

Course Exercises Guide

# Process Implementing with IBM Business Process Manager Standard V8.5.7 - II

Course code WB824 / ZB824 ERC 2.1



## April 2018 edition

### Notices

This information was developed for products and services offered in the US.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing  
IBM Corporation  
North Castle Drive, MD-NC119  
Armonk, NY 10504-1785  
United States of America*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

### Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

© Copyright International Business Machines Corporation 2016, 2018.

**This document may not be reproduced in whole or in part without the prior written permission of IBM.**

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Trademarks</b> .....	<b>v</b>
<b>Exercises description</b> .....	<b>vi</b>
<b>Exercise 1. Exploring the REST API</b> .....	<b>1-1</b>
General exercise information .....	2
How to follow the exercise instructions .....	2
Section 1. Start the IBM Business Process Manager Process Center Server and the designer tools ....	1-5
1.1. Start the Process Center server .....	1-5
1.2. Import a process application by using the Web Process Designer .....	1-9
Section 2. Examine a failed instance .....	1-17
2.1. Explore the process .....	1-17
2.2. Run the instance .....	1-23
2.3. Explore the Move the Token client-side human service implementation .....	1-28
2.4. Use the Move the Token service to complete the process .....	1-39
<b>Exercise 2. Handling content events in a process</b> .....	<b>2-1</b>
Section 1. Implement a document start event on a process .....	2-2
1.1. Change the time-based start event to a document start event .....	2-2
1.2. Create the event subscription .....	2-4
1.3. Configure the start event by using the undercover agent and map a variable to the output ...	2-10
Section 2. Test the start content event .....	2-21
2.1. Test the start content event in the Hiring Request Process .....	2-21
2.2. View the instance details in the browser .....	2-23
<b>Exercise 3. Localizing a coach</b> .....	<b>3-1</b>
Section 1. Create a localization resource .....	3-2
1.1. Create a localization resource .....	3-2
1.2. Bind the localization to a coach or coach view .....	3-4
1.3. Create resource keys .....	3-6
1.4. Assign keys to the labels on the coach .....	3-8
1.5. Import resource keys .....	3-10
1.6. Localize the department menu .....	3-14
Section 2. Test the localization .....	3-21
2.1. Change the localization preference in the portal .....	3-21
<b>Exercise 4. Implementing the "four eyes" policy</b> .....	<b>4-1</b>
Section 1. Create the four eyes policy .....	4-2
1.1. Modify the process .....	4-2
1.2. Create the team filter service .....	4-8
1.3. Apply the team filter service to the activity assignment .....	4-13
Section 2. Test the four eyes policy .....	4-14
2.1. Test the four eyes policy in the process portal .....	4-14
<b>Exercise 5. Building a cancellation pattern</b> .....	<b>5-1</b>
Section 1. Implement a cancel event .....	5-2
1.1. Create the event .....	5-3
1.2. Create an undercover agent .....	5-5
1.3. Implement the boundary event with the UCA .....	5-6
Section 2. Create a service to trigger the UCA .....	5-10

2.1. Create the wrapper service .....	5-10
2.2. Create a test harness .....	5-12
Section 3. Cancel an instance .....	5-18
3.1. Test the cancellation UCA .....	5-18
<b>Exercise 6. Implementing a multi-instance loop .....</b>	<b>6-1</b>
Section 1. Create a multi-instance loop .....	6-2
1.1. Delete the unneeded activity .....	6-2
1.2. Add the selection list to the approval task .....	6-3
1.3. Convert the approval step to a multi-instance loop and map the output variables .....	6-13
1.4. Test the multi-instance loop with the condition feature .....	6-20
<b>Exercise 7. Building web service connections .....</b>	<b>7-1</b>
Section 1. Create an inbound integration .....	7-2
1.1. Create an inbound web service to trigger the UCA .....	7-2
Section 2. Create an outbound integration .....	7-4
2.1. Test the inbound integration by creating an outbound integration .....	7-4
2.2. Test the integrations .....	7-8
<b>Appendix A. Data dictionary .....</b>	<b>A-1</b>
Training database .....	1
Part 1: Departments .....	A-2
Part 2: Divisions .....	A-2
Part 3: JobLevels .....	A-2
Part 4: Positions .....	A-4
Part 5: IncidentCategory .....	A-4
Part 6: IncidentType .....	A-5

---

# Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide:

DB™	DB2®	developerWorks®
Express®	Initiate®	PartnerWorld®
Rational®	Redbooks®	WebSphere®

Intel and Intel Core are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

VMware and the VMware "boxes" logo and design, Virtual SMP and VMotion are registered trademarks or trademarks (the "Marks") of VMware, Inc. in the United States and/or other jurisdictions.

Other product and service names might be trademarks of IBM or other companies.

---

# Exercises description

This course includes the following exercises:

- Exercise 1: Exploring the REST API
- Exercise 2: Handling content events in a process
- Exercise 3: Localizing a coach
- Exercise 4: Implementing the “four eyes” policy
- Exercise 5: Building a cancellation pattern
- Exercise 6: Implementing a multi-instance loop
- Exercise 7: Building web service connections

In the exercise instructions, you can check off the line before each step as you complete it to track your progress.

Most exercises include required sections, which should always be completed. It might be necessary to complete these sections before you can start later exercises. If you have sufficient time and want an extra challenge, some exercises also might include optional sections that you can complete.



## Important

The exercises in this course use a set of lab files that might include scripts, applications, files, solution files, PI files, and others. The course lab files can be found in the following directory:

C:\labfiles for the Windows platform

/usr/labfiles for the Linux platform

The exercises point you to the lab files as you need them.



## Important

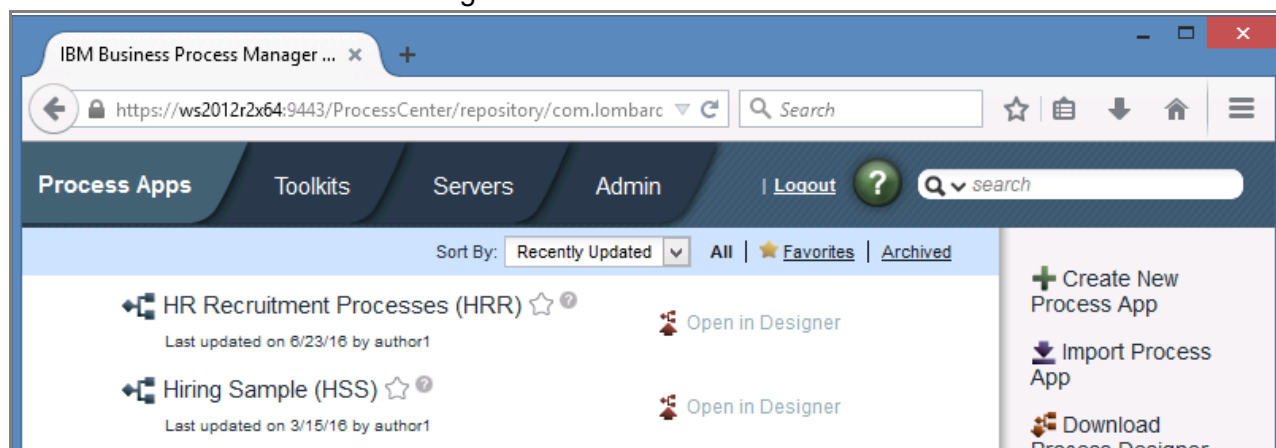
Online course material updates might exist for this course. To check for updates, visit the Instructor wiki at <http://ibm.biz/CloudEduCourses>.

---



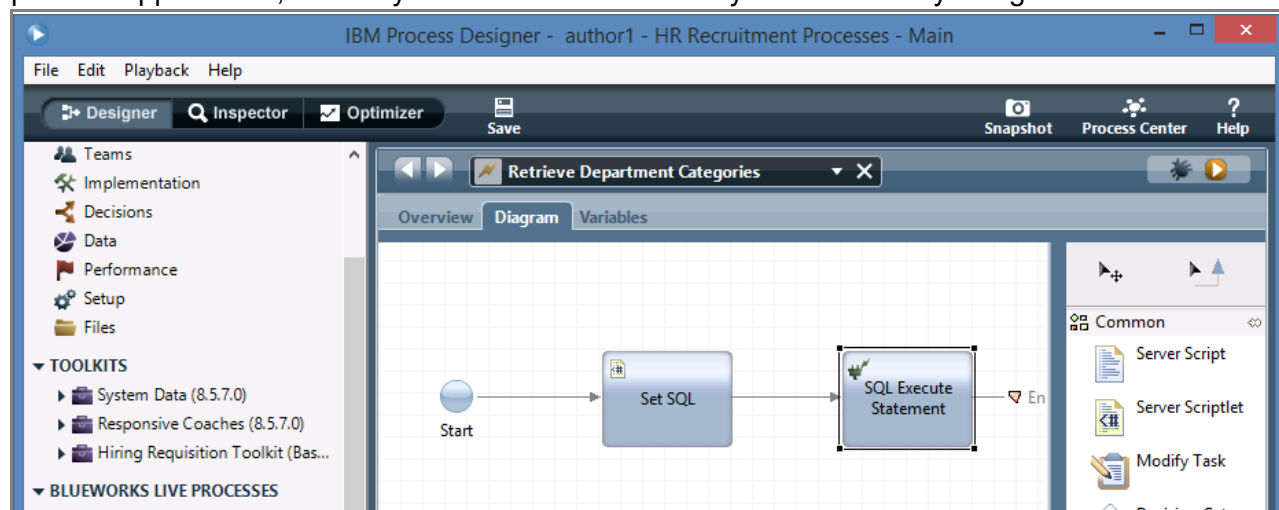
## Important

Throughout this course, you use two tools to create your process assets. It is important to distinguish between the two tools because of the similar design and names. You create most of the process modeling and front-end (coach) development by using what is referred to throughout this course as the web Process Designer.



This browser-based tool uses your desktop browser to create most of your process application. The web Process Designer is cross-browser compatible and the designer runs on multiple operating systems. For more information about browser and system compatibility, see the *Detailed System Requirements Supported Software* page for IBM Business Process Manager.

The second tool is used to implement services in your library along with some configuration options that are not found in the web Process Designer. This tool is called the Process Designer client application because this tool is downloaded and installed on a Windows based host. The web designer is a web page that you load inside your Internet browser, the client application must be launched as an executable file on your computer. You use both tools simultaneously to create your process applications, so always be aware of which tool you are currently using.



---

# Exercise 1. Exploring the REST API

## Estimated time

01:00

## Overview

In this exercise, you create an unstructured ad hoc activity and use the REST API to move a token from a failed instance.

## Objectives

After completing this exercise, you should be able to:

- Examine a failed instance in Process Inspector
- Restart a failed instance and complete the process
- Explore REST APIs by using the REST API Tester

## Introduction

In this exercise, you work with a process in which some of the instances are succeeding while some are failing. The business unit wants to target some of those failed instances to move the token on the process. One of the failed instances is critical; you need this instance to pass beyond the failed step. Since it is a business emergency, you need to move the token and get the process beyond that failed activity. You solve this problem of moving the token by using an external implementation of the REST API.

You can complete the exercise regardless of your proficiency in these languages and technologies. Answers and screen captures are provided for you throughout the exercise. Read each step of the exercise and determine whether you can complete the requirements without assistance from the code provided. Attempt to troubleshoot your own code before using the answer given, as it simulates your real-world experience.

## Requirements

None.



## General exercise information

This section provides general information about the exercises in this course. Review this section before starting the exercises.

### User IDs and passwords

This table contains a list of user ID and password information for this course.

Entry point	User ID	Password
VMware image	Administrator	web1sphere
Windows Server 2012 R2	Administrator	web1sphere
IBM BPM author user	author1	author01
IBM BPM user 1	user1	user01
IBM BPM user 2	user2	user02
IBM BPM WebSphere Application Server cell administrator	Administrator	web1sphere



#### Note

#### For IBM Business Process Manager on Cloud users

To follow along with the exercise instructions, you must create the author1, user1, and user2 accounts in your IBM Business Process Manager on Cloud environment. Use the administrative console to add the users as local accounts.

The data in the data dictionary must also be added to the database in [Appendix A, "Data dictionary"](#) in this exercise guide. Contact your cloud administrator for assistance in setting up your environment.

## How to follow the exercise instructions

### Exercise structure

Each exercise is divided into sections with a series of steps and substeps. The step represents an action to be completed. If required, the substeps provide guidance on completing the action.

#### Example:

- \_\_\_ 1. Create a user account named **ADMIN**.
    - \_\_\_ a. Right-click **My Computer** and click **Manage**.
    - \_\_\_ b. Expand **Local Users and Groups**.
- ... *continue*

In this example, the creation of a user account is the action to be completed. The substeps underneath provide specific guidance on how to create a user account. (In this example, the

instructions are for the Windows operating system.) Words that are highlighted in bold represent menu items, field names, and other screen elements.

Based on the actions that are mentioned in the lab, you can complete the steps on your own without using the substeps that are provided or you can use all the substeps. And when you complete the action, you should review the substeps that are mentioned to verify your steps.

If you encounter an error while working on the exercise, go back and review your steps to find out whether you successfully completed all the steps.

After completing the exercise, you are encouraged to go back over all the steps to make sure that you understand why you did the steps and how you did it.

An underscore precedes each step and substep. You are encouraged to use these markers to track your progress. As you complete a step, place an X or a check mark on the underscore to indicate that it is completed. By tracking your progress in this manner, you can stay focused if you are interrupted during a lengthy exercise.

## Text highlighting in exercises

Different text styles indicate various elements in the exercises.

Words that are highlighted in **bold** represent GUI items that you interact with, such as:

- Menu items
- Field names
- Icons
- Button names

Words that are highlighted with a `fixed font` include the following items:

- Text that you type or enter as a value
- System messages
- Directory paths
- Code

The exercises in this course use a set of lab files that might include scripts, applications, files, solution files, and others. The course lab files are available in the `C:\labfiles` directory. The exercises point you to the lab files as you need them.

### For IBM Business Process Manager on Cloud Users

The exercises in this book are designed for the on-premises version of IBM Business Process Manager. If you are taking the course that uses the IBM Business Process Manager on Cloud product, download the labfiles assets from the Cloud Education website. Substitute the local files that you extract from the labfiles compressed file when the text references files that are contained in the `c:\labfiles` folder.

---

**Stop*****Course updates and errata***

A Course Corrections document might be available for this course.

If you are taking the class with an instructor, the instructor can provide this document to you.

If you are taking the course in a self-paced environment, the course corrections document is provided with the other manuals.

To check whether a Course Corrections document exists for this course:

1. Go to the following URL: [http://www.ibm.com/developerworks/connect/middleware\\_edu](http://www.ibm.com/developerworks/connect/middleware_edu).
  2. On the web page, locate, and click the **Course Information** category.
  3. Find your course in the list and click the link.
  4. Click the **Attachments** tab to see whether an errata document exists with updated instructions.
  5. To save the file to your computer, click the document link and follow the dialog box prompts.
-

## Section 1. Start the IBM Business Process Manager Process Center Server and the designer tools

Before you can start IBM Business Process Manager, three server configurations must be started. After logging on to the lab environment, start the Deployment Manager profile, the Node Agent profile, and the Deployment Environment.

All three server configurations must be started in order: starting with the Deployment Manager profile, then the Node Agent profile, and followed last by the Deployment Environment. To accomplish it quickly, IBM Business Process Manager provides a Quick Start menu to automatically run the server start in order. Follow the instructions to start the servers by using the Quick Start menu in step 1.1.

### 1.1. Start the Process Center server



#### Note

#### For IBM Business Process Manager on Cloud users

You do not need to start the Process Center. It is already running in your cloud environment.

\_\_\_ 1. Start the Process Center server from the quick start menu.

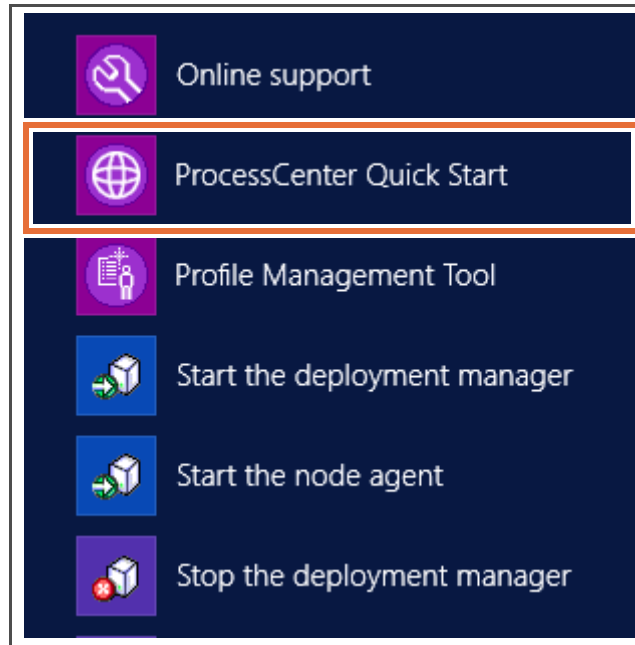
\_\_\_ a. Click the Windows **Start** menu.



\_\_\_ b. Click the **Down arrow**.



\_\_\_ c. Click **IBM > ProcessCenter Quick Start**.



### Important

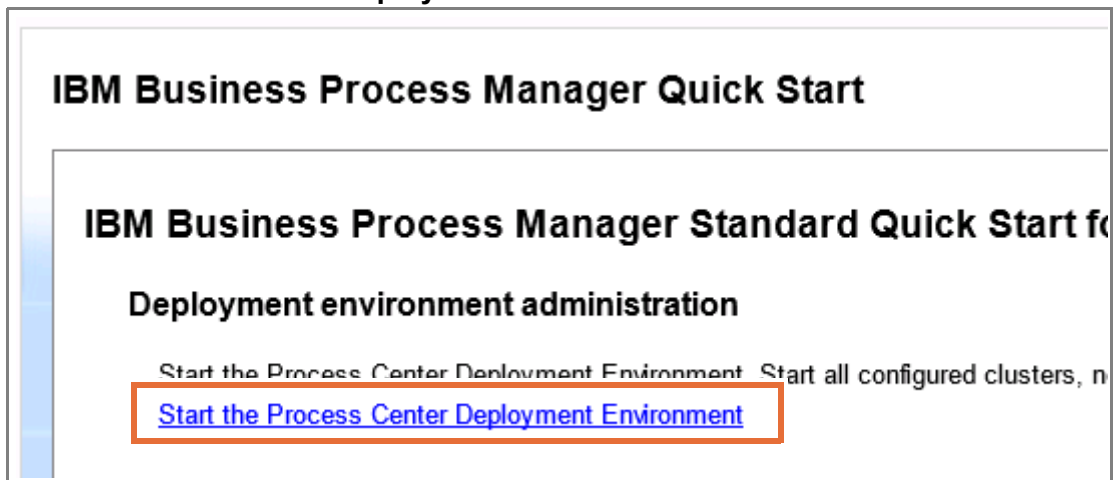
Some of the artifacts, such as the Process Designer, are labeled 8.5 even though the product version is 8.5.7. You can verify the server version with the System Data toolkit version that is installed on the Process Center later in the course.

\_\_\_ d. A ProcessCenter Quick Start command prompt window appears.

A screenshot of a Windows command prompt window titled 'ProcessCenter Quick Start'. The window has a black background with white text. The text shows the following commands and their output:

```
C:\IBM\BPM\v8.5\bin>set local
C:\IBM\BPM\v8.5\bin>set WAS_USER_SCRIPT=
C:\IBM\BPM\v8.5\bin>set USER_INSTALL_ROOT=
C:\IBM\BPM\v8.5\bin>set mode=0
C:\IBM\BPM\v8.5\bin>call "C:\IBM\BPM\v8.5\bin\setupCmdLine.bat" ProcessCenter Dm
grProfile
```

- \_\_\_ e. In the **IBM Business Process Manager Standard Quick Start** browser, click the **Start the Process Center Deployment Environment** link.



It takes a few moments to start the deployment environment.

A command prompt window runs through a start of the **Deployment Manager** profile, **Node Agent** profile, and **Deployment Environment**. Allow the entire start to complete. It can take 15 – 20 minutes, so make sure that you provide ample time for this initial start.

- \_\_\_ f. Press any key when prompted.

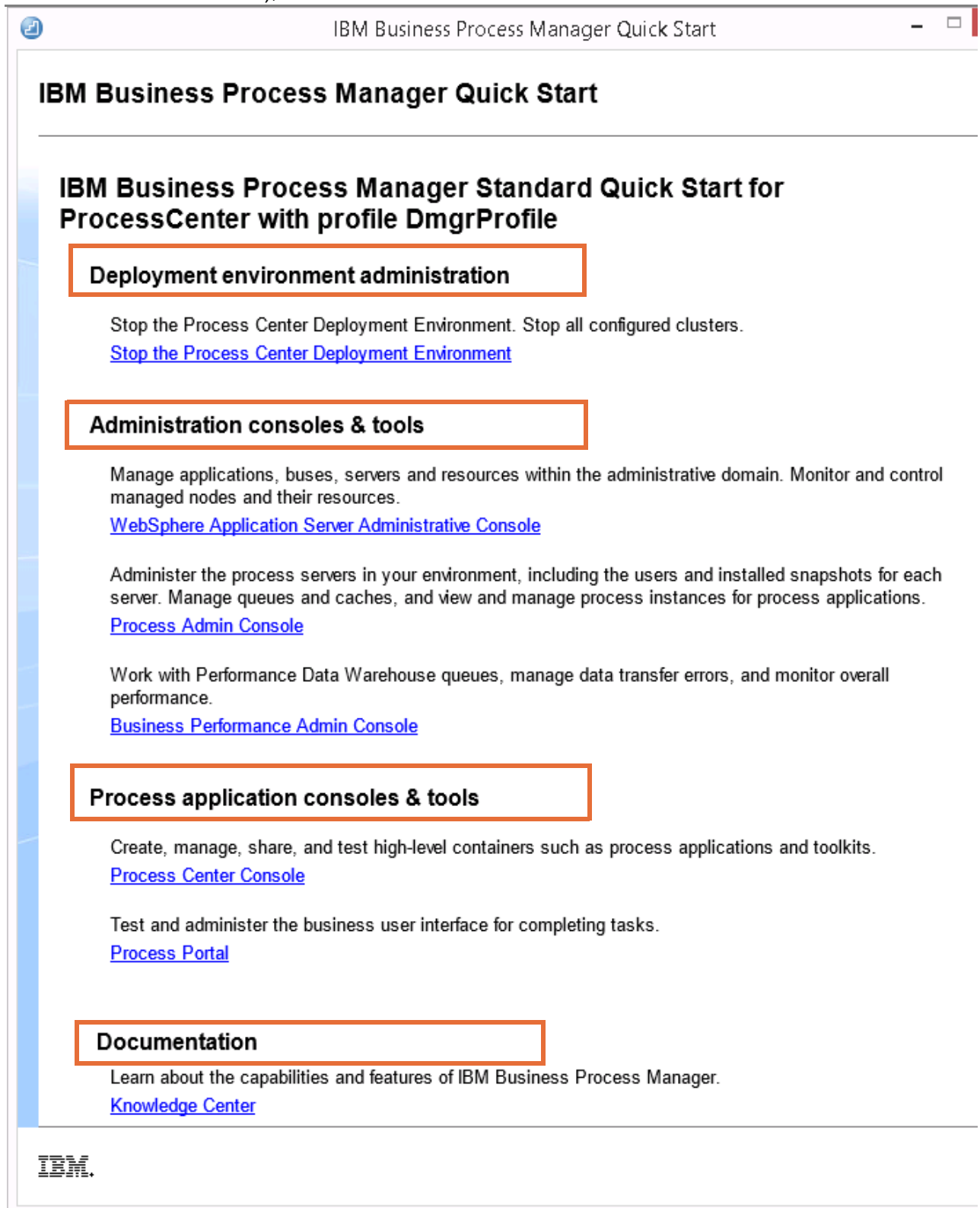
```

C:\Windows\system32\cmd.exe

CWUP00001I: Running configuration action detectNewProducts.ant
ADMU0116I: Tool information is being logged in file
           C:\IBM\BPM\v8.5\profiles\DmgrProfile\logs\dmgr\startServer.log
ADMU0128I: Starting tool with the DmgrProfile profile
ADMU3100I: Reading configuration for server: dmgr
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server dmgr open for e-business; process id is 1416
Starting node Node1.
CWUP00001I: Running configuration action detectNewProducts.ant
ADMU0116I: Tool information is being logged in file
           C:\IBM\BPM\v8.5\profiles\Node1Profile\logs\nodeagent\startServer.log
ADMU0128I: Starting tool with the Node1Profile profile
ADMU3100I: Reading configuration for server: nodeagent
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server nodeagent open for e-business; process id is 2232
Starting cluster SingleCluster.
When the BPMConfig command is used to start a deployment environment, it invokes
the processes that are used to start the associated clusters. If the command is
successful in invoking the processes, it returns a message to report that the c
ommand completed successfully. However, to determine whether the cluster members
were all started successfully, you need to check the log files of the cluster m
embers. The log files are located in <profile_root>/logs.
The 'BPMConfig.bat -start -profile DmgrProfile -de ProcessCenter' command comple
ted successfully.
Press any key to continue . . . _

```

- \_\_\_ g. The Quick Start menu populates. It might take up to 20 minutes to fully start the servers. When the following four sections populate on the **IBM Business Process Manager Standard Quick Start** browser window (Deployment environment administration, Administration consoles & tools, Process application consoles & tools, and Documentation), the server is started.



IBM Business Process Manager Quick Start

---

**IBM Business Process Manager Standard Quick Start for ProcessCenter with profile DmgrProfile**

**Deployment environment administration**

Stop the Process Center Deployment Environment. Stop all configured clusters.  
[Stop the Process Center Deployment Environment](#)

**Administration consoles & tools**

Manage applications, buses, servers and resources within the administrative domain. Monitor and control managed nodes and their resources.  
[WebSphere Application Server Administrative Console](#)

Administer the process servers in your environment, including the users and installed snapshots for each server. Manage queues and caches, and view and manage process instances for process applications.  
[Process Admin Console](#)

Work with Performance Data Warehouse queues, manage data transfer errors, and monitor overall performance.  
[Business Performance Admin Console](#)

**Process application consoles & tools**

Create, manage, share, and test high-level containers such as process applications and toolkits.  
[Process Center Console](#)

Test and administer the business user interface for completing tasks.  
[Process Portal](#)

**Documentation**

Learn about the capabilities and features of IBM Business Process Manager.  
[Knowledge Center](#)

IBM.

**Note****For IBM Business Process Manager on Cloud users**

Click the Launch in the Process Portal tile of your IBM Business Process Manager on Cloud home screen to access the links to the tools shown in the quick start menu.

## 1.2. Import a process application by using the Web Process Designer

- \_\_\_ 1. Open the Process Center Console.
  - \_\_\_ a. Click the **Process Center Console** link to open it in the default browser.

**Process application consoles & tools**

Create, manage, share, and test high-level containers such as process applications and toolkits.

[Process Center Console](#)

Test and administer the business user interface for completing tasks.

[Process Portal](#)

- \_\_\_ b. Use the credentials `author1` for the user name and `author01` for the password and click **Log In**.

**IBM** Process Center

User name

author1

Password

.....

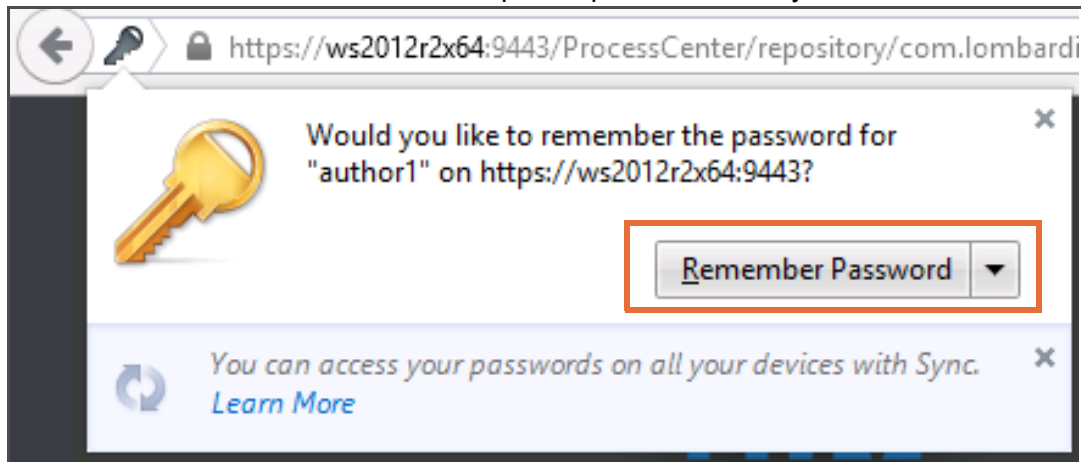
**Log In**

Licensed Materials - Property of IBM. © Copyright IBM Corporation 2000, 2016.

