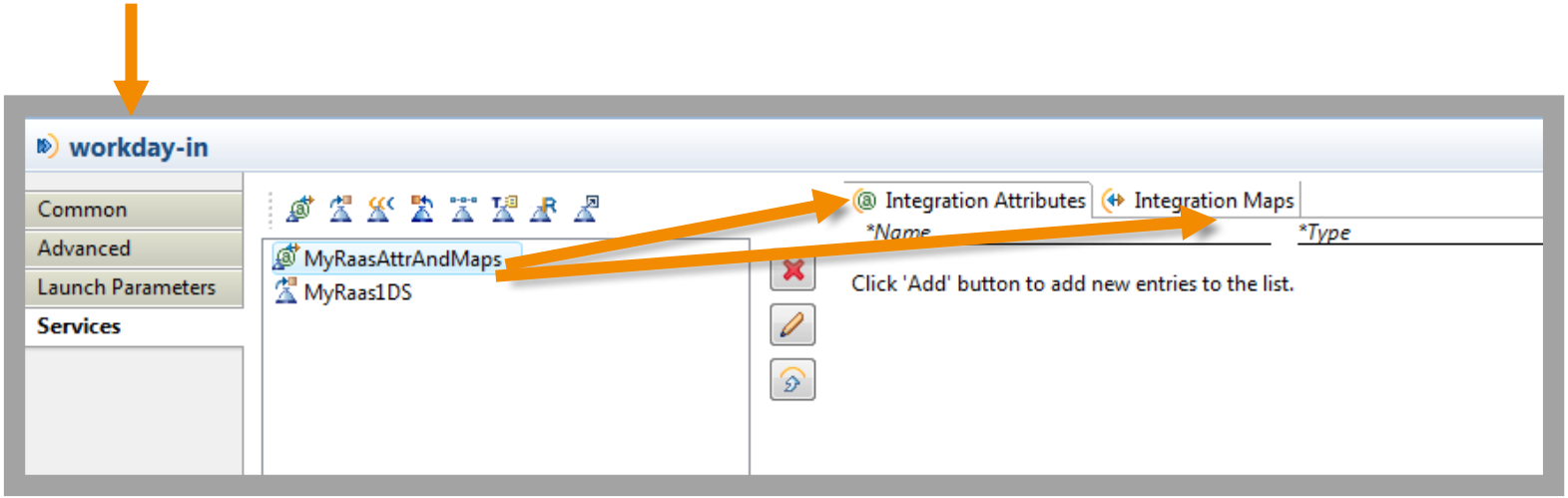
1. Attributes are values required for the integration that a functional user may need to change occasionally.
2. Maps are mappings of values in Workday to their equivalent values in another system.

You define services as part of the workday-in, but you configure their runtime values within the Workday application. You must first deploy the assembly, then log in to the Workday application and configure values for the service.



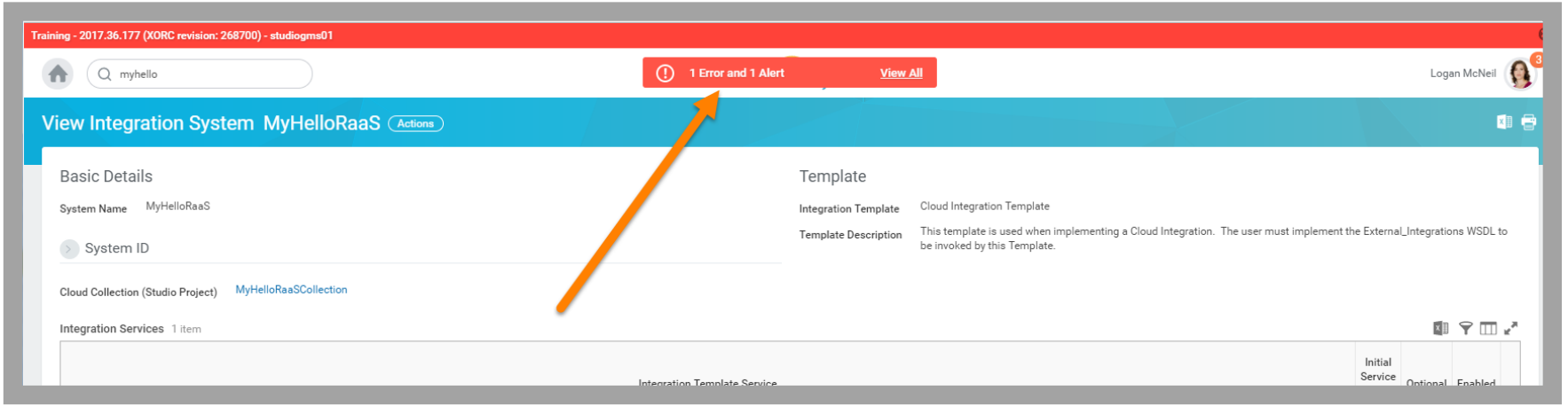
Attributes and Maps Tabs

Go to properties on Workday-In, then view the Services tab.



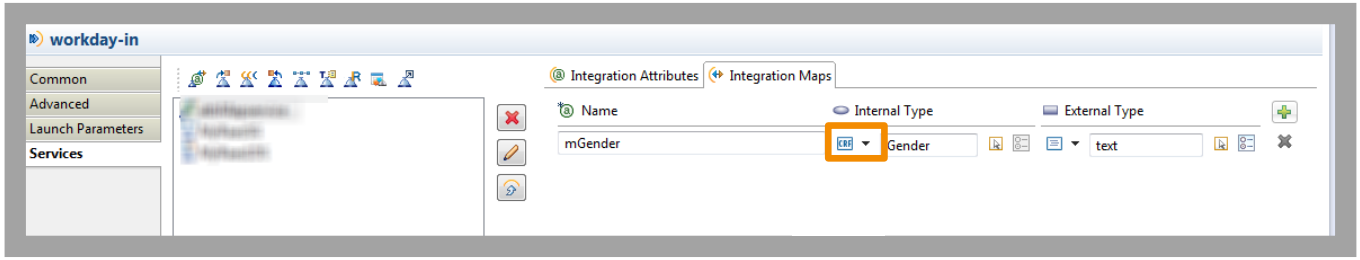
Configuring the Attribute Value in the Tenant

Note: Critical error.

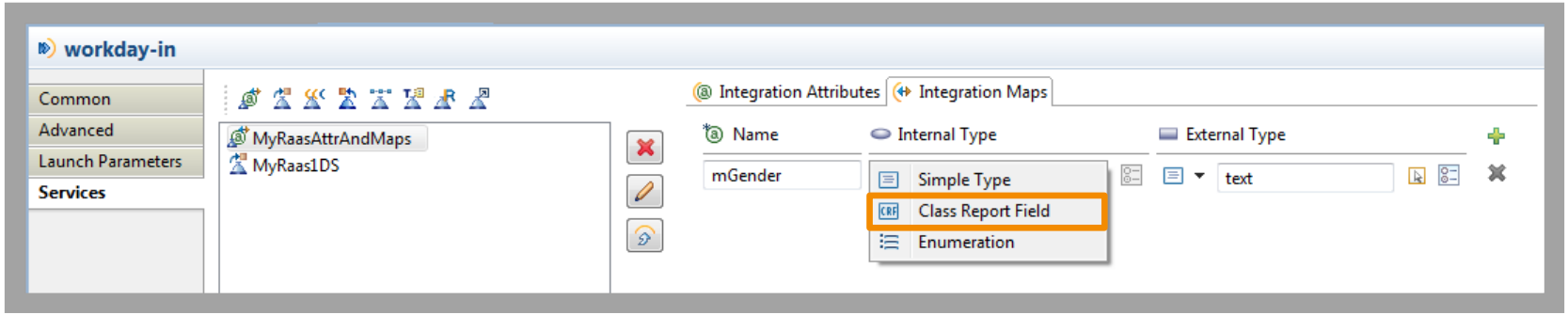


Adding an Integration Map

- Go to properties on Workday-In, then view the Services tab.

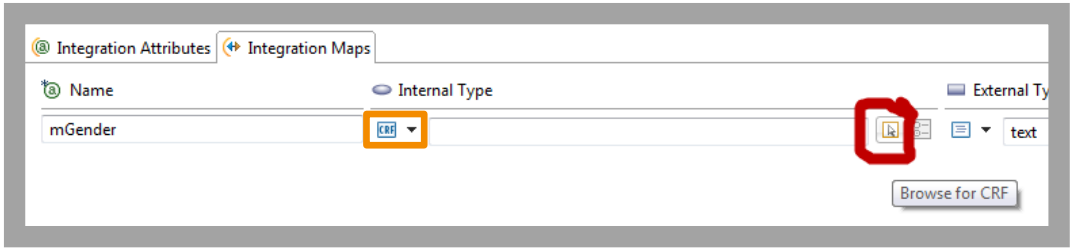


- Select **Class Report Field** from the drop down list.



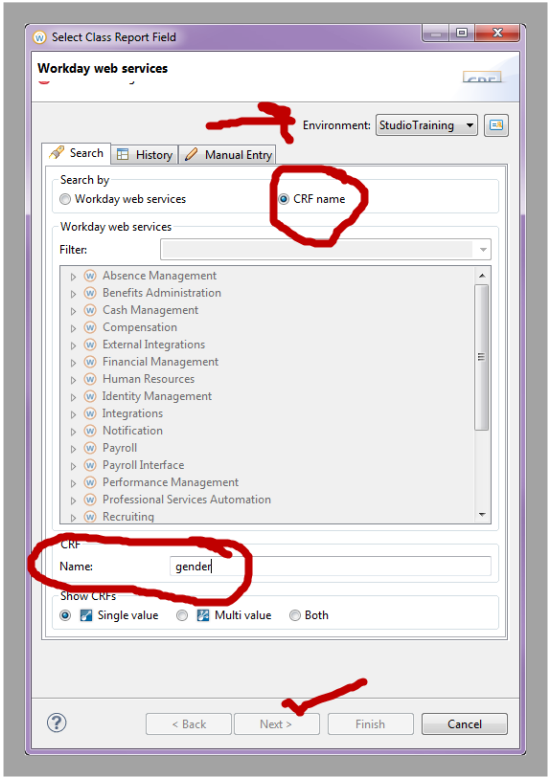
Adding an Integration Map

Select the **Browse for CRF button**.



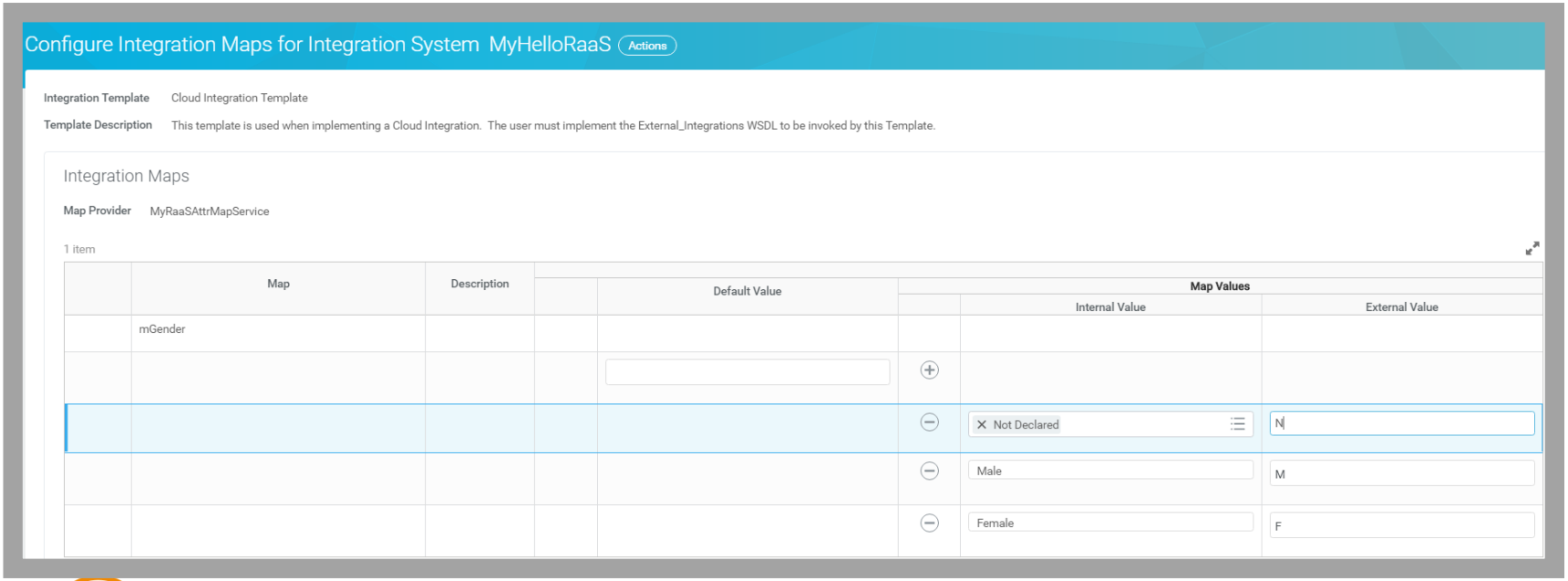
Select your Environment and then search for the business object.

Then click **Next**.



Configuring the Integration Map Value in the Tenant

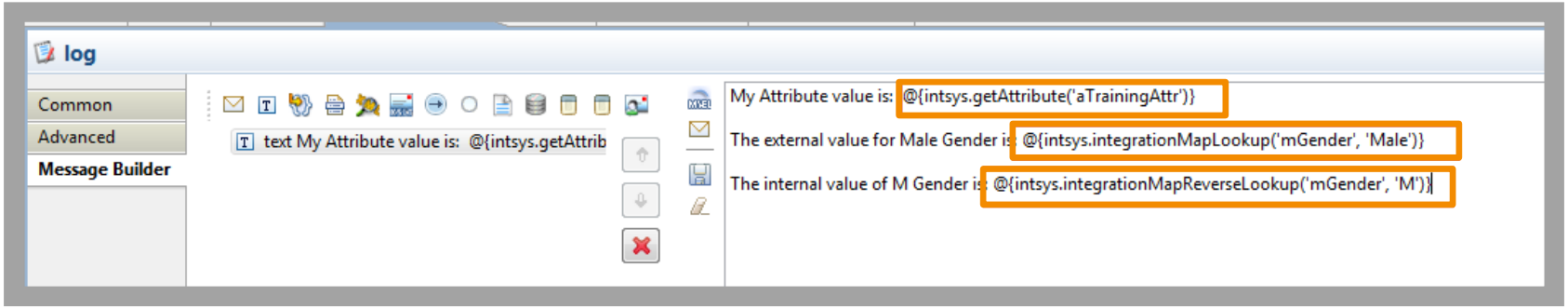
- Internal Value is a Workday's value.
- External Value is a non-Workday value.



Referencing Attribute and Integration Maps in Assembly Using MVEL Expression

MVEL Syntax

- **Attribute**: intsys.getAttribute('attribute_name')
- **External Map Lookup**: intsys.integrationMapLookup('map_name', 'key')
- **Internal Map Lookup**: intsys.integrationMapReverseLookup('map_name', 'external_value')

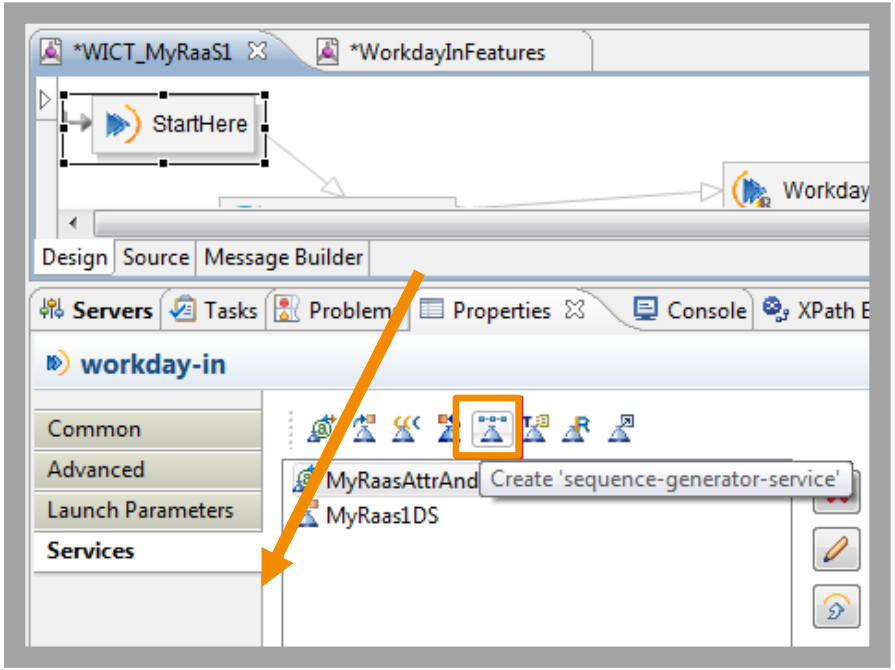


Sequence Generator

- One of the integration system services is the Sequence Generator. You can use this service to generate unique sequenced names.

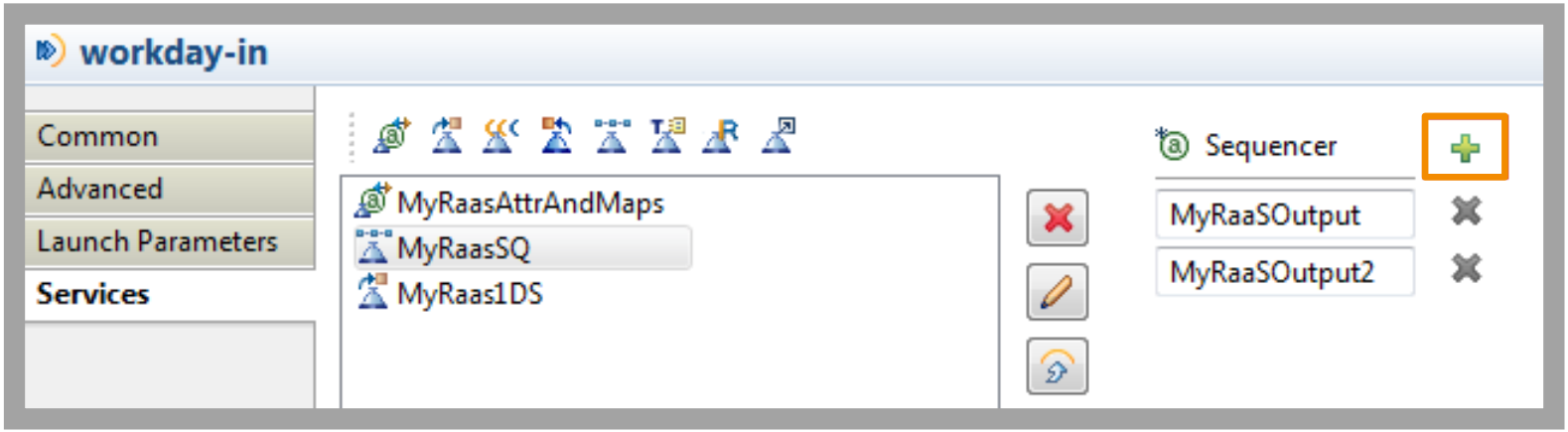
Example: If an integration needs to create a different file name each time it runs.

- You define the service as part of the workday-in transport with one or more named sequencers, then configure it in the Workday tenant.



Sequence Generator

You can create multiple sequences using one sequence generator service.



Sequence Generator

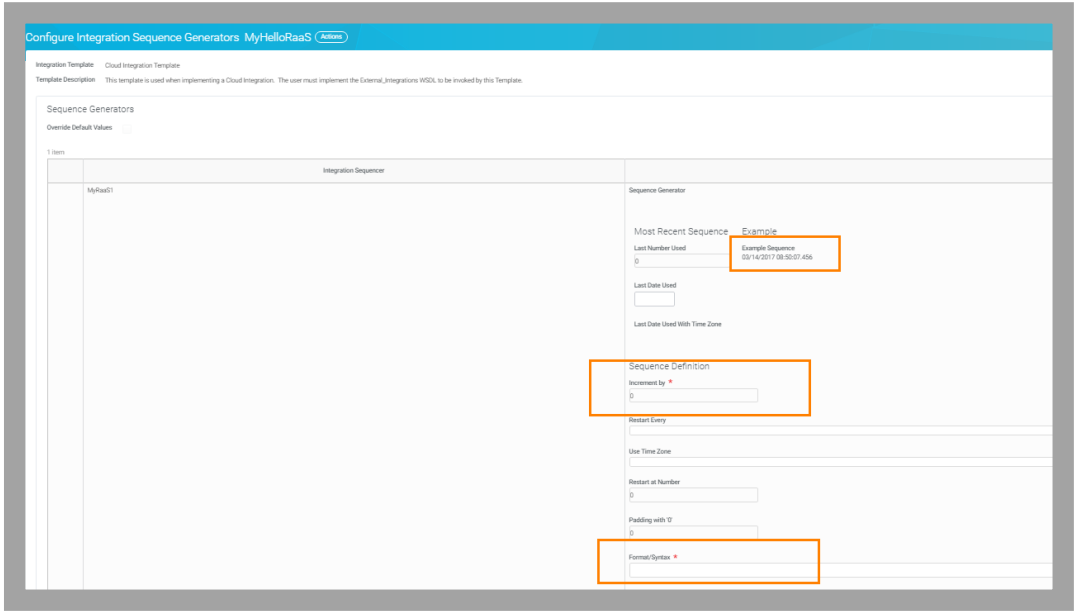
Each sequence can be defined using the allowed sequence, date, and time patterns.