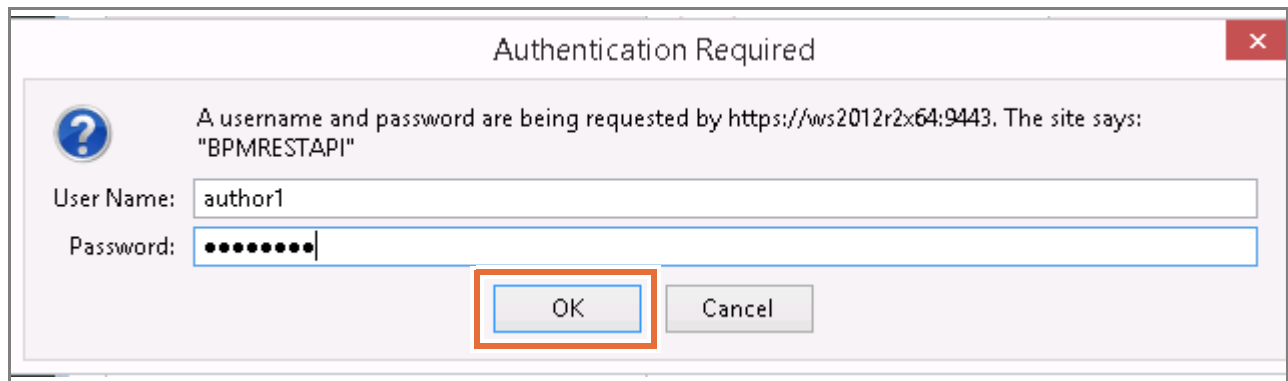


**Hint**

Click the **Remember Password** button to speed up the next time you enter the site.

**Important**

At certain times when working inside the web-based Process Designer, the session times out due to inactivity. If at any time during this course you see an Authentication Required prompt, specify the same credentials and click **OK**.



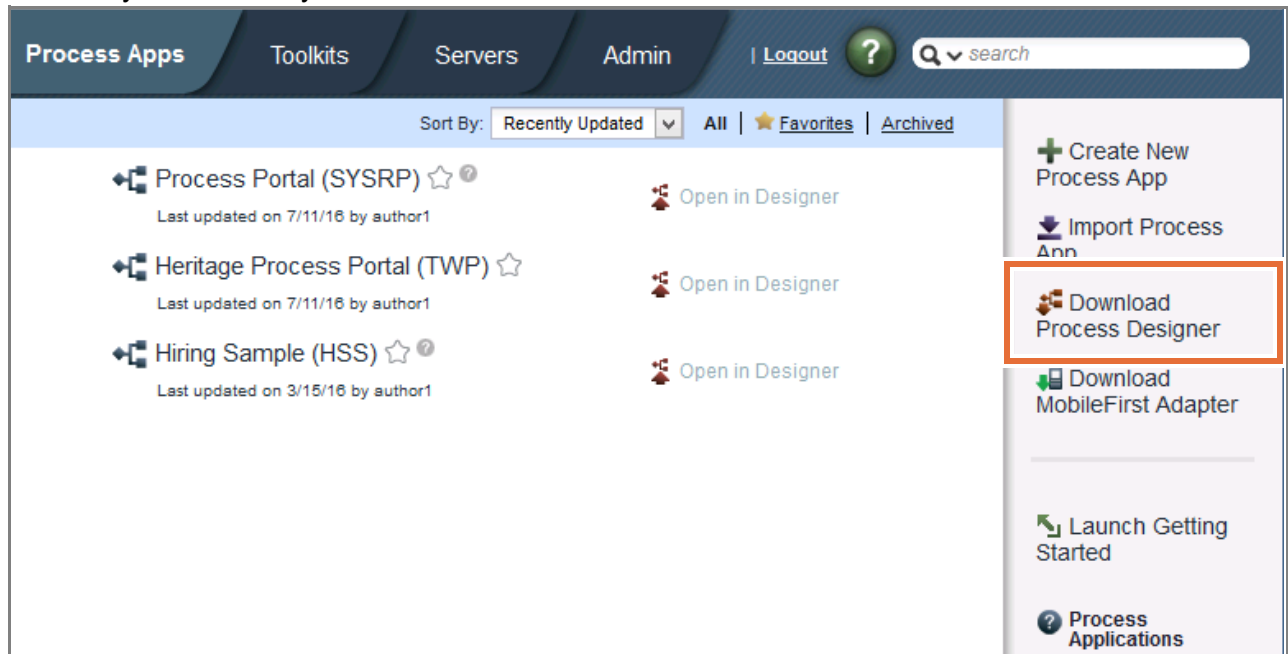
- \_\_\_ c. You see the Getting Started window the first time you open the Process Center console. Close this window.





## Information

The Process Center console contains a web-based utility to manage, create, and edit your Process Applications and the Process Server. Normally you download the IBM Process Designer client application from the Process Center Console. The IBM Process Designer client application is already installed on your virtual machine.

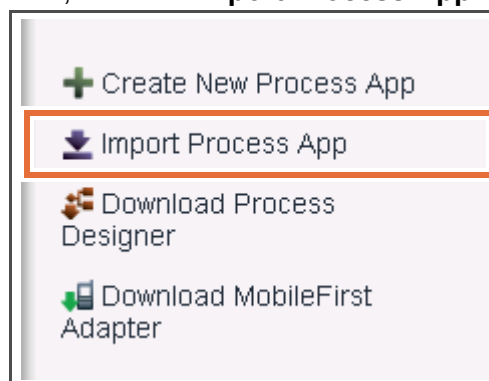


### For IBM Business Process Manager on Cloud users

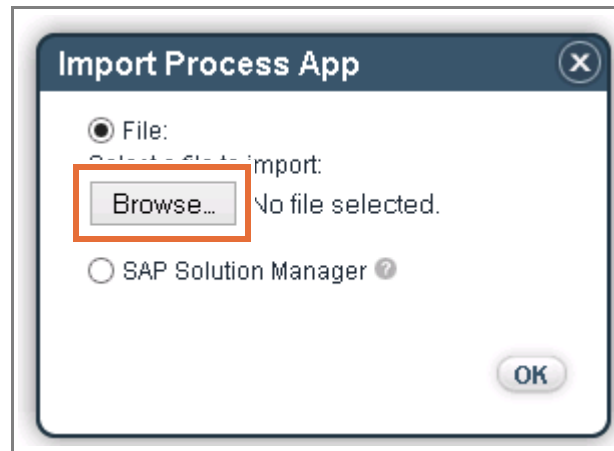
To obtain and install the desktop version of Process Designer, you must log in to your IBM Business Process Manager on Cloud instance. At the bottom of the Process Center tile, you find a link to download Process Designer. You must install the Process Designer client application on your Windows based client. The version that you download from IBM Business Process Manager on Cloud on cloud is already configured to communicate with your Cloud Process Center.

\_\_ 2. Import the HR Recruitment Process process application.

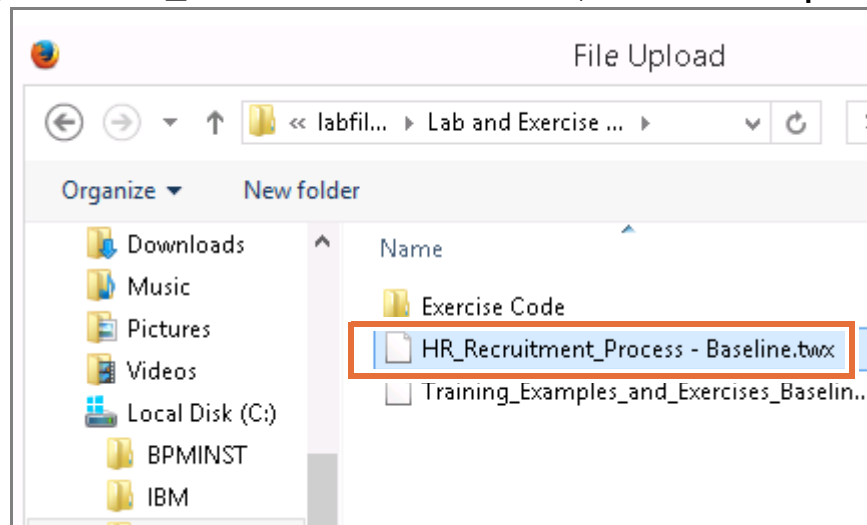
\_\_ a. In the Process Console, click the **Import Process App** link.



- \_\_\_ b. Click **Browse**.



- \_\_\_ c. Browse to the C:\labfiles\Lab and Exercise Assets folder, click the HR\_Recruitment\_Process - Baseline.twx file, and then click **Open** to add the file.



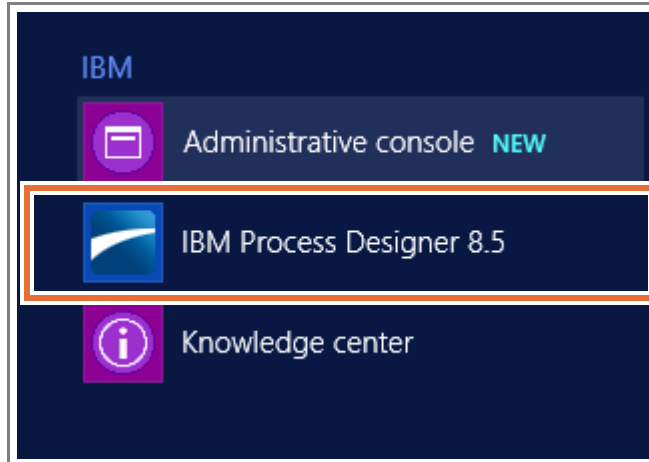
### Note

#### For IBM Business Process Manager on Cloud users

All the files that are referenced in the C:\labfiles folder can be downloaded from the Cloud Education website. Extract these files to a folder on your computer, and then substitute the path that is shown here with the path on your computer.

- \_\_\_ d. Click **OK** to begin the import of the file.
- \_\_\_ e. Click **Import** to import the file and all associated assets.
- \_\_\_ f. Wait until the import is completed and the last updated date is shown below the name of the process application. The imported process application is now part of the **Process Apps** repository.
- \_\_\_ 3. Import the Training Examples and Exercises process application.
- \_\_\_ a. Click **Import Process App**.

- \_\_\_ b. Click the **Browse** icon, select the C:\labfiles\Lab and Exercise Assets\Training\_Examples\_and\_Exercises\_Baseline.twx file, and click **Open**.
- \_\_\_ c. Click **OK** to begin the import of the file.
- \_\_\_ d. Click **Import** in the **Import Process App** dialog box to import the process application.  
The baseline process application is now ready to be modified with tracking points, tracking groups, and timing intervals.
- \_\_\_ 4. Start the IBM Process Designer client application.
  - \_\_\_ a. Click **Start > Down arrow > IBM > IBM Process Designer 8.5** to start the application.



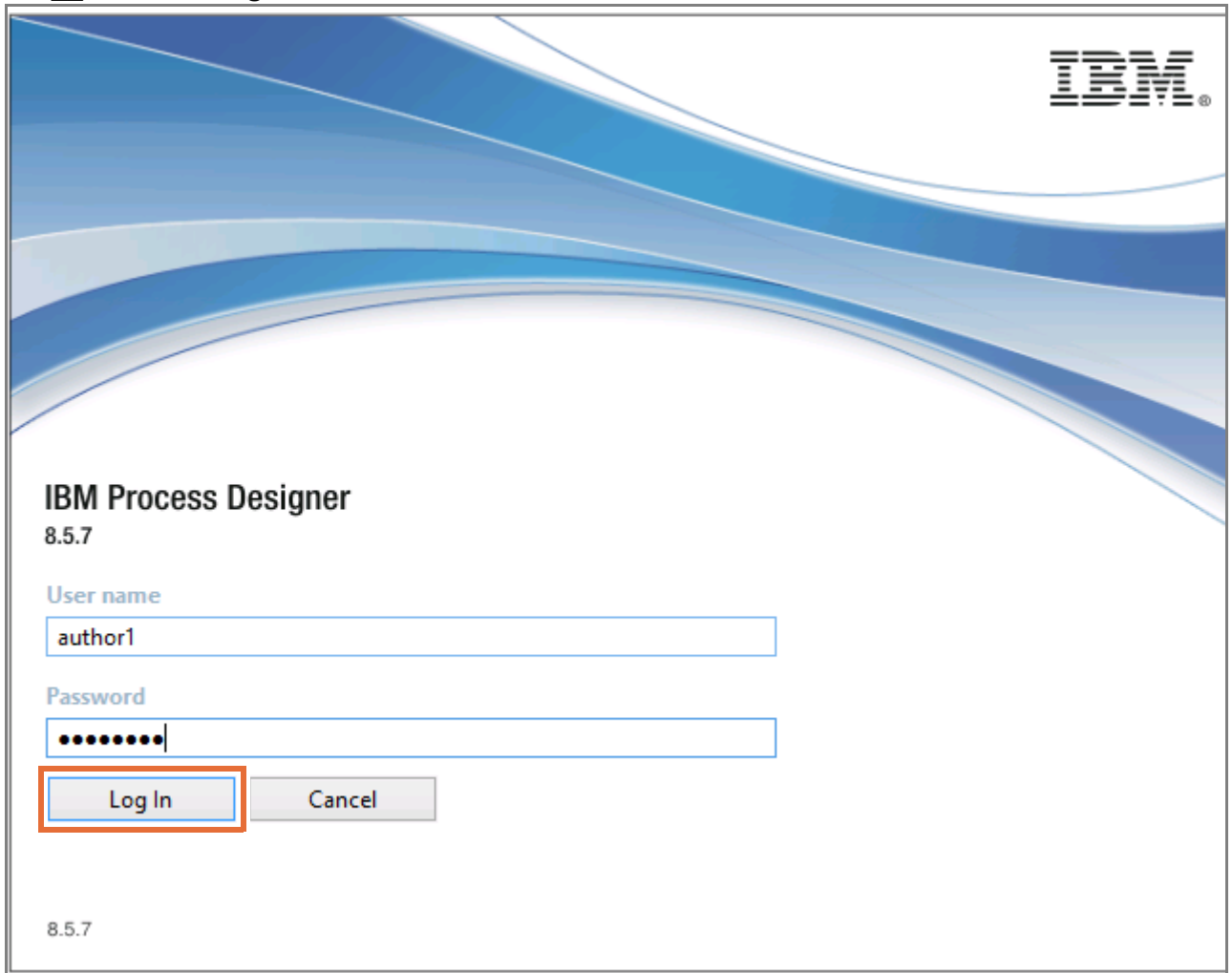
#### Note

#### For IBM Business Process Manager on Cloud users

Start the IBM Process Designer you installed to your computer. Depending on which version of Windows you are using, the path in the start menu to open the IBM Process Designer might be different.

- \_\_\_ b. In the login screen, enter `author1` for the user name and `author01` for the password.

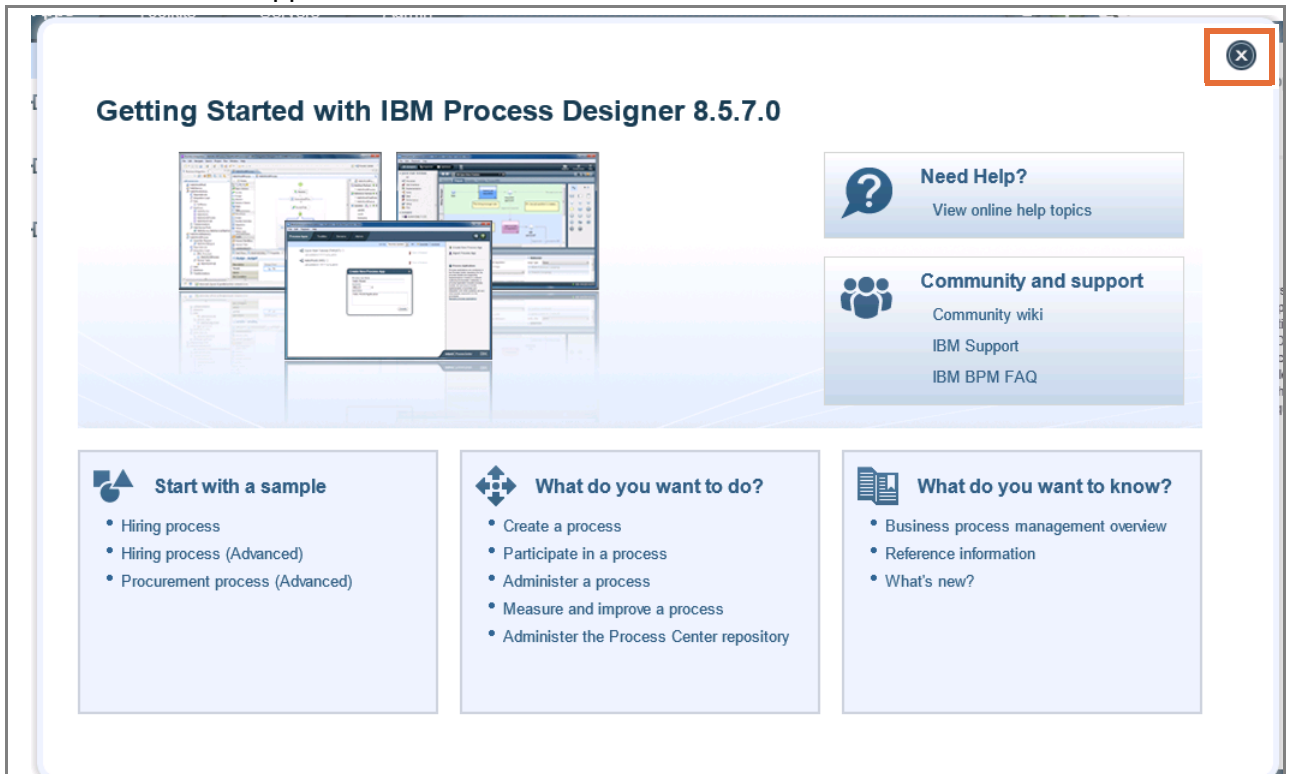
\_\_\_ c. Click **Log In**.



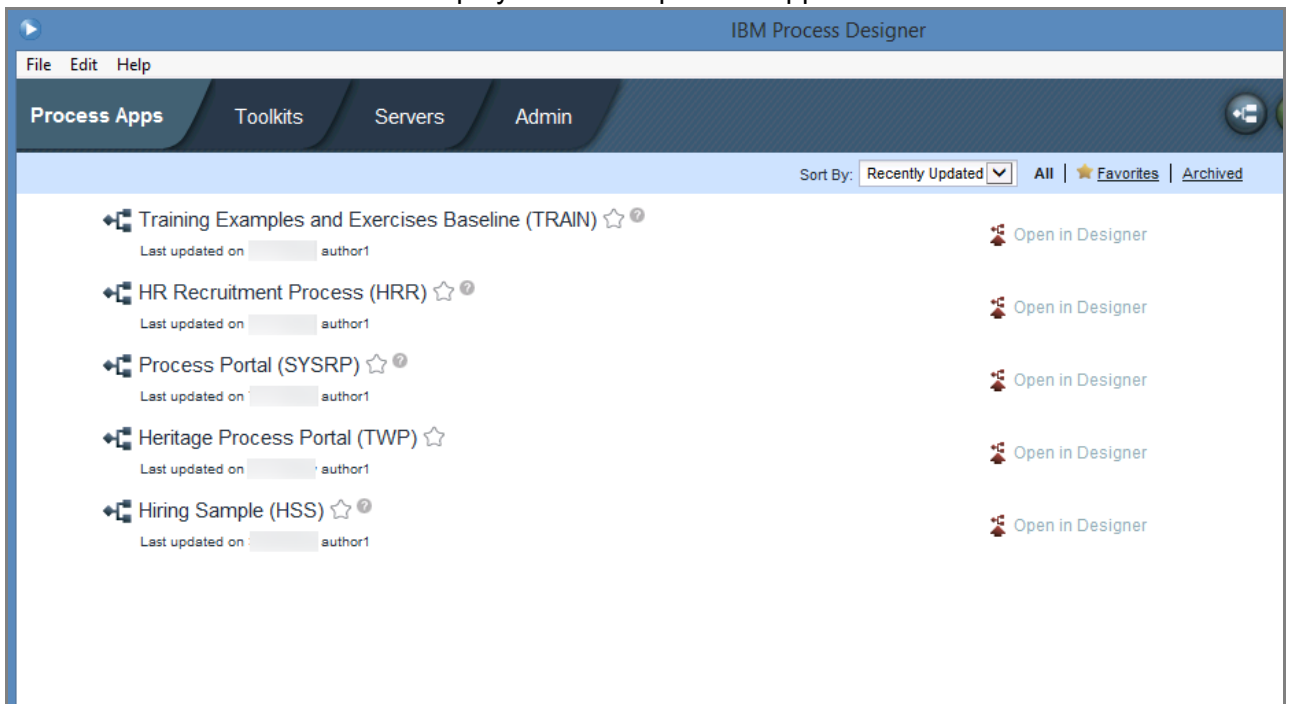
The image shows the IBM Process Designer 8.5.7 login dialog box. It features a blue and white background with the IBM logo in the top right corner. The title "IBM Process Designer" and version "8.5.7" are displayed on the left. Below the title, there are two input fields: "User name" with the text "author1" and "Password" with masked characters. At the bottom, there are two buttons: "Log In" (highlighted with a red border) and "Cancel". The version "8.5.7" is also displayed in the bottom left corner.

\_\_\_ d. When you log in for the first time, you are presented with security alerts. Click **Yes** to proceed.

- \_\_\_ e. The first time that you enter the Process Designer client application, the Getting Started window appears. Close this window.



The Process Center displays the list of process applications that are available.



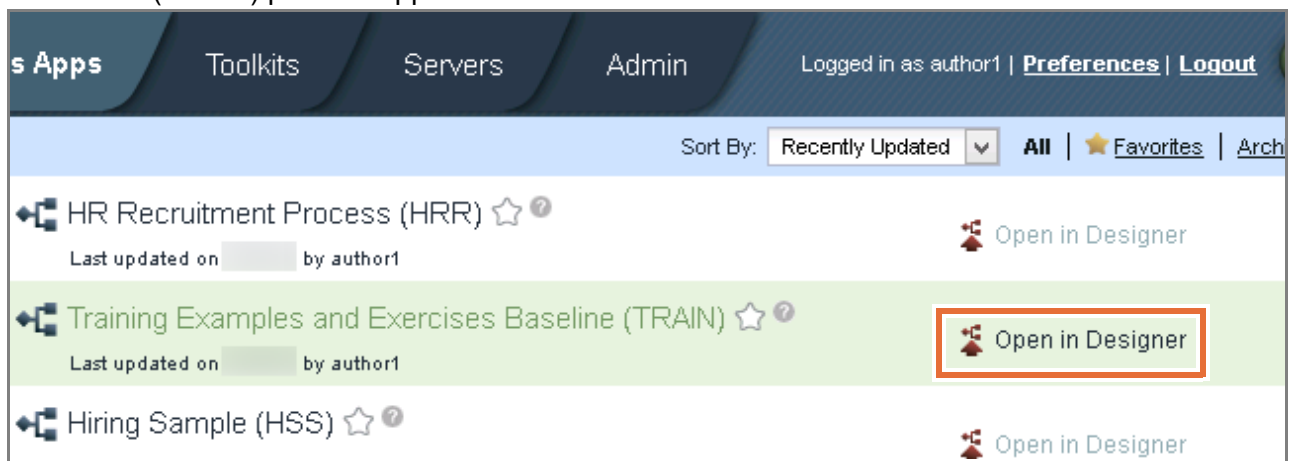
- \_\_\_ f. Minimize the Process Designer client application window. You use this tool later in this exercise.

## Section 2. Examine a failed instance

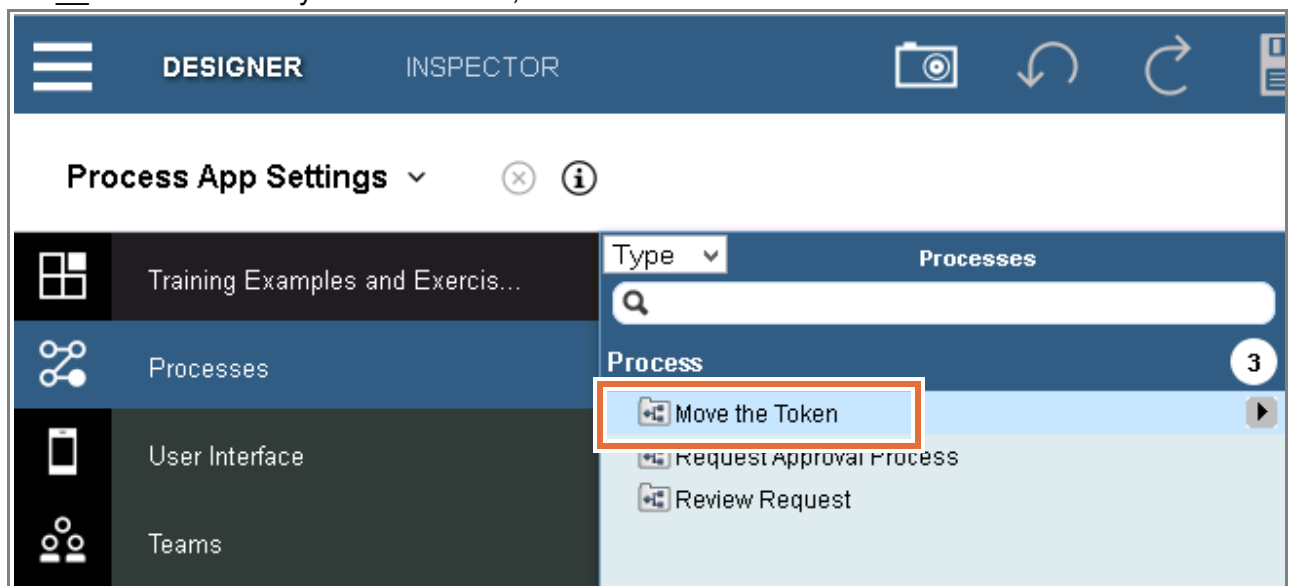
In this section, you work with a process in which some of the instances are succeeding while some are failing. The business unit wants to target some of those failed instances to move the token on the process. One of the failed instances is critical; you need this instance to pass beyond the failed step. Since it is a business emergency, you need to move the token and get the process beyond that failed activity. You solve this problem by using an external implementation of the REST API.

### 2.1. Explore the process

- \_\_\_ 1. Explore the **Move the Token** user interface.
  - \_\_\_ a. Make sure that you are now working on the Process Center Console page that you opened in the Firefox browser.
  - \_\_\_ b. Click the **Open in Designer** link next to the Training Examples and Exercises Baseline (TRAIN) process application.

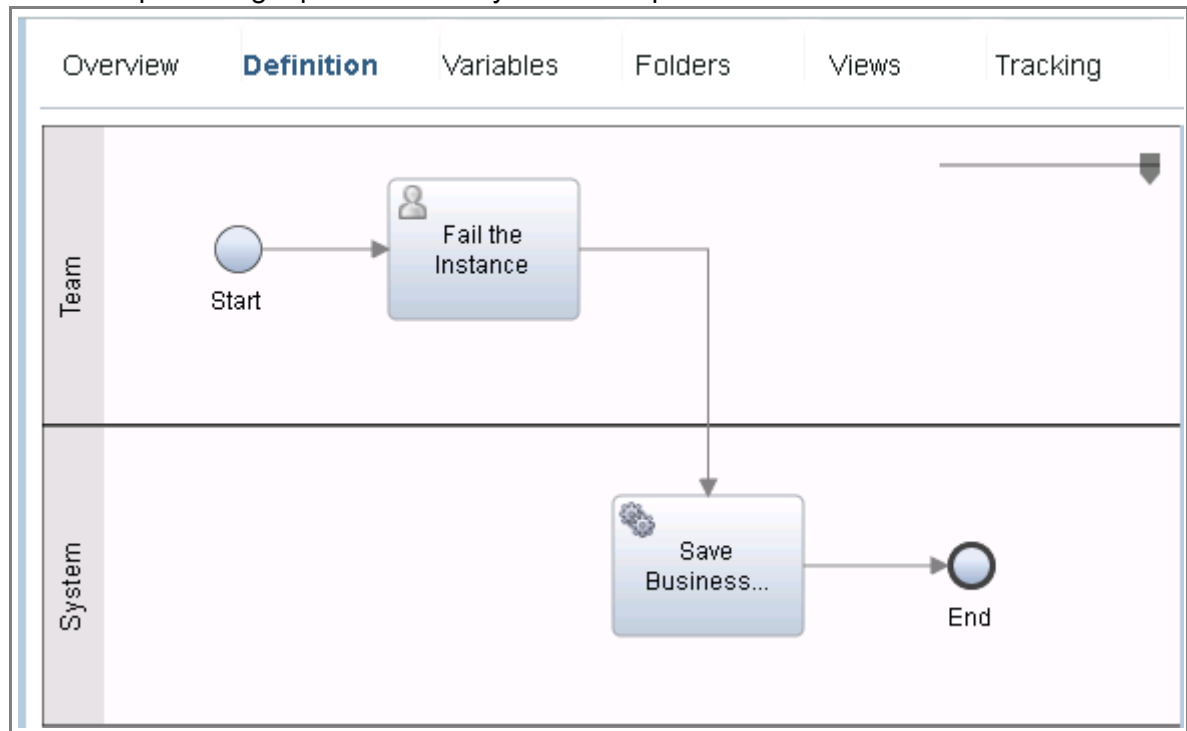


- \_\_\_ c. In the library on the left side, click **Processes > Move the Token**.

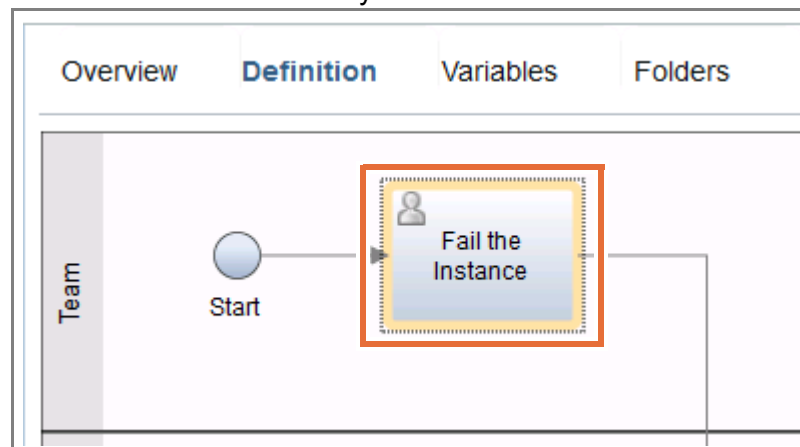




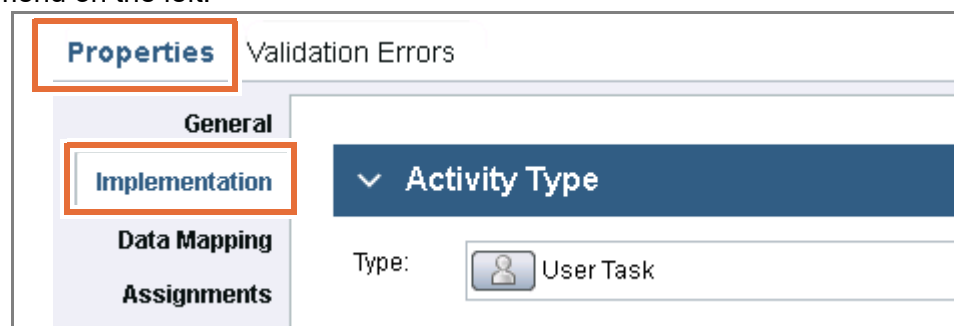
- \_\_\_ d. The **Move the Token** process is displayed on the **Definition** tab. Examine the process. The **Move the Token** process is a simple process that is having problems at run time. The **Fail the Instance** activity is failing for several process instances, and you are required to get past this activity for some specific failed instances.



- \_\_\_ e. Click the **Fail The Instance** activity on the canvas.



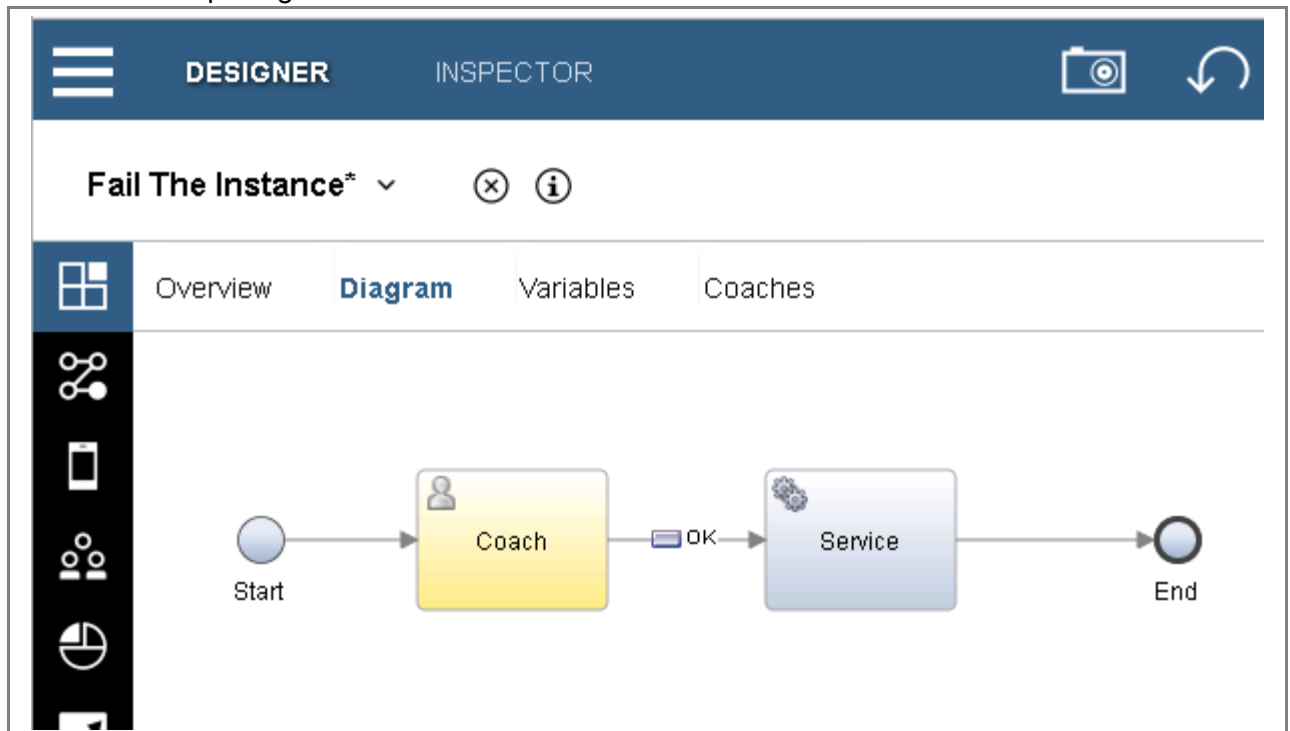
- \_\_\_ f. Click the **Properties** tab at the bottom of the window. Then, click the **Implementation** menu on the left.



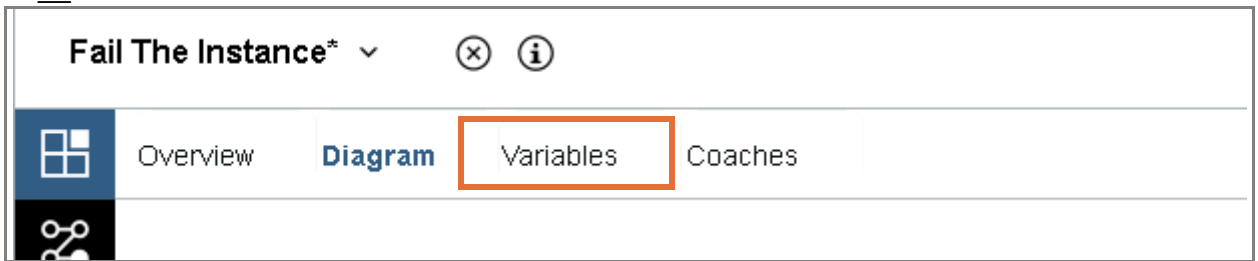
- \_\_\_ g. In the Implementation section, click the **Fail The Instance** link to open the client-side human service.

The screenshot shows the 'Implementation' section of a software tool. On the left, a sidebar lists 'Implementation', 'Data Mapping', 'Assignments', 'Pre & Post', 'Tracking', and 'Conditions'. The 'Implementation' section is active, showing 'Activity Type' as 'User Task'. Below this, the 'Implementation' section lists 'Fail The Instance' as the selected implementation, which is highlighted with a red box. There are also 'Select...' and 'New...' buttons.

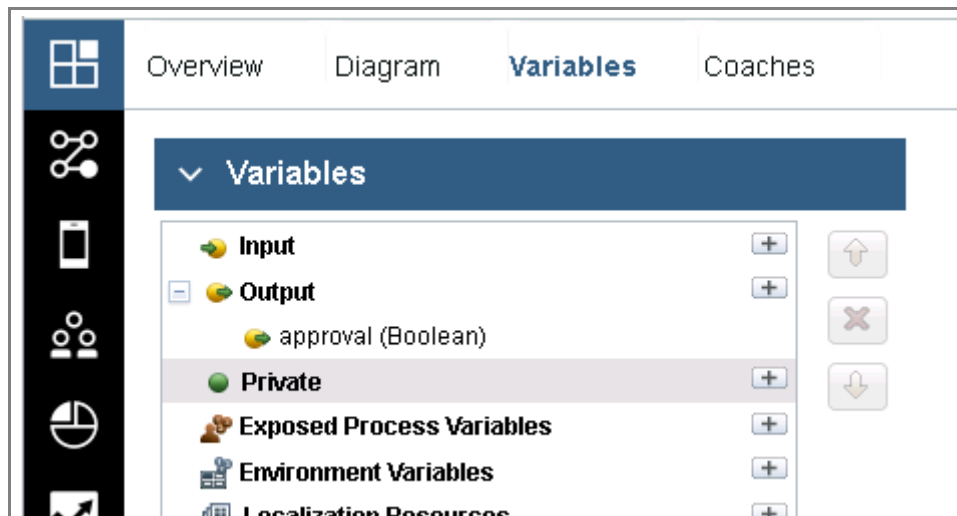
- \_\_\_ h. The Fail The Instance service diagram is shown. This service simulates a connectivity problem for the process that doesn't allow the task to complete. The service is a simple approval activity for the process, and contains only two steps. The first is the coach to ask the user for an approval, and the second step is a general system service to persist the decision immediately to a database. The simulated problem is either the network is down or the database is unavailable for this step to complete its operation. When you run the service, the service throws an error to simulate this connectivity problem. Later in this exercise, you resolve the problem by completing this step in the process by completing the task for the user.



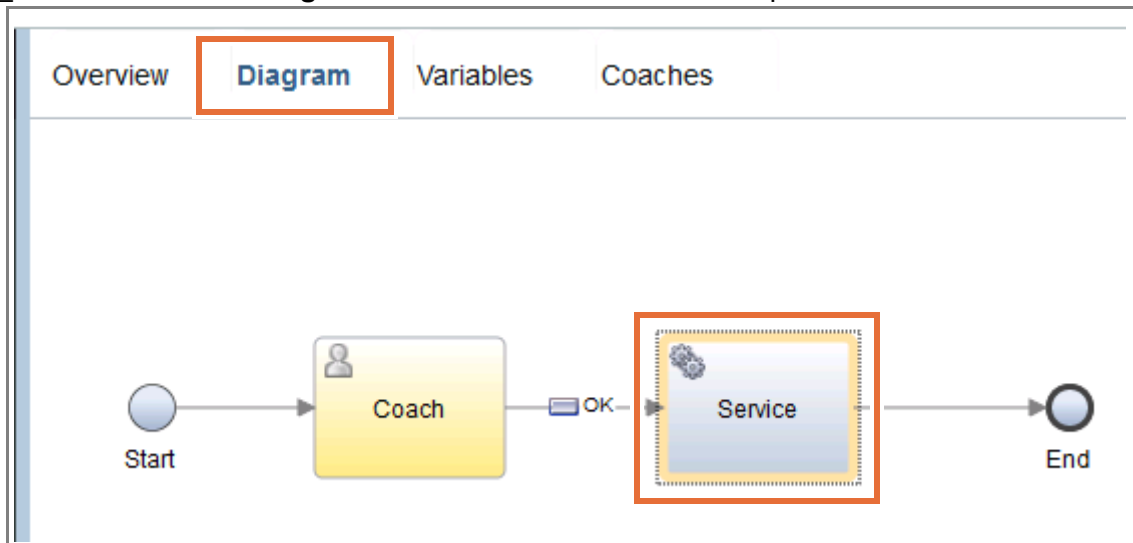
- \_\_\_ i. Click the **Variables** tab.



- \_\_\_ j. The private variable `approval` of type Boolean is defined. Because the general system service fails, when the process runs, the result of the validation Boolean variable never is output from the service back to the process level. Therefore, the task cannot complete and the process does not receive the approval variable. By using a REST call to complete the task, you can close the task and send the approval variable from an administrative dashboard service.

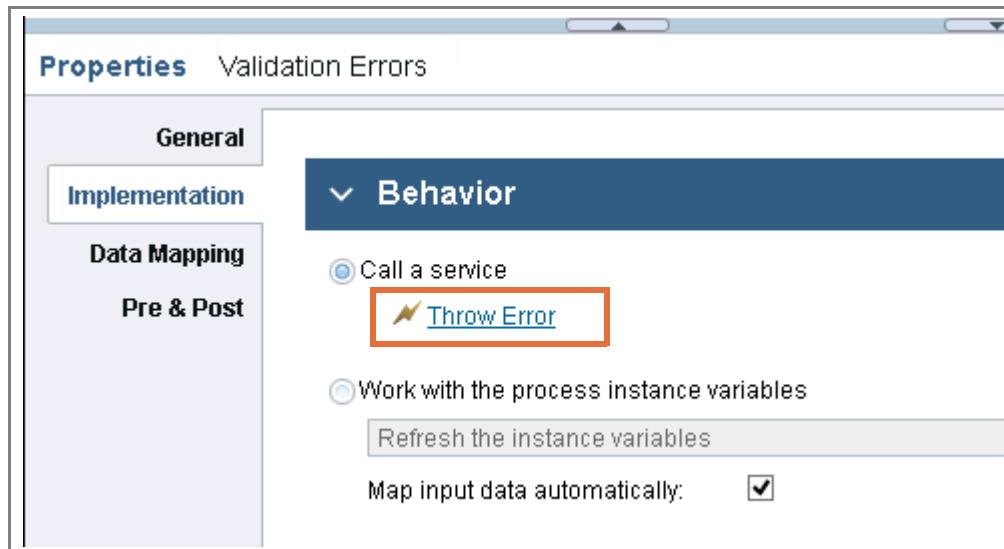


- \_\_\_ k. Return to the **Diagram** tab and select the **Service** step on the canvas.

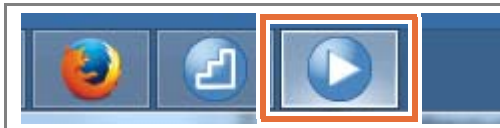


- \_\_\_ l. Click the **Properties > Implementation** menu to view the implementation details for the service.

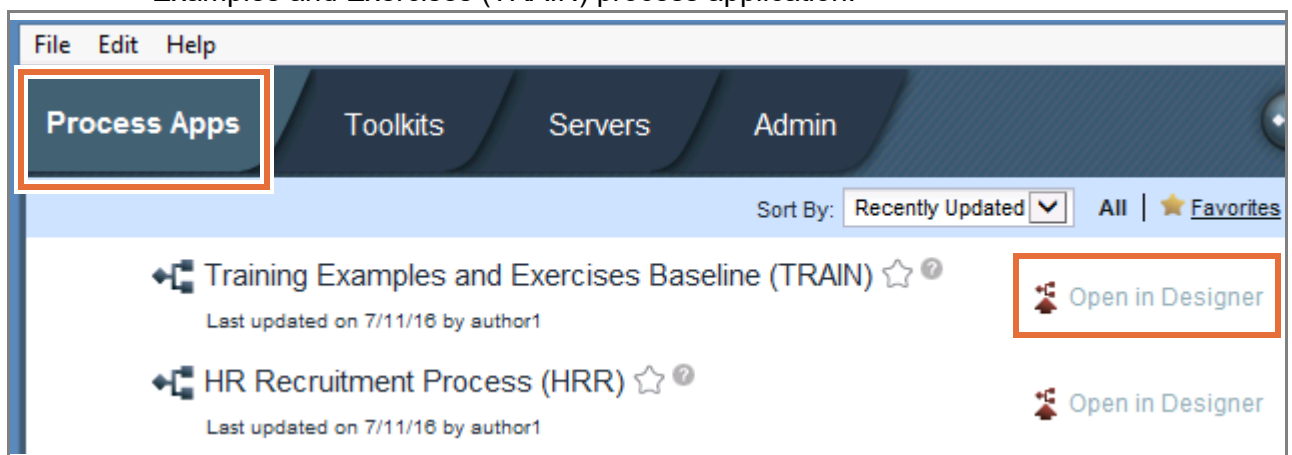
- \_\_\_ m. The **Throw Error** general system service implements this step of the service. In a real-world example. This step should save the approval to a database, but the Throw Error service is designed to throw an error every time. By using a REST API call, you can recover instance failures in your organization's environment similar to the one simulated here.



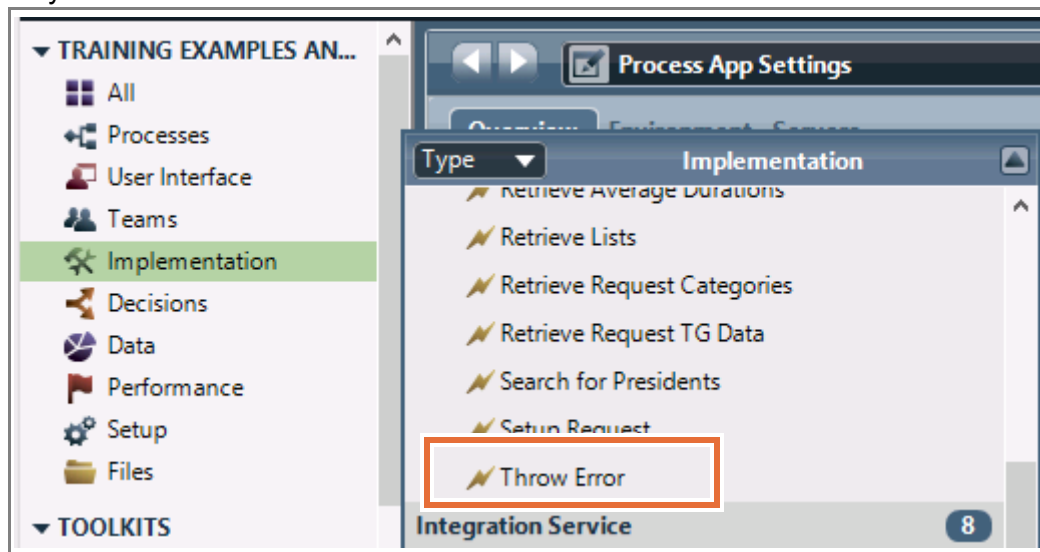
- \_\_\_ n. In the windows taskbar area at the bottom, switch to the Process Designer client application.



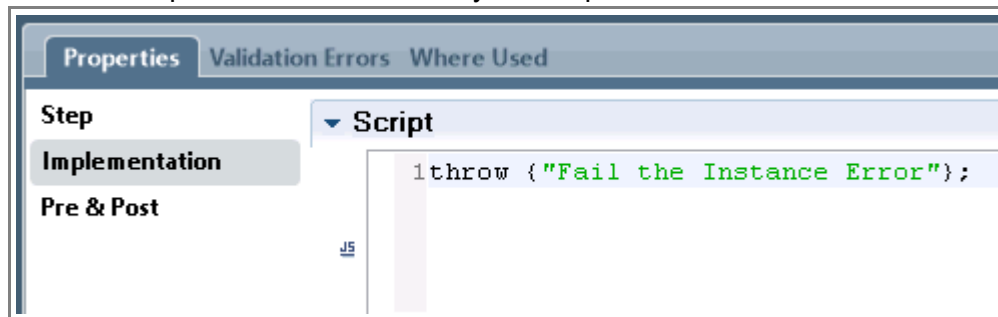
- \_\_\_ o. On the Process Apps tab, click the **Open in Designer** link to the left of the Training Examples and Exercises (TRAIN) process application.



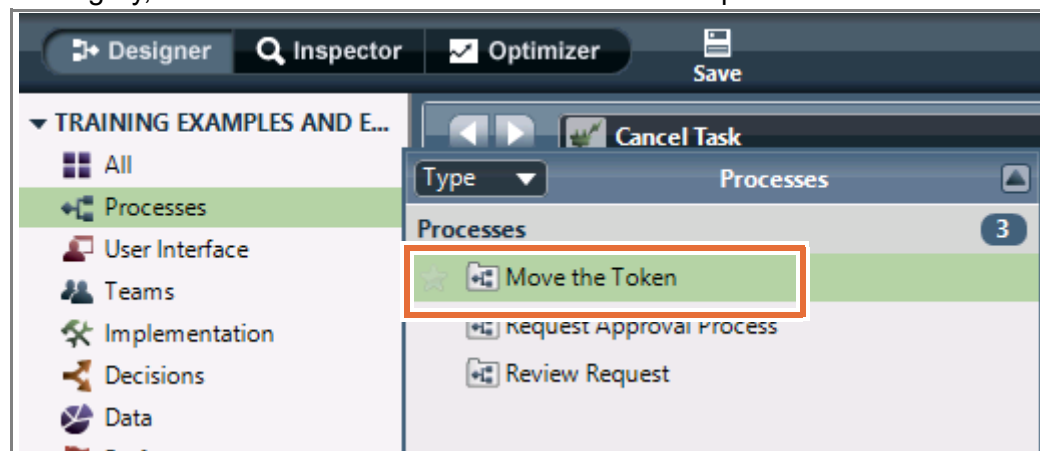
- \_\_\_ p. In the library on the left, double-click the **Implementation > Throw Error** general system service.



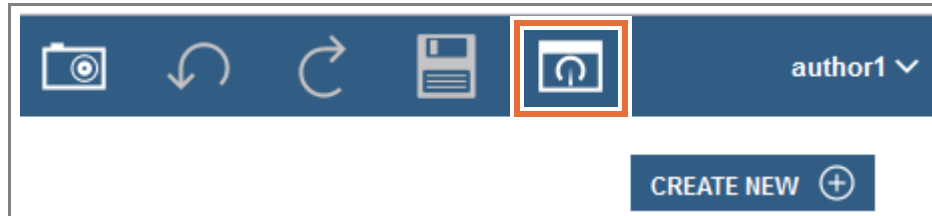
- \_\_\_ q. Click the **Fail The instance** step on the diagram.
- \_\_\_ r. Open the **Properties > Implementation** menu. This server script throws an error and simulates a problem with this activity on the process.



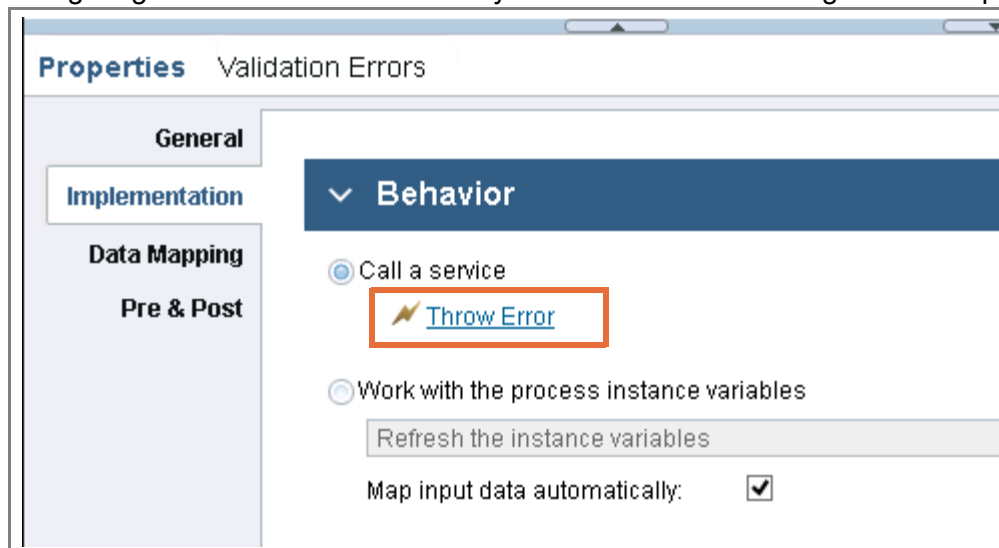
- \_\_\_ s. To return to the web Process Designer, in the library on the left, click the **Processes** category, and then double-click the **Move the Token** process.



- \_\_\_ t. The Move the Token process opens inside the web Process Designer on a new tab, and now two tabs are open in your browser. The first is the original tab that you opened the Process Center console from the Quick Start menu. The new tab is only a web Process Designer window. You can tell the difference between the two because the icon to view the Process Center is shown on the Process Center Console, and is missing on the web Process Designer tab.

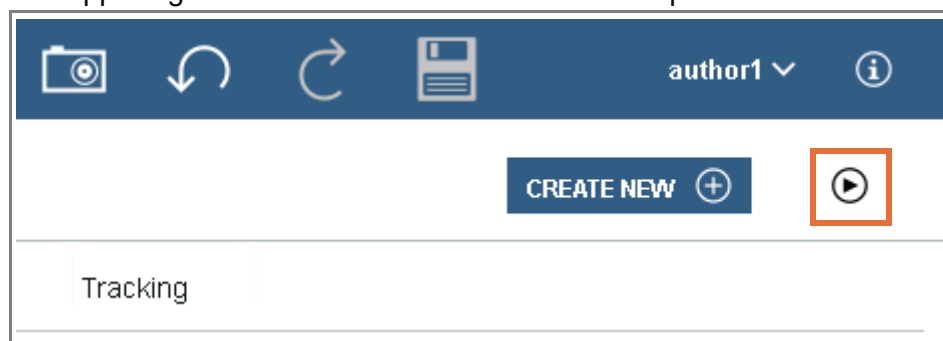


- \_\_\_ u. You can open artifacts directly in the Process Designer client application by using the web Process Designer and clicking the link to the service. You are required to flip between the two tools frequently, so work on the web Process Designer tab to avoid navigating to all the services manually inside the Process Designer client application.

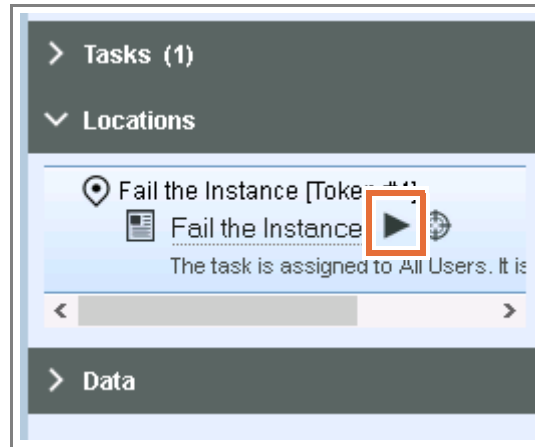


## 2.2. Run the instance

- \_\_\_ 1. Create an instance of the process.
- \_\_\_ a. In the Move the Token process that you opened in the web Process Designer, click **run** in the upper-right corner to create an instance of the process.



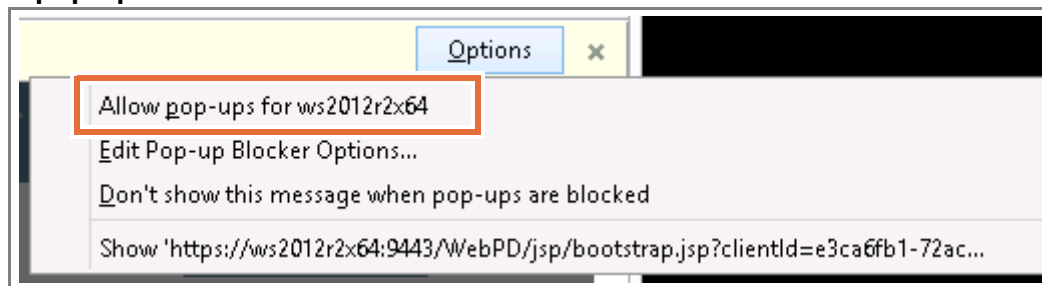
- \_\_\_ b. The web Process Designer switches to the Inspector view and the instance details are shown on the right. A token is shown for the task that is created for the Fail the Instance activity on the process definition. In the instance details section on the right, under the Locations section, click **Start** next to the Fail the Instance task.



- \_\_\_ c. The Inspector attempts to open the task inside a new window, but Firefox prevents the site from opening the window. An error message is shown. Click the **OK** button to close the error.

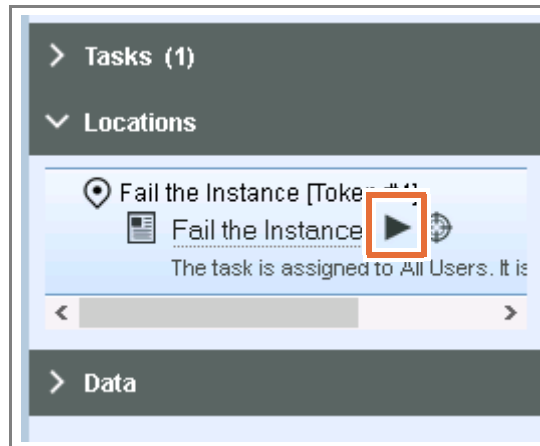


- \_\_\_ d. To correct the problem, in the yellow bar at the top of the browser, click **Options > Allow pop-ups for ws2012r2x64**.

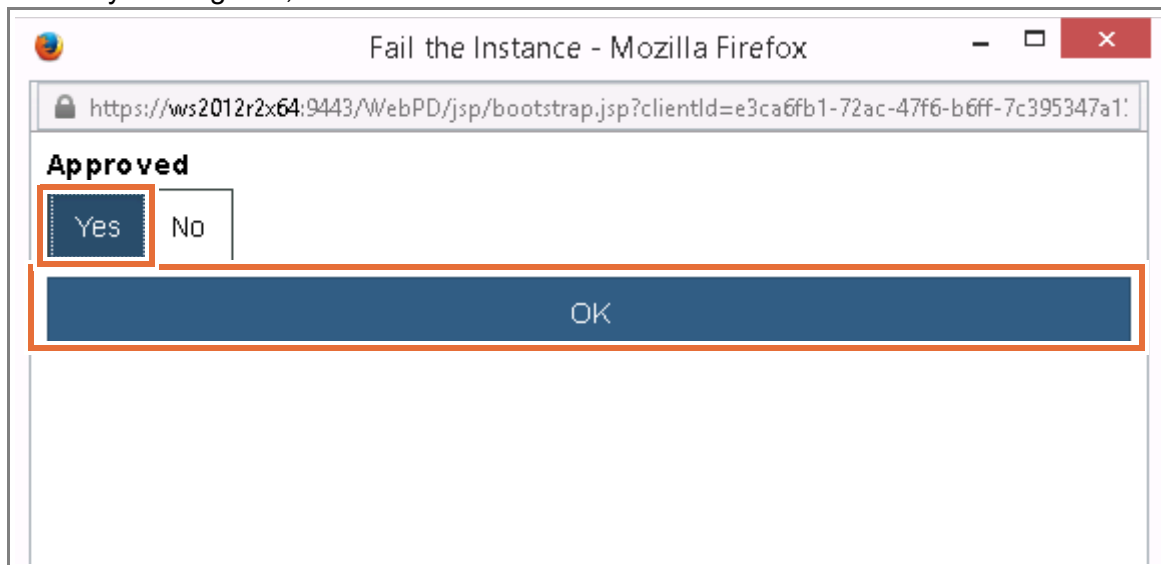


- \_\_\_ e. The window opens, but shows another instance of the Process Portal. Close this window.