Define the Format using any string constant plus any of the following Patterns.
(Note: each Pattern needs to be surrounded by square brackets. Example: FILE[Seq][yyyy][MM][dd].csv)

Sequence Generator Patterns:
Seq - next Sequence Number

Date/Time Patterns:

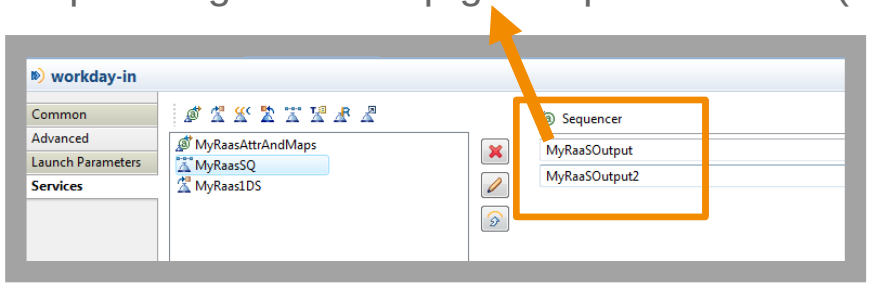
y	-	Year	(e.g. 08; 2008)
M	-	Month in Year	(e.g. 07; Jul; July)
w	-	Week in Year	
W	-	Week in Month	
D	-	Day in Year	
d	-	Day in Month	
F	-	Day of Week in Month	
E	-	Day in Week	(e.g. Tue; Tuesday)
a	-	Am/pm marker	
H	-	Hour in day (0-23)	
h	-	Hour in	

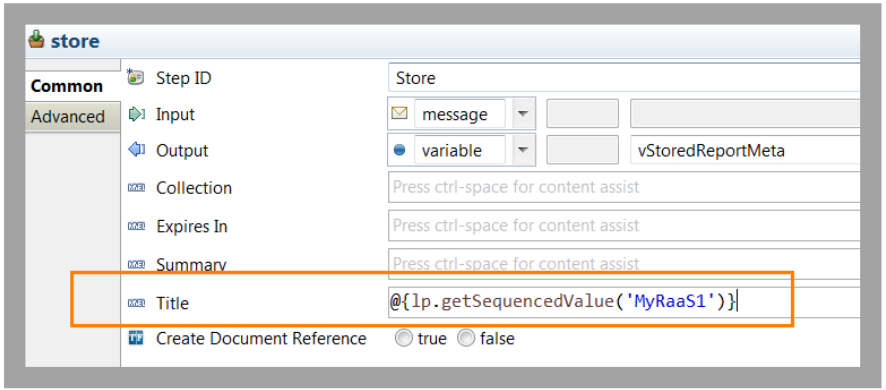


Referencing in MVEL

MVEL Syntax

Sequence generator: lp.getSequencedValue('name')





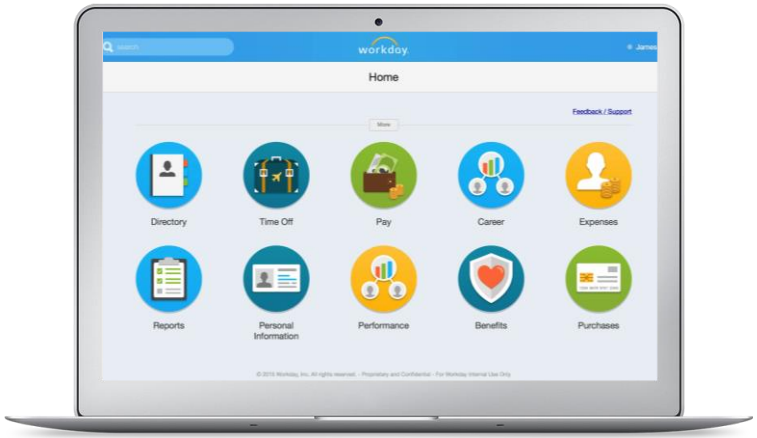
Activity 9

Create, Configure, and Test Integration System Services

Estimated Time: 20 Minutes

Workday In Transport

1. Create a new attribute map service called MyRaasAttributeMapService.
2. Add an Attribute and a Map.
3. Create a Sequence Generator Service.
4. Add a sequencer and use it to name the output file.



Parameters in XSLT

workday. | EDUCATION

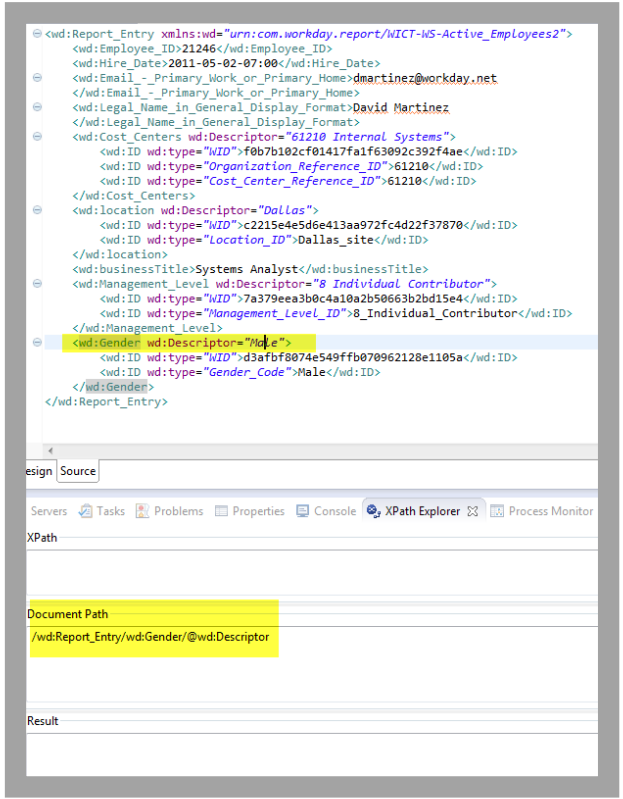
Learning Objectives

- Use property values in stylesheet to change the integration message.
- Implement integration maps and attributes to dynamically manipulate an integration output file.

Reminder: XPath Explorer

Once you have created or imported an XML file into Studio, you can open the file and click on an element.

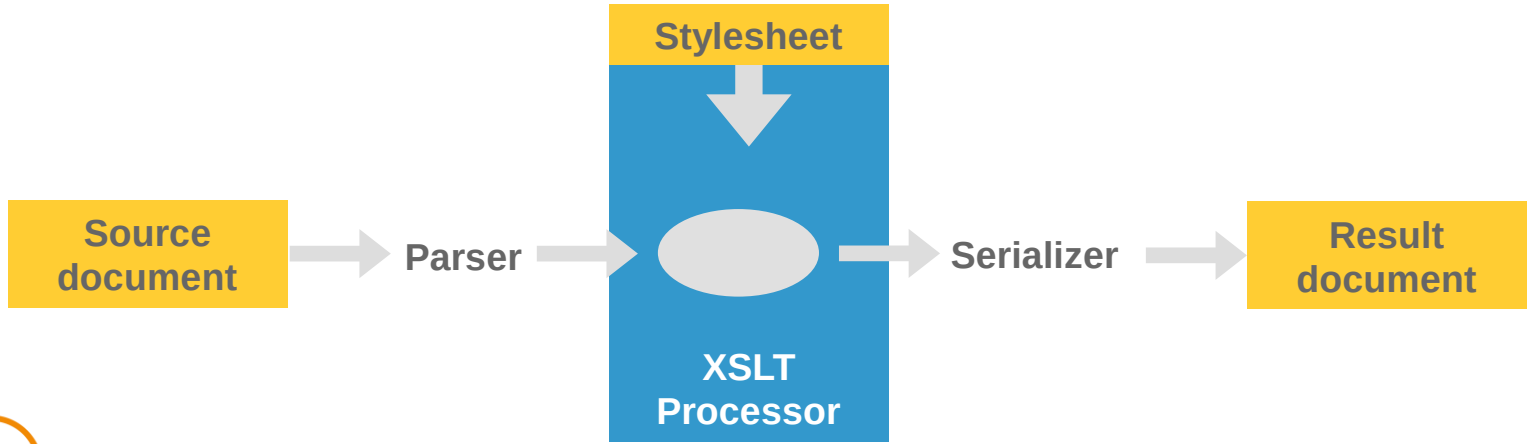
Using the XPath Explorer view, you can review the XPath for that element.



Review: XSLT

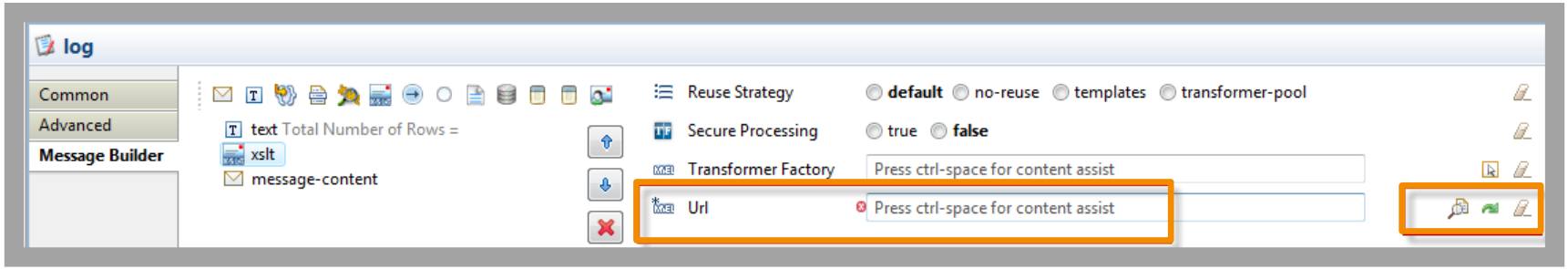
There are times when Workday does not present output in the desired format for simple integration tasks. In this case, an Extensible Stylesheet Language Transformation (XSLT) could be implemented with integration systems.

Note: **Workday does not generate or validate your XSLT attachments. These attachments can be used in EIB, Studio and Document Transformation.**



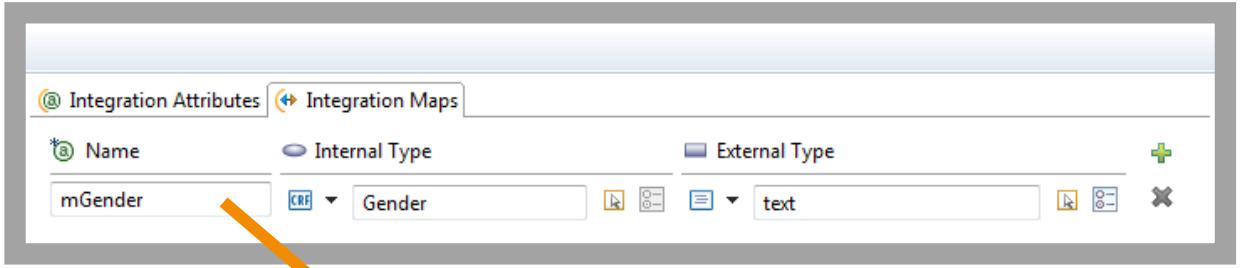
Create xsl File

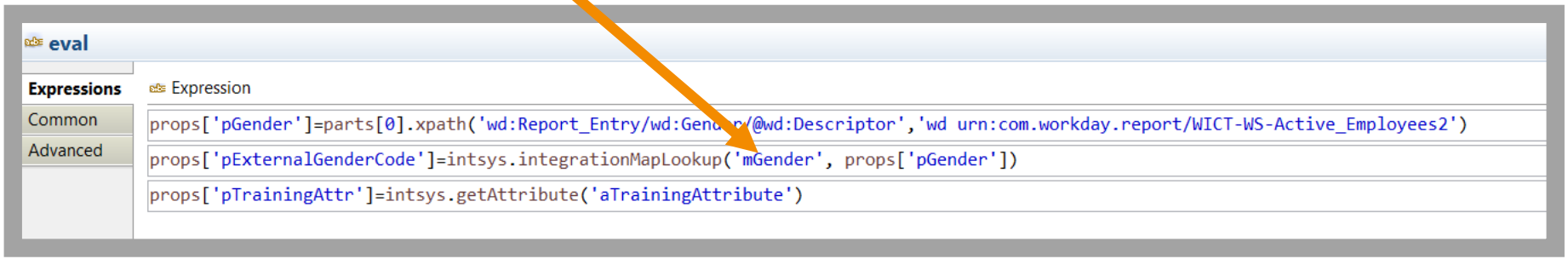
- You can use the browse to in this case as the XSL has been provided.
- If you are building from scratch, you can create the XSL file first under the **WS > WSAR-INF** folder, then add it to the XSLT step.



In MyHelloRaaS – Define a Property in Eval Step and Reference in XSLT as a Parameter

Set a Property from a Launch Parameter, Attribute, or even a mapped value.





Reference Property in XSLT as a Parameter

Set the parameter in the XSLT using the property name.

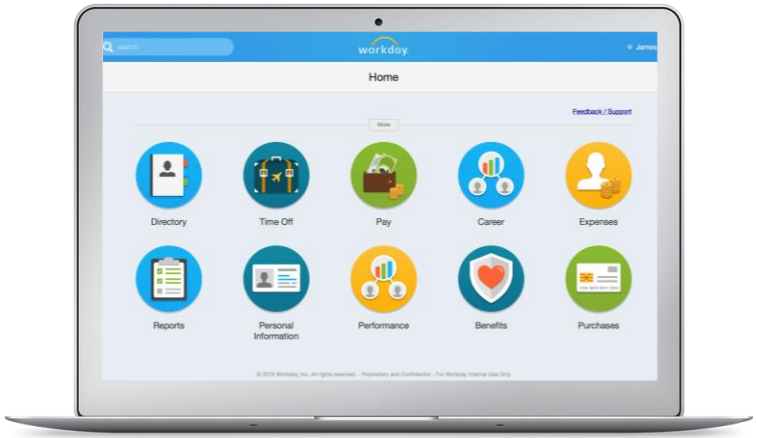
```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:wd="urn:com.workday.report/WICT-WS-Active_Employees2">
<xsl:param name="pExternalGenderCode"></xsl:param>
<xsl:param name="pTrainingAttr"/>

    <xsl:template match="/wd:Report_Entry">

        <wd:WorkerSummary xmlns:wd="urn:com.workday.report/WICT-WS-Active_Employees2">

            <wd:ReferenceID><xsl:value-of select="wd:Employee_ID"></xsl:value-of></wd:ReferenceID>
            <wd:HireDate><xsl:value-of select="substring(wd:Hire_Date,1,10)"></xsl:value-of></wd:HireDate>
            <wd:Name><xsl:value-of select="wd:Legal_Name_in_General_Display_Format"></xsl:value-of></wd:Name>
            <wd:Location><xsl:value-of select="wd:location/@wd:Descriptor"></xsl:value-of></wd:Location>
            <wd:Gender><xsl:value-of select="$pExternalGenderCode"></xsl:value-of></wd:Gender>
             <wd:Custom><xsl:value-of select="$pTrainingAttr"></xsl:value-of></wd:Custom>
        </wd:WorkerSummary>

    </xsl:template>

</xsl:stylesheet>
```

Activity 10

Include External Gender Code in Workday Summary

Estimated Time: 15 Minutes

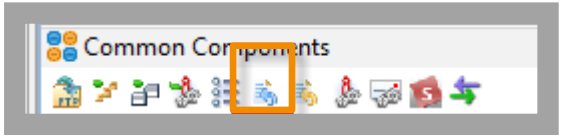
Modify the MyHelloRaaS assembly so that the output file displays the gender codes M, F, or N, rather than the gender descriptor.

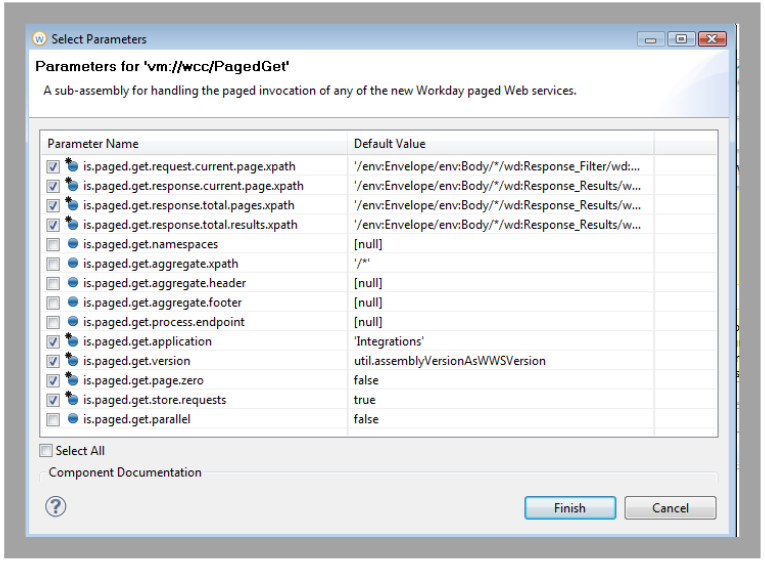


The PagedGet Common Component

PagedGet Common Component

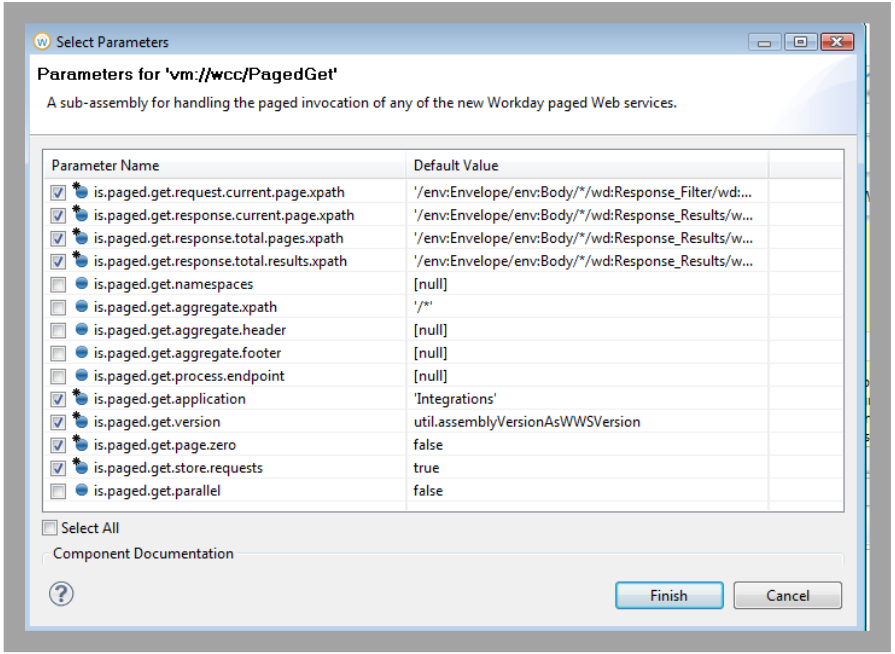
- The PagedGet component handles the paged invocation of any of the Workday web services and allows data to be retrieved and processed page by page.
- The PagedGet component has both in and out parameters.



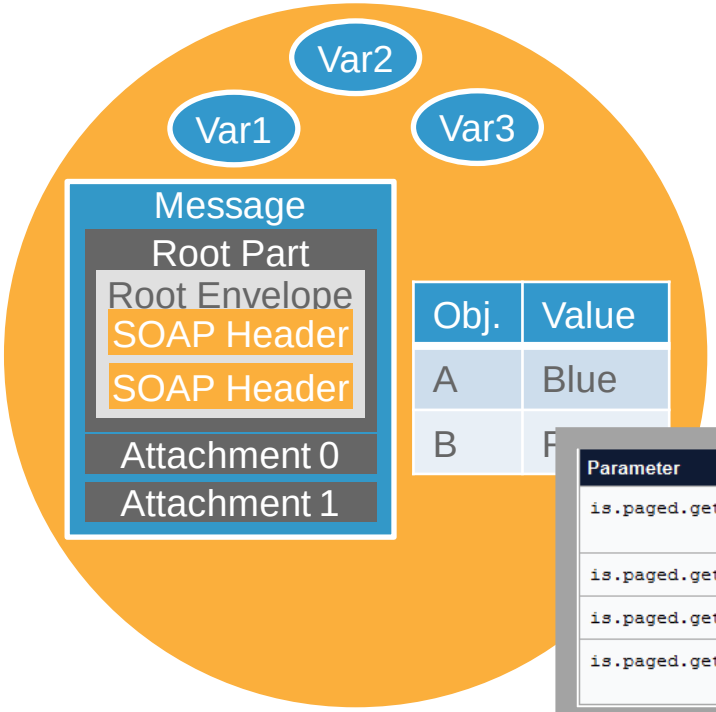


PagedGet Common Component

- Note the use of /*.
- This level has the name of the Web Service.
- Generally it is good form to update to the WS you are using.