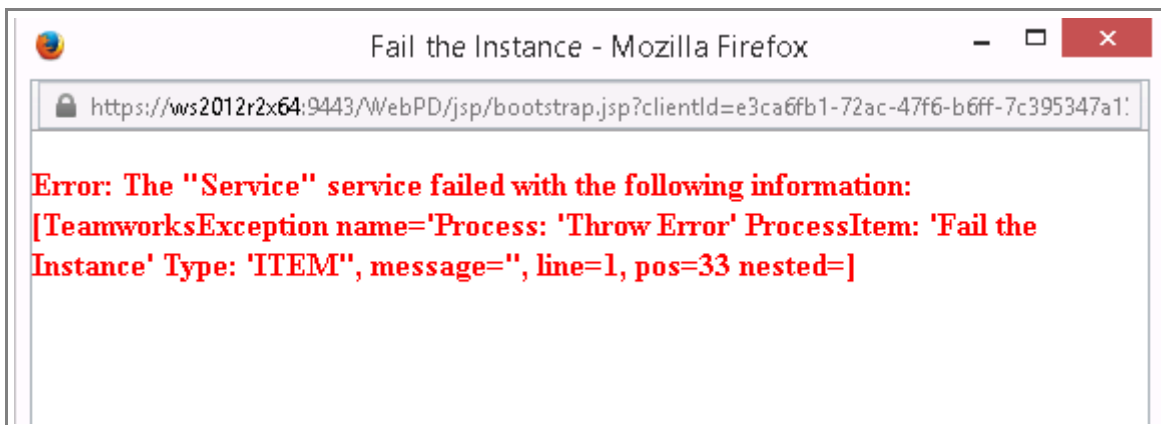
- \_\_\_ f. Return to the original web Process Designer window. Click **Start** next to the Fail the Instance task.



- \_\_\_ g. This time the window opens and shows the simple approval coach. Approve the request by clicking **Yes**, and then click **OK**.

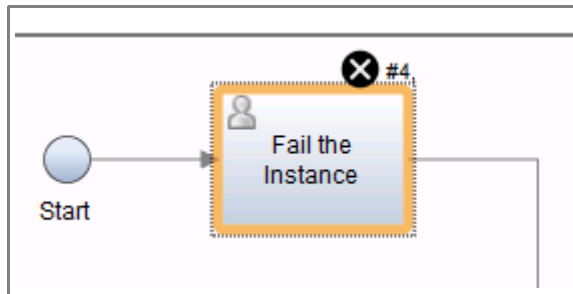


- \_\_\_ h. An error message is shown in the browser because of the Throw Error general system service. Close the browser window.

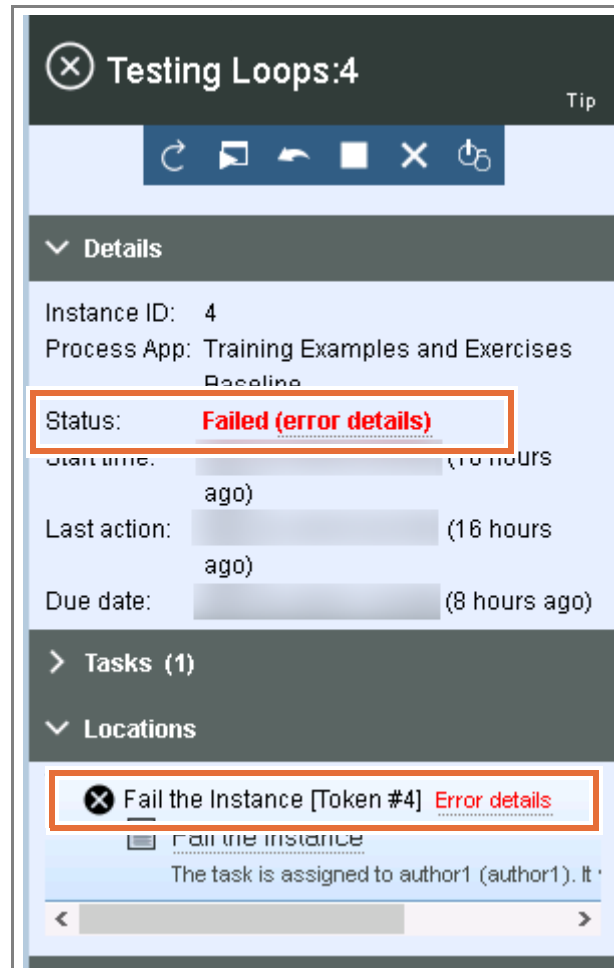




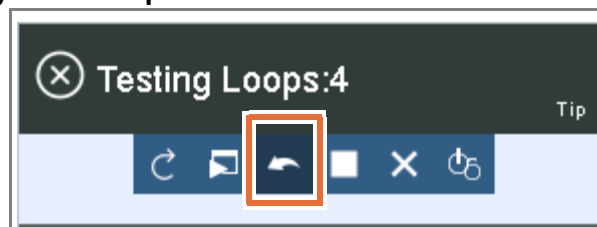
- \_\_\_ i. When the instance details page refreshes, the token on the activity is now an X to show that the task failed.



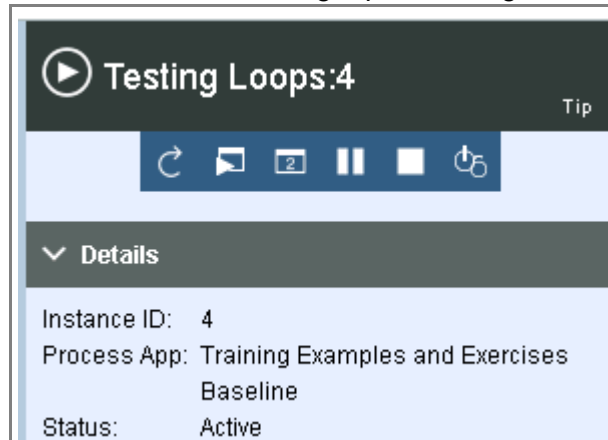
- \_\_\_ j. In the Details section on the Inspector view for this instance, the status changed to Failed. In the Locations section, the Fail the Instance task is now in a failed state.



- \_\_\_ 2. Try to restart the instance and see whether you can get past this error.
- \_\_\_ a. Click the **Retry failed steps** icon.

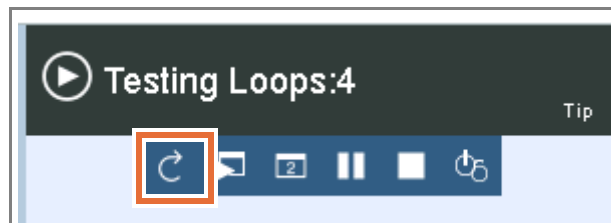


- \_\_\_ b. Notice that the instance status in the right pane changes from `Failed` to `Active`.

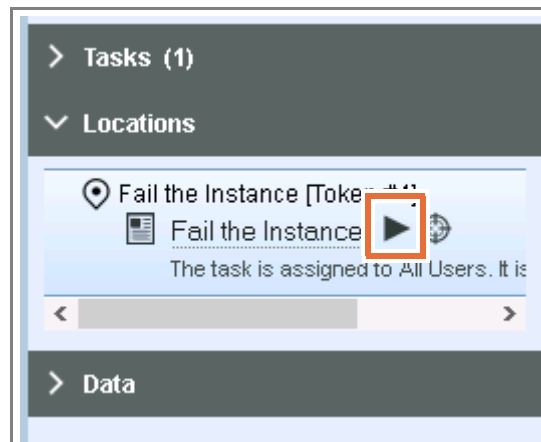


### Hint

If the Fail the Instance task remains in the failed state or the system does not show the Start icon for the task, click the **Refresh** button to refresh the task.



Click the **Start** icon next to the Fail the Instance task.



Verify that the error is thrown again.



### Information

This error is the same error that you saw earlier. Notice that this time when you ran the task, you did not see the coach for approval. The service went straight to the error. The process remembers the step where you were when the service failed. Suppose that a database call was made in that service, and you resumed the instance and ran it again. It would go directly to the step after the

coach, attempt to make that database call each time, and eventually fail. The token is at the Fail the Instance server script step inside the service. Each time that you try to restart that failed instance it starts from that failed step, and you do not see the coach. The system tries to resume the service from the point where it failed. It does not restart the entire service.

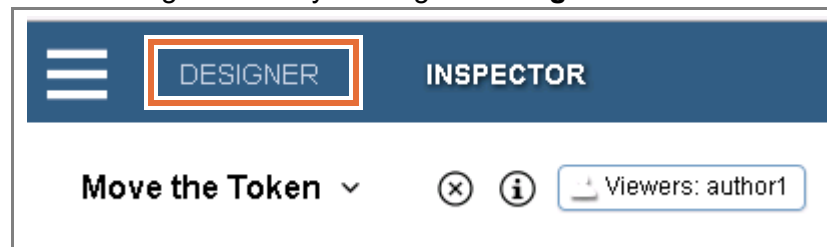
The user is associated with that failed task. If this task is in a failed state for a few days, then it might reflect badly upon the administrator as far as IBM Business Process Manager process performance metrics are concerned. Something that should take 5 minutes is now taking 5 days and is still not complete.

- \_\_\_ c. Close the task browser.

## 2.3. Explore the Move the Token client-side human service implementation

- \_\_\_ 1. Explore the **Move the Token** client-side human service.

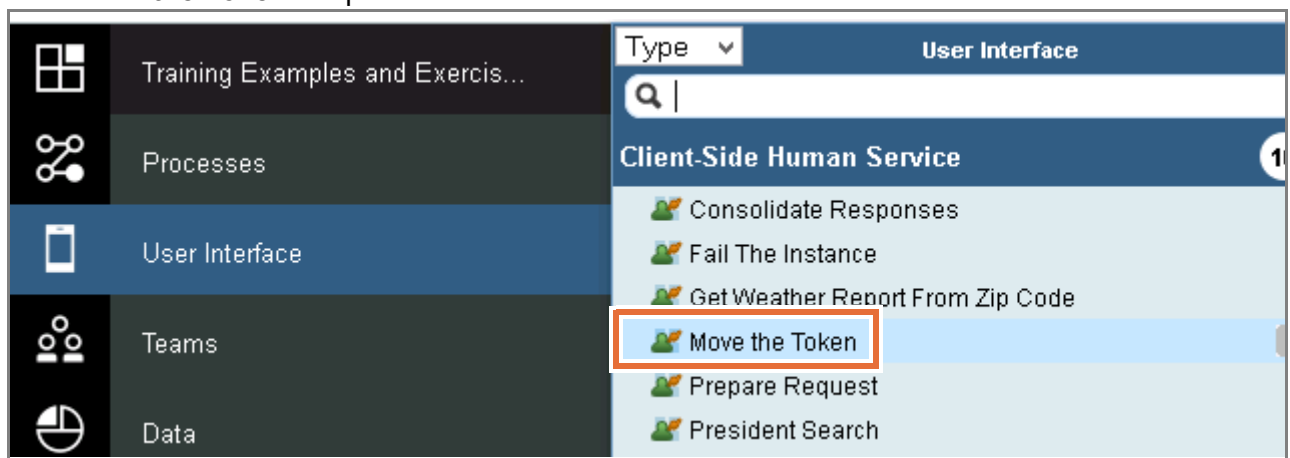
- \_\_\_ a. Return to the designer view by clicking the **Designer** tab.



- \_\_\_ b. In the web Process Designer, you can expand and contract the library on the left by clicking the **Library** button in the top area of the designer. You can also hover your cursor over one of the icons to expand the library section on the left.

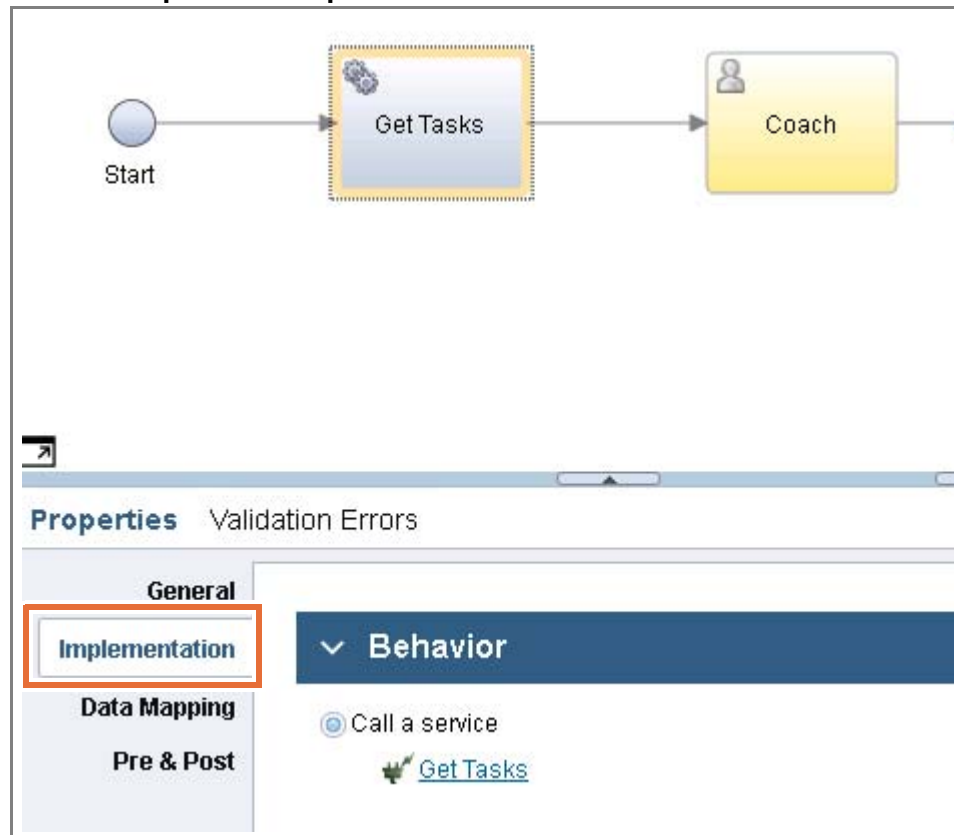


- \_\_\_ c. Expand the library on the left. Click the **User Interface** category and then click **Move the Token** to open the client-side human service.

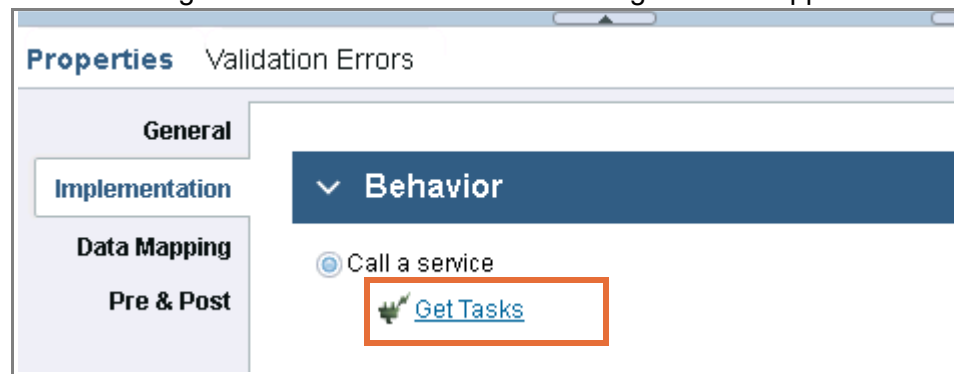


- \_\_\_ d. Click the **Get Tasks** service on the canvas.

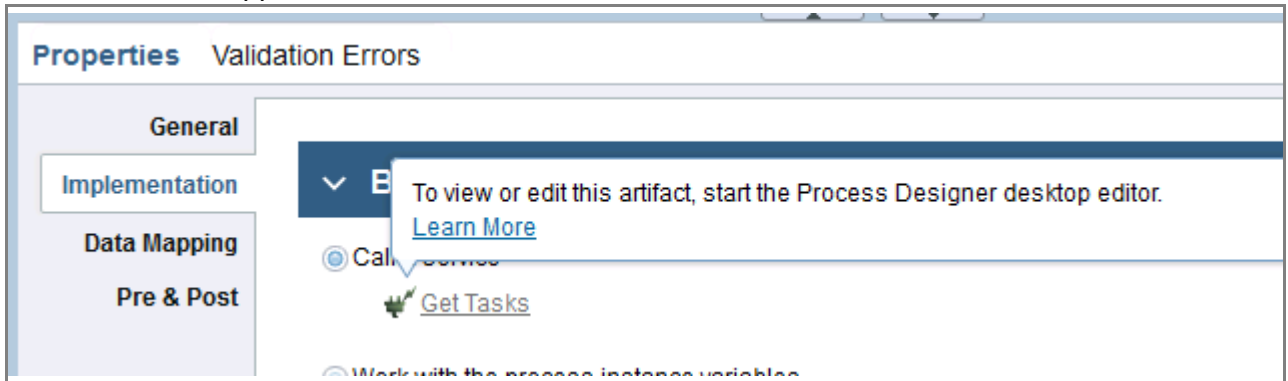
- \_\_\_ e. Click the **Properties > Implementation** menu.



- \_\_\_ f. The Get Tasks service implements this step. Get Tasks retrieves all the tasks in the system. Click the **Get Tasks** link inside the web Process Designer. The link opens the Get Tasks integration service in the Process Designer client application.



- \_\_\_ g. When you click the link to the service, it might show the To view or edit this artifact, start the Process Designer desktop editor message. You are in the Process Center Process Designer tool, and it is not linked to your Process Designer client application.

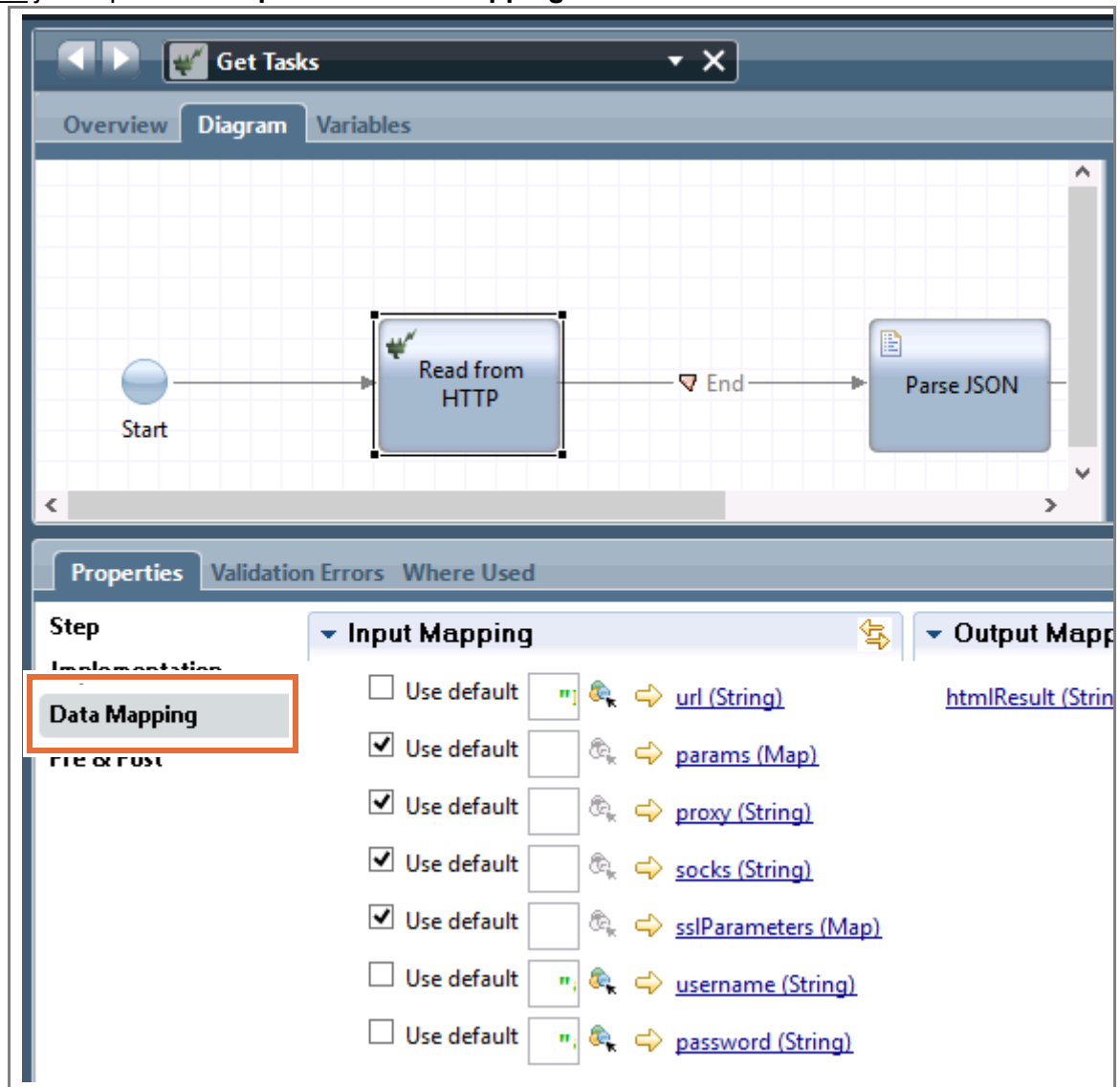


- \_\_\_ h. Switch to the tab that contains the web Process Designer that is linked to the Process Designer client application. When you hover over the link, your cursor changes, and when you click the link, it opens the service inside the Process Designer client application.



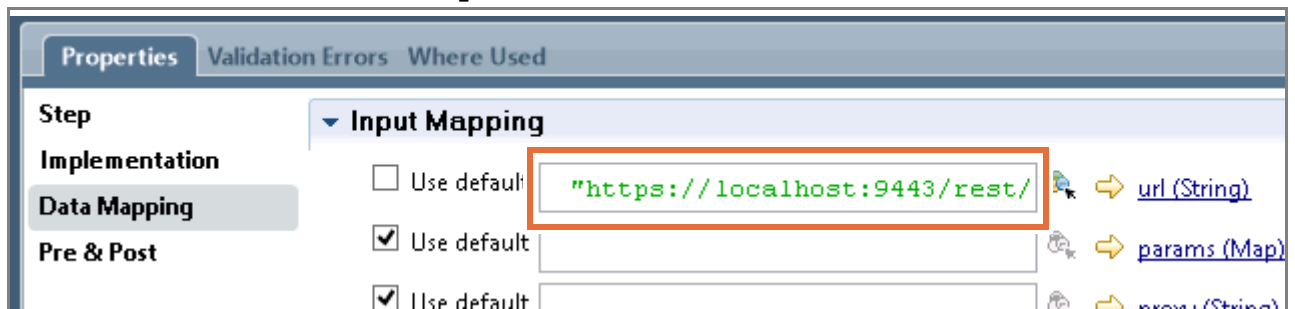
- \_\_\_ i. Select the **Read from HTTP** step on the canvas.

- j. Open the **Properties > Data Mapping** menu.



- k. Examine the Input Mapping section. The URL on the left side is a REST API call to the system.

"https://localhost:9443/rest/bpm/wle/v1/tasks?interaction=all&queryFilter=STATE%3C%3E3+AND+STATE%3C%3E5&filterByCurrentUser=false&calcStats=false&size=20"





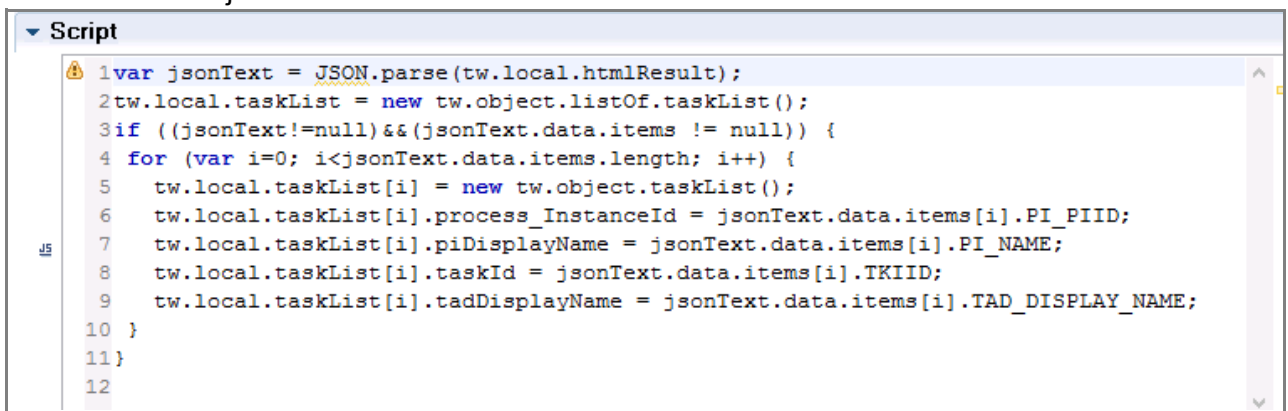
## Important

For testing purposes, you can use string literals in your variables, but a good practice is to move these strings into an environment variable when you are ready for a final Playback.

- \_\_\_ l. Examine the Output Mapping section.



- \_\_\_ m. The REST API call returns a JSON representation of an object that is then turned into a string, `tw.local.htmlResult`. Select the **Parse JSON** step and click the **Properties > Implementation** menu. Examine the script. The return result set is dumped into the task list object.

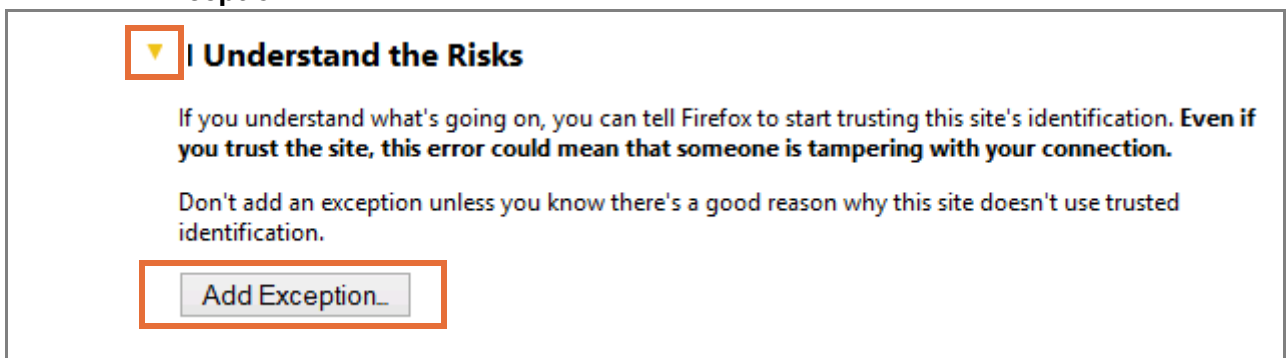


- \_\_\_ 2. Test a REST API call you can use to complete the failed activity and bypass the failing step in the service.

- \_\_\_ a. Open a new browser tab and go to the following web address:

<https://localhost:9443/bpmrest-ui/>

- \_\_\_ b. When prompted, expand the section for **I Understand the Risks** and click **Add Exception**.



- \_\_\_ c. Accept the default settings and click **Confirm Security Exception** to add the security exception to the browser.
- \_\_\_ d. Log in to the API Tester by entering `author1` in the **User name** field and `author01` in the **Password** field. Click the **Log In** button.



**Business Process Manager REST API Tester**

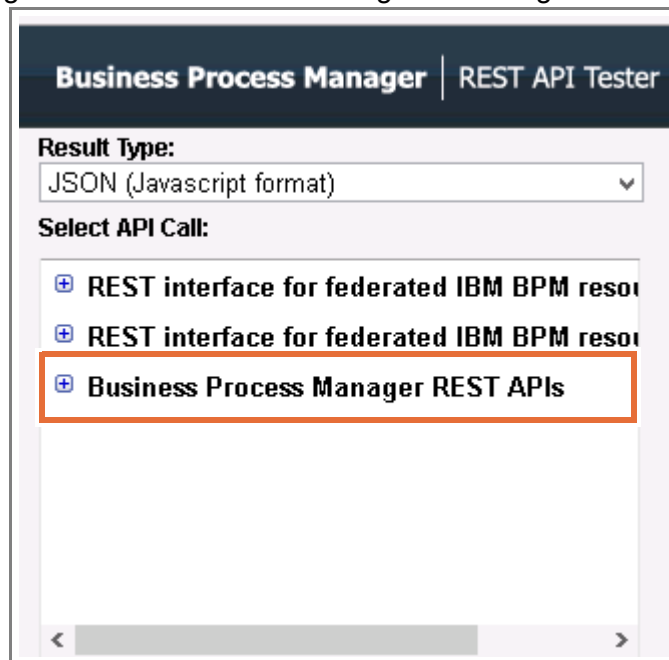
User name

Password

**Log In**

Licensed Materials - Property of IBM. © Copyright IBM Corporation 2000, 2016. All Rights Reserved. US Government Users Restricted Rights- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

- \_\_\_ e. On the left, expand the **Business Process Manager REST APIs** category. Your list of API calls might not use the same ordering as the image shown here.