Review – PagedGet Out Parameters

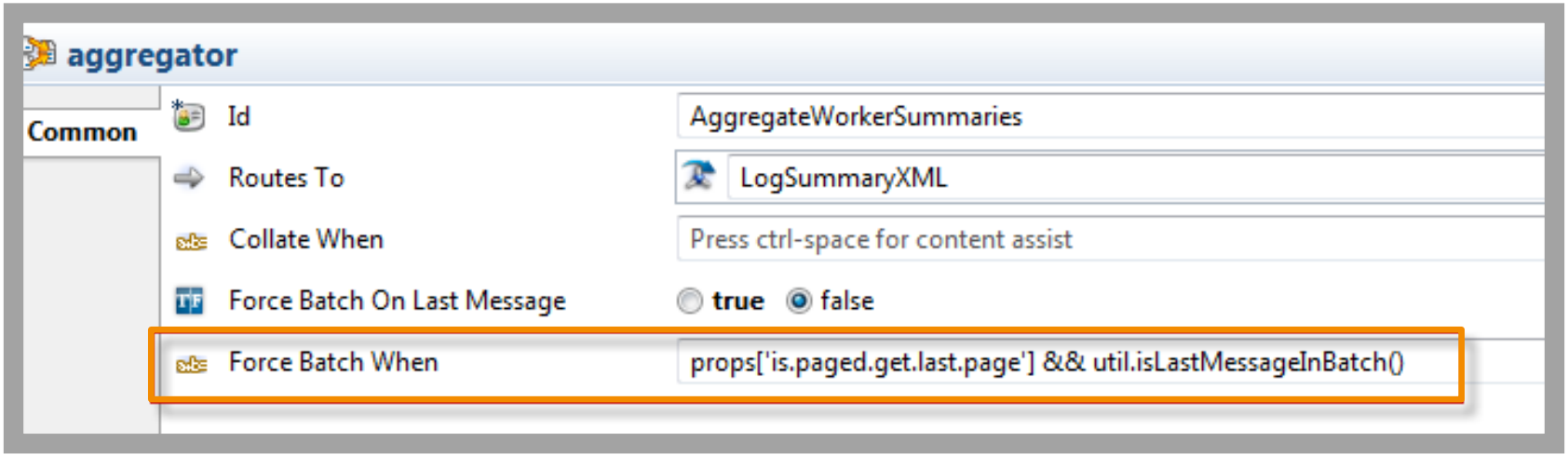


Parameters are properties set in the mediation context as properties by the PagedGet subassembly. These can be evaluated in downstream components.

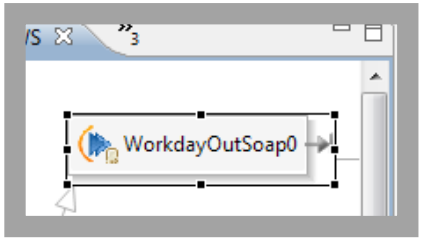
Parameter	Description
`is.paged.get.last.page`	A Boolean property that is set to `true` when processing the final page of data returned by the Web service operation.
`is.paged.get.current.page`	An integer property describing the current page of data being processed.
`is.paged.get.total.pages`	An integer property describing the total number of pages to be processed.
`is.paged.get.total.results`	An integer property describing the total number of results returned by the Web service operation.

Aggregating Multiple “Pages”

You can use the following expression in Aggregator to determine when to force the batch: **props['is.paged.get.last.page'] && util.isLastMessageInBatch()**.

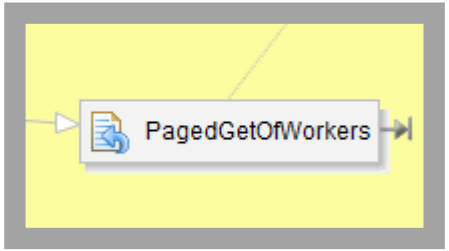


Workday Out SOAP vs. PagedGet



Workday Out SOAP:

- Requires Request Message.
- One call to the web service-operation specified in the request message.
- One response page received back in assembly.
- It is useful for specific object calls to get data and to set data.



PagedGet Common Component**:**

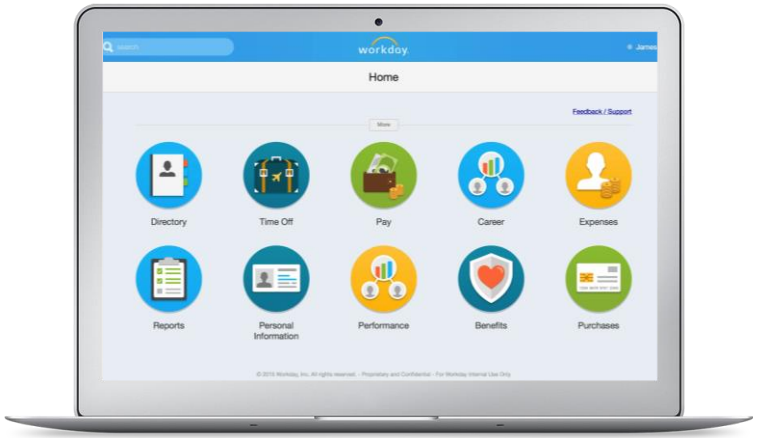
- Requires Request Message.
- Handles the paged invocation of any of the paged Workday web services-operation and allows data to be retrieved and processed page-by-page.
- Implements the paging logic required by Workday Studio paged-get web service-operations.
- Useful to get large sets of data out of Workday.
- Can be used to get data only.

Activity 11

PagedGet

Estimated Time: 15 Minutes

Modify the PagedGet assembly to allow it to return all of the results.



Homework: Review Assemblies 101 Sample

workday. | EDUCATION

Learning Objectives

- Assemblies101 contains multiple Workday-In transports that represent an integration system Pattern.
- Import and Review patterns.
- Deploy and Launch individual integration systems to review specific pattern behavior.

Assemblies 101 Sample

Benefits

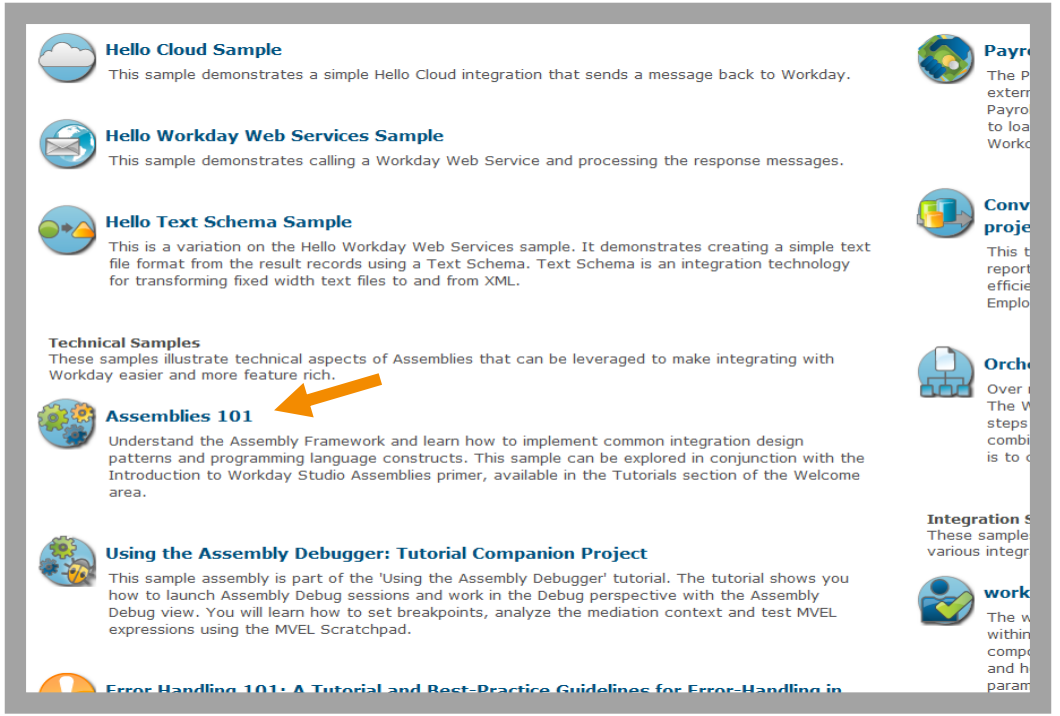
- Learn about Assembly concepts, or reinforce learning
- Design patterns

- Programming Language Constructs
  - If-then-Else
  - Looping
  - Sub-routines
  - Error Handling

This sample provides a basic introduction to Workday Studio Assemblies, illustrating how many common integration design patterns and programming language constructs can be represented. The sample can be explored in conjunction with the Introduction to Studio Workday Assemblies primer. Each of the examples starts with a workday-in component so they can be launched and examined independently.

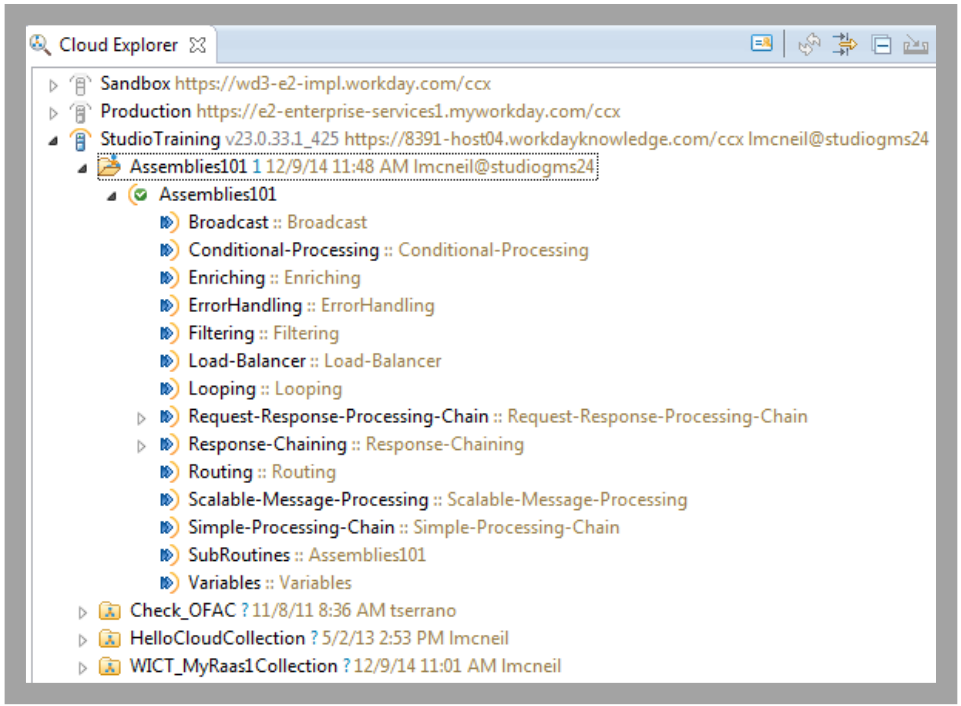
Technical Samples: Assemblies 101

Help > Welcome > Samples



Deploy Sample and View Cloud Explorer

- When deployed, each Workday In Transport represents an integration system.
- The integration systems in this sample represent different patterns and are independent of each other.



Many Small Examples

Assemblies101 sample contains 14 integrations (visually separated with a thick black line):

- Simple-Processing-Chain
- Request-Response-Processing-Chain
- Response-Chaining
- Routing
- Looping
- Broadcast
- Filtering
- Load-Balancer
- Conditional-Processing
- Variables
- ErrorHandling
- Scalable-Message-Processing
- Enriching
- Assemblies101 (SubRoutines > A-SubRoutine)

Homework

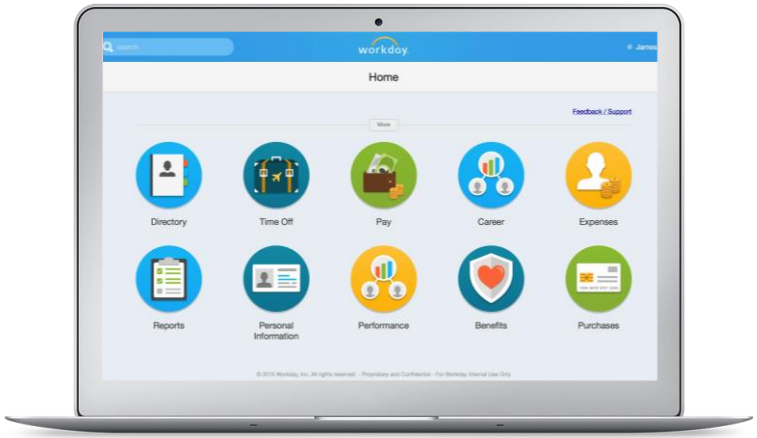
Assembly 101 Review

Estimated Time: 20 Minutes

Help > Welcome > Samples > Assembly 101

Revisit the following examples and perform TO DO steps (see following pages):

3. Response-Chaining
5. **Looping**
6. Broadcast
7. Filtering
8. Load-Balancer
9. Conditional-Processing
10. **Variables**
13. **Enriching**



workday. | EDUCATION

ducation
resource
is where
learning

Please log in to your tenant:

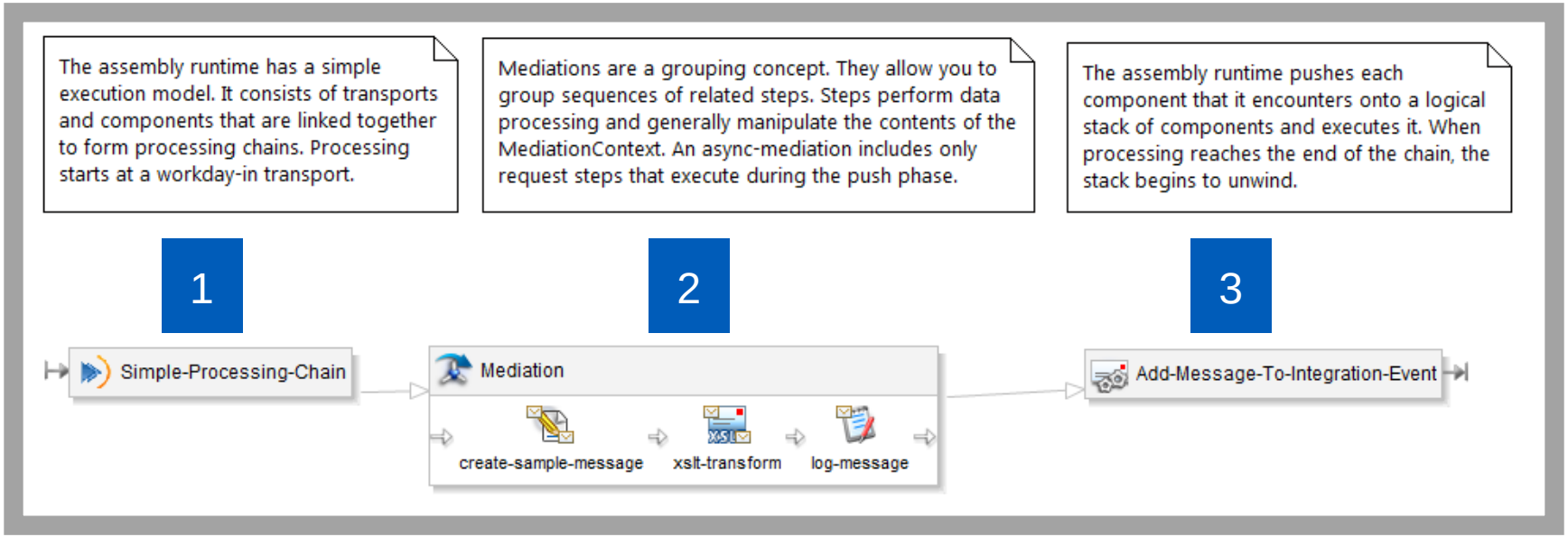
Username:

Password:

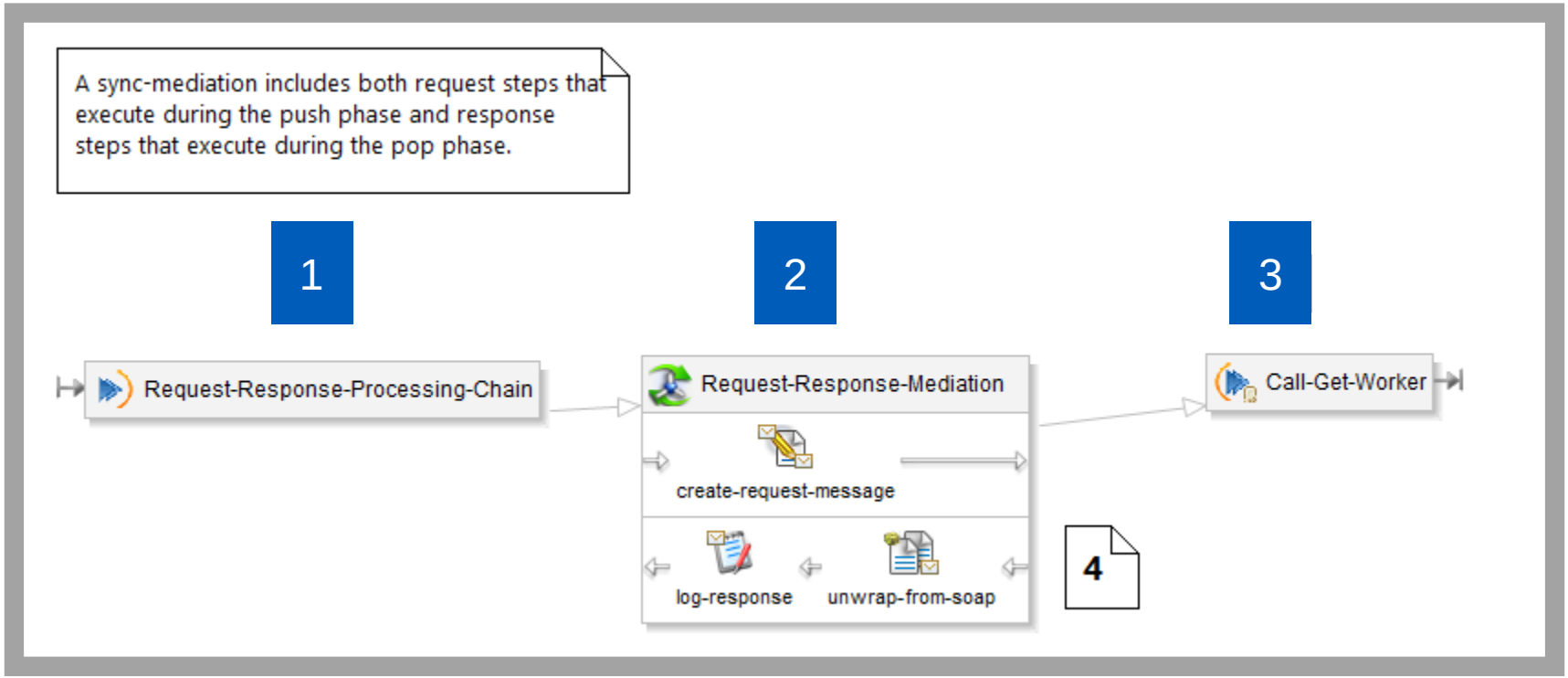
Advanced Workday Studio

Day 3

1. Simple-Processing-Chain



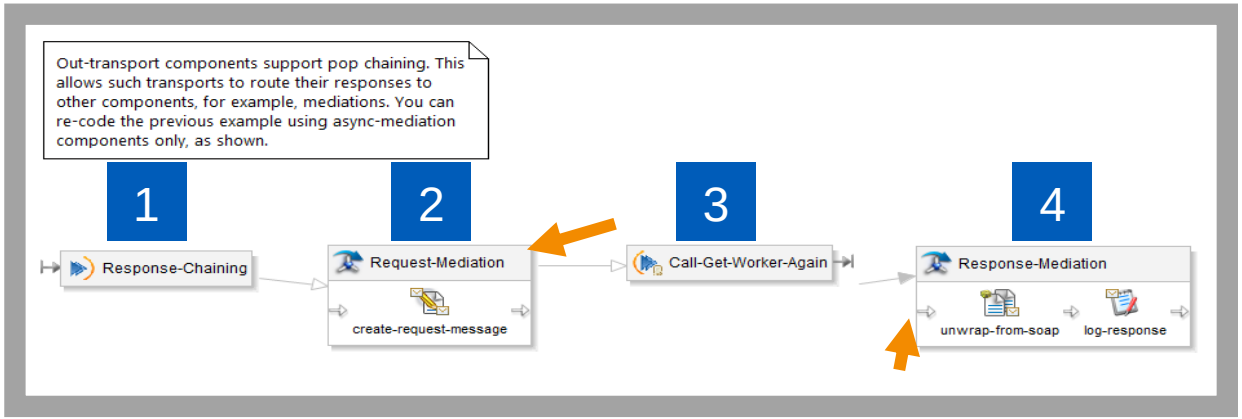
2. Request-Response-Processing-Chain



3. Response-Chaining

To Do:

1. Launch. Try to anticipate result in Log output.
2. Modify. Change [2] to a Sync Mediation and add a Log step in the return.
3. Deploy, Launch: Try to anticipate result in Log output.