**Business Process Manager | REST API Tester**

**Result Type:**  
JSON (Javascript format) ▾

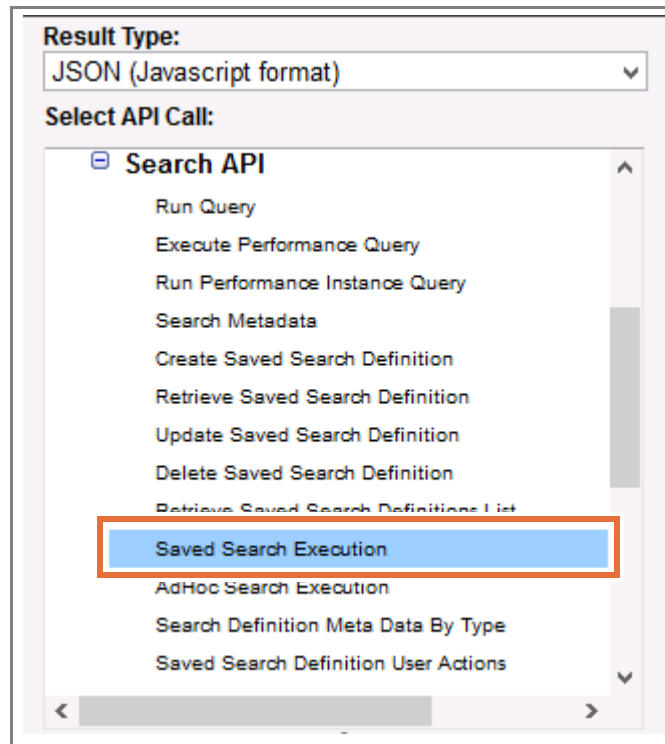
**Select API Call:**

- ⊕ REST interface for federated IBM BPM resou
- ⊕ REST interface for federated IBM BPM resou
- ⊕ **Business Process Manager REST APIs**

<  >



- \_\_\_ f. Expand **Search API** and click **Saved Search Execution**.

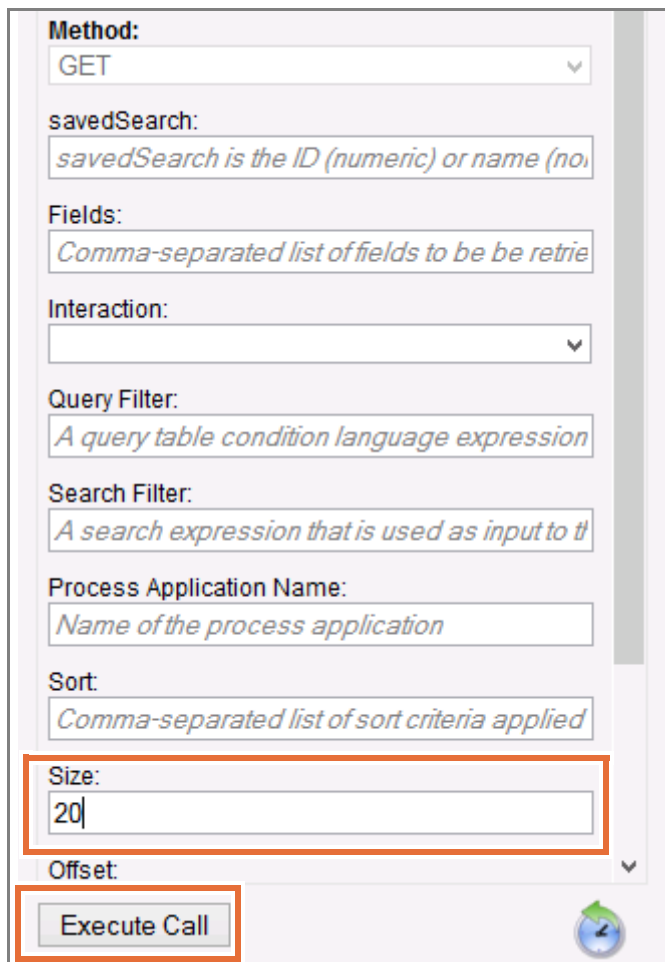


**Result Type:**  
JSON (Javascript format) ▼

**Select API Call:**

- Search API
  - Run Query
  - Execute Performance Query
  - Run Performance Instance Query
  - Search Metadata
  - Create Saved Search Definition
  - Retrieve Saved Search Definition
  - Update Saved Search Definition
  - Delete Saved Search Definition
  - Retrieve Saved Search Definitions List
  - Saved Search Execution**
  - AdHoc Search Execution
  - Search Definition Meta Data By Type
  - Saved Search Definition User Actions

- \_\_\_ g. Enter 20 in the **Size** field and click **Execute Call**.



**Method:**  
GET ▼

**savedSearch:**  
*savedSearch is the ID (numeric) or name (no)*

**Fields:**  
*Comma-separated list of fields to be be retrie*

**Interaction:**  
▼

**Query Filter:**  
*A query table condition language expression*

**Search Filter:**  
*A search expression that is used as input to th*

**Process Application Name:**  
*Name of the process application*

**Sort:**  
*Comma-separated list of sort criteria applied*

**Size:**  
20

**Offset:**  
▼

**Execute Call**



## Information

This method executes a default saved search that is stored in the system. When you leave the savedSearch field empty, the system uses the default search “IBM.DEFAULTALLTASKSLIST\_75” (or you can use the saved search ID number 6), Which returns all the tasks in the system you are allowed to view. You can view the other saved searches in the system by running the **Search API > Retrieve Saved Search Definitions List** method.

You can use the Search Definition Meta Data By Type method in the Search API section to help you create constraints for your query in your environment. The “all” interaction type is used by default, and returns a list of all the tasks in the system. You can read more about the Saved Search Query method by searching the IBM BPM Help System documentation for the Search Execution REST API call.

\_\_ h. Examine the request in the right pane.

**Request:** `https://localhost:9443/rest/bpm/wle/v1/tasks?size=20&filterByCurrentUser=false&calcStats=false`

**Method:** GET

**Status:** 200 - OK

### Header:

X-Powered-By - Servlet/3.0  
 BPM\_GENERIC\_HEADER - SERVED  
 Cache-Control - no-cache, no-store, max-age=0  
 Content-Type - application/json  
 Content-Encoding - gzip  
 Content-Language - en-US  
 Transfer-Encoding - chunked  
 Date -   
 Expires -

### Result:

```
{
  status: "200",
  data: {
    identifier: "TASK.TKIID",
    query: "IBM.DEFAULTALLTASKSLIST_75",
    entityType: "TASK",
    queryExecuteTime: " ",
    taskIndexUpdateInterval: 5,
    attributeInfo: [
      {
        name: "ORIGINATOR",
        type: "STRING",
        content: "TASK.ORIGINATOR",
        isArray: false
      }
    ]
  }
}
```



## Important

Status 200 means that the call was successful. The result lists all the tasks in the current environment. The result is a JSON object that is parsed to get all the IDs for every task whether it is closed, stopped, or deleted in the system.

```
offset: 0,
size: 1,
requestedSize: 20,
totalCount: 1,
countLimitExceeded: false,
countLimit: 500,
items: [
  {
    TASK.TKIID: "3",
    PROCESS_INSTANCE.PIID: "3",
    DUE: "",
    IS_AT_RISK: true,
    NAME: "Fail the Instance",
    ACTIVATED: "",
    PI_NAME: "Testing Loops:3",
    ORIGINATOR: "author1",
    COMPLETED: null,
    ASSIGNED_TO_ROLE_DISPLAY_NAME: null,
    TAD_DISPLAY_NAME: "Step: Fail the Instance",
    STATE: "STATE_TERMINATED",
    SNAPSHOT_ID: "2064.8b37d2a1-44ac-4591-...",
    PROCESS_APP_ACRONYM: "TRAIN",
  }
]
```

Recall that the REST API call is made in the **Properties > data mapping** section of the **Move the Token** service that you explored earlier. The request is:

```
https://localhost:9443/rest/bpm/wle/v1/tasks?interaction=all&queryFilter=STATE%3C%3E3+AND+STATE%3C%3E5&filterByCurrentUser=false&calcStats=false&size=20
```

Compare that request with the one you see in the right pane:

```
https://localhost:9443/rest/bpm/wle/v1/tasks?size=20&filterByCurrentUser=false&calcStats=false
```

You can refine and filter the result set depending on your requirements. For this exercise, this request is modified so any task that is not active is not included in the result. Similarly, deleted tasks are not listed either.

You can create the same request by adding a filter to the test query by adding `STATE<>3 AND STATE<>5` (STATE\_RUNNING and STATE\_FINISHED) in the **Query Filter** field. Because the failed task is in `STATE=7` (STATE\_TERMINATED), the failed task is returned.

The `size=20` argument in the JSON request tells the system to return the first 20 records. If you leave this field blank, the JSON response object `totalCount` field responds with the number of

records, but the requestedSize defaults to zero and no items are returned in the response. The items data must be populated to perform the next steps.

Feel free to examine the remaining contents of the result.



### Information

The different task states are: STATE\_INACTIVE=1, STATE\_READY=2, STATE\_RUNNING=3, STATE\_SKIPPED=4, STATE\_FINISHED=5, STATE\_FAILED=6, STATE\_TERMINATED=7, STATE\_CLAIMED=8, STATE\_TERMINATING=9, STATE\_FAILING=10, STATE\_WAITING=11, STATE\_EXPIRED=12, STATE\_STOPPED 13. The system allows query filter states: 2, 3, 5, 6, 7, and 8.

---



---

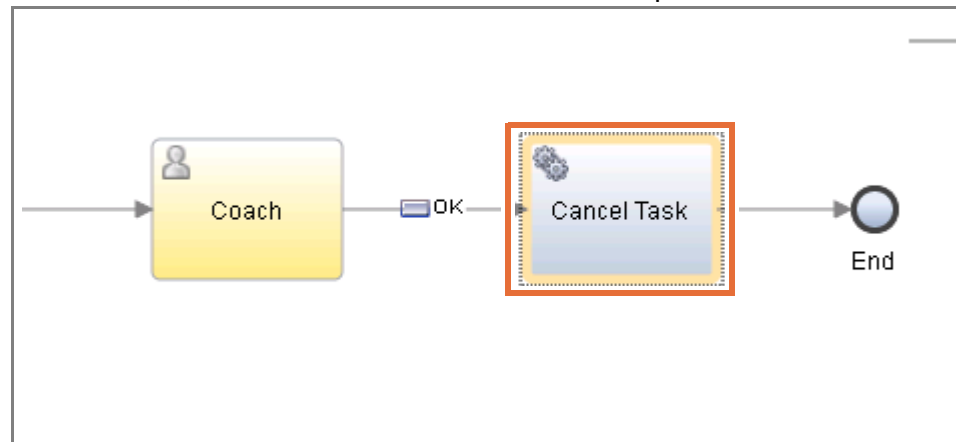


### Optional

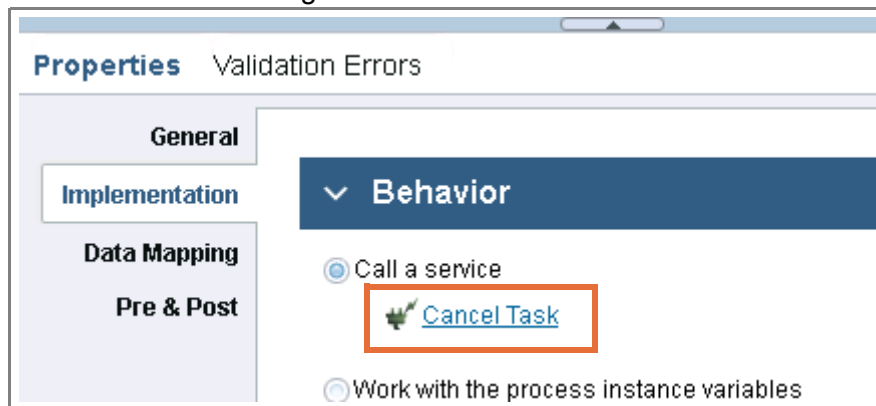
Add the number 6 to the savedSearch field and view the results. This saved search returns all the process instances in the system and lists your failed instance.

---

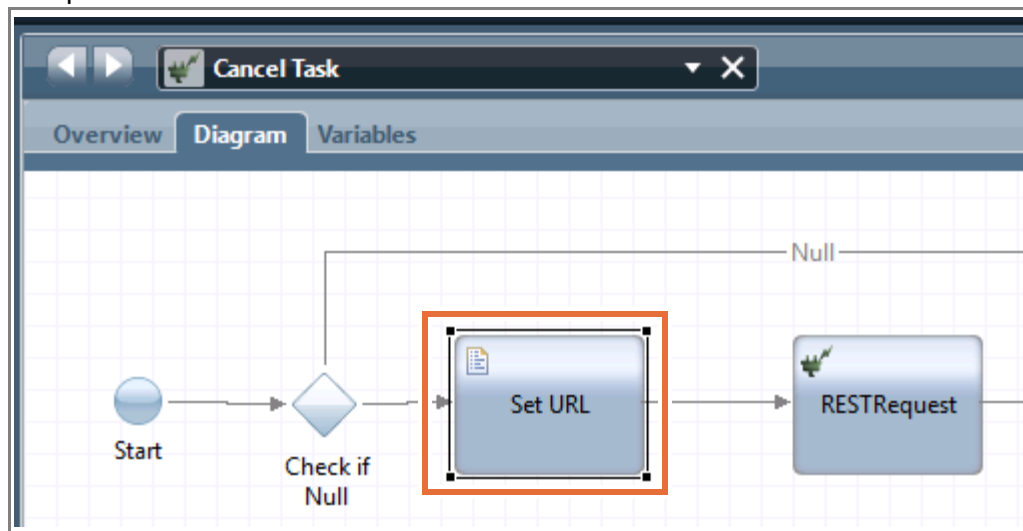
- \_\_\_ i. When you are done examining the REST APIs, close the browser tab.
- \_\_\_ j. Return to the web **Process Designer** tab, return to the **Move the Token** client-side human service, and then click the **Cancel Task** step.



- \_\_\_ k. Click the **Properties > Implementation** menu, and under the Behavior section, click to open the **Cancel Task** integration service.



- \_\_\_ l. When the service opens in the Process Designer client application, click the **Set URL** step.



- \_\_\_ m. Click the **Properties > Implementation** menu.

```
var param = {'approval':
tw.local.taskList[tw.local.taskList.listSelectedIndex].approved};
tw.local.url = 'https://localhost:9443/rest/bpm/wle/v1/task/' +
tw.local.taskList[tw.local.taskList.listSelectedIndex].taskId +
'?action=finish&params=' + JSON.stringify(param);
```



### Information

This script creates the URL that is used to complete a task. The user selects a task on the coach and provides the approval status that is an output value for the task. That value is added to a JSON object that represents the output variable of the activity. The object is converted to a string, and added to the end of the finish task REST API URL. The REST API call is to finish or complete the

task. You can return to the REST API page to view the parameters, test this call, and view more information about this call. When the URL is created, the next step uses an HTTP PUT action with the formatted URL. The system cancels the task that is included as a parameter as part of the URL string.

## 2.4. Use the Move the Token service to complete the process

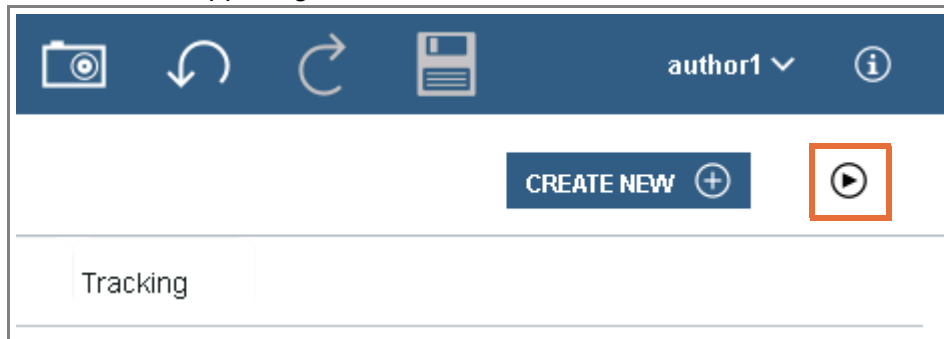
- \_\_\_ 1. Now that you explored the process and the service, you create several failed instances so you can close the ones that you want.
  - \_\_\_ a. Return to the web Process Designer. If not already open, in the library, click **Processes** and then click **Move the Token** to open the process.



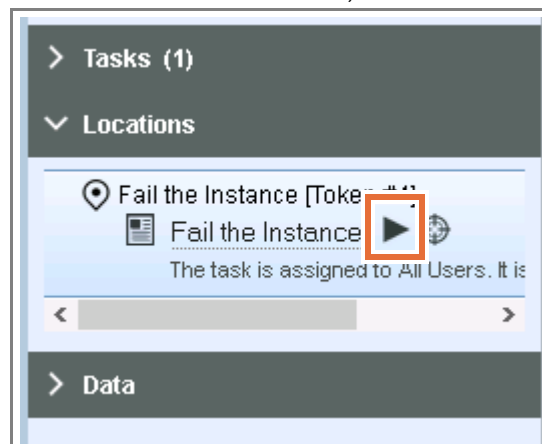
### Hint

Ensure the **Move the Token** process is displayed in the **Definition** tab.

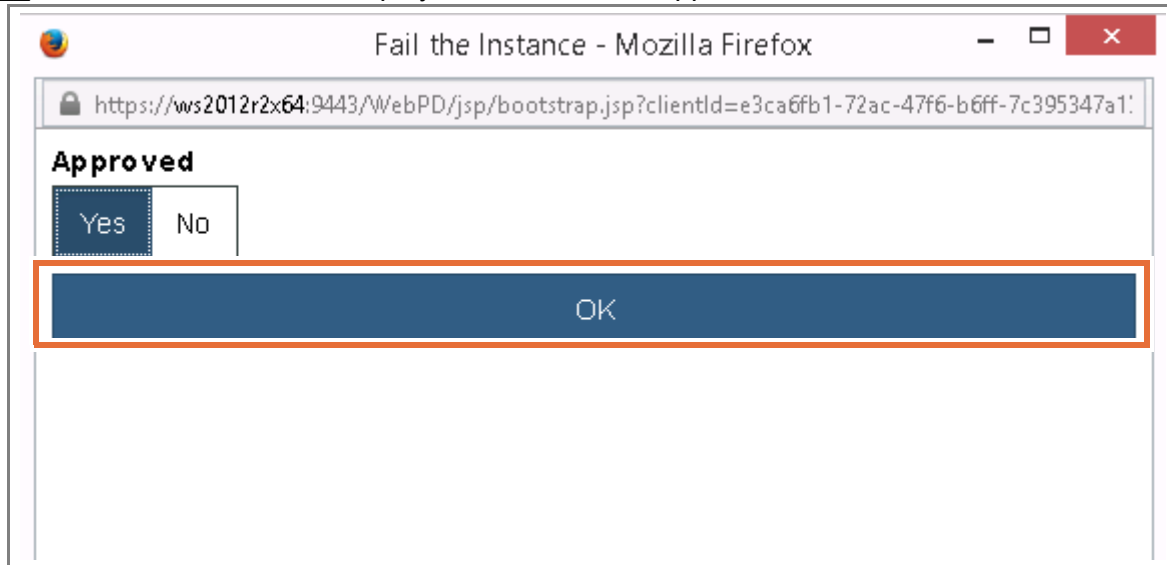
- \_\_\_ b. Click **run** in the upper-right corner.



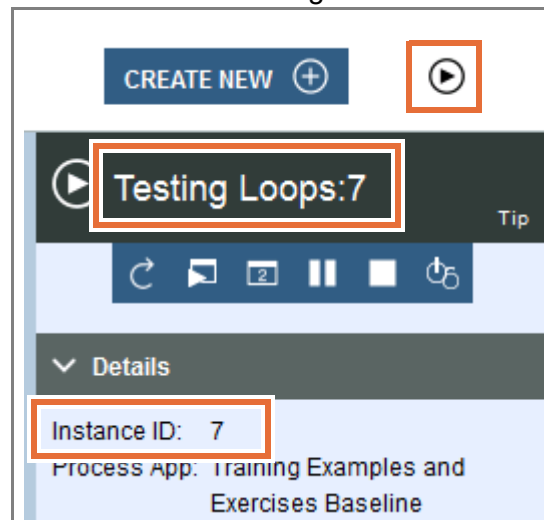
- \_\_\_ c. The web Process Designer switches to the Inspector view and the instance details are shown on the right. In the Locations section, click **start** next to the Fail the Instance task.



- d. When the coach is displayed, select **Yes** to approve and click **OK**.



- \_\_\_ e. Verify that a runtime error is thrown.
- \_\_\_ f. Close the new browser window and return to web Process Designer. Notice that the instance status is **Failed**.
- \_\_\_ g. You can remain in the Inspector view to create more instances of the process. Repeat steps 2.3.1.a through f to create two more failed instances. The instance ID changes to reflect the new instance ID when creating instances from the inspector view.



- \_\_\_ h. When you are done, click the **Search** icon at the top of the Inspector page.



- \_\_\_ i. This page provides a top-level view of all the instances in your Process Server environment. Click the **Refresh** button on the left to refresh the view.

- \_\_\_ j. Several failed instances are listed in the Inspector view. Depending on your environment and the number of instances run, the exact number of failed instances that are shown might not match your environment.

**Testing Loops:7**
**Move the Token**

[<216message:Error: The "Service" service failed with the following information: [TeamworksException name='Process: 'Throw Error' ProcessItem: 'Fail the Instance' Type: 'ITEM', message='<JSScript>', line=1, pos=33 nested=<none>]><103flowObjectID:/25.a0c16880-3d5a-4033-96b5-f5d4eb615960/f624e398-065d-4af3-b399-16e683b2a165/Step (Fail the Instance)>]

Last modified

---

**Testing Loops:6**
**Move the Token**

[<216message:Error: The "Service" service failed with the following information: [TeamworksException name='Process: 'Throw Error' ProcessItem: 'Fail the Instance' Type: 'ITEM', message='<JSScript>', line=1, pos=33 nested=<none>]><103flowObjectID:/25.a0c16880-3d5a-4033-96b5-f5d4eb615960/f624e398-065d-4af3-b399-16e683b2a165/Step (Fail the Instance)>]

Last modified

---

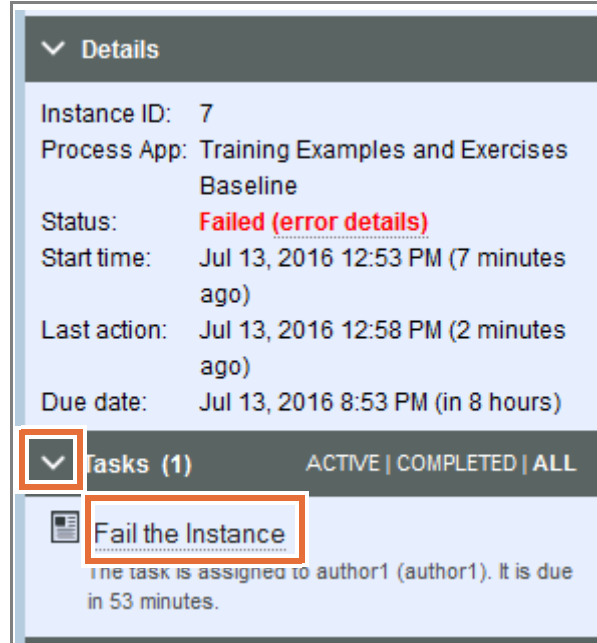
**Testing Loops:5**
**Move the Token**

[<216message:Error: The "Service" service failed with the following information: [TeamworksException name='Process: 'Throw Error' ProcessItem: 'Fail the Instance' Type: 'ITEM', message='<JSScript>', line=1, pos=33 nested=<none>]><103flowObjectID:/25.a0c16880-3d5a-4033-96b5-f5d4eb615960/f624e398-065d-4af3-b399-16e683b2a165/Step (Fail the Instance)>]

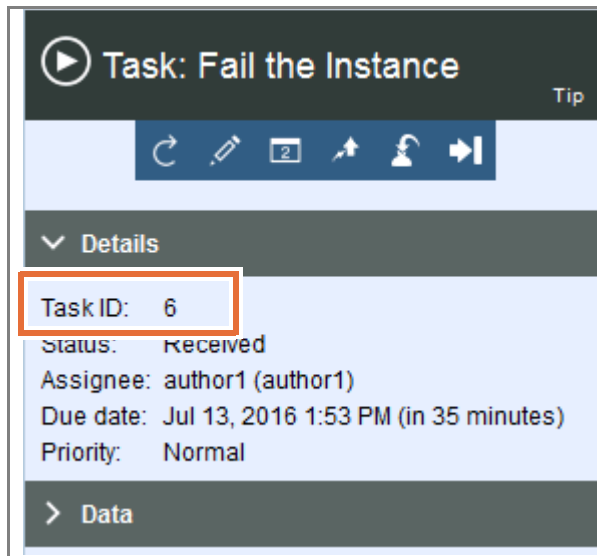
Last modified



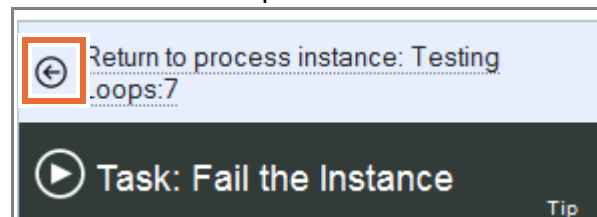
- \_\_\_ k. Click one of the instances and view the failed instance information on the right. In the Tasks section contains a failed task that is associated with this instance. Expand the **Tasks** section and click the **Fail the Instance** task to view the task details.



- \_\_\_ l. This ID is used to cancel tasks and move the token in the next part of the exercise.

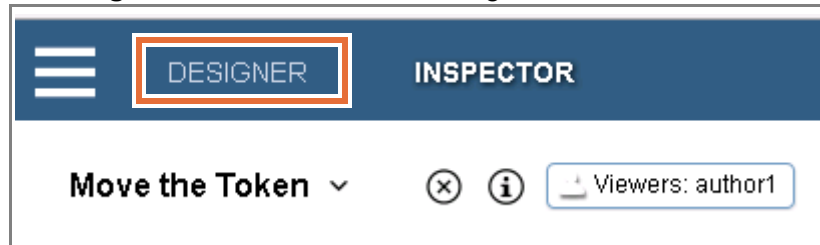


- \_\_\_ m. Click the left arrow to return to the process instance information.

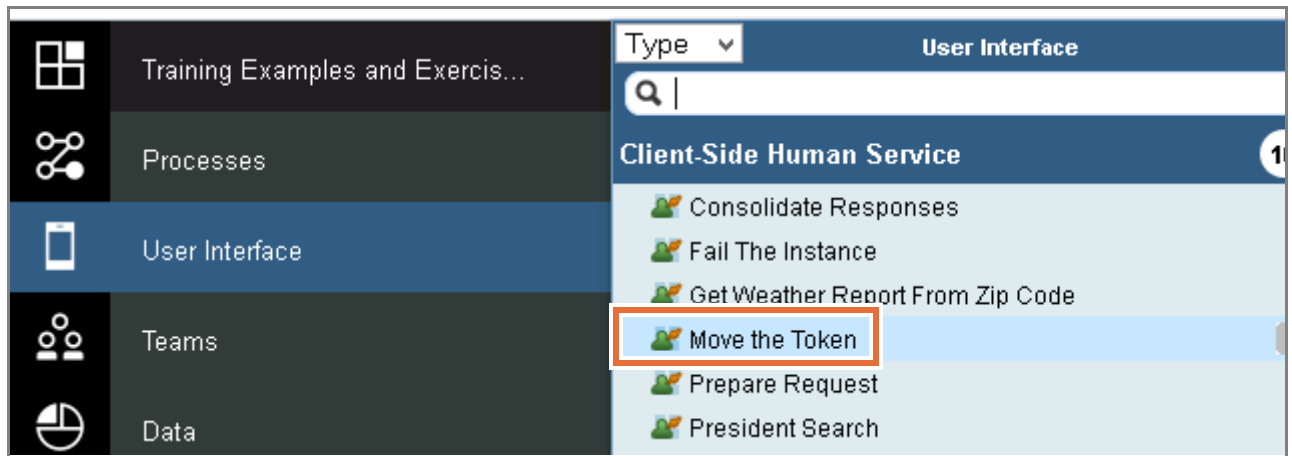


\_\_\_ 2. Move the token of a failed instance.

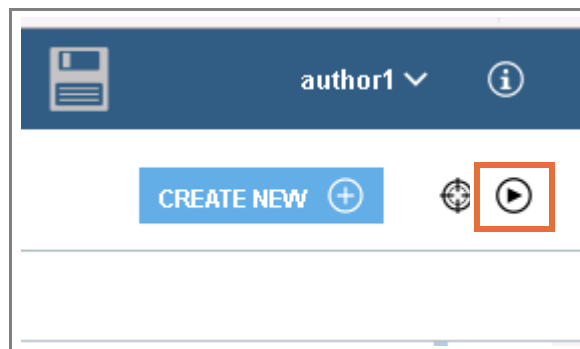
\_\_\_ a. Click the **Designer** tab to return to the Designer view.