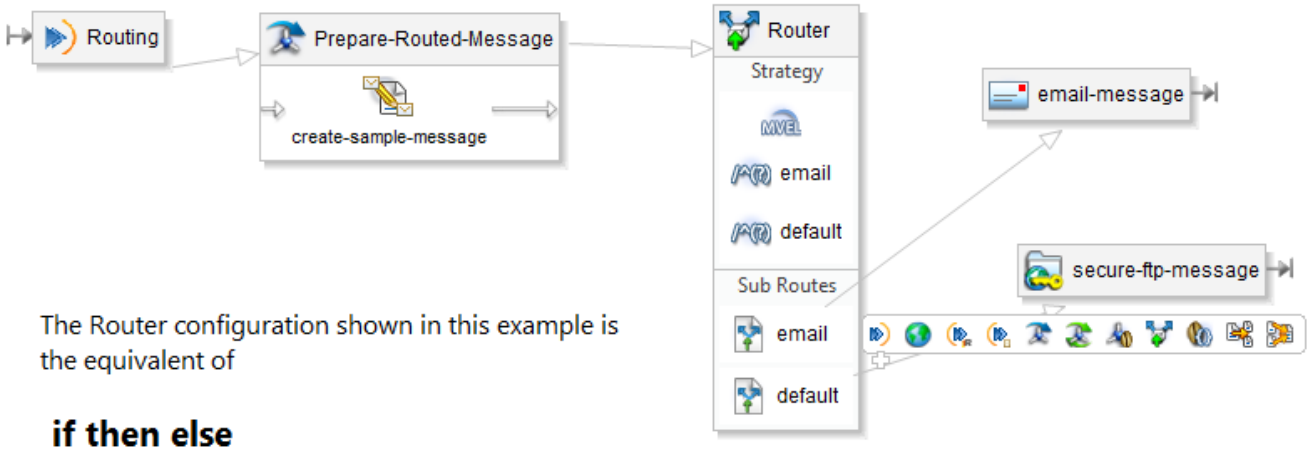


4. Routing

Assemblies support conditional routing using the Router component.

You can configure the Router component with several different routing strategies, and any number of routes.

Note that this example does not work out-of-the-box. It requires email and/ or sftp configuration.



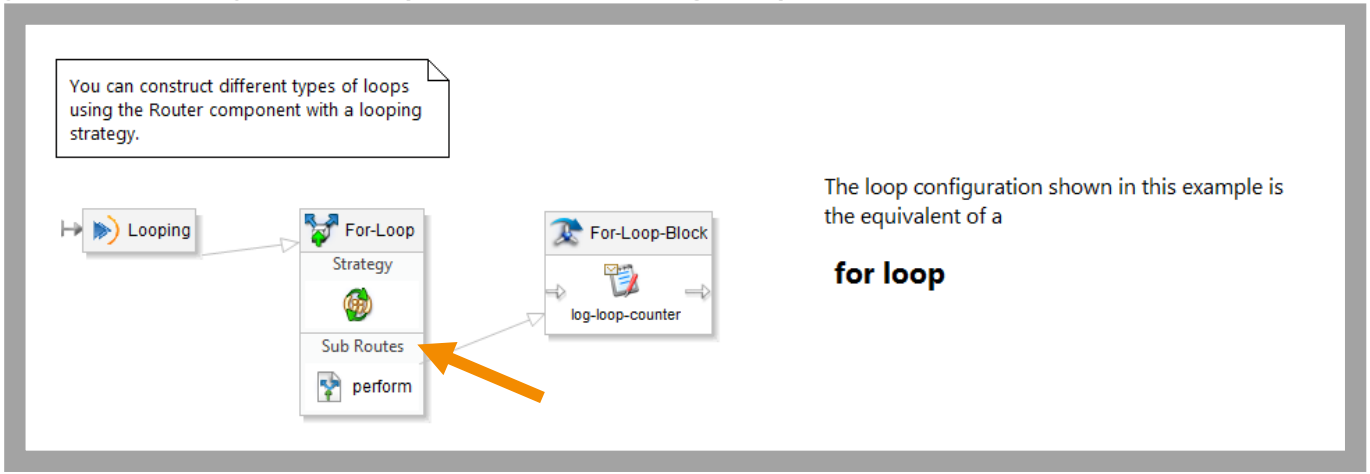
The Router configuration shown in this example is the equivalent of

if then else

5. Looping

To Do:

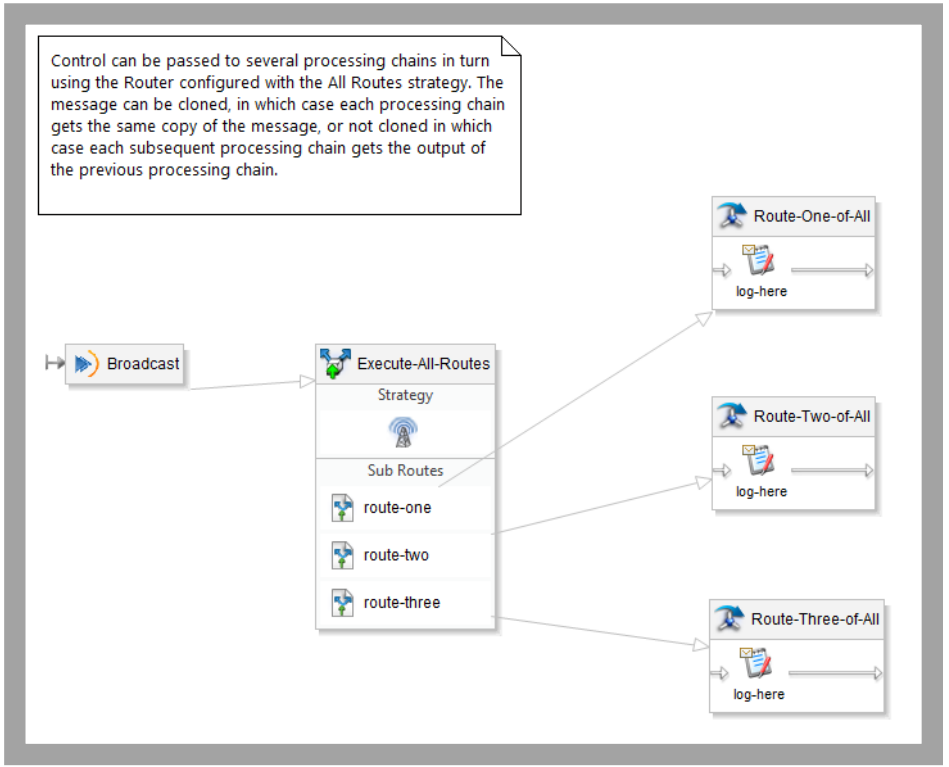
1. Launch: Try to anticipate result in Log output.
2. Modify: After the For Loop completes, add a Log of loop counter.
3. Deploy, Launch: Try to anticipate result in Log output.



6. Broadcast

To Do:

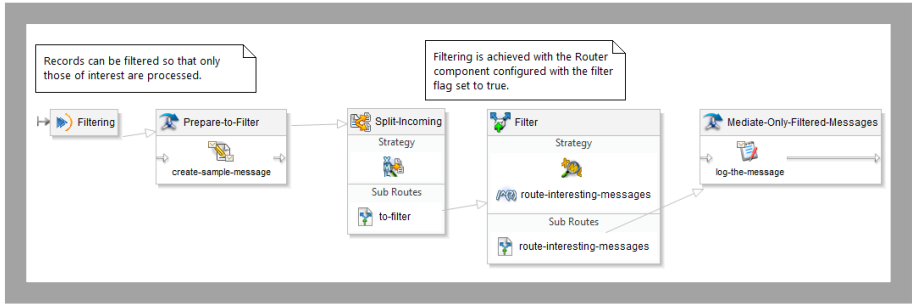
1. Launch: Try to anticipate result in log output.
2. Modify: Set Clone Request property of the route component ALL strategy to false.
3. Deploy & Launch: Try to anticipate result in log output.

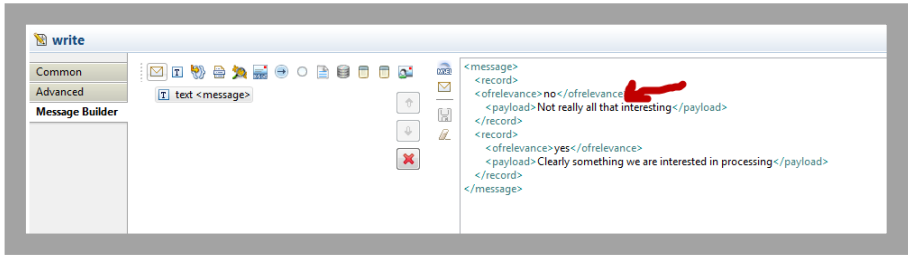


7. Filtering

To Do:

- Launch: Try to anticipate result in Log output.
- Modify: Add two <of relevance> element to first record in write step message builder.
  - <ofrelevance>no</ofrelevance>
    <!-- no instead of yes -->
- Deploy, Launch: Try to anticipate result in Log output.
- Modify: Edit the choose-route expression to evaluate value: /record/ofrelevance = ‘yes’.
- Deploy, Launch: Try to anticipate result in Log output.



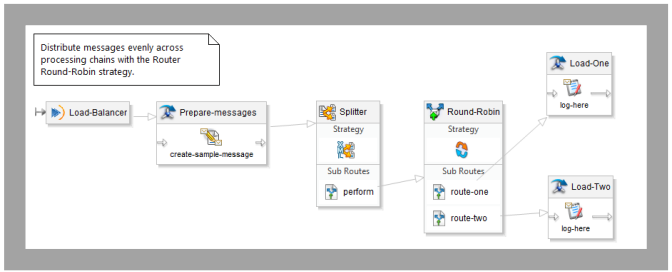


8. Load-Balancer

To Do:

1. Launch: Try to anticipate result in Log output.
2. Modify: Add another sub route that is wired to an async mediation with a Log step.
3. Deploy and Launch: Try to anticipate result in Log output.
4. Modify: Add another <message> element to the Write step message builder.

```
<messages>
   <message>some content</message>
   <message>more content</message>
   <message>other content</message>
   <message>my new content</message>
</messages>
```

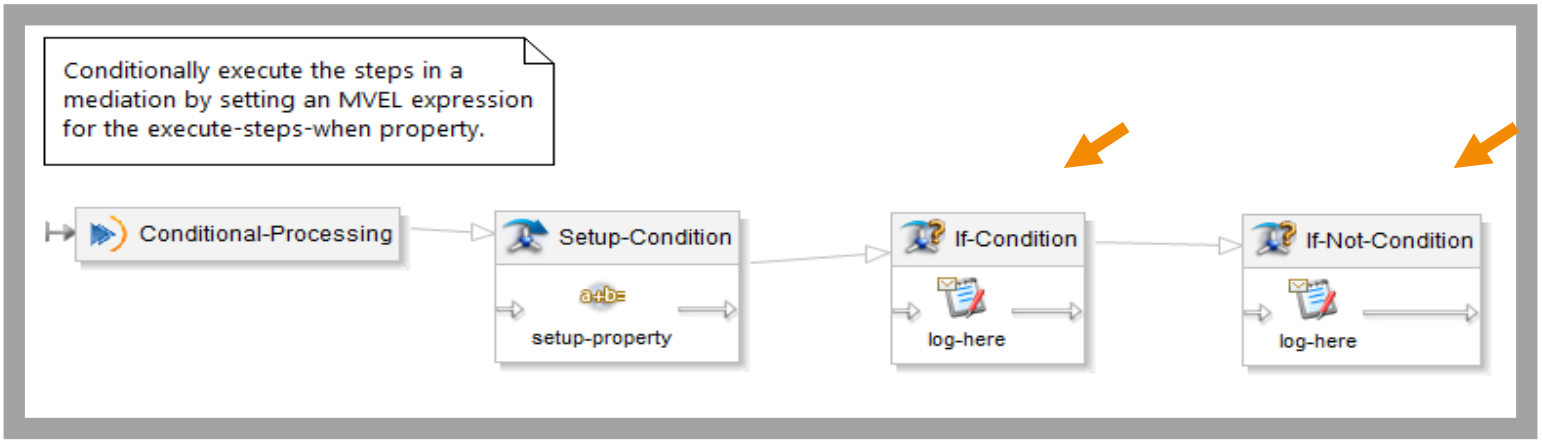


5. Deploy and Launch: Try to anticipate result in Log output.

9. Conditional-Processing

To Do:

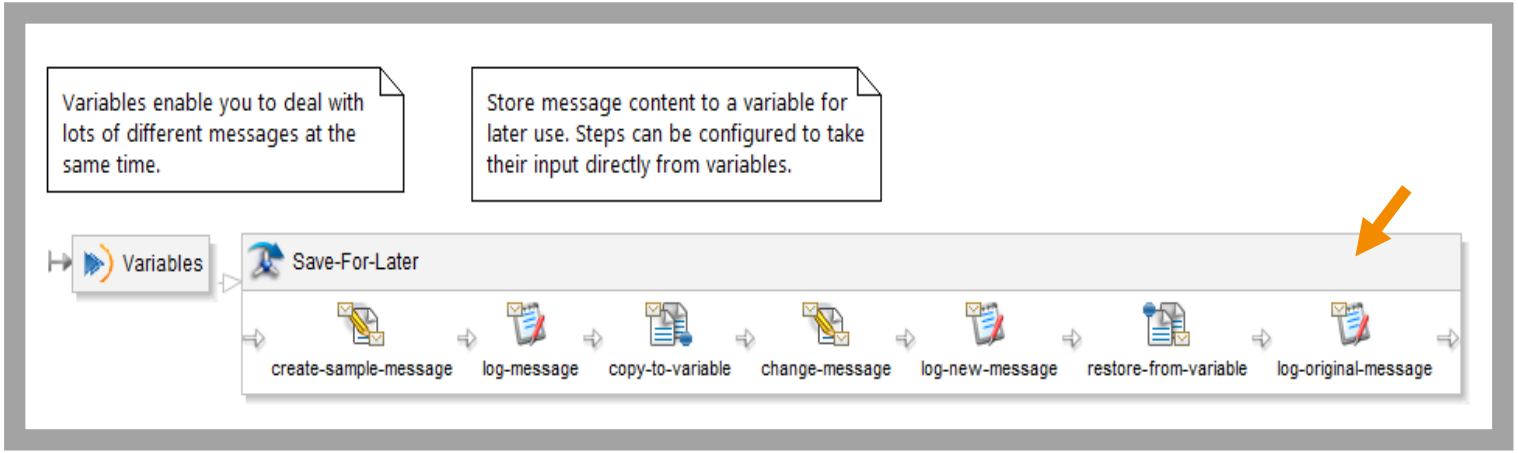
- Launch: Try to anticipate result in Log output.
- Modify: Change value of property in Eval step to true.
- Deploy & Launch: Try to anticipate result in Log output.



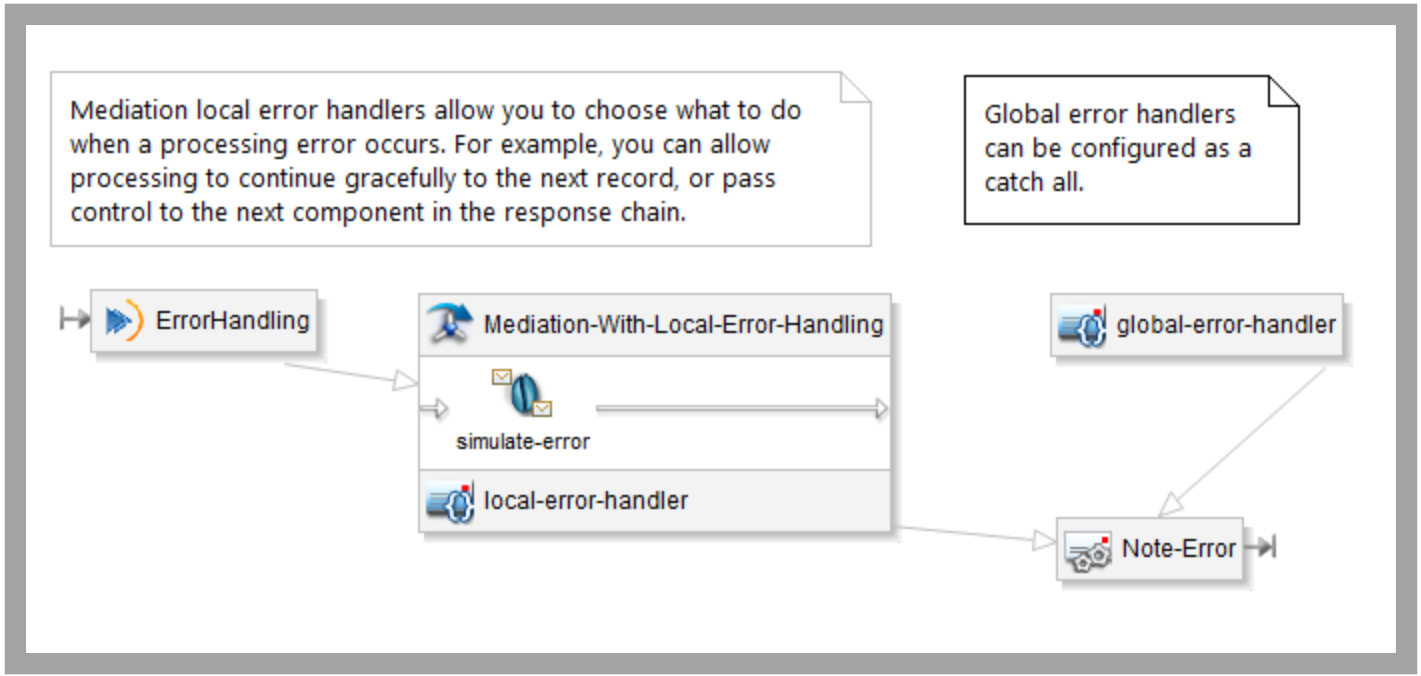
10. Variables

To Do:

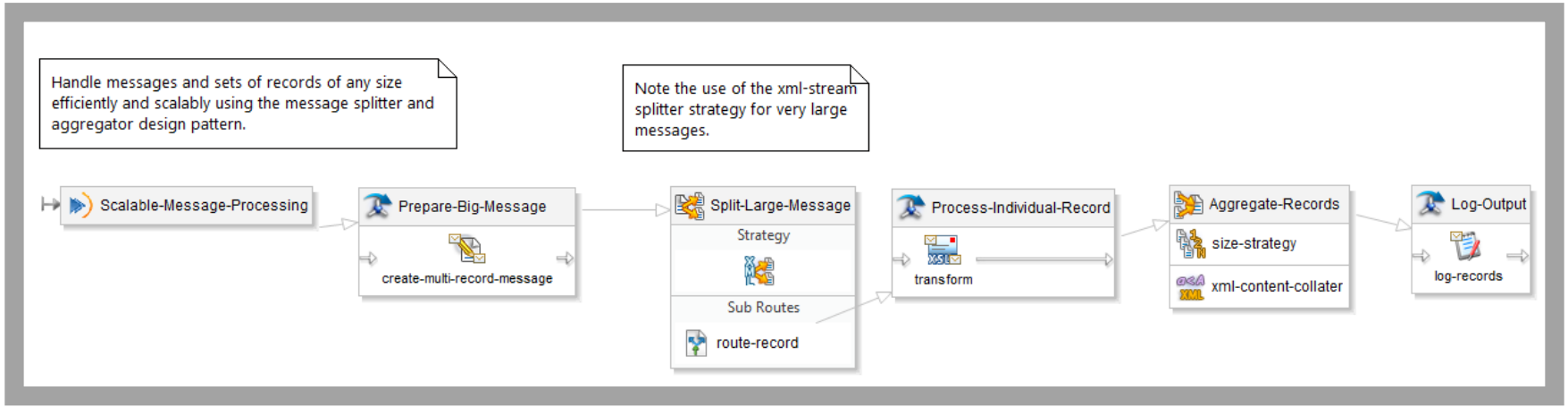
1. Review.
2. Launch: Try to anticipate result in Log output.



11. Error Handling



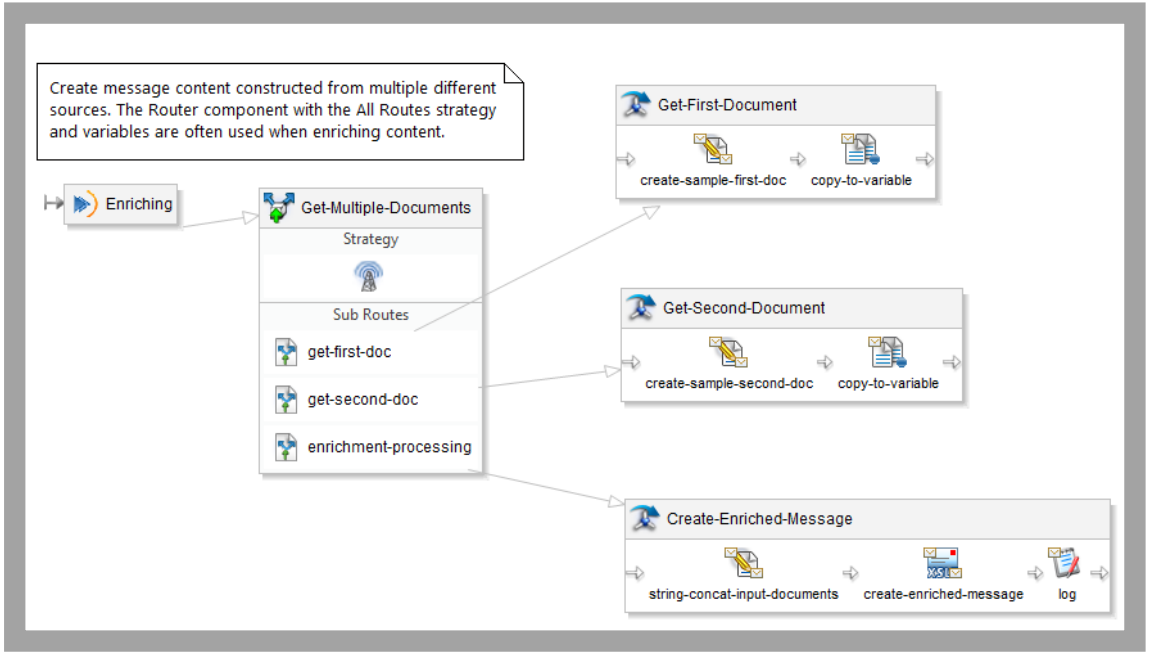
12. Scalable-Message-Processing



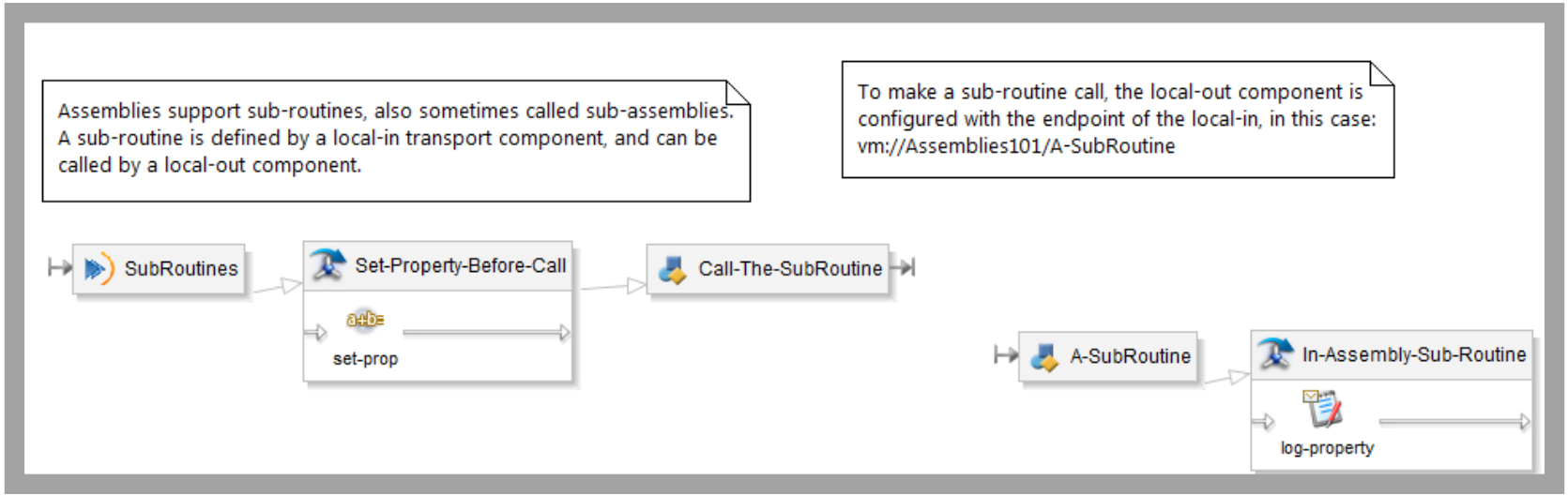
13. Enriching

To Do:

1. Review.
2. Launch: Try to anticipate result in Log output.



14. Subassembly



Route Component

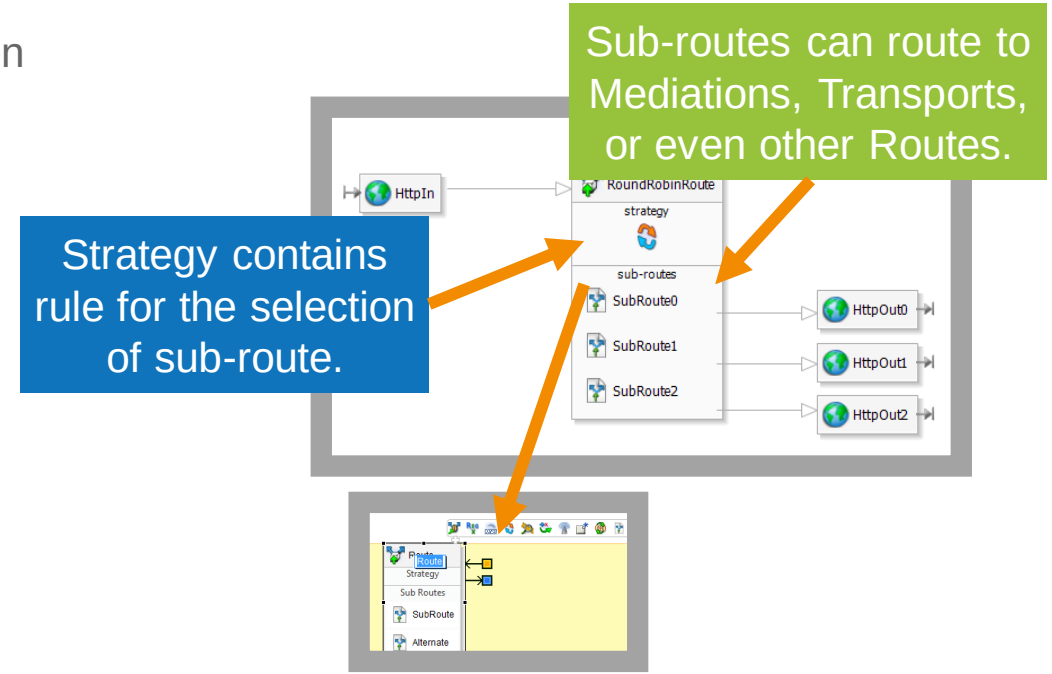
workday. | EDUCATION

Objectives

- Introduce and identify the purpose of route components.
- Discuss Route Component Strategies and identify properties.
- Discuss how to manage multiple requests and responses in the assembly.
- Design an integration to implement route component to make multiple web service calls.

The Route Component

Routes add support for iteration and conditional routing to the assembly runtime in order to process a list.



Route Mediation Component

- Router supports one or more wired paths (called sub-routes).
- Router is a “fan-out” component.
- Router offers different flavors of routing:
  - **content-based-routing**: choose sub-route based on message content
  - **all**: send your message to all the sub-routes (possibly “cloned” message)
  - **failover**: try one sub-route, and if it fails try another
  - **load-balancing**: alternating between all the sub-routes
  - **looping**: iterating through sub-routes until some condition is met
- These different flavors are called Strategies.

Some of the Routing Strategies

XPath – Select first sub-route with an XPath expression that evaluates to true.

MVEL – Select first sub-route with an MVEL expression that evaluates to true.

Regular Expression – Select first sub-route with a regex that matches.

Round Robin – Distribute messages evenly between sub-routes.

Failover – If error occurs sending to first sub-route, try the next, and so on.

All – Send message to all sub-routes.

Looping – Call sub-route until specified condition ends.

Strategies

Visually, the wiring is a Fan-Out, but the strategy select determines how the list is processed.

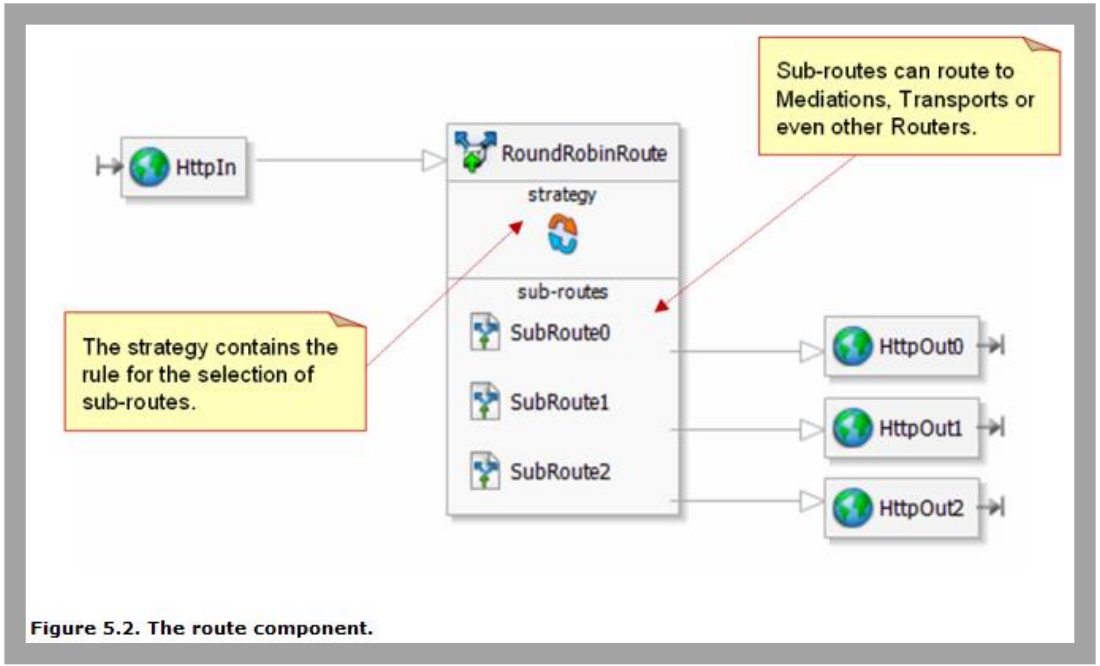


Figure 5.2. The route component.

Route Strategy: All

Same message sent to all three subroutes:

clone-request = true.

Result from one subroute sent to the next subroute:

clone-request = false.

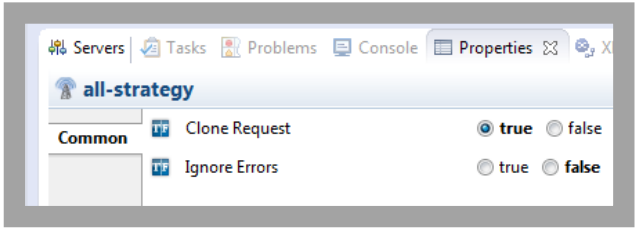
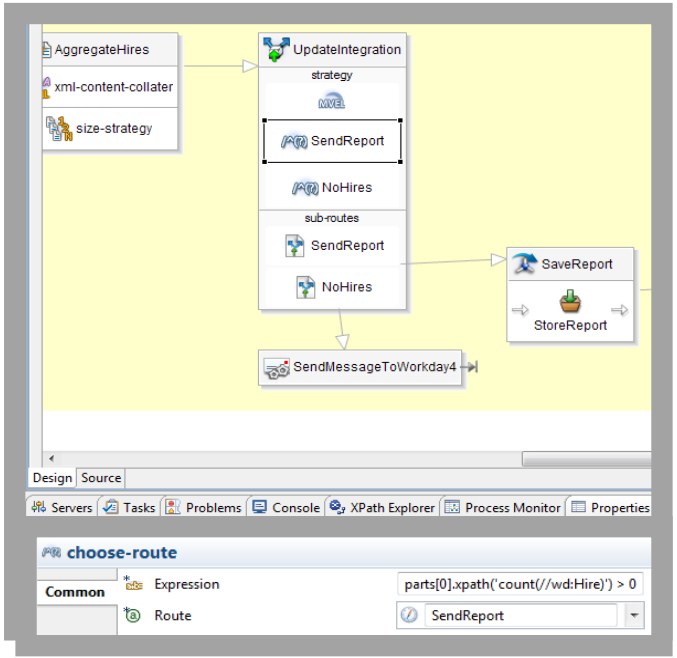


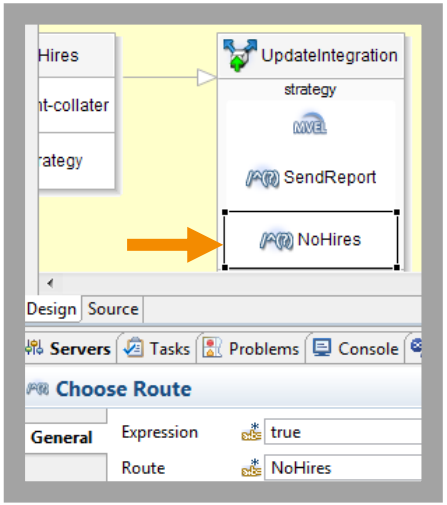
Table 5.5. Route Strategies

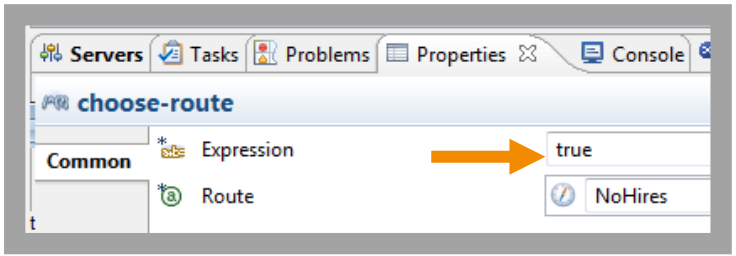
Icon	Strategy	Description
	`all-strategy`	This strategy routes the incoming message to each of the sub-routes configured within the component. You can set the following properties: • `ignore-errors`: to ignore any errors thrown by any sub-routes and continue the "fan-out", set this property to `true`. • `clone-request`: by default, each sub-route gets a copy of the original request. To chain processing between sub-routes, set `clone-request` to `false`.

Route Strategy: MVEL If (with Else)

Note: There are two choices, but the second is always “true.”

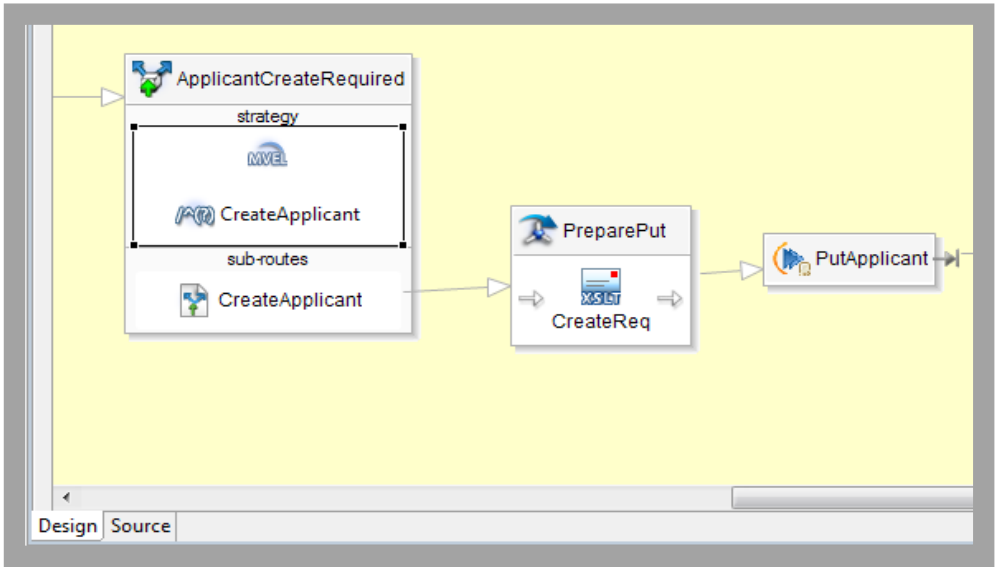


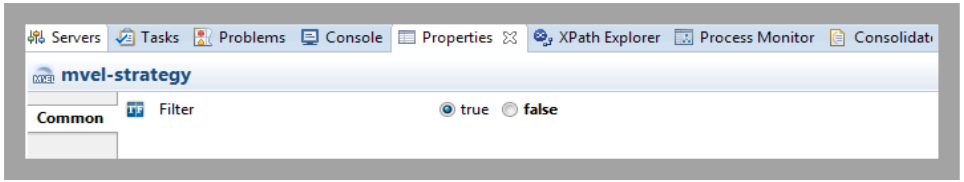




Route Strategy: MVEL If (no Else)

- Note only one choice listed.
- But filter is set to “true.”
- Allows inbound message to not leave the route component without throwing an error.



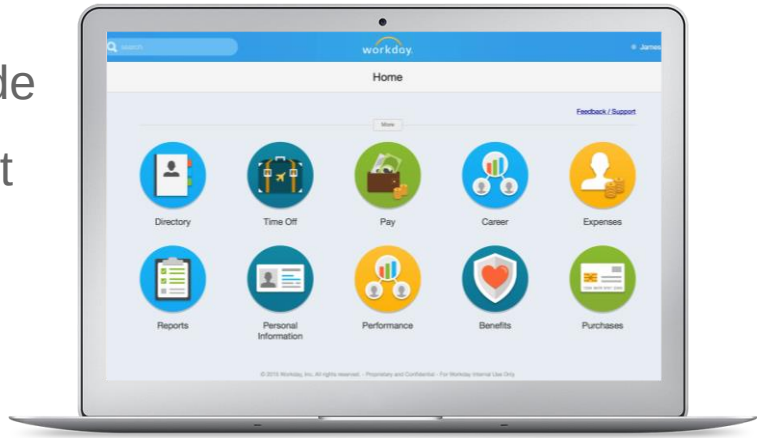


Activity 12 – Route Component

Route Component

Estimated Time: 15 Minutes

Modify the myHelloRaaS Assembly to include a route component which controls the output file format.



Error Handling 101

workday. | EDUCATION

Learning Objectives

- Why use Error Handling?
- How is Error Handling Implemented
  - How to throw Exceptions
  - How to Catch Exceptions
- Exception Messages
- Error Handling Patterns
  - Local vs. Global
    - Mediation Properties for Local Error Handling
  - Handling Downstream Errors
  - All or Nothing
  - SOAP Faults
  - Rethrow Error

Error Handling

- Sufficient error handling and integration messages should be used to allow end-users to diagnose and resolve as many integration issues as possible. Error messages should be readable to the end-users of your integration.
- Two cases:
  - Global Error Handler – Unforeseen Errors.
  - Local Error Handler – Errors we saw coming OR scenario-driven errors.

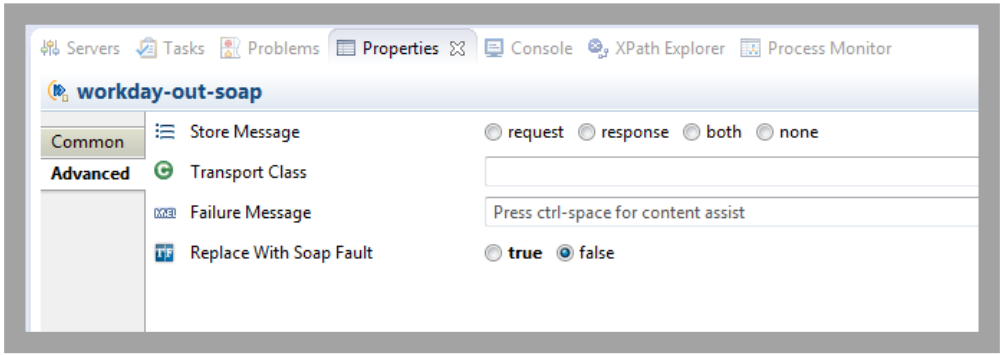
Building Error Messages

Error messages provide information to help you determine the cause and solution for integration errors:

- **Context**: Identifies the source of the problem, such as the name of the worker or integration map that caused an error.
- **Condition**: Describes the error condition such as invalid data, missing fields or attributes, or empty map values.
- **Requirement**: Specifies the valid values or conditions that must be met.
- **Response**: Provides a suggested solution to the problem.

Failure Messages

- Some components have a property called Failure Message. If the web service call fails, the contents of the failure message will be included in the exception thrown.
- The text of the original message will be appended to the failure message you provide. This is a useful way to augment the error messages returned by the web services.

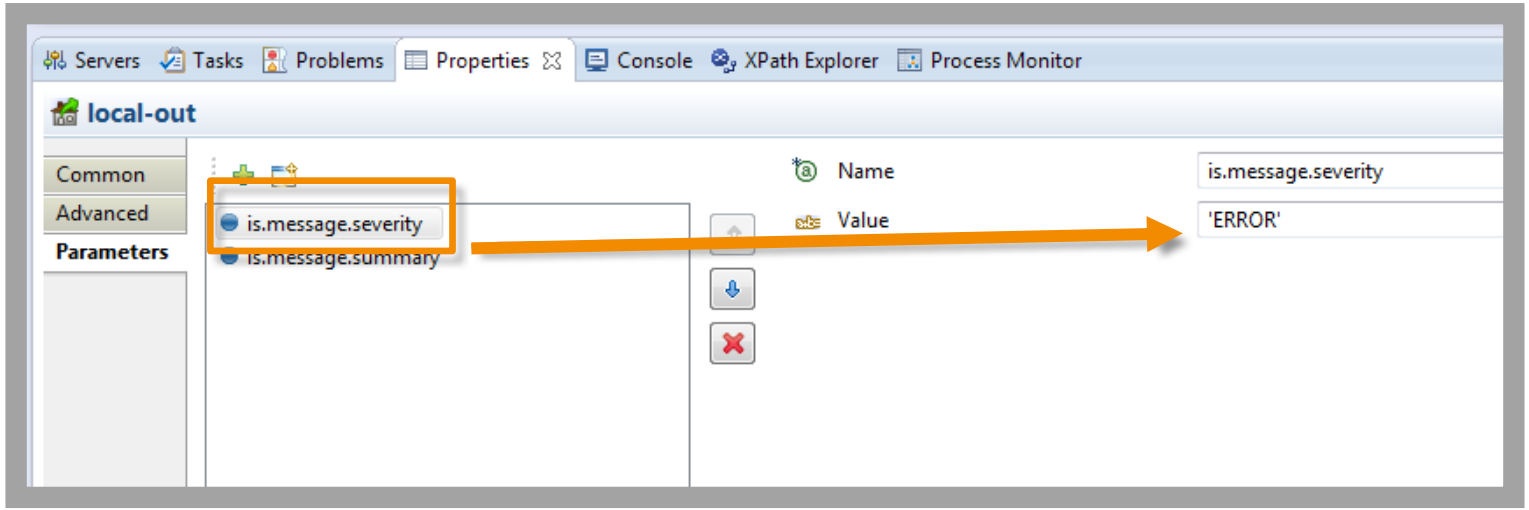


For example, you might provide the following as a failure message:

Employee ID @{props['employee.id']} had a problem while attempting to retrieve contact details. Additional details:

Message Severity

- Be sure you think through the definition of "Completed," "Completed with Errors," and "Completed with Warnings" statuses for your integration.
- Design your error handling such that the final status reported to the user through the Integration Event Messages accurately reflects what occurred during the execution of the integration.