\_\_\_ b. In the library, click **User Interface** and then click **Move the Token** to open the client-side human service.



\_\_\_ c. Click **run**.



- \_\_\_ d. The new coach opens in the browser. The coach lists several failed tasks with their instance ID and task ID details. The instance IDs and task IDs that are shown here might not match your environment.

Task List					
	Process Instance ID	PI Display Name	Task ID	Tad Display Name	Approved
<input type="checkbox"/>	4	Testing Loops:4	3	Step: Fail the Instance	No
<input type="checkbox"/>	5	Testing Loops:5	4	Step: Fail the Instance	No
<input type="checkbox"/>	6	Testing Loops:6	5	Step: Fail the Instance	No
<input type="checkbox"/>	7	Testing Loops:7	6	Step: Fail the Instance	No
OK					



### Information

The only data that is not coming from the database is the value for the Approved variable. The remaining contents are pulled from the database by using the REST API call. The coach is set up so that you can set the value of the approved variable when you move the token. Therefore, the value you set during the failed instance is not shown on this coach because the service did not complete.



### Note

In the next step, instance ID 4 is selected and is used as a reference for the remainder of the exercise. In your lab, make sure to keep a note of the instance ID you select for the next step.

- \_\_\_ e. Select any check box next to the Process Instance ID and remember the Process Instance ID number. This instance is the failed instance that you tried to restart, but it still failed. Select the check box for that instance on the left, change the Approved Status to **Yes** by clicking to the left of No, and click **OK**.

Task List					
	Process Instance ID	PI Display Name	Task ID	Tad Display Name	Approved
<input checked="" type="checkbox"/>	4	Testing Loops:4	3	Step: Fail the Instance	<input type="radio"/> Yes <input type="radio"/> No
<input type="checkbox"/>	5	Testing Loops:5	4	Step: Fail the Instance	No
<input type="checkbox"/>	6	Testing Loops:6	5	Step: Fail the Instance	No
<input type="checkbox"/>	7	Testing Loops:7	6	Step: Fail the Instance	No

OK

- \_\_\_ f. The service finishes successfully without any errors. Close the pop-up browser.
- \_\_\_ g. Return to the **Inspector** view in web Process Designer.
- \_\_\_ h. Click **Refresh** to view the updated statuses of all the instances listed.
- \_\_\_ i. Click the **completed instance**.

[Select shown instances](#) | [Select all instances](#) | [Clear selection](#)

Showing 4 of 4 instances  
 Sort by: Date of last action

✔

**Testing Loops:4** ➡ Move the Token  
Last modified Mar 25, 2016 Due Mar 25, 2016

⚠

**Testing Loops:7** ➡ Move the Token  
Last modified Mar 25, 2016 Due Mar 26, 2016

[<216message:Error: The "Service" service failed with the following information: [TeamworksException name="Process: 'Throw Error' ProcessItem: 'Fail the Instance' Type: 'ITEM", message='<JSScript>', line=1, pos=33 nested=<none>]><103flowObjectID:/25.a0c16880-3d5a-4033-96b5-f5d4eb615960/f624e398-065d-4af3-b399-16e683b2a165/Step (Fail the Instance)>]

⚠

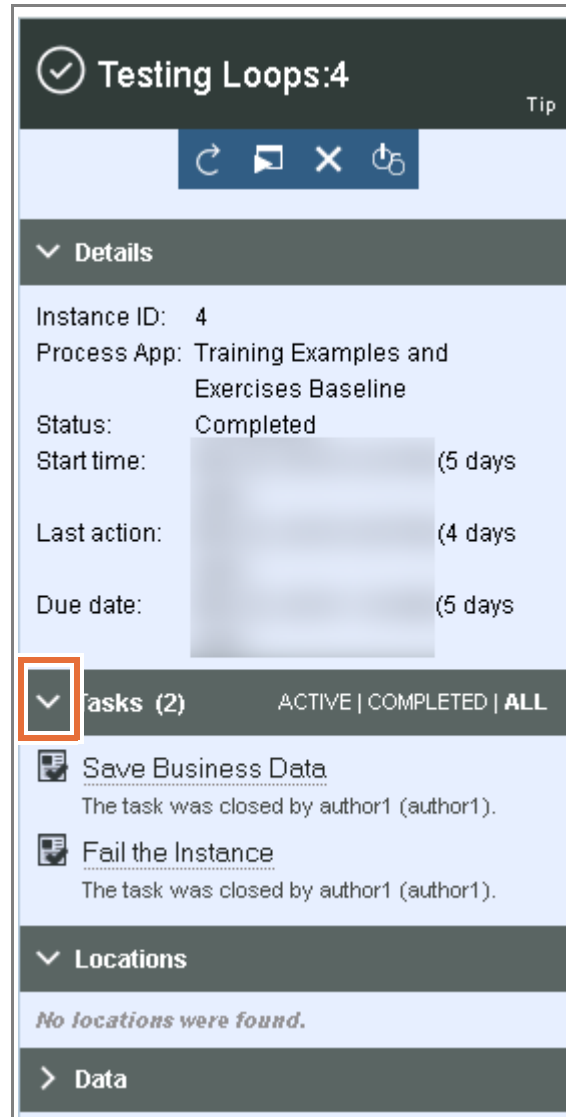
**Testing Loops:6** ➡ Move the Token  
Last modified Mar 25, 2016 Due Mar 26, 2016

[<216message:Error: The "Service" service failed with the following information: [TeamworksException name="Process: 'Throw Error' ProcessItem: 'Fail the Instance' Type: 'ITEM", message='<JSScript>', line=1, pos=33 nested=<none>]><103flowObjectID:/25.a0c16880-3d5a-4033-96b5-f5d4eb615960/f624e398-065d-4af3-b399-16e683b2a165/Step (Fail the Instance)>]

**Note**

The task is now completed. It is not in the failed state.

The Step: Fail the Instance is now in a completed status because the REST API call closed the task. If you open the **Tasks** section, you see that the system also closed the Save Business Data task, which means all the activities in the process are complete, and the instance is in the completed status.

**Information**

You successfully moved the token without having to submit a new version of the process into production. To maintain this instance and complete the activity in a failed or inactive process, you can use the REST API calls to complete tasks. It proves you can complete any failed or active task in the system.

## End of exercise

---



### Optional

Create another process instance and move the token when the instance fails.

You can use this test to complete a task even though the instance might be in a failed state. You can restart the instance to move beyond the failed task.

---

## Exercise review and wrap-up

You worked with a process in which some of the instances are succeeding while some are failing. The business unit wanted to target some of those failed instances to move the token on the process. One of the failed instances was critical; you needed this instance to pass beyond the failed step. Since it was a business emergency, you needed to move the token and get the process beyond that failed activity. You solved this problem of moving the token by using an external implementation of REST API.

---

# Exercise 2. Handling content events in a process

## Estimated time

01:00

## Overview

In this exercise, you learn how to use the CMIS capabilities in IBM Business Process Manager.

## Objectives

After completing this exercise, you should be able to:

- Define ECM folders
- Implement content events on a process

## Introduction

In this exercise, you start with the Hiring Requisition Process. Further enhancements are needed to provide capability for people to create instances when users upload a document to the document store, and affecting the flow of a process by using documents.

## Requirements

Completion of the previous exercise in this book.



## Section 1. Implement a document start event on a process

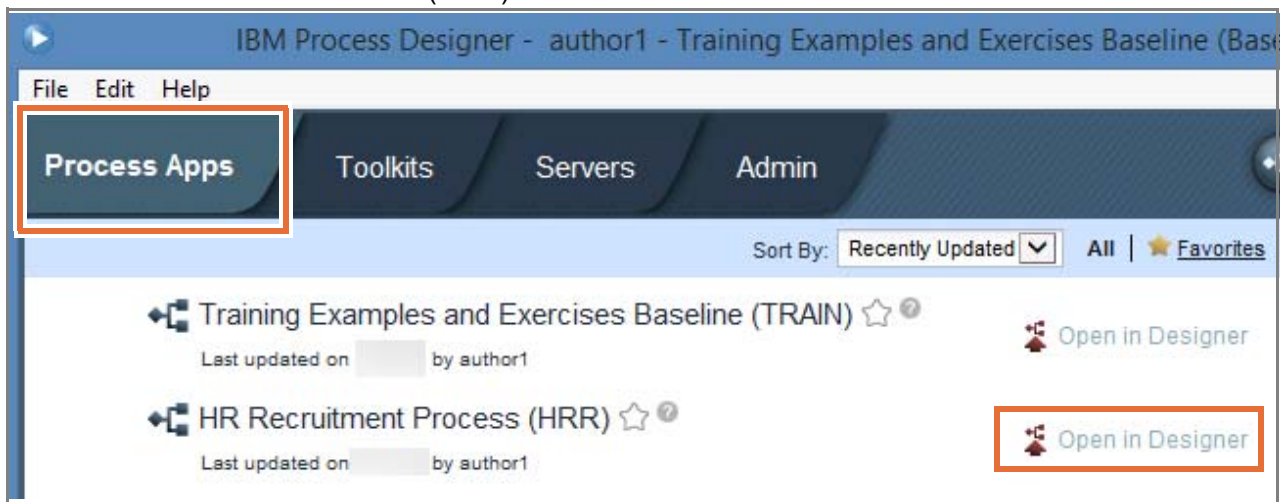
### 1.1. Change the time-based start event to a document start event

\_\_\_ 1. Open the **Hiring Request Process** in the web Process Designer.

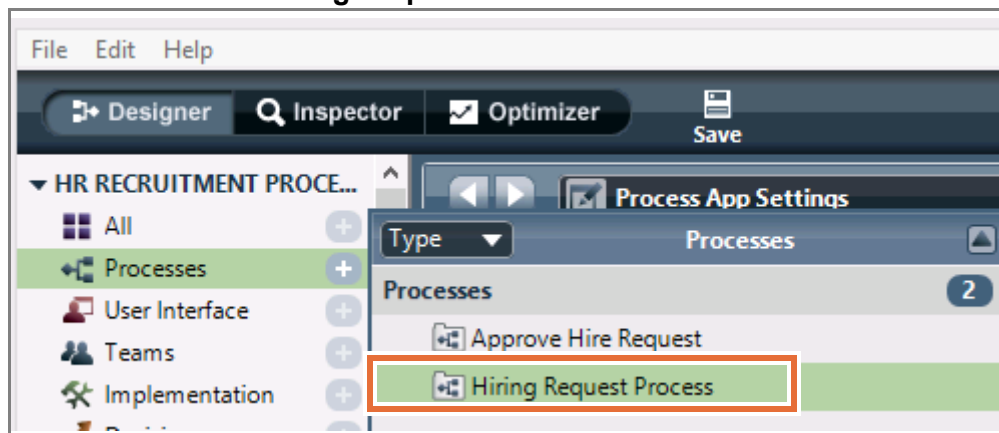
\_\_\_ a. Open the **Process Designer** client application.



\_\_\_ b. With the **Process Apps** tab selected, click the **Open in Designer** link next to HR Recruitment Process (HRR).



\_\_\_ c. Click **Processes > Hiring Request Process**.



#### Hint

The process opens inside a web Process Designer tab that is connected to your Process Designer client application. You can quickly switch between the tools.

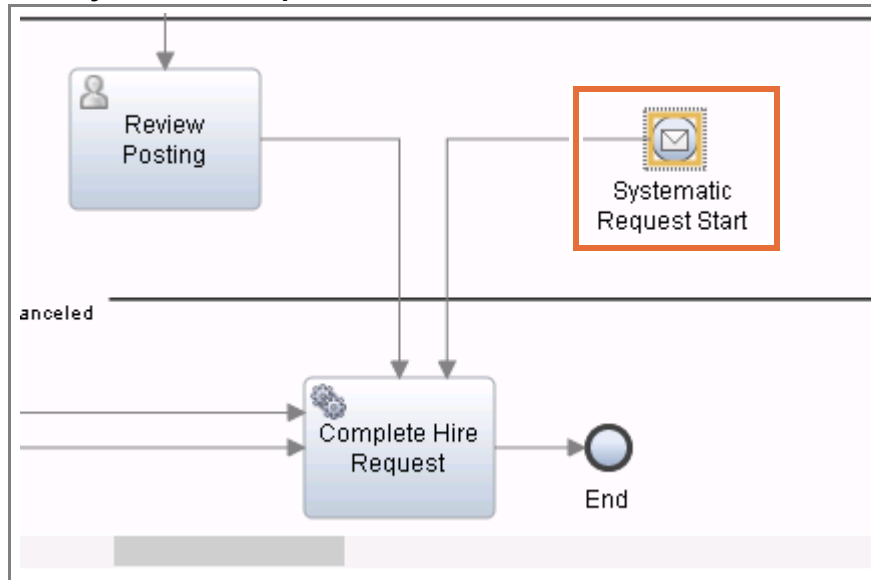
\_\_\_ 2. Change the name of the start event on the definition.



## Information

View the process on the **Definition** tab. The process has a Systematic Request Start that creates an instance of the process on a periodic basis. Management wants a hiring request document that is uploaded to the repository automatically to trigger an instance of the process.

- \_\_\_ a. Click the **Systematic Request Start** event.



- \_\_\_ b. From the **Properties > General** menu, change the name of the event to Hiring Request Document Start.

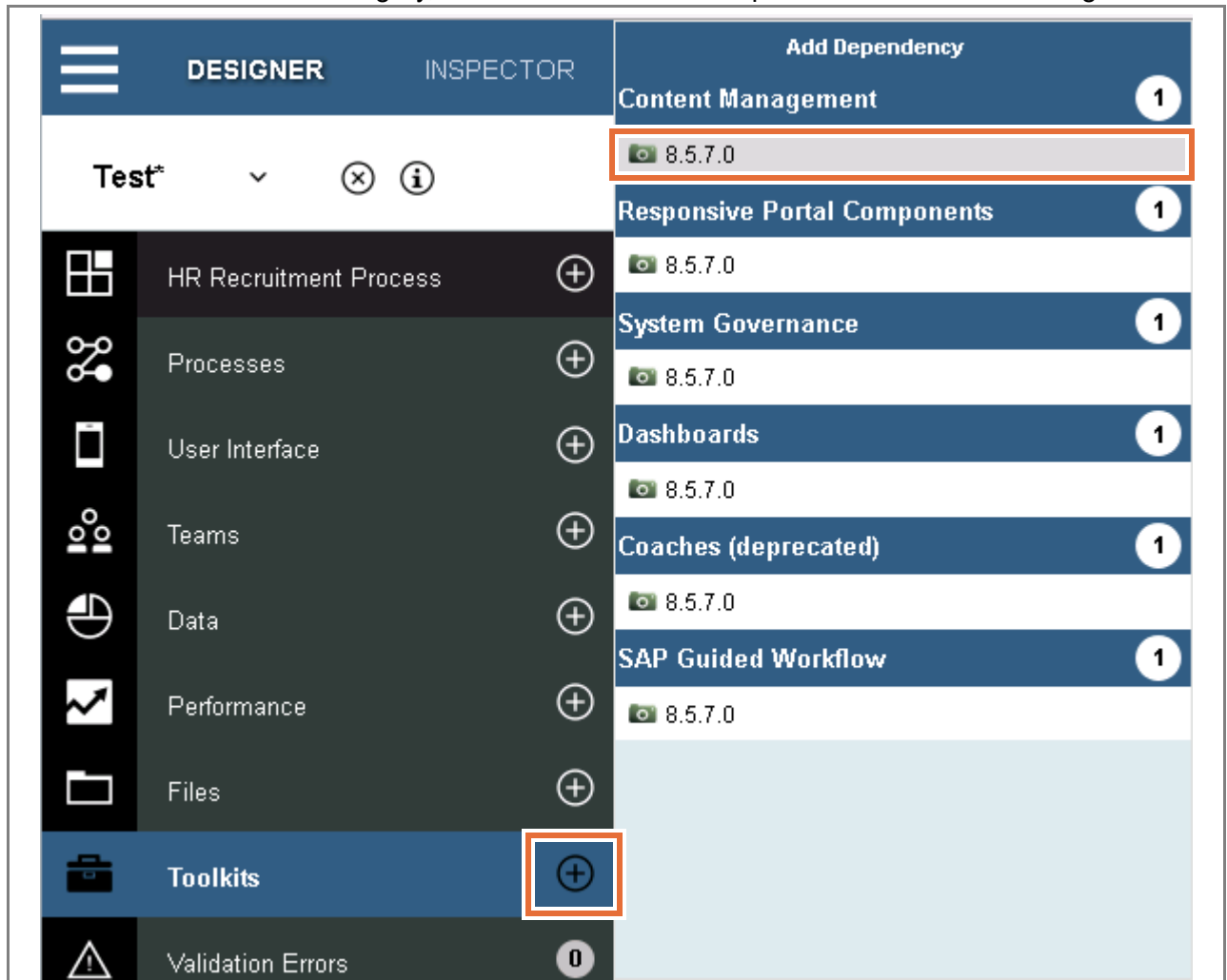
The screenshot shows the BPMN editor interface. The 'Properties' panel is open, and the 'General' tab is selected. The 'Name' field is highlighted with a red box and contains the text 'Hiring Request Document Start'. The 'Documentation' field is empty. The 'Implementation' tab is also visible, showing 'Data Mapping' and 'Pre & Post' sections. The main diagram area shows the same process flow as in the previous diagram, but the 'Systematic Request Start' event is now labeled 'Hiring Request Document Start'.

- \_\_\_ c. Save your work.



- \_\_\_ 3. Create a dependency to the Content Management toolkit snapshot.

- \_\_\_ a. In the web Process Designer, open the library on the left and click the **plus sign (+)** next to the Toolkits category, then click the **8.5.7.0** snapshot under Content Management.

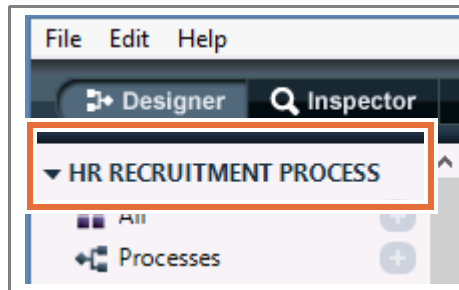


## 1.2. Create the event subscription

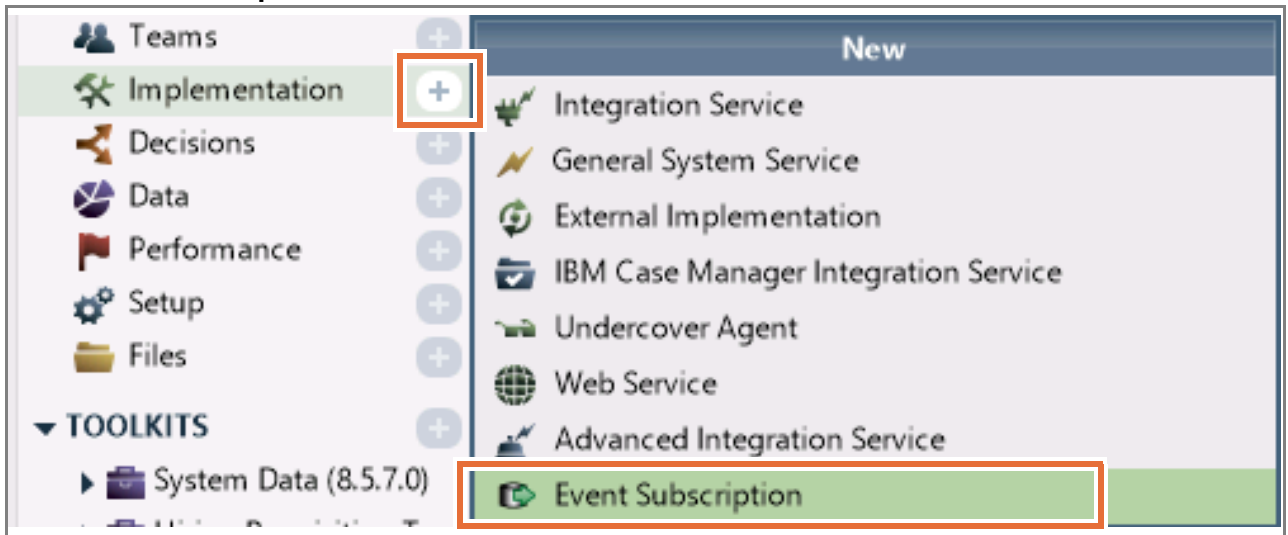
- \_\_\_ 1. Add and configure an event subscription.
- \_\_\_ a. Return to the Process Designer application.



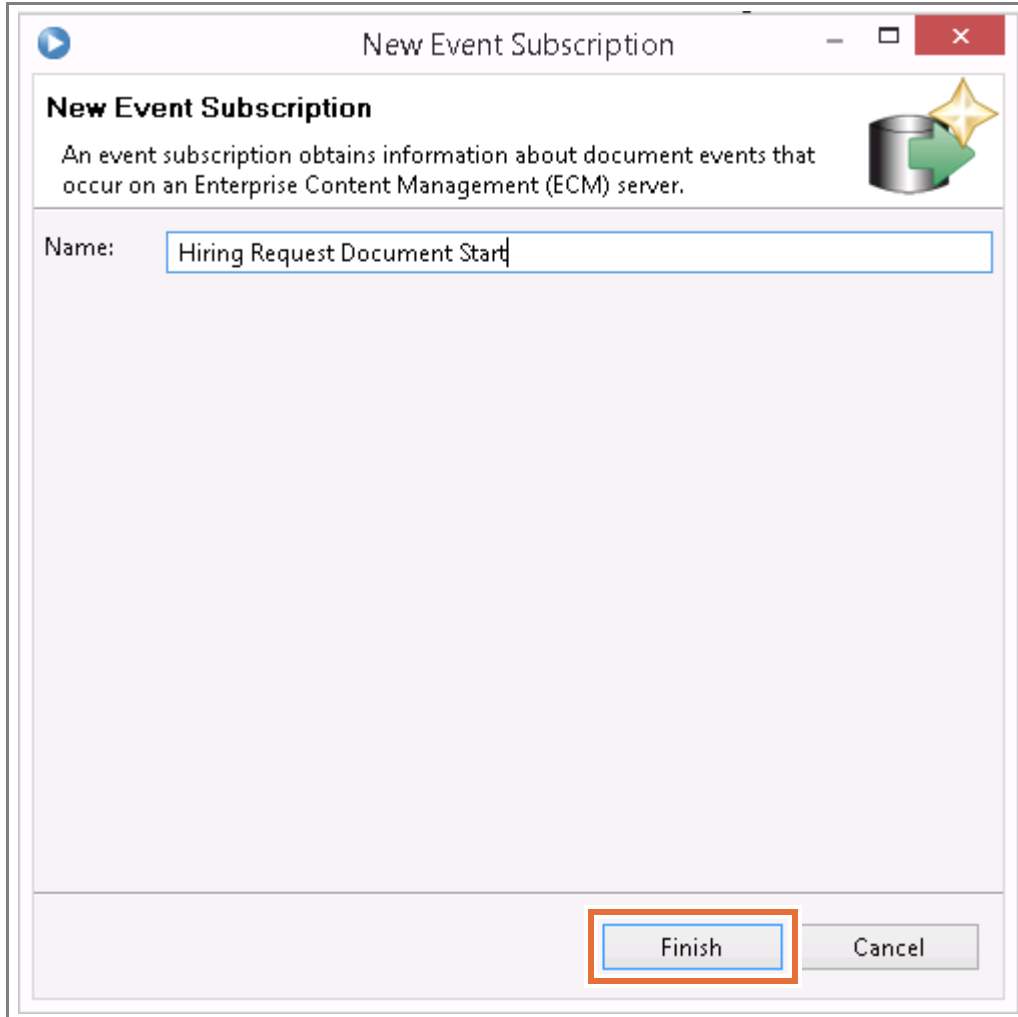
- \_\_\_ b. Make sure that you are editing the HR Recruitment Process in the Process Designer client application.



- \_\_\_ c. In the library, click the **plus sign (+)** next to Implementation and then click **Event Subscription**.



- \_\_\_ d. Name the event subscription **Hiring Request Document Start** and click **Finish**.



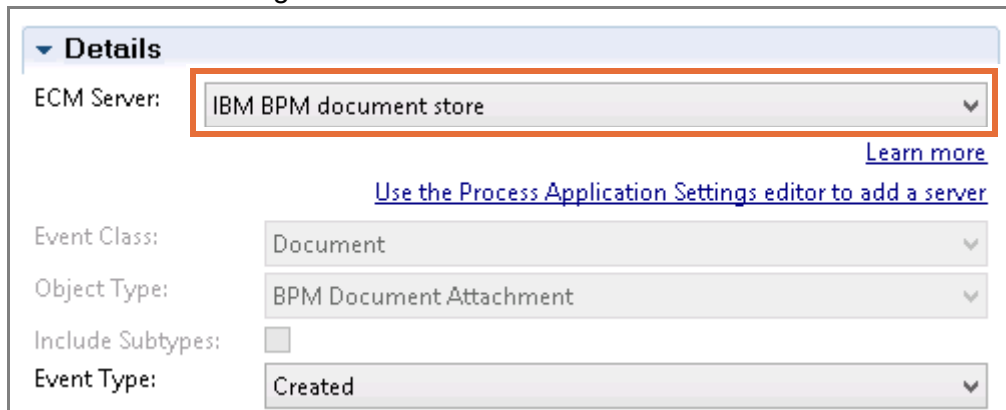
**New Event Subscription**

An event subscription obtains information about document events that occur on an Enterprise Content Management (ECM) server.

Name:

**Finish** **Cancel**

- \_\_\_ e. The event subscription opens in the Process Designer client application. On the right, in the Details section, select **IBM BPM document store** for the ECM Server setting. Leave the rest of the settings as the default.



**Details**

ECM Server:

[Learn more](#)

[Use the Process Application Settings editor to add a server](#)

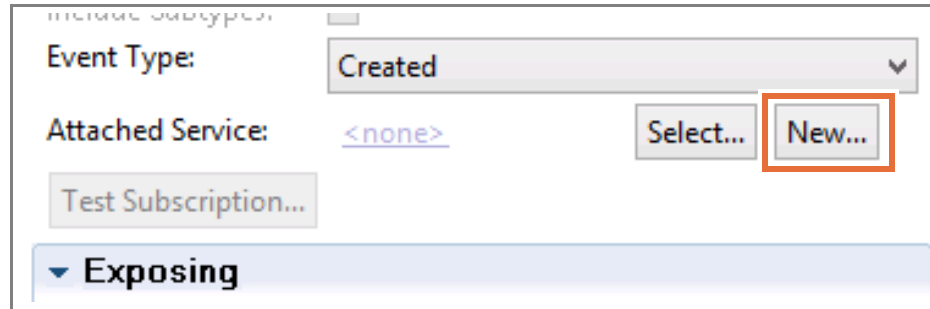
Event Class:

Object Type:

Include Subtypes: ☐

Event Type:

- \_\_\_ f. At the bottom of the Details section, click the **New** button next to Attached Service.



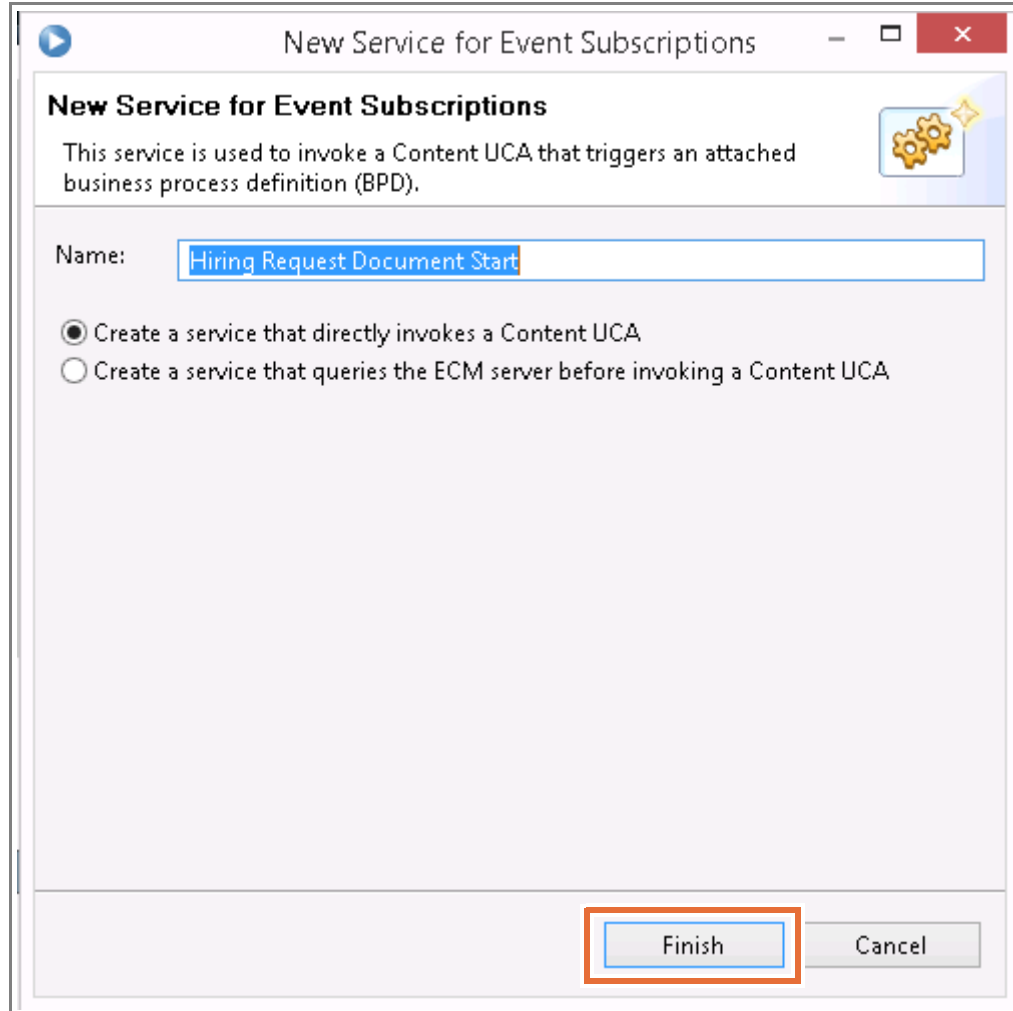
Event Type: Created

Attached Service: <none> Select... **New...**

Test Subscription...

▼ Exposing

- \_\_\_ g. Leave the New Service for Event Subscription options as the default and click **Finish**.



New Service for Event Subscriptions

This service is used to invoke a Content UCA that triggers an attached business process definition (BPD).

Name: Hiring Request Document Start

☒ Create a service that directly invokes a Content UCA

☐ Create a service that queries the ECM server before invoking a Content UCA

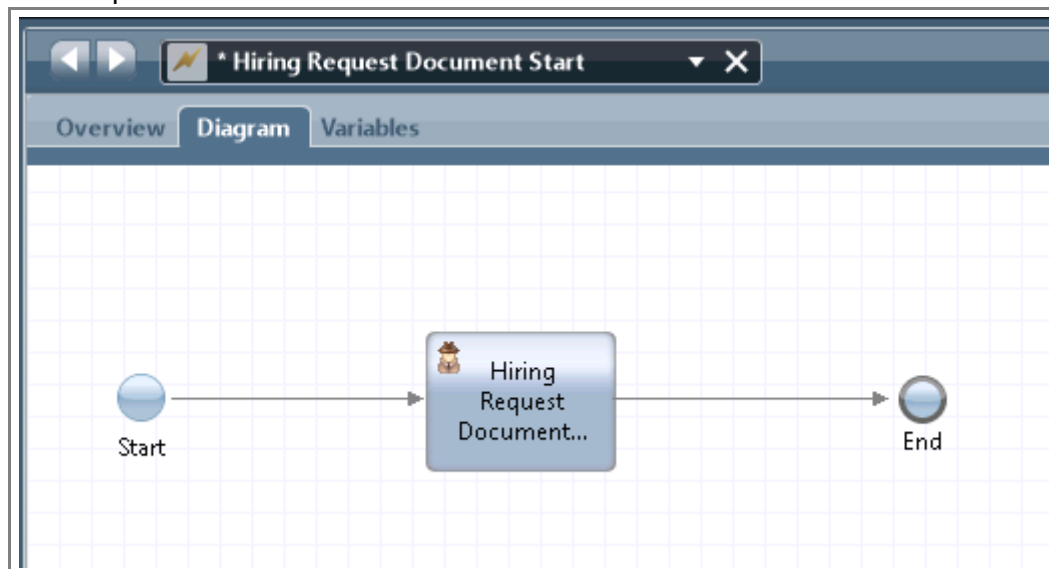
**Finish** Cancel



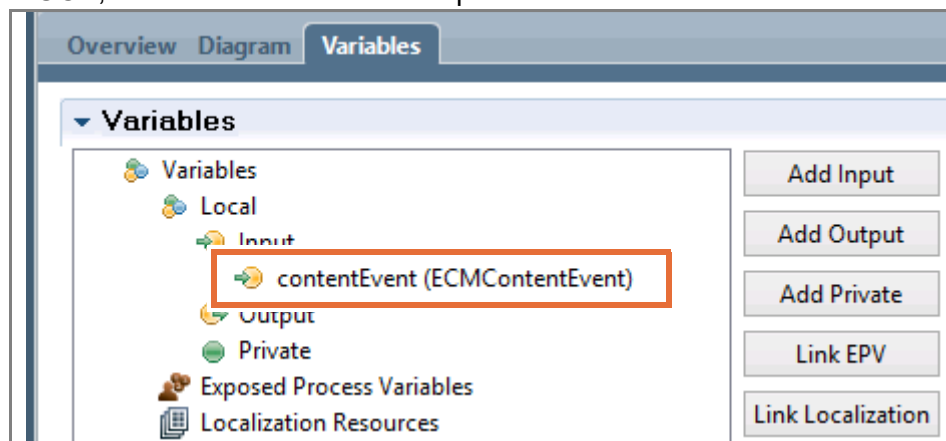
## Information

When you chose the first option, the system creates a fully implemented general system service. The service consists of a Start event, an End event, and a step that invokes the new content undercover agent (UCA). The UCA listens for messages from the system. By invoking the UCA, the system creates an integration service for the second option. The service is partially implemented, and it consists of several components: an Integration step, a decision gateway step, and an Invoke UCA step to invoke the content UCA.

When you choose the first option, the system creates the Hiring Request Document Start general system service and opens it in the designer. The system also creates an undercover agent with the same name and places it on the canvas.

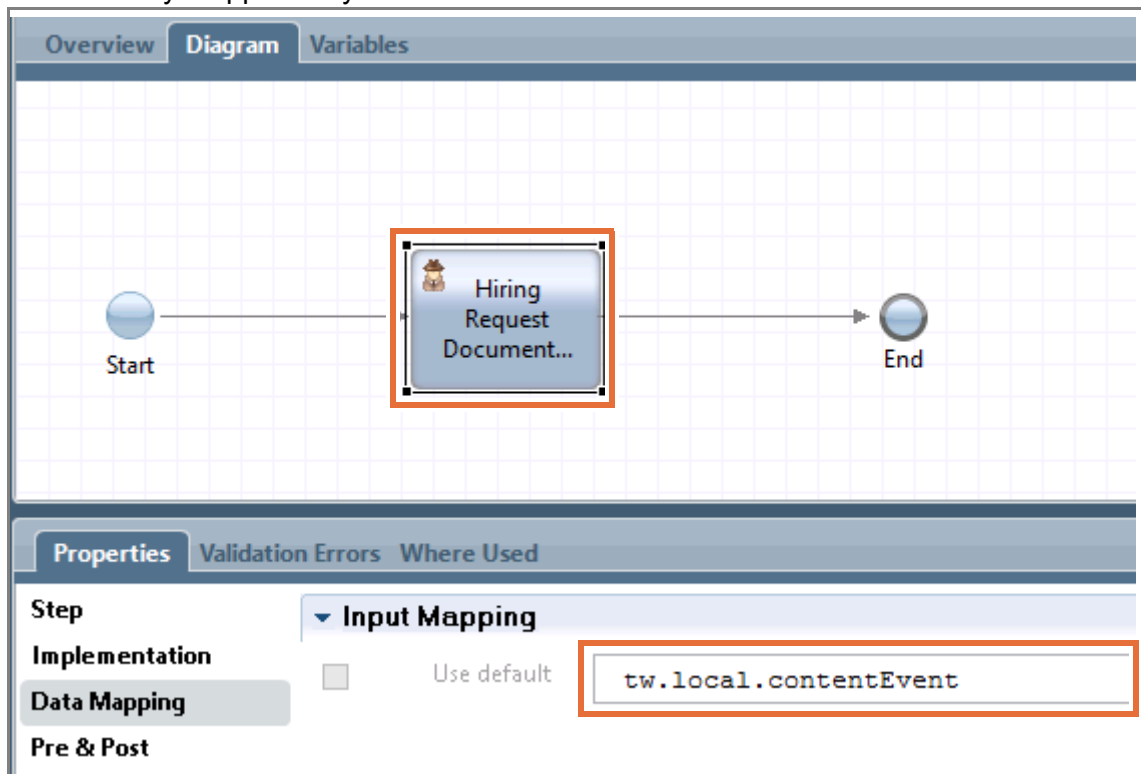


- \_\_\_ h. After the Hiring Request Document Start service is loaded, click the **Variables** tab.
- \_\_\_ i. The system created an input variable `contentEvent` (`ECMContentEvent`). The system requires this type of variable when the content start event is triggered. When you trigger the UCA, it creates an instance of a process.

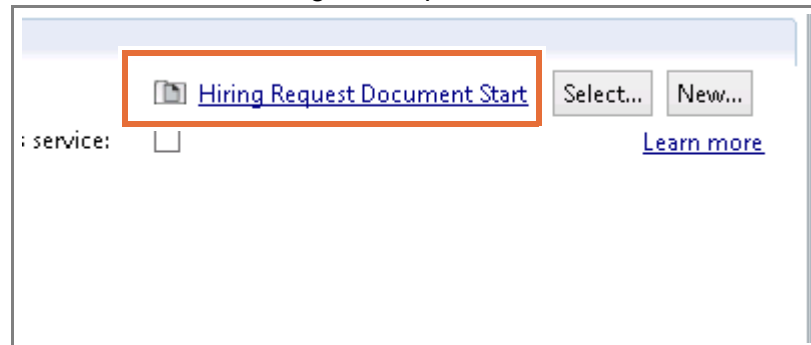


- \_\_\_ j. Return to the **Diagram** tab.
- \_\_\_ k. Click the **Hiring Request Document Start** step on the canvas.

- \_\_\_ l. Open the **Properties > Data Mapping** menu. The `tw.local.contentEvent` variable is already mapped for you.



- \_\_\_ m. Open the **Properties > Implementation** menu and click the **Hiring Request Document Start** link for the undercover agent to open the UCA.





- \_\_\_ n. Under the Details section, the Undercover Agent is already configured for you. The Event Marker is **Content**, which makes it available to implement a Content Start Event. The Undercover Agent requires an input variable of type `ECMContentEvent`, and this variable is sent to the process that triggers the Undercover Agent.

**Scheduler**

Schedule Type: On Event

Event Marker: Content Select...

Run now Add Event Subscription...

**Details**

Queue Name: Async Queue

Implementation: ☒ Variable ☐ Service

Variable Type: [ECMContentEvent](#) [Content Management](#) Select... New...

Enabled: ☒

**Parameter Mapping**

☐ Use default  ⇒ [Input \(ECMContentEvent\)](#)

- \_\_\_ o. Always change the event message to better identify UCA events that occur in the logs. You should always change the event message identifiers from the default hash to a relevant event label when creating undercover agents. At the bottom, in the Event section, change the Event Message for the UCA to `HiringRequestDocumentStart`.

**Event**

Event Message:

**Properties** **Validation Errors** **Where Used**

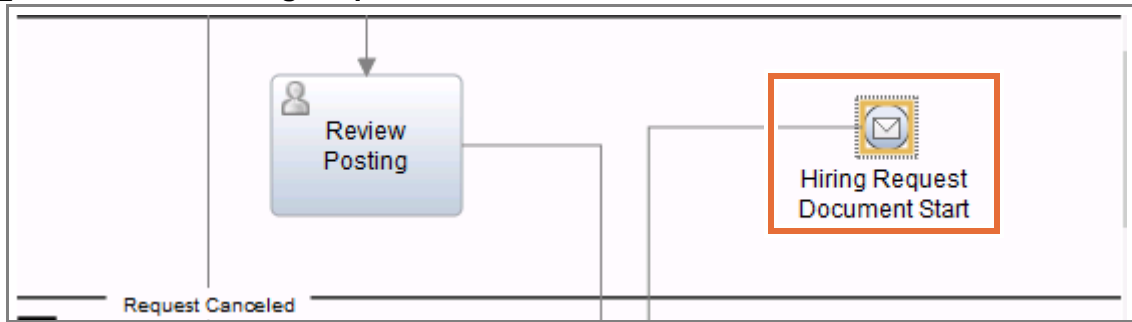
Property	Value

- \_\_\_ p. Save your work.

### 1.3. Configure the start event by using the undercover agent and map a variable to the output

- \_\_\_ 1. Configure the start event
- \_\_\_ a. Return to the web Process Designer. Make sure that the **Hiring Request Process** is open on the **Definition** tab.

- b. Select the **Hiring Request Document Start** event.



- c. Click the **Properties > Implementation** menu. Change the Start Event Type to **ECM Content**.
- d. Select the **Hiring Request Document Start** undercover agent.

**Properties** Validation Errors

**General**

**Implementation**

**Data Mapping**

**Pre & Post**

**Tracking**

Start Event Type

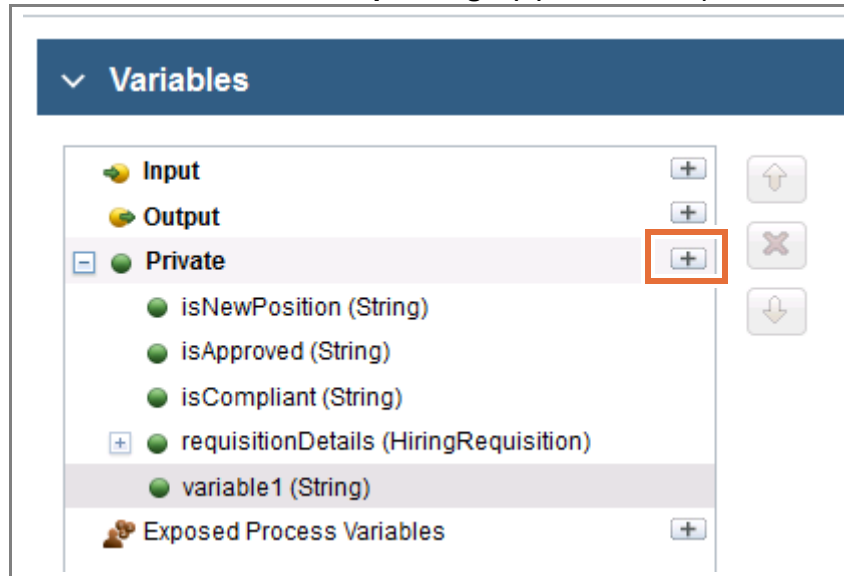
ECM Content

Event Properties

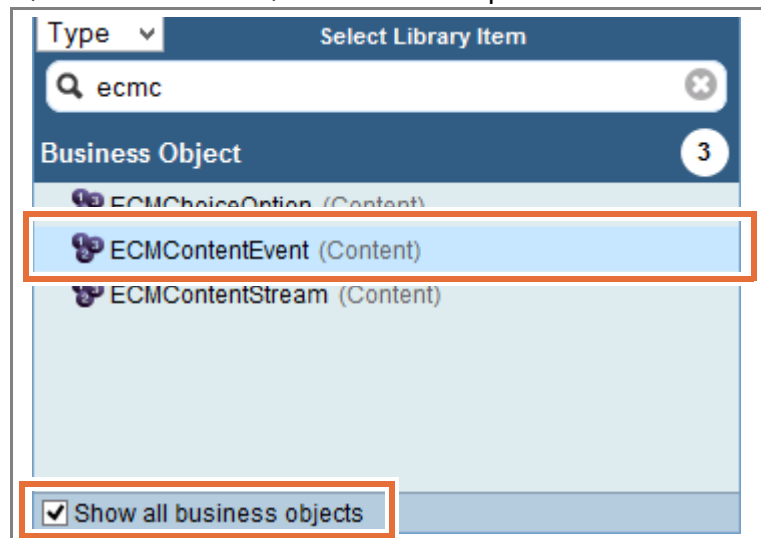
Attached content UCA: Hiring Request Document Start

Condition: 1

- \_\_\_ e. Click the **Variables** tab. Click the **plus sign (+)** to create a private variable.



- \_\_\_ f. Name the variable `content`. Next to the Variable type option, click **Select**.
- \_\_\_ g. Select the **Show all business objects** check box, then select **ECMContentEvent**. The `content (ECMContentEvent)` variable is complete.



- \_\_\_ h. Return to the **Definition** tab.

- \_\_\_ i. Open the **Systematic Request Start Event > Properties > Data Mapping** menu. Map the output of the start event to the `tw.local.content` private variable. All the rest of the child objects are populated when you map the parent object. If you want to overwrite one of the fields inside the parent object, you can map the child object to a different value.

**Output Mapping**

<u>Output (ECMContentEvent)</u>	→	tw.local.content	
<u>serverName (String)</u>	→		
<u>repositoryId (ECMID)</u>	→		
<u>eventClass (ECMEvent...)</u>	→		
<u>objectTypeId (ECMID)</u>	→		
<u>eventType (ECMEventTy...)</u>	→		
<u>objectId (ECMID)</u>	→		

- \_\_\_ j. Save your work.



### Optional

You can test the UCA by switching to the **Process Designer client application**. In the **Hiring Request Document Start Undercover Agent**, click the **Run now** button.

**Scheduler**

Schedule Type: On Event

Event Marker: Content Select...

**Run now** Add Event Subscription...

**Details**

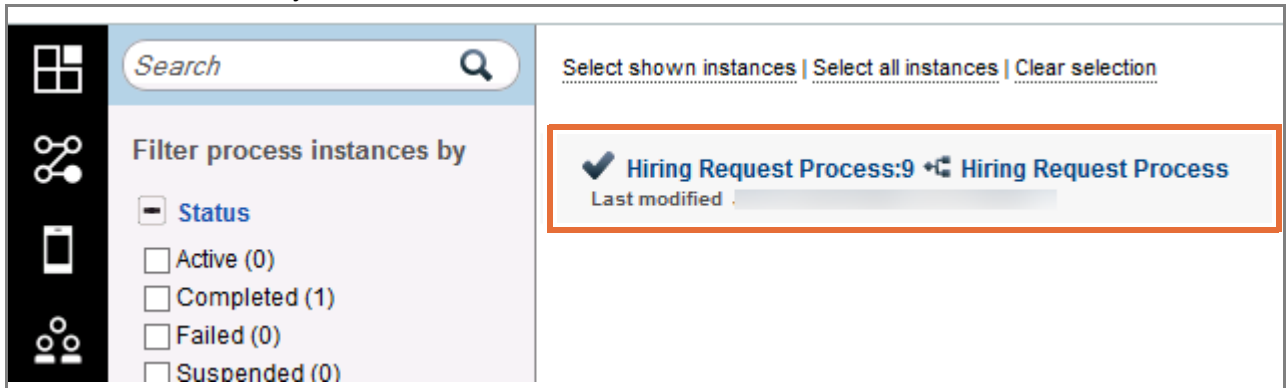
Queue Name: Async Queue

Implementation: ☐ Variable ☒ Service

Click the **OK** button on the Information window when it confirms that the UCA is scheduled for immediate execution. Verify the creation of a Hiring Request Process instance by returning to the web Process Designer and clicking the **Inspector** tab, then click the **Search** icon.



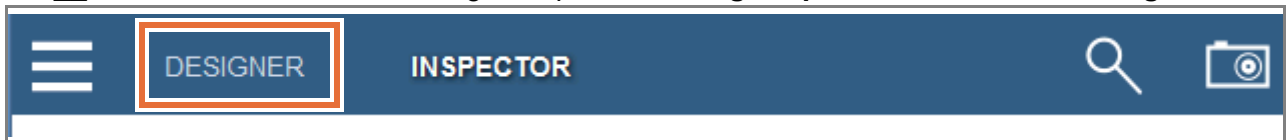
If the instance you created is not shown, click the **Refresh** button.



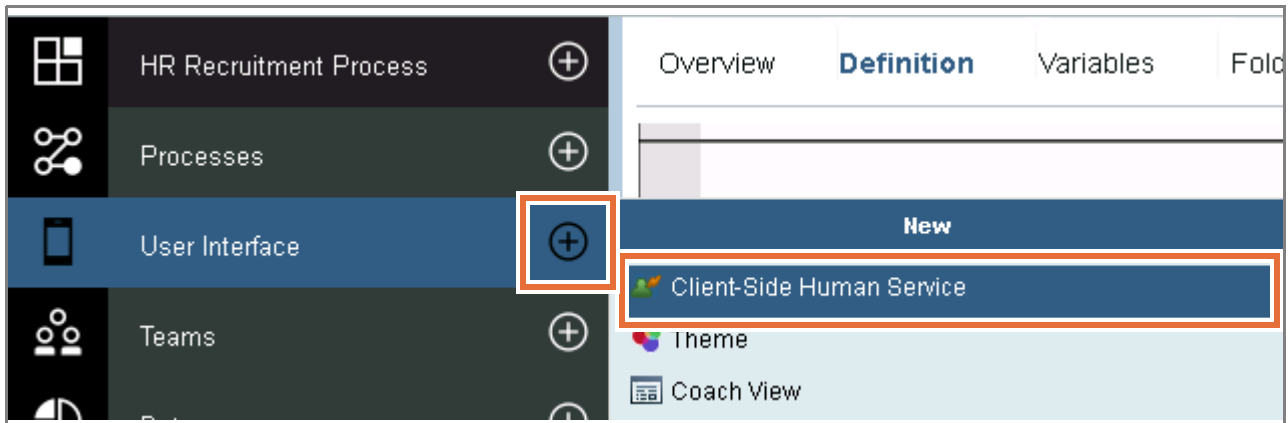
You created an instance of the process, but you didn't upload a document to create an instance of the process. You create the client-side human service to upload your document later in this exercise.

\_\_ 2. Create a client-side human service to trigger the UCA

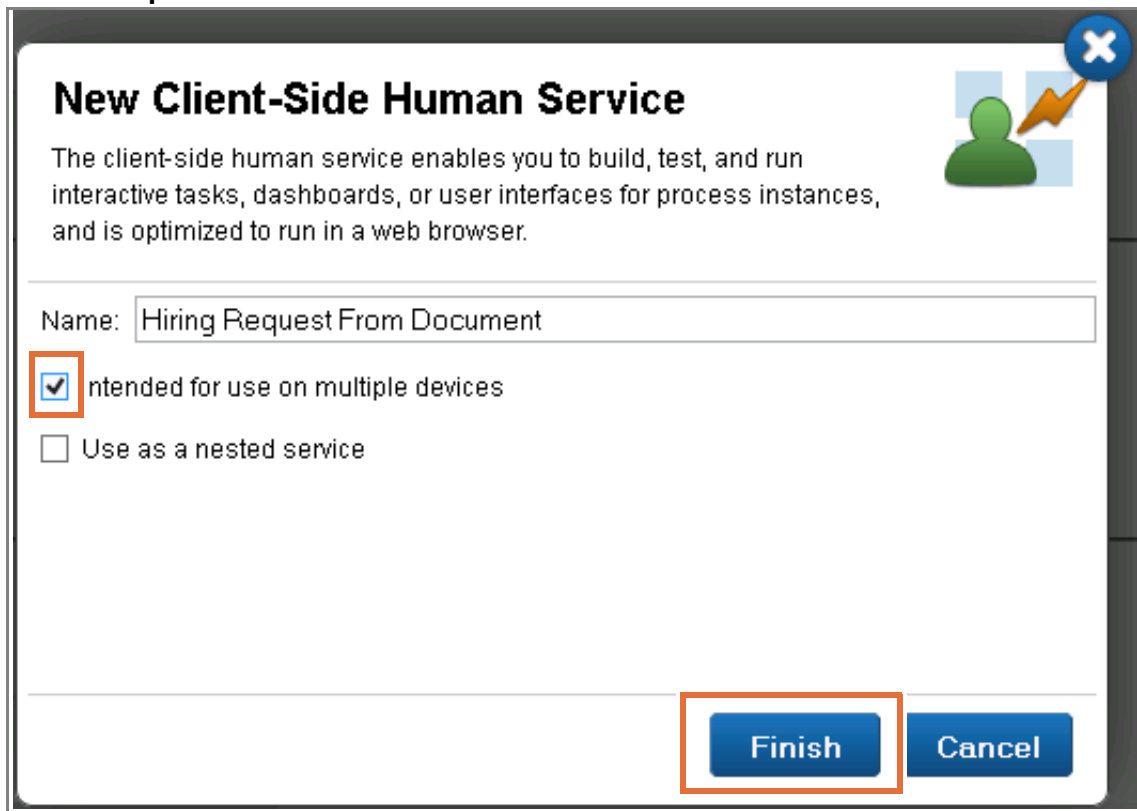
\_\_ a. In the web Process Designer, open the **Hiring Request Process** in the **Designer** tab.



\_\_ b. In the web Process Designer library, click the **plus sign (+)** next to User Interface, and then click **Client-Side Human Service**.



- \_\_\_ c. Name the service **Hiring Request From Document**. Select the **Intended for use on multiple devices** check box and click **Finish**.



**New Client-Side Human Service**

The client-side human service enables you to build, test, and run interactive tasks, dashboards, or user interfaces for process instances, and is optimized to run in a web browser.

Name:

☒ Intended for use on multiple devices

☐ Use as a nested service

**Finish** **Cancel**



## Troubleshooting

If you forget to select the check box during the initial creation, or if you decide later to change to a responsive coach, you can change the service. You can mark any Client-side Human Service as a responsive coach service in the **Overview** tab of the service. Select the **Intended for use on multiple devices** check box in the Common section. By selecting **Intended for use on multiple devices** check box, you are enabling the responsive capabilities on the coach.

The screenshot shows the 'Overview' tab of a service configuration interface. The 'Common' section is expanded, showing the following fields:

- Name:** Hiring Request From Document
- Label:** (with 'Select...' and 'Clear' buttons)
- Documentation:** (with a rich text editor toolbar containing Bold, Italic, Underline, and various list and link icons)

At the bottom of the 'Common' section, the checkbox 'Intended for use on multiple devices:' is checked, and this entire section is highlighted with a red rectangular box.

- \_\_\_ d. Click the **Variables** tab.
- \_\_\_ e. Create a private variable document (ECMDocumentInfo) (List).

**Details**

Name:

Documentation: 

**B I U** | [List Icons]

Is list: ☒

Variable type: [ECMDocumentInfo](#) [Content Management](#) [Select...](#) [New...](#)



### Important

Choose the `ECMDocumentInfo` variable type and not the `ECMDocument` variable type, and ensure that you enable the **Is list** option.

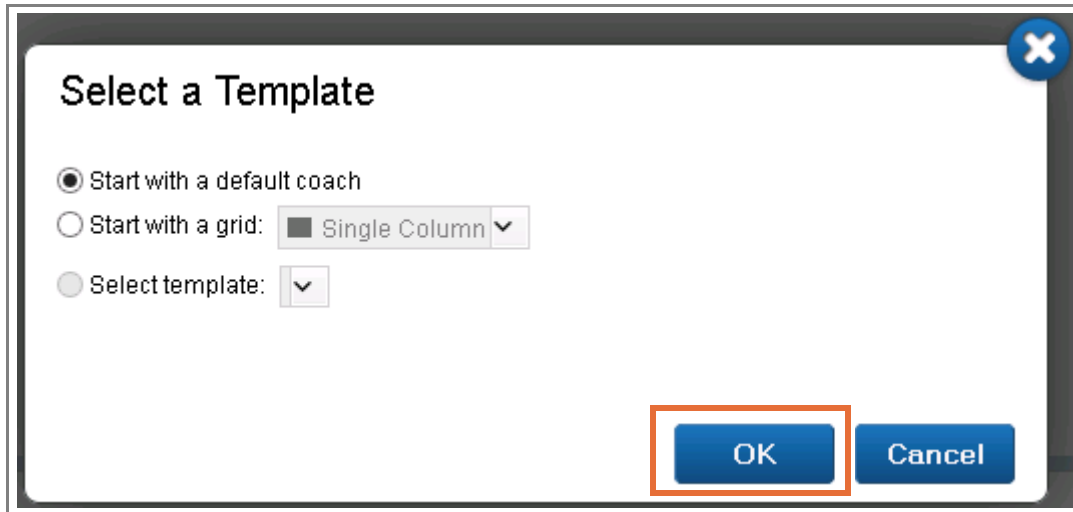
- \_\_\_ f. Save your work.
- \_\_\_ 3. Create the coach
  - \_\_\_ a. Click the **Coaches** tab, and then click the **Coach** on the left.

Overview | Diagram | Variables | **Coaches**

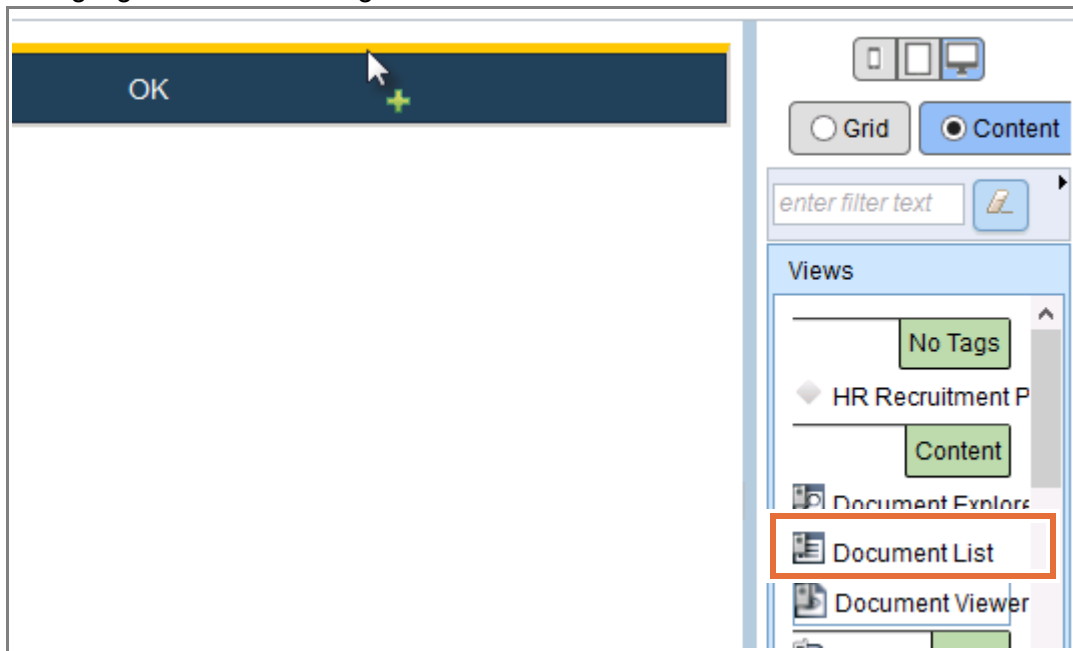
Coach



- b. Click **Start with a default coach** and click **OK**.