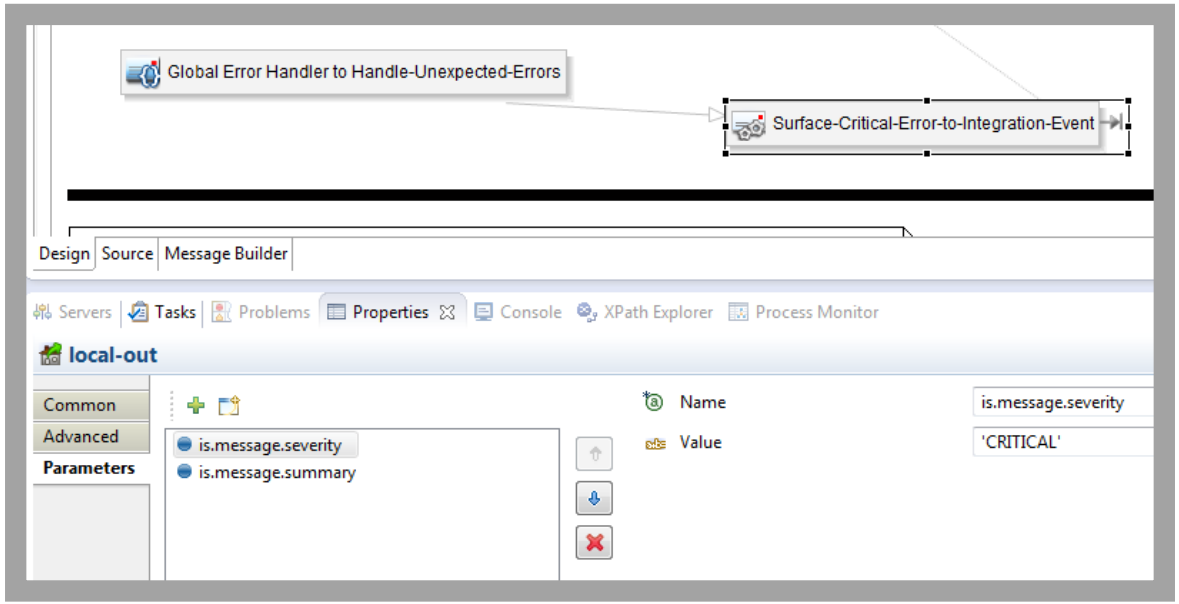


Message Severity

Severity	Message Example	Processing Behavior	Integration Event Status
INFO	Initiated process.	Processing continues.	Completed.
WARNING	Ann Smith has missing or invalid data. The value 123-456-78910 is too long for "Phone Number" and will be truncated in the output. Phone Number should have a length equal to or shorter than 12.	Processing continues.	Completed with warnings.
ERROR	Errors during data load. See report for details.	Processing continues unless the error threshold has been met.	Completed with errors.
CRITICAL	Unable to complete integration processing.	Processing stops.	Failed.

Unforeseen Errors – Global Error Handler

- All integrations should have a global error handler.
- The global error handler should link to a PutIntegrationMessage which posts an Integration Message with a severity of 'CRITICAL.'

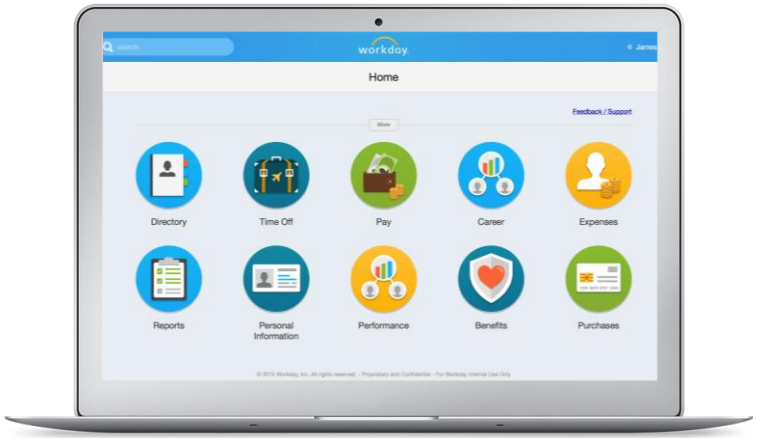


Activity 13

Global Error Handler

Estimated Time: 10 Minutes

Modify the MyHelloRaaS assembly so that unhandled errors are caught by a Global Error handler.



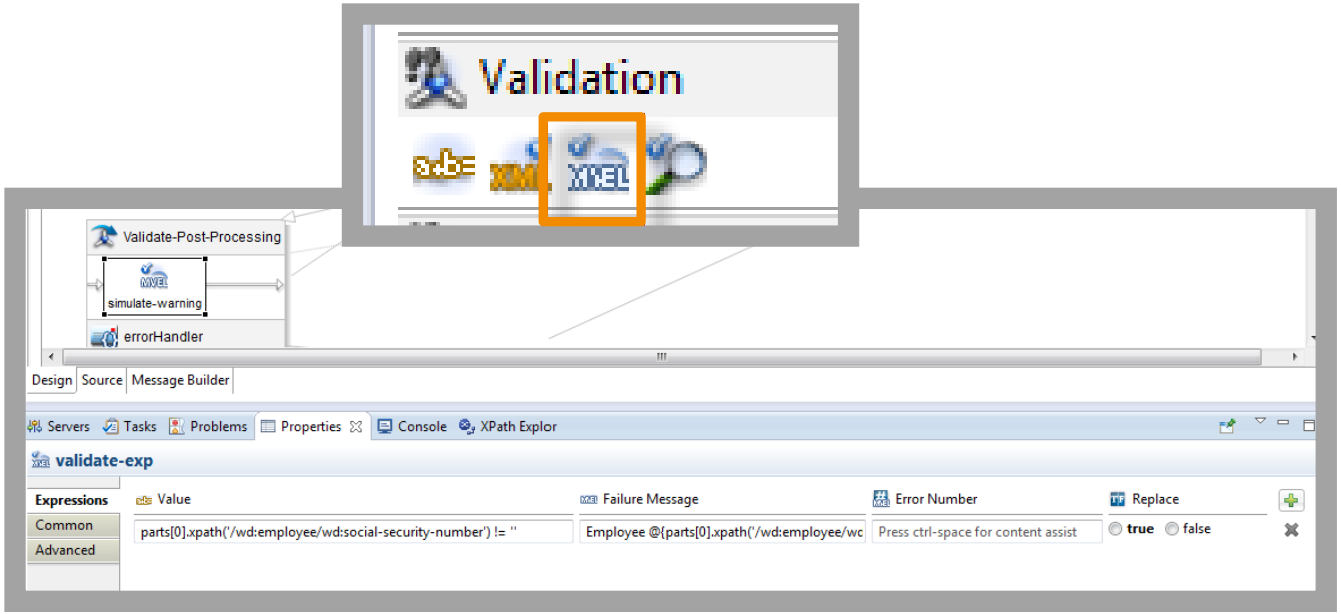
Known Problems – Throwing and Catching Errors

Two parts to trapping issues you saw might come up:

- Throw
  - Detecting the problem
  - MVEL, XSD, xPath
- Catch
  - Send to PIM
  - Send to Log
  - Send to a custom file (text log or Excel)
  - Rethrow to Global

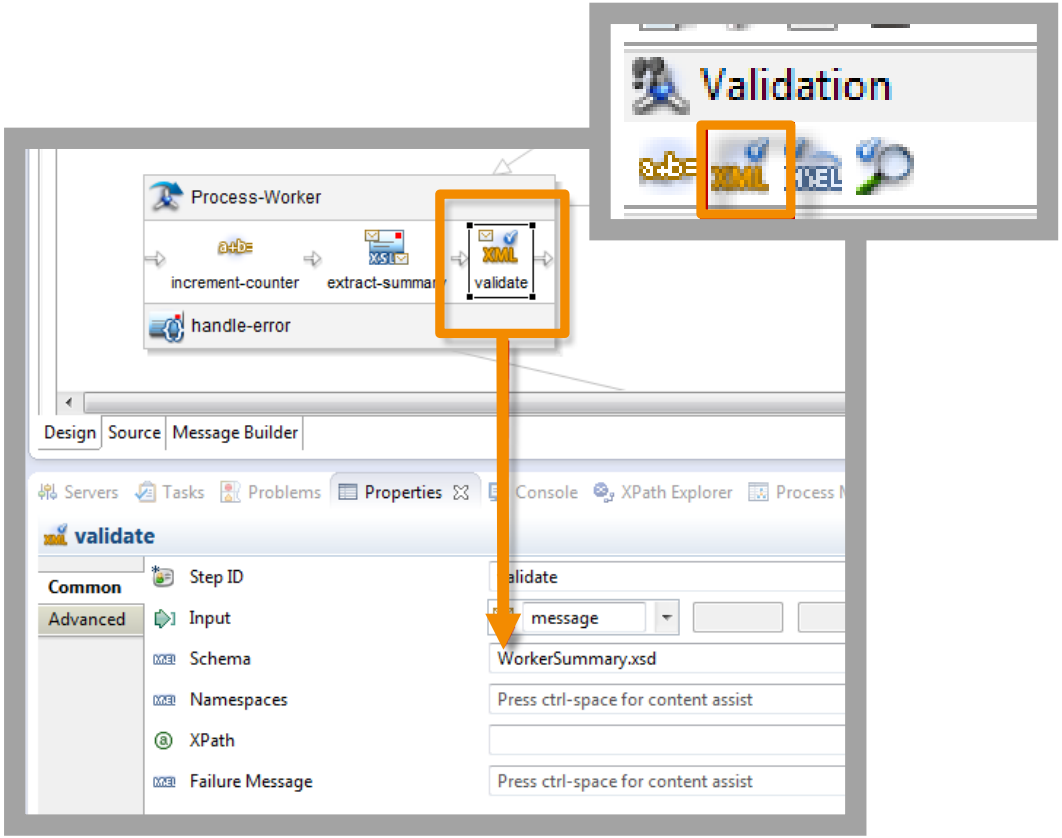
Throwing Errors – MVEL Validations

- MVEL validations are configured with one or more MVEL expressions that will cause the assembly to throw an error if they evaluate to false.
- The Validation Message can be customized to provide a meaningful error message.



Throwing Errors – XML Validation

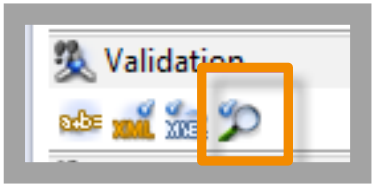
- These can be used to validate against an XSD schema.
- Be careful using XSD schemas to validate large messages. These can be configured with a custom error message using the "Failure Message" and "Replace Message" properties of the step.



Throwing Errors – XPATH Validation

These steps can be used to validate a message by applying an XPath expression to it.

This is less commonly used over the other types of validation.



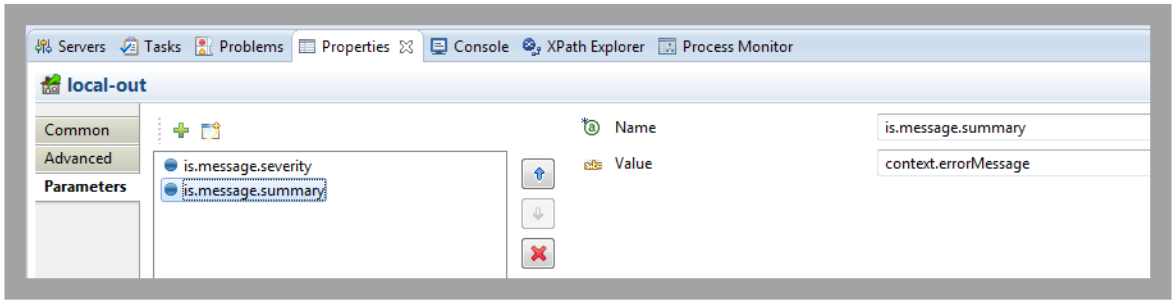
Catching Errors

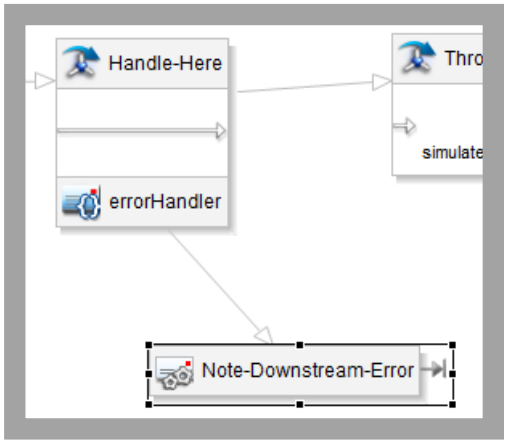
Two types of error handling components:

- **send-error**: Catches an error and passes it to an out-transport.
- **log-error:** Creates an ERROR log message. By default, this causes the error to be logged to the server log file.

Catching Errors – Messages

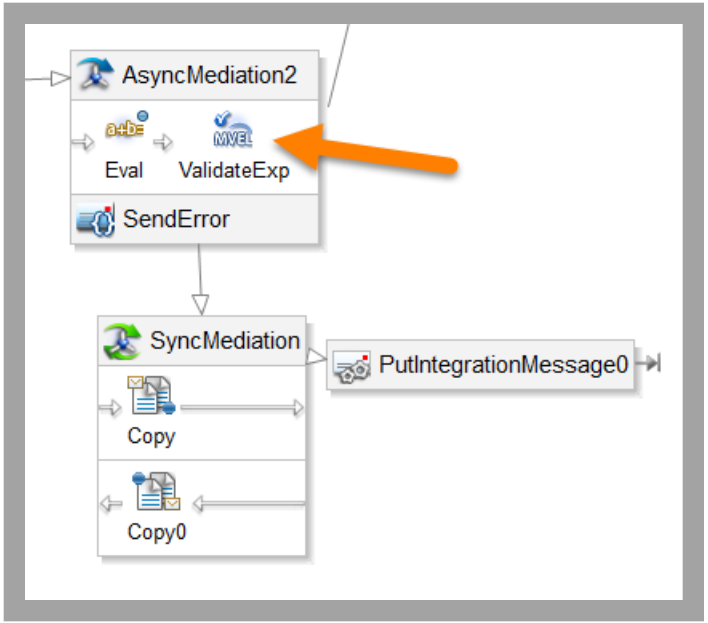
- The errorHandler should extract the error, report it to the user using a PutIntegrationMessage, trap or rethrow the error.
- Use this MVEL expression to extract the details of an error: *context.errorMessage*.



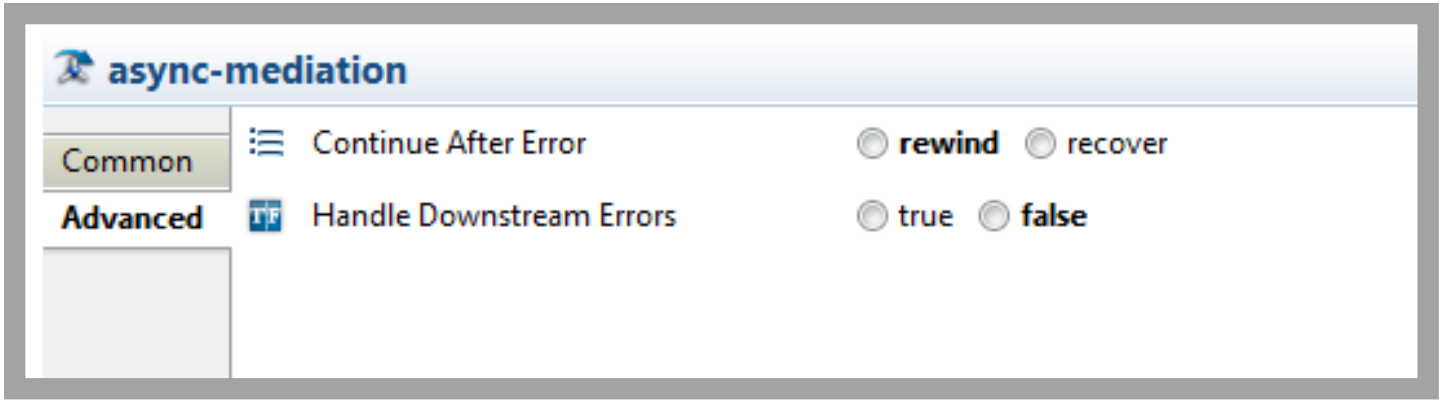


Catching Errors – Local

- Set error handlers for asynchronous and synchronous mediations where appropriate.
- The handler will only catch errors thrown within the steps of the mediation.

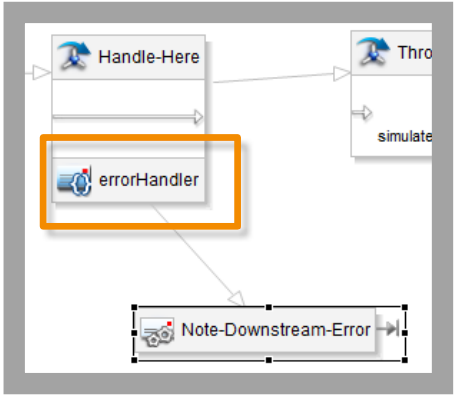


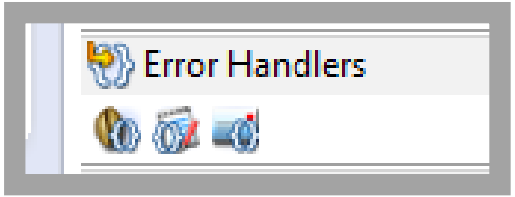
Mediation Error Handler Options



Mediation Errors – Handle Downstream Errors

- Set error handlers for asynchronous and synchronous mediations where appropriate.
- The Handle Downstream Errors property of the mediation allows you to apply the handler to only the steps within the mediation, or to handle any errors thrown by other downstream mediations or transports.

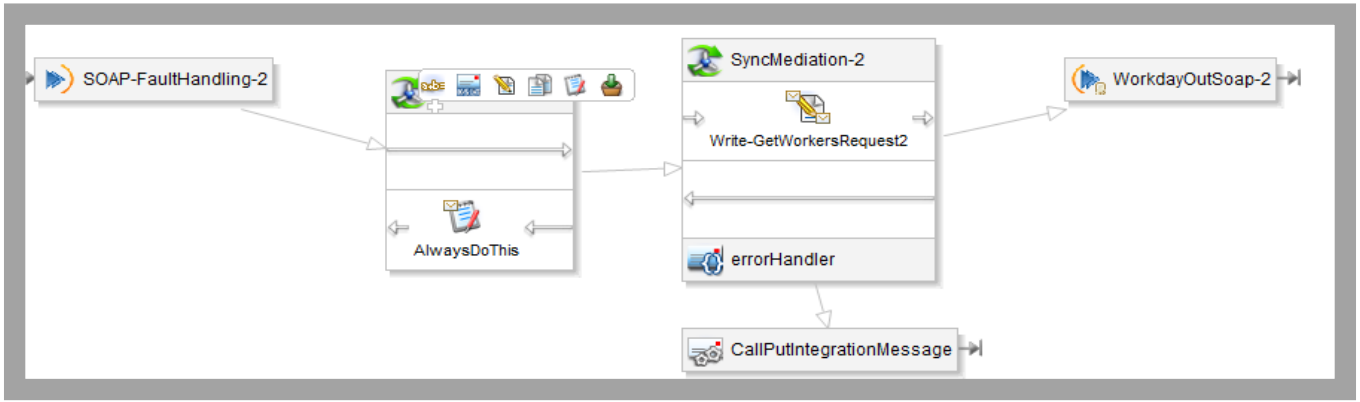




Soap Fault Errors

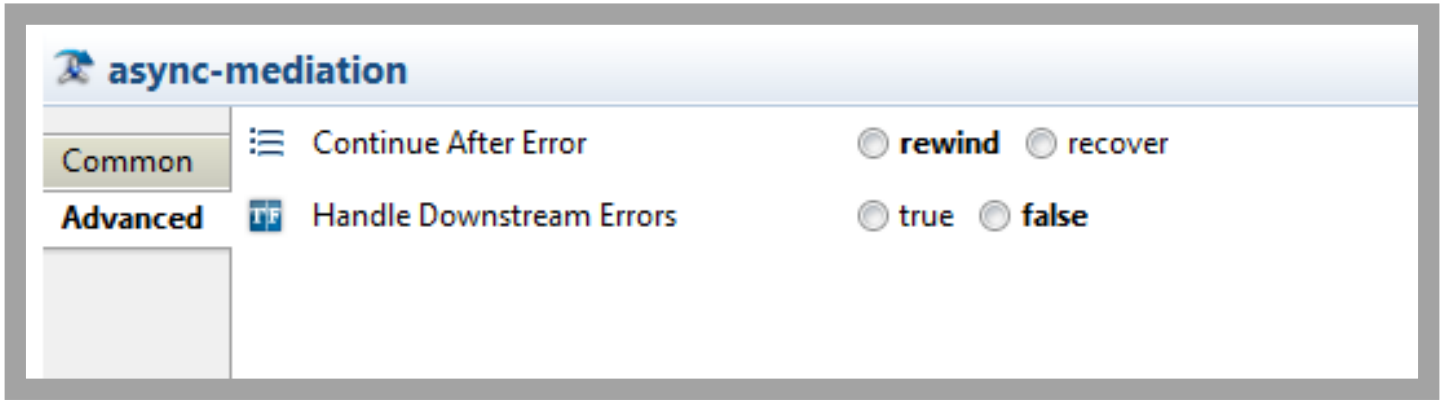
If the WorkdayOutSoap transport could return a SOAP Fault.

- To ensure that a step or mediation will always be executed, even if a workday-soap-out transport returns a SOAP Fault, a mediation component is configured to handle the SOAP Fault and continue processing the response chain.
- The mediation component properties should be set to handle-downstream-errors - true. The default is false.



Mediation Error Handler Options

- Rewind is used for errors.
- Recover is used for warnings.



All or Nothing Integrations

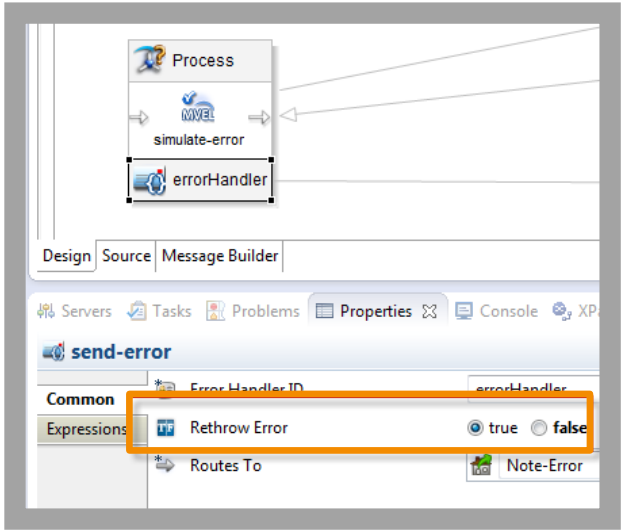
- Unless there is an explicit requirement to do so, integrations should never fail based on a problem with a single transaction.

  Example: If you are extracting worker data and one worker is missing a required SSN, your integration should have the capability to report that one worker as an error or warning, and then continue processing and producing the final output file.

- Exactly how that problem worker gets handled (e.g., omitted from file, included in file in some situations) should be dictated by requirements.

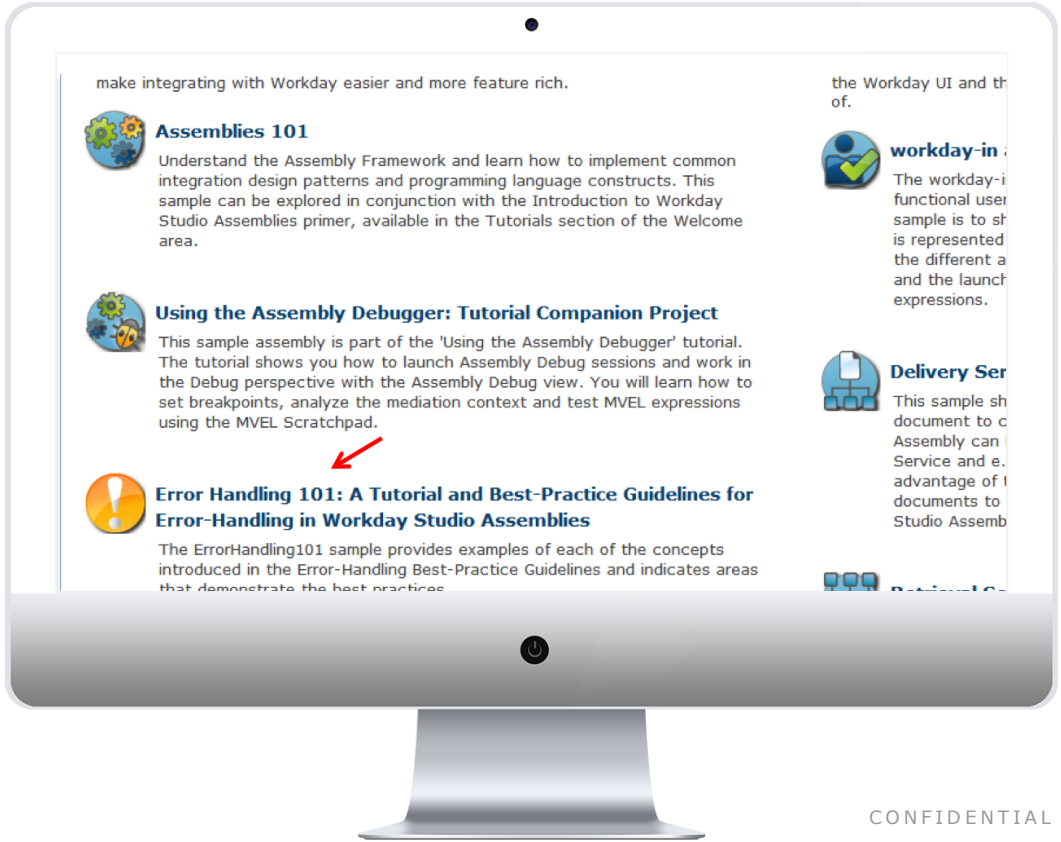
All or Nothing – Rethrow Error

- A local (Mediation) send-error handler component can define the Rethrow Errorproperty as true. The error will be rethrown.
- The Assembly runtime will send an integration message with a message-severity level of CRITICAL for the unhandled exception, so the integration will fail.



Error Handling – Best Practice

Sample contains six examples of Best-Practice Guidelines.



Error Handling Guidelines

Consider the following best-practice guidelines when implementing error handling in Workday Studio integrations.

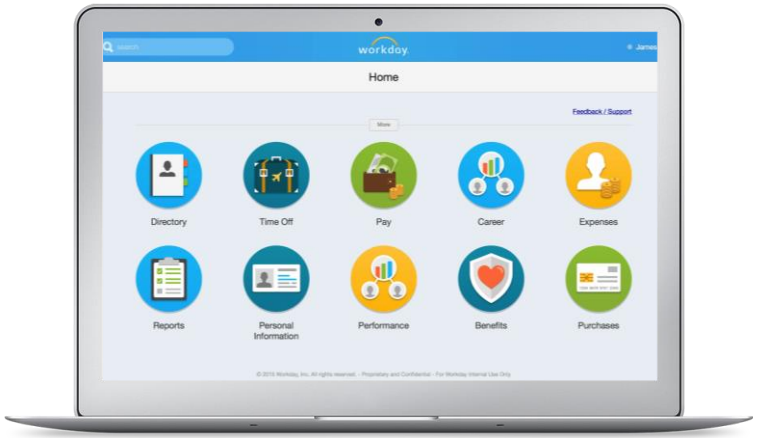
1. Surface all error-related messages to the Workday Integration Event.
2. Set message-severity levels appropriately.
3. Do not create “all-or-nothing” integrations, unless specifically required.
4. Define actionable integration messages that provide functional users, and not just Studio developers, with enough information to fix a problem.
5. Determine whether you want a mediation to handle downstream errors.
6. Ensure that the Rethrow Error error-handler property is set to false when you want the integration to report errors, but continue processing.

Activity 14

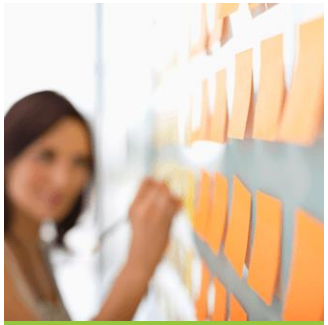
Local Error Handling

Estimated Time: 10 Minutes

- Add a Local Error Handler to MyHelloRaaS in order to detect employees in Calgary



Error Handling – Write to Error File



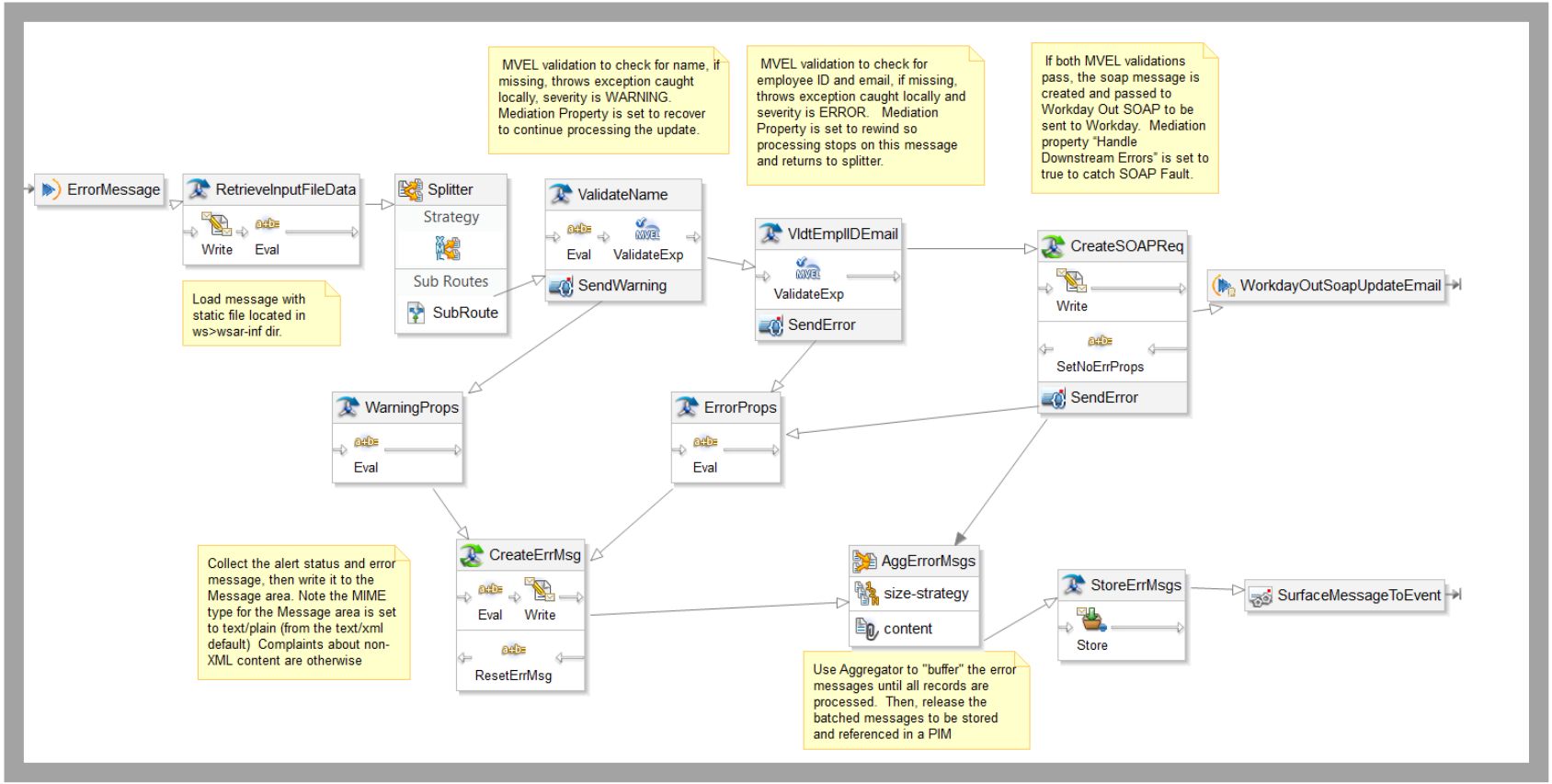
When working with large volumes of data in an integration system, using PIM to write errors and warnings can be very expensive performance-wise. Furthermore, PIMs may only raise a maximum of 500 integration events after which the events are sent to the Console Log.

You may find it more efficient to collect the errors and warnings in a file and attach it to the integration event in an output file.

In your student activity files provided by the instructor, you will find an error handling example that creates an error file, rather than using the PutIntegationMessage to log each exception.

EXTRA_DemoErrorHandlingUsingFileCollection.clar

Demo: Additional Solution



Inbound Retrieval Service

Learning Objectives

- Using the Retrieval Service
- Access documents retrieved using the Retrieval Service in the assembly
- Copy the documents into Message so assembly can process
- Implement Retrieval Service

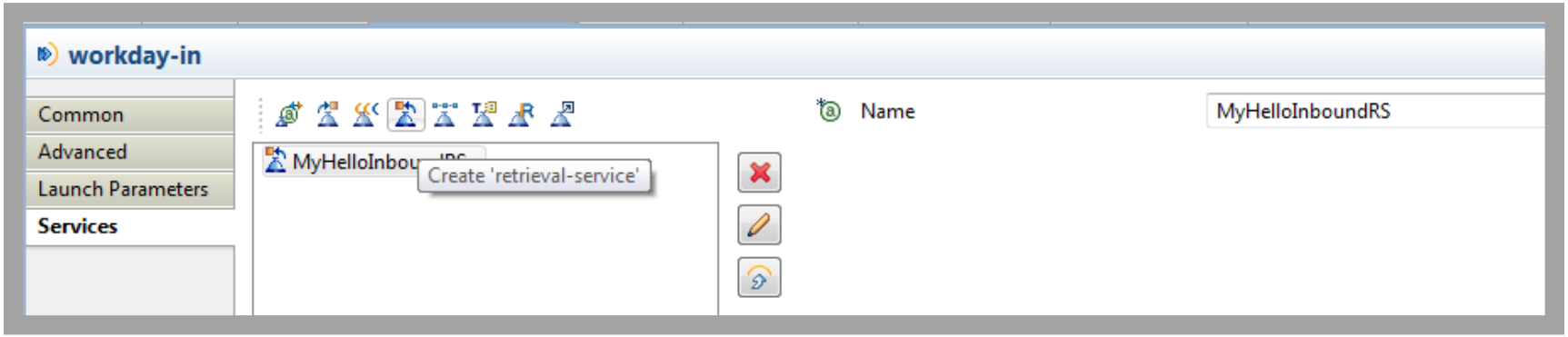
Retrieval Service

To get a file into the Mediation Context, define a retrieval service for the Workday-In that retrieves the file from an external location.

- The retrieval service runs before the integration event.
- The files are retrieved and stored in the Workday Blobitory.
- These files can then be retrieved and placed into a mediation message in your Studio Assembly.

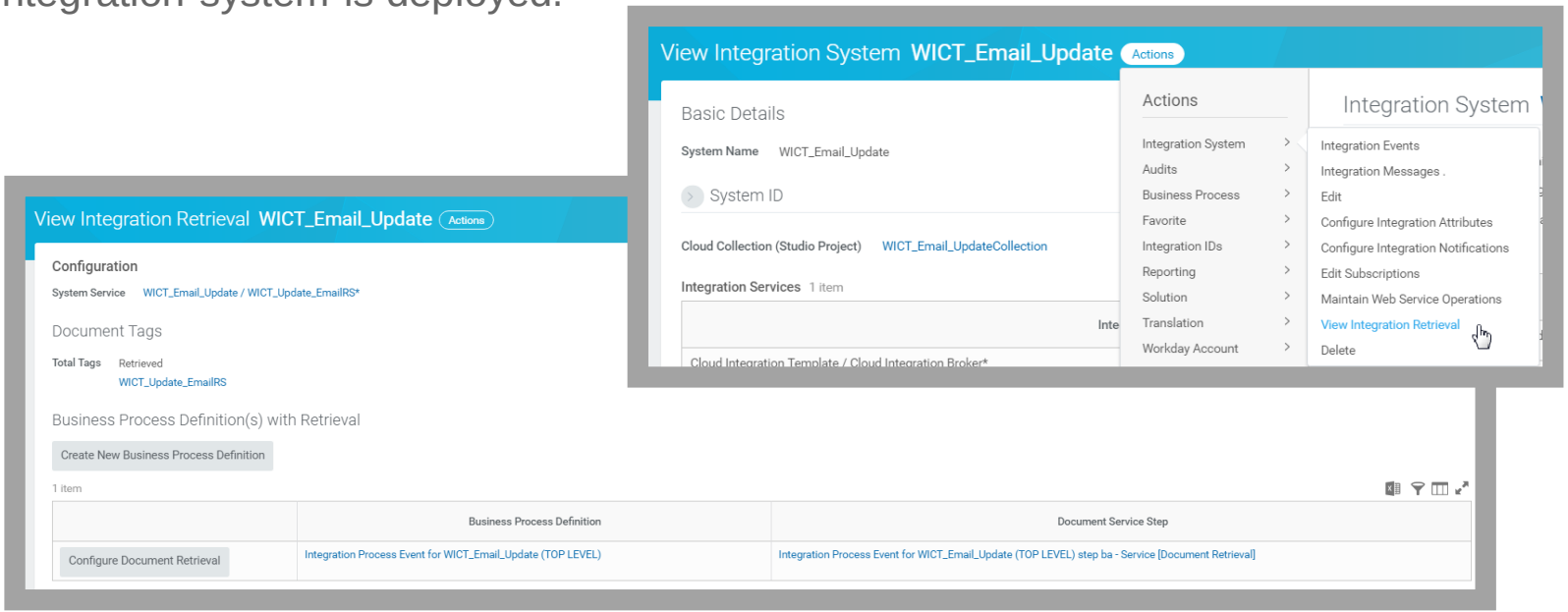
Retrieval Service

Each workday-in transport can have one or more retrieval-services defined. You need to configure those services in the Workday application after you deploy it.

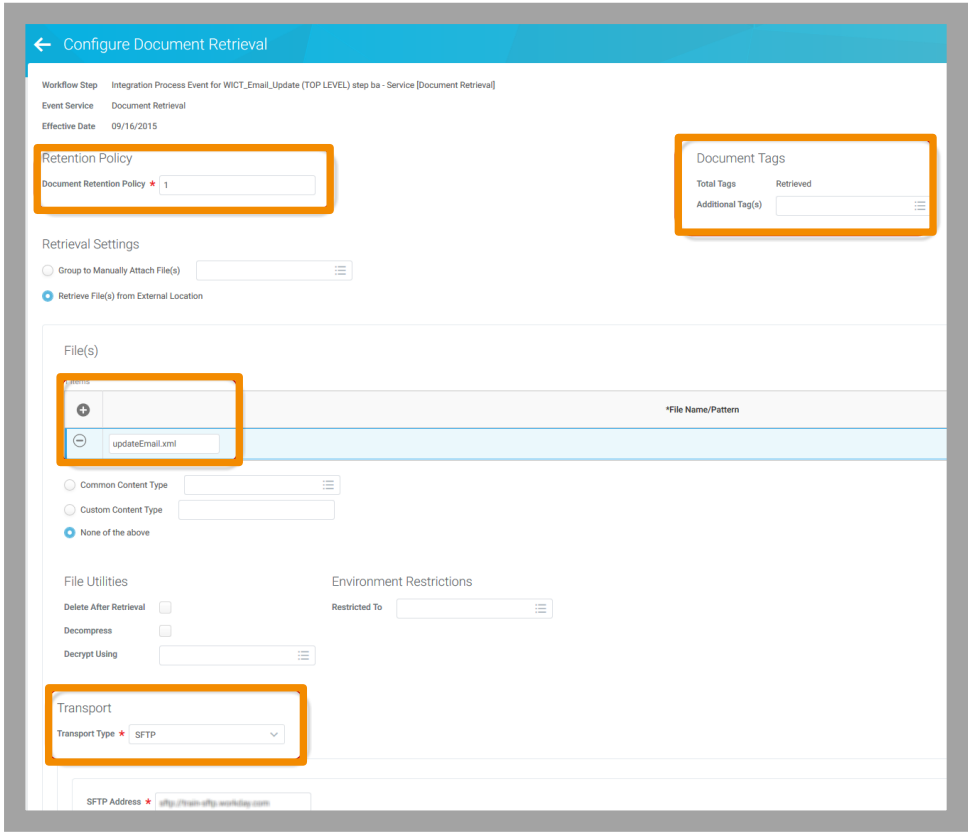


Configure Retrieval Service

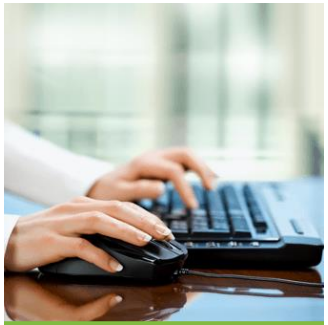
The retrieval service is configured as part of the business process in the tenant once the integration system is deployed.



Configure Retrieval Service



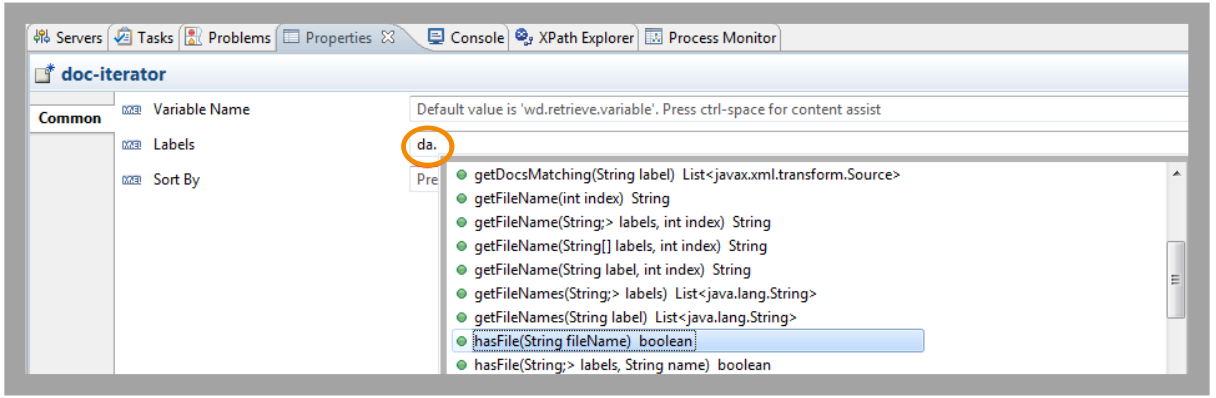
Accessing Retrieved Documents in Assembly



- If a Workday In has a retrieval service defined, then documents retrieved are stored in the Workday Blobitory.
- Put the document into your assembly and in a message so you can process it.

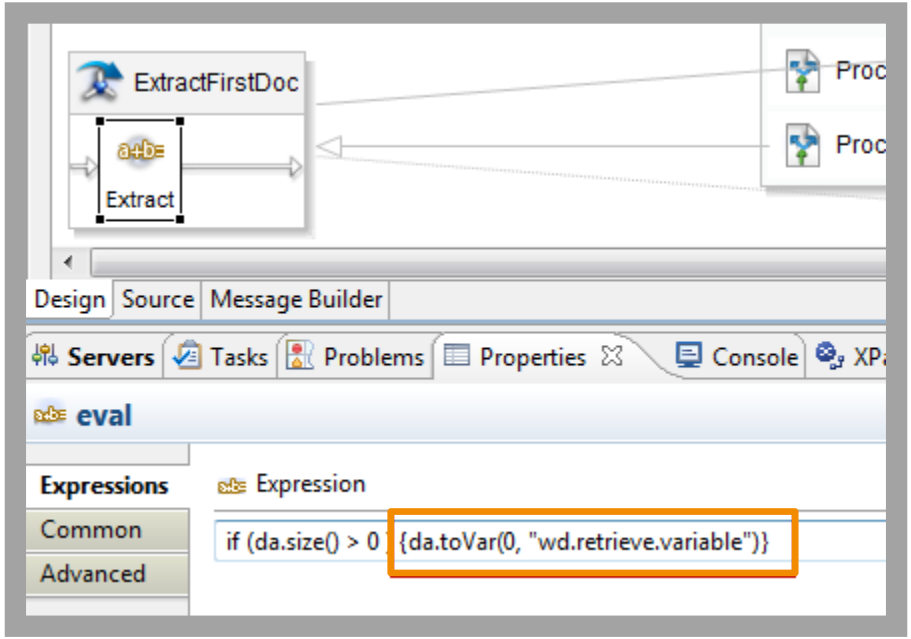
Document Accessor (da) Variable

- Workday stores integration message content as repository documents on the integration event with document accessor labels. You can specify one or more Labels, each of which references a particular document to retrieve. To retrieve all documents, specify*.
- You can express these properties in terms of MVEL template expressions. Use content assist to view the methods that the document accessor (da) MVEL variable supports.
- You can specify a name for the variable in which to store the retrieved document.



Processing One Document

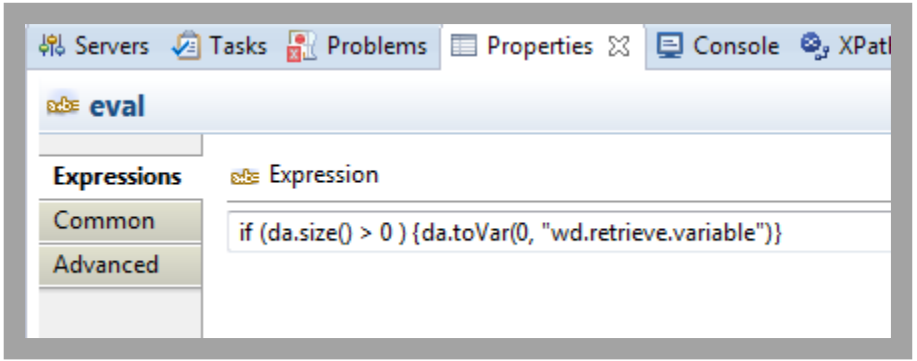
Use an eval step to copy the document into a variable for processing.



Eval Step

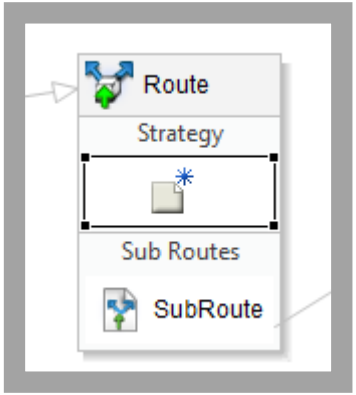
The eval step's MVEL expression uses the Document Accessor (da) MVEL variable with the following methods to extract the first document:

- **size()**: Returns the number of retrieved documents.
- **toVar(int index, String)**: Copies the data from the retrieved document with the given index to the named variable.



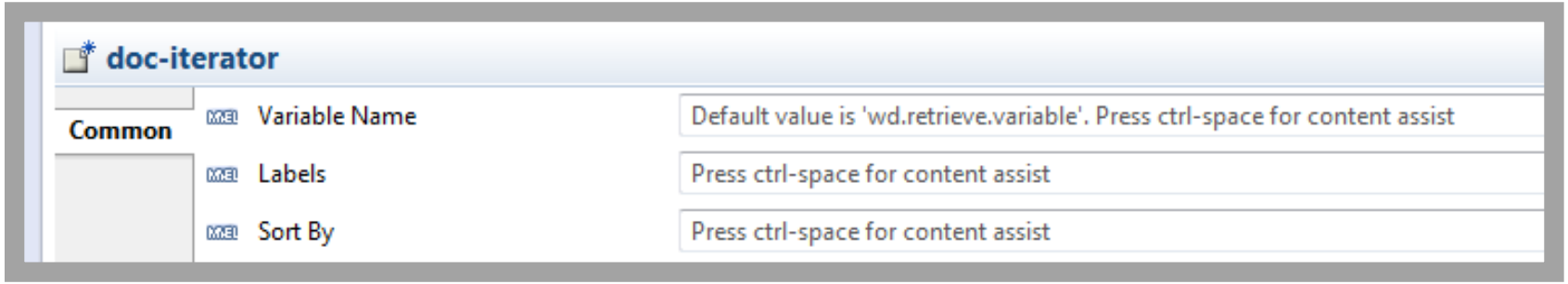
Processing Multiple Documents
Route Component – Document Iterator Strategy

This strategy allows the route component to iterate over multiple documents that your assembly retrieves using the Retrieval service.



Route Component – Document Iterator Strategy

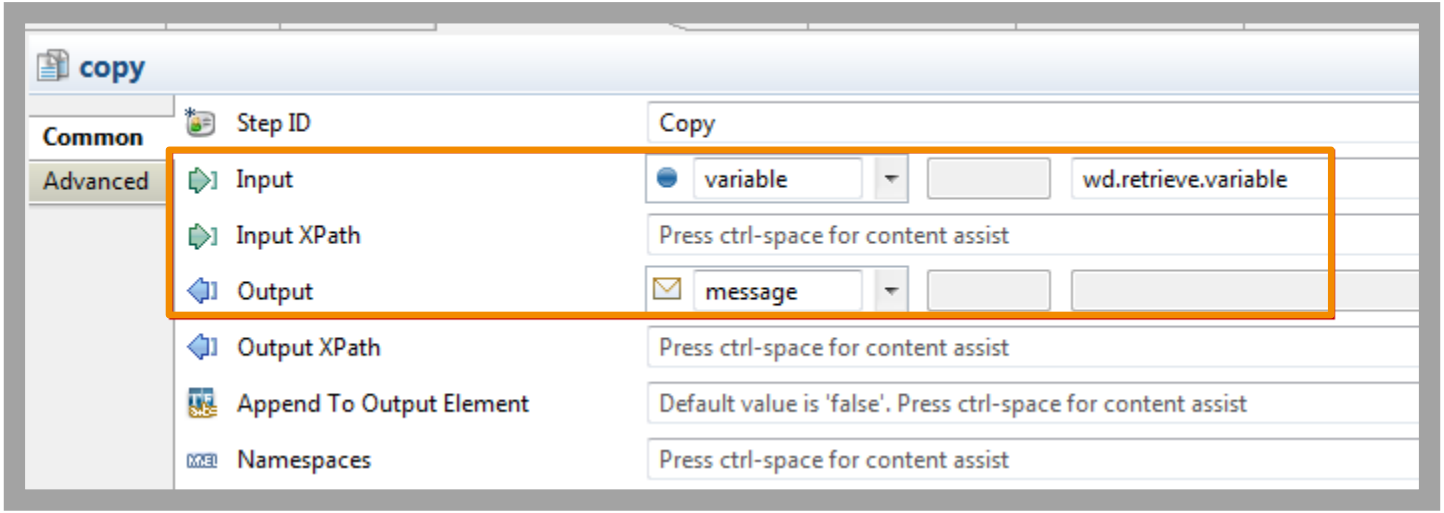
- **variable-name**: Specifies a name for the variable in which to store the retrieved document. The default value is **wd.retrieve.variable**.
- **labels**: Workday stores integration message content as repository documents on the integration event with document accessor labels. You can specify one or more labels, each of which references a particular type of document to retrieve.
- **sort-by**: An optional setting that controls the sort order for retrieving and presenting documents. The options include FILENAME_ASCENDING, FILENAME_DESCENDING, and NONE.



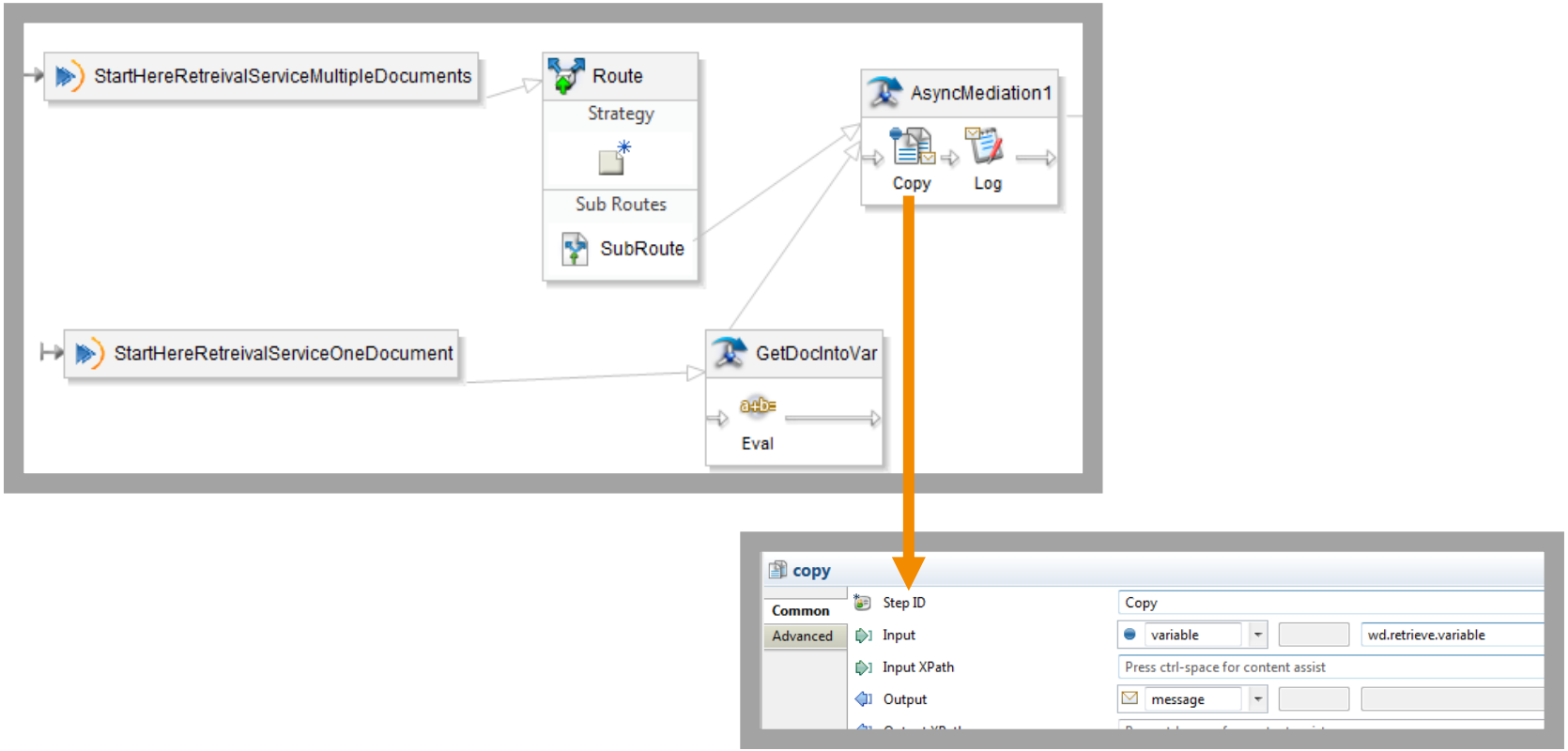
Copy Variable into a Message

Once you retrieve the document into variable (either via Doc Iterator or the Eval step), you need to copy the variable into a Message so the assembly can process it.

To do so, use the Copy step. Set the input property to variable, then specify wd.retrieve.variable as the input-variable. Set the output property to message.



Copy a Variable into a Message

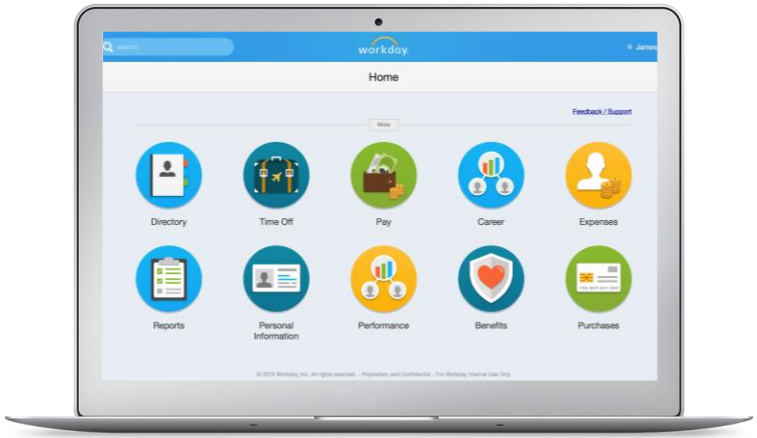


Activity 15

Build an Assembly to Update Email

Estimated Time: 15 Minutes

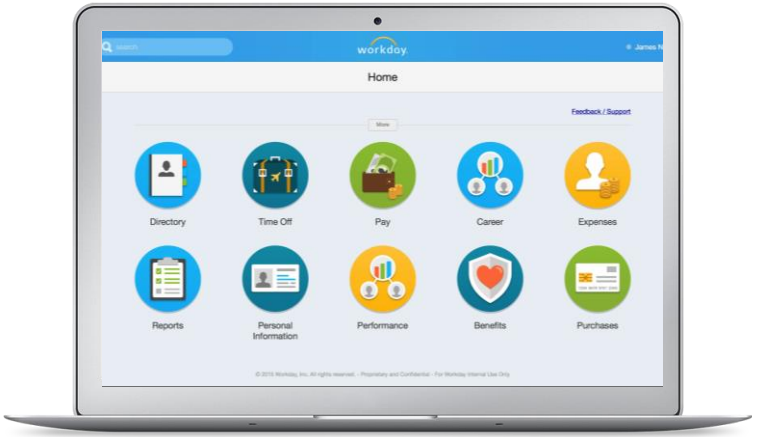
- Use the retrieval service to update employee's email addresses. The file to retrieve is called updateEmail.xml located on the instructor's SFTP Server.
- Retrieve the document from the SFTP Server using the Retrieval Service. We will use a router with a Doc Iterator strategy to process the retrieved document.



Activity 16

Design and Whiteboard

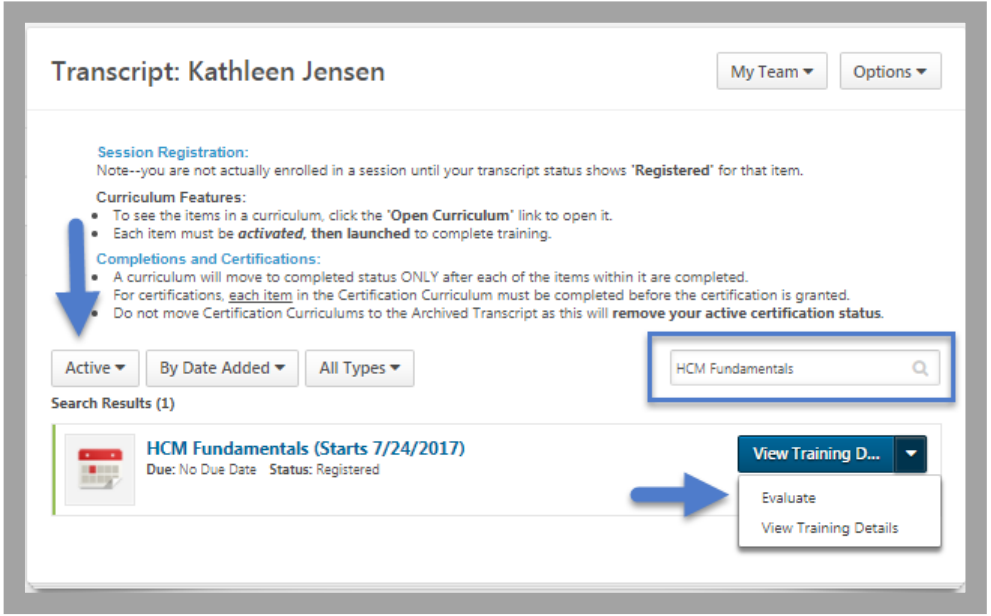
Estimated Time: 30 Minutes



Whiteboard – Activity 14

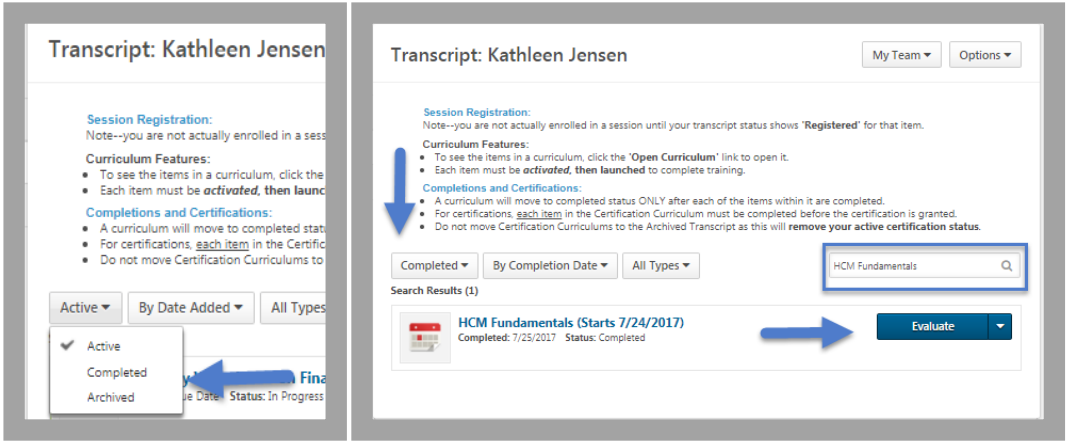
Class Evaluation:
Available at the Start of the Last Day of Class

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**.



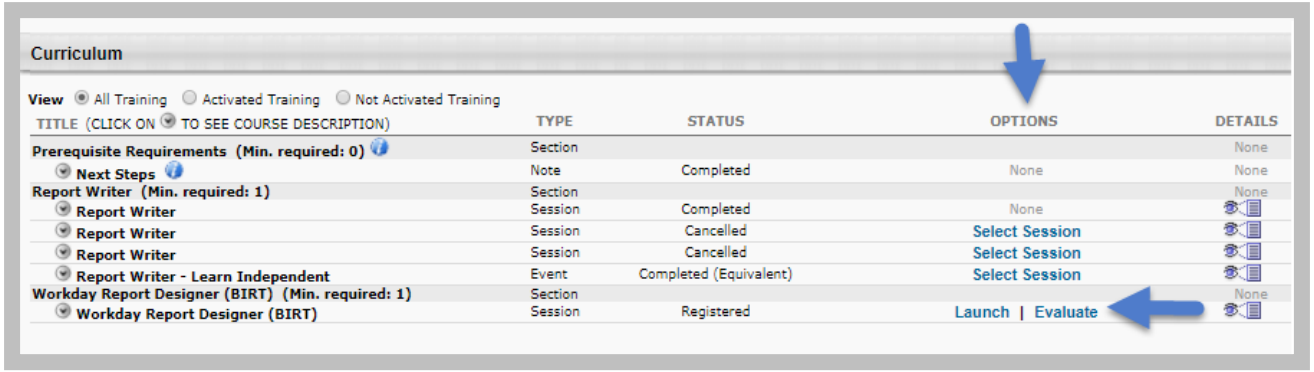
Class Evaluation:
Available After Class Ends and Roster Submitted

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**.



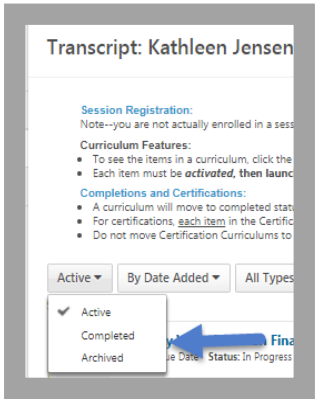
Class Evaluation (Session Within a Curriculum): Available at the Start of the Last Day of Class

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.
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**.



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