- c. Under the Views tray in the palette on the right, drag the **Document List** coach view to the canvas above the OK button. When dragging the coach view from the palette to the canvas, the cursor turns to a green plus sign (+). If the cursor does not change, the system did not recognize your action. The target where the element is placed is highlighted with an orange bar on the canvas.



- d. Click the coach view on the canvas.

- \_\_\_ e. In the **Properties > General > Behavior** section, click the **Select** button next to Binding. Select the **document (ECMDocumentInfo) (List)** variable to bind the variable to the coach view.

The screenshot displays the IBM Business Process Manager interface. At the top, a table titled 'Document List' shows two documents:

Name	File name or URL	Major Version Number	Date Last Modified	Last modified by	Actions
Document 1	Document1.txt	1		administrator	[Search] [Edit] [Delete]
Document 2	Document2.txt	1		administrator	[Search] [Edit] [Delete]

Below the table, the 'Common' and 'Behavior' sections are visible. The 'Behavior' section is highlighted with a red box, showing the following configuration:

- Binding:** document[] (ECMDocumentInfo)
- View:** Document List (ECMDocumentInfo (List))

- \_\_\_ f. Click the **Properties > Configuration** menu.



### Information

The **Configuration** menu contains a large amount of data and sometimes can take up to a minute to load. Sometimes the Configuration menu times out and doesn't load completely, leaving a blank section. If you cannot see the section, open the Firefox menu and click **History > Clear Recent History > Clear Now**. Then, refresh the browser.

- \_\_\_ g. Select the **Allow create**, **Allow update**, and **Allow delete** check boxes.

The screenshot shows the 'Properties' dialog box with the 'Configuration' tab selected. The left sidebar contains the following categories: General, Positioning, Configuration (selected), Visibility, and HTML Attributes. The main area lists several configuration options, each with a circular icon and a checkbox:

Property	Icon	Checkbox
Collapsible:		<input type="checkbox"/>
Collapsed:		<input type="checkbox"/>
Allow create:		<input checked="" type="checkbox"/>
Allow update:		<input checked="" type="checkbox"/>
Allow delete:		<input checked="" type="checkbox"/>
Open in new window:		<input type="checkbox"/>

The 'Allow create', 'Allow update', and 'Allow delete' rows are highlighted with a red rectangular box.

- \_\_\_ h. Save your work.

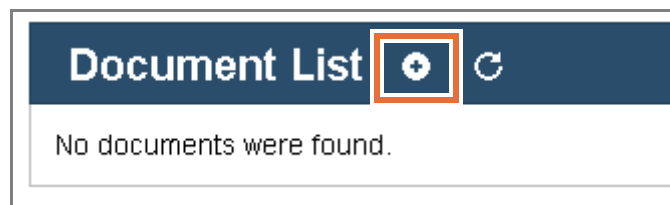
## Section 2. Test the start content event

### 2.1. Test the start content event in the Hiring Request Process

- \_\_\_ 1. Upload a document by using the coach.
- \_\_\_ a. In the Hiring Request From Document client-side human service, click **Run**.



- \_\_\_ b. Wait until the Document List coach view completes its request for documents and lists that no documents were found. Click the **plus sign (+)** next to the Document List section header.



- \_\_\_ c. Click the **folder button** next to Browse and double-click the file C:\labfiles\Lab and Exercise Assets\Exercise Code\Exercise 2\New Hire Request.doc file.
- \_\_\_ d. Use New Hire Request as the document name. Click **OK**.

**Add Document**


Select either a local document with a file as content or a local document that links to a URL. Then, specify the file or URL, and a name.

**Content**

File

▼

New Hire Request.doc



**Properties**

**Name**

New Hire Request





☐ **Add another document**

OK

Cancel

- \_\_\_ e. Normally the Ajax request times out between the initial load and the document upload completion on the first try. The second upload attempt is usually successful. If the document upload fails, or the document is not listed after clicking OK, repeat steps 2.1.1.b - 2.1.1.d and attach the document a second time.
- \_\_\_ f. When the document is uploaded and is shown in the Document List, click **OK** to submit the request.

**Document List**
⊕ ↺

Name	File name or URL	Major Version Number	Date Last Modified	Last modified by	Actions
New Hire Request	New Hire Request.doc	1	10/1/2016 1:00:00 PM	author1	   

OK

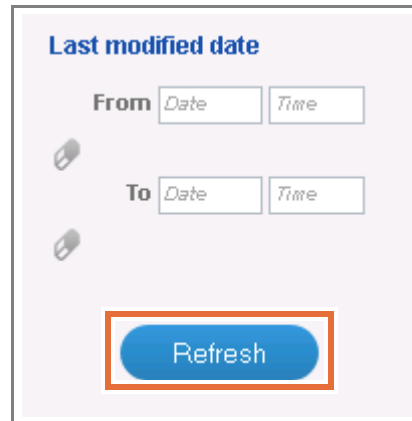
- \_\_\_ g. The service completes. Close the browser.

## 2.2. View the instance details in the browser

- \_\_\_ 1. View the instance by using the Inspector view.
  - \_\_\_ a. Open the **Hiring Request Process** in the web Process Designer.
  - \_\_\_ b. In the web Process Designer window, open the **Inspector** view.

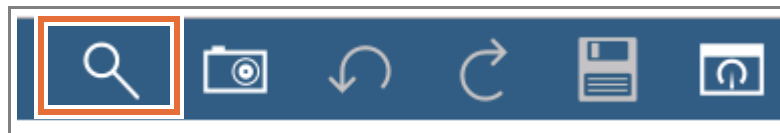


- \_\_\_ c. Click the **Refresh** button to refresh the instances that are shown.

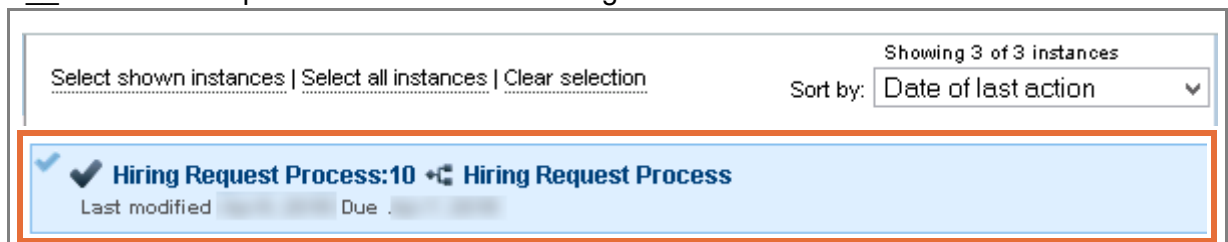


### Hint

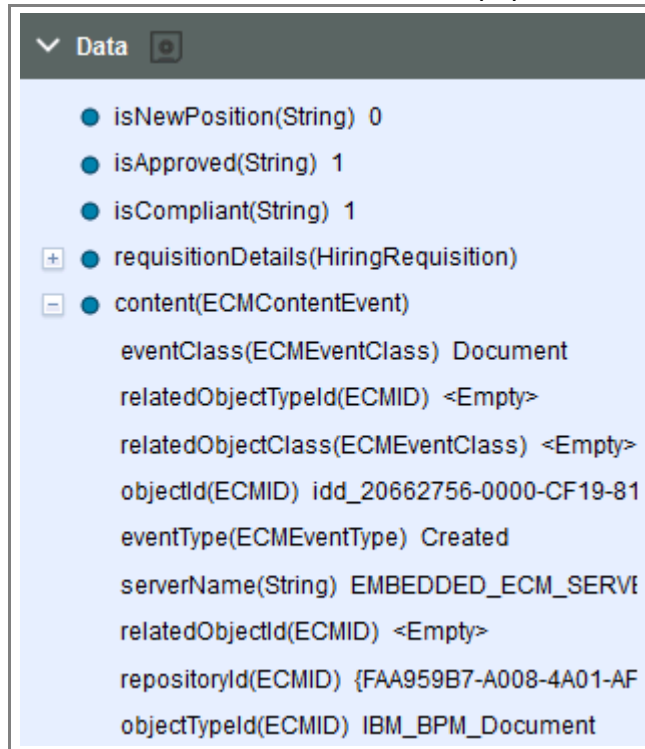
If you do not see the Refresh button, click the **Search** icon.



- \_\_\_ d. Click the process instance with the highest number.



- \_\_\_ e. Expand the **Data** section. The content variable is populated, and the objectId is shown.



- \_\_\_ f. Return to the **Designer** tab.



### Information

When you run the coach, the server created an instance of the process. The creation of the document in the internal document repository triggers the creation of the instance. If you created this functionality as part of a production development effort, the next step inside the process is to associate the document with the instance. You would then create the step when the instance is complete, persist the document to an ECM server.

## End of exercise

## Exercise review and wrap-up

In the first part of the exercise, you modified an existing process to set up a document start event. You created the event subscription and verified the user-generated settings. You then created a coach and tested the ECM subscription. You proved that the document was sent as part of the start event output variable.



---

# Exercise 3. Localizing a coach

## Estimated time

01:00

## Overview

In this exercise, you create a localization bundle and add the keys to the localization. These localization keys expand the coach to include language options.

## Objectives

After completing this exercise, you should be able to:

- Create a localization resource
- Implement the localization on a coach
- Test the localization

## Introduction

Management decided to expand the existing HR Recruitment Process to include the company's HR recruiting efforts outside the United States. Your job is to change one coach for the pilot to demonstrate localization of the entire process to management.

## Requirements

Completion of the previous exercise in this book.

## Process narrative

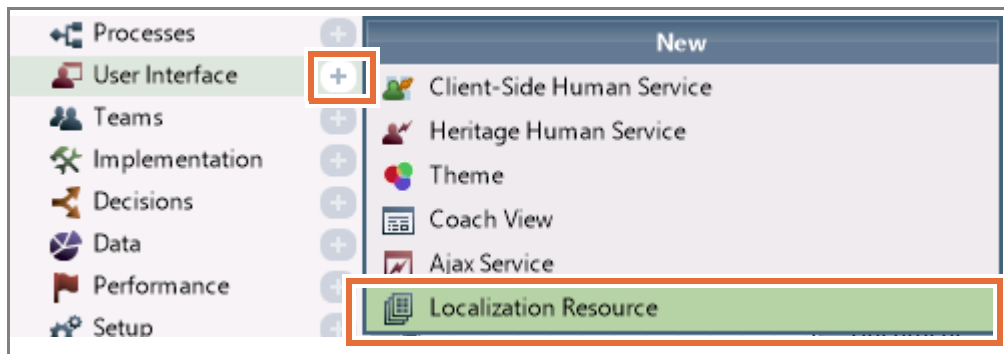
The process owner wants to test the ability of a user to create an ad hoc report. The next IBM BPM team meeting has a block of time that is scheduled to present to the subject matter experts how ad hoc reports are created. It is not currently important to establish the specific user groups with the ability to create the reports.

## Section 1. Create a localization resource

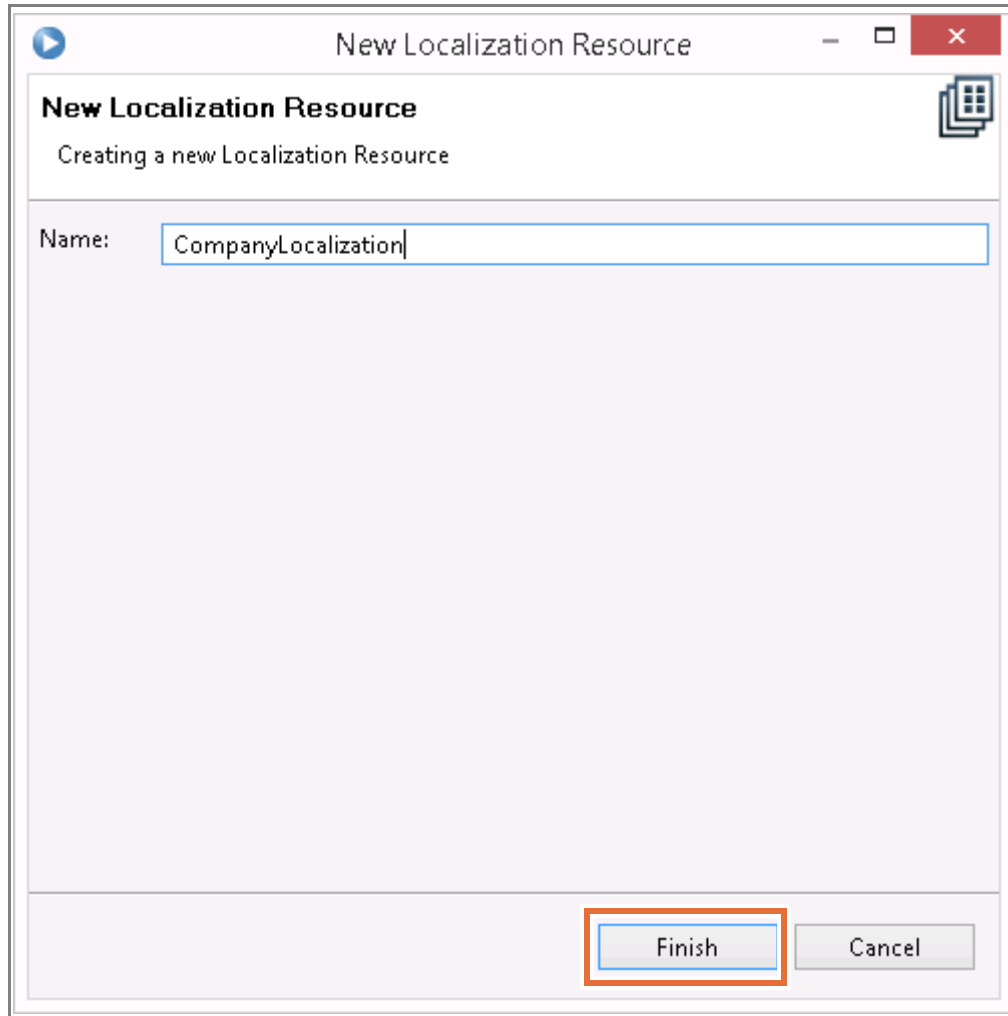
Use the localization resource to change the language of coach fields. The first step to localizing coaches is to create a localization resource.

### 1.1. Create a localization resource

- \_\_\_ 1. Open the **HR Recruitment Process** process application in the Process Designer client application.
- \_\_\_ 2. Create a localization resource.
  - \_\_\_ a. Click the **plus sign (+)** sign next to the **User Interface** category, and click **Localization Resource**.



- \_\_\_ b. Name the new resource `CompanyLocalization` and click **Finish**.



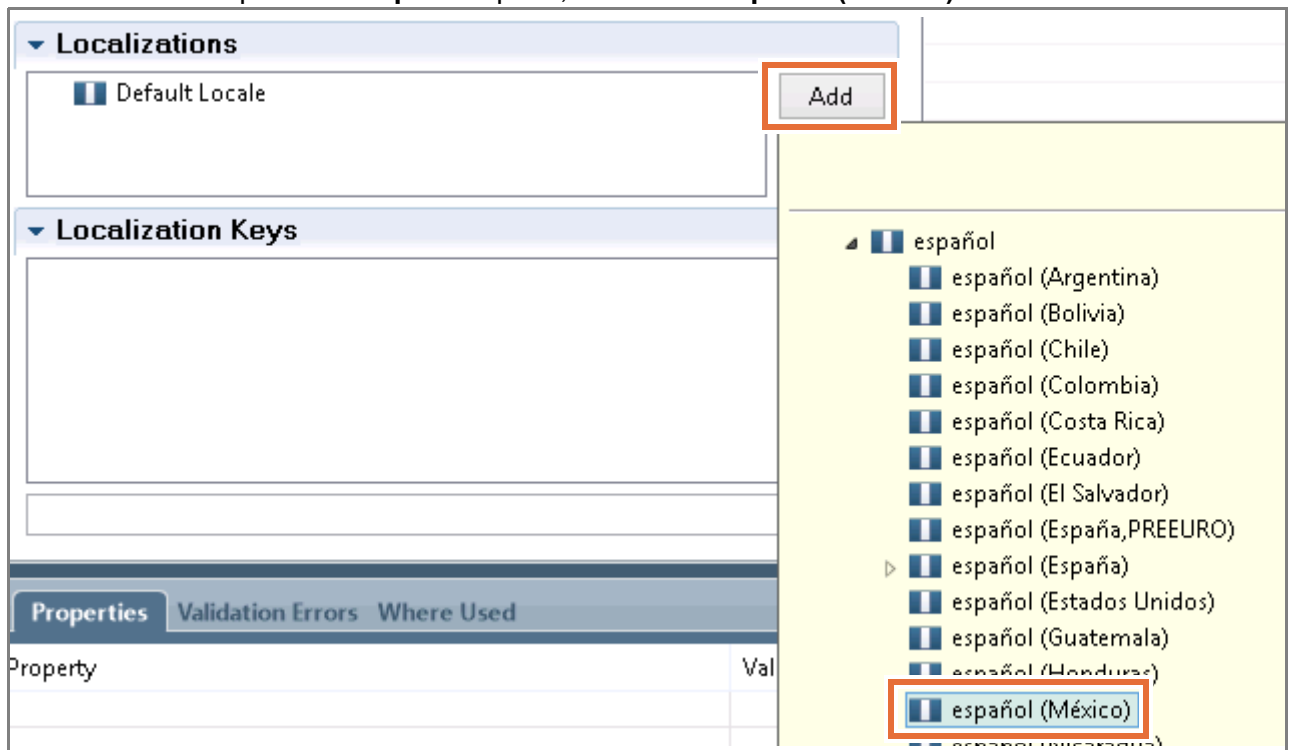
The screenshot shows a Windows-style dialog box titled "New Localization Resource". Inside the dialog, there is a subtitle "Creating a new Localization Resource". Below this, there is a "Name:" label followed by a text input field containing the text "CompanyLocalization". At the bottom right of the dialog, there are two buttons: "Finish" and "Cancel". The "Finish" button is highlighted with a red rectangular border.



### Information

If you plan to use this localization resource inside this process application or site-wide, name the localization resource with appropriate naming conventions.

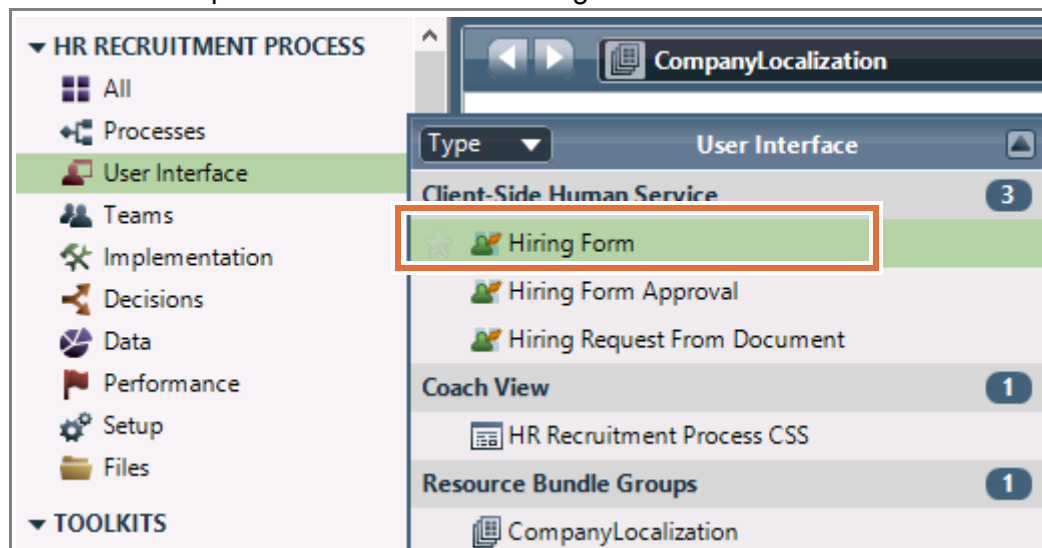
- \_\_\_ 3. Add a localization by clicking **Add** in the Localizations section, and clicking a language of choice. Expand the **español** option, and select **español (México)**.



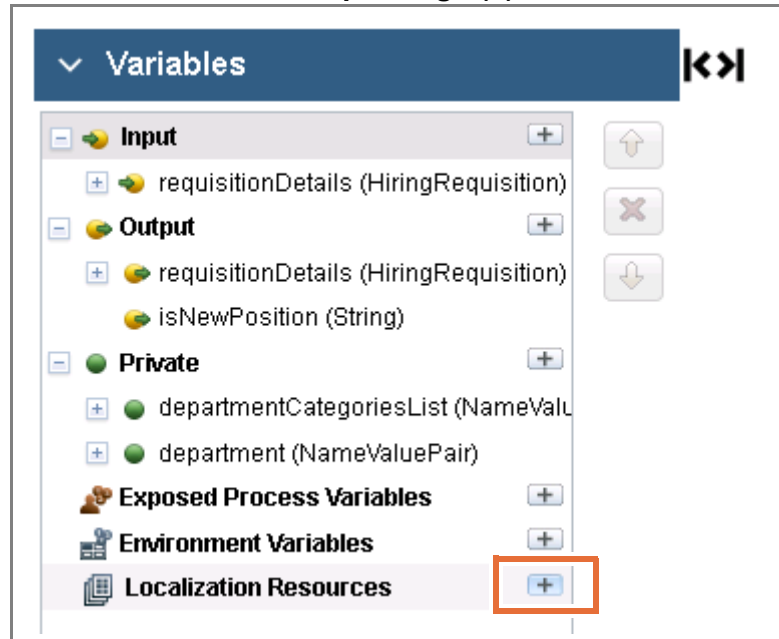
- \_\_\_ 4. Save your work.

## 1.2. Bind the localization to a coach or coach view

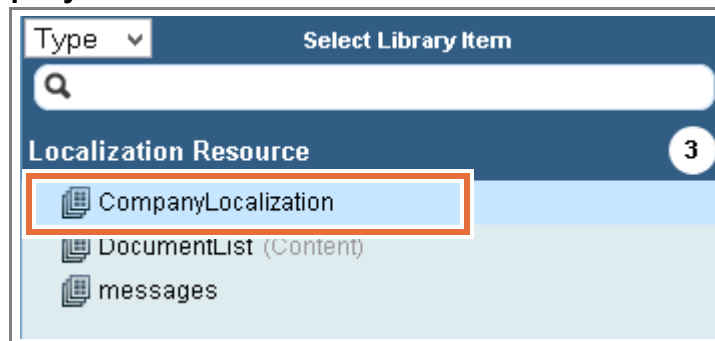
- \_\_\_ 1. Bind the localization to the coach view.
- \_\_\_ a. In the library, double-click the **User Interface > Hiring Form** client-side human service. The service opens in a web Process Designer tab.



- \_\_\_ b. Click the **Variables** tab. Click the **plus sign (+)** next to **Localization Resources**.



- \_\_\_ c. Click **CompanyLocalization** to add the localization resource bundle to the coach view.



- \_\_\_ d. Save your work.
- \_\_\_ e. Click the **Coaches** tab, and click the **Hiring Form** coach. Browse through the tabs and view all the input fields on the coach.

- \_\_\_ f. All the input fields contain labels that must be translated to a different language when the user designates a different locale as part of their profile. You create the key-value pairs next.

The screenshot shows the 'Hiring Form' interface. The 'Coaches' tab is selected, and the 'Requisition Number' label is highlighted with a red box. The interface includes a sidebar with icons for Overview, Diagram, Variables, and Coaches. The main area shows the 'Requisition Details' section with input fields for 'Requisition Number' and 'Requestor'.

### 1.3. Create resource keys

- \_\_\_ 1. Create the resource keys for the coach view.
- \_\_\_ a. Return to the **CompanyLocalization** resource in the Process Designer client application.
- \_\_\_ b. Ensure that the **Spanish (Mexico)** localization is selected under the Localizations section. Under the Localization Keys section, type `RequisitionNumber` and click **Add**.

The screenshot shows the 'Localization Keys' section. The 'Spanish (Mexico)' localization is selected. The 'RequisitionNumber' key is entered in the input field, and the 'Add' button is highlighted with a red box.



### Important

Localization keys cannot contain spaces.

- \_\_\_ c. Under the Localization Values section on the right, click the **Default Locale** row and enter: Requisition Number
- \_\_\_ d. Press Ctrl+Enter to move to the next key.
- \_\_\_ e. Enter the translated key Requisición Número into the second entry, and press Alt+Enter.



### Note

If necessary, use an Internet translation service to assist you in this exercise. Two sites that you can use are Babelfish (<http://www.babelfish.com>) or Google Translate (<http://translate.google.com>). Translation websites are helpful for this exercise, but use caution when using them for your company development efforts.

Also, some characters might cause problems with IBM Business Process Manager or your browser, so test your code thoroughly.

- \_\_\_ f. Save your work.



### Important

You must save your work after you create keys to make them visible for selection on a coach or coach view.

Localization Values	
Mode:	Show All Keys
Key	Value
Default Locale	Requisition Number
Spanish (Mexico)	Requisición Número

## 1.4. Assign keys to the labels on the coach

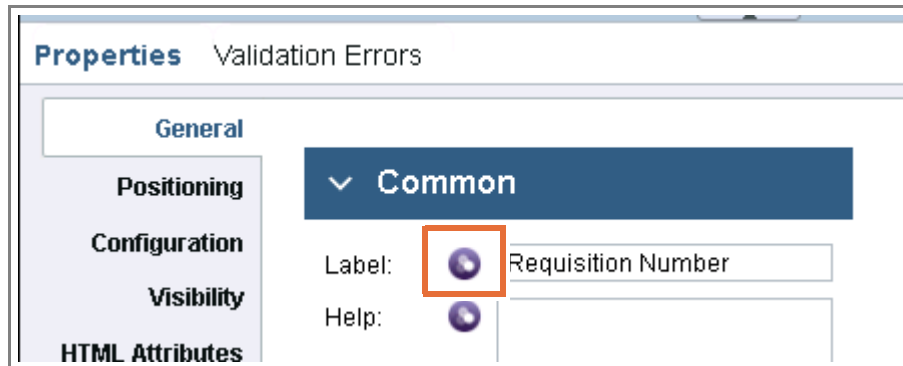
- \_\_\_ 1. Localize the coach view.
  - \_\_\_ a. If it is not already open, return to the web Process Designer and open the **Hiring Form**.
  - \_\_\_ b. Click the **Requisition Details** tab on the canvas, and then click the **Requisition Number** input control.

The screenshot shows the Hiring Form canvas with the following elements:

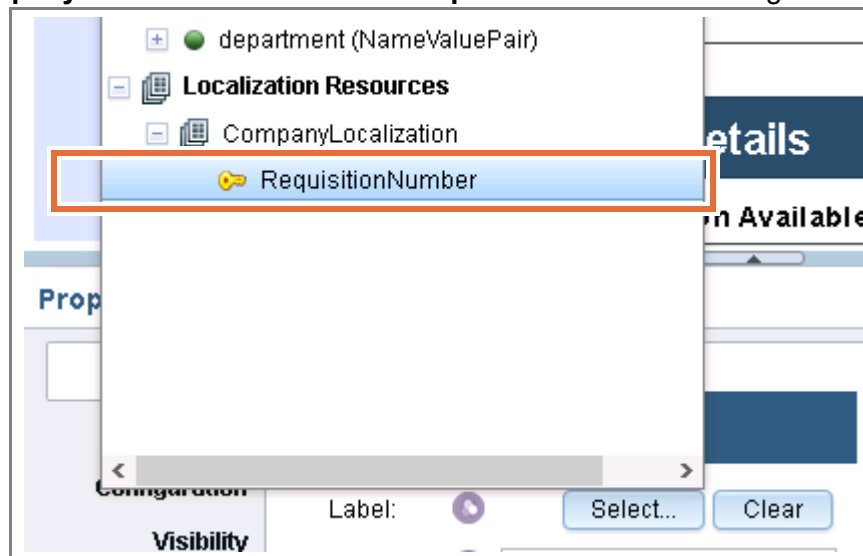
- At the top, there are four tabs: **Requisition Details** (highlighted with an orange box), **Position Details**, **Recruiting Details**, and **Compensation Details**.
- Below the tabs, there is a **Department Details** section with a green plus icon.
- The main content area contains the **Requisition Number** input control, which is highlighted with an orange box.
- Below the Requisition Number input, there is a **Requestor** label and an empty input field.



- \_\_\_ c. In the **Properties > General** menu, click the icon next to **Label** to assign a variable to this field.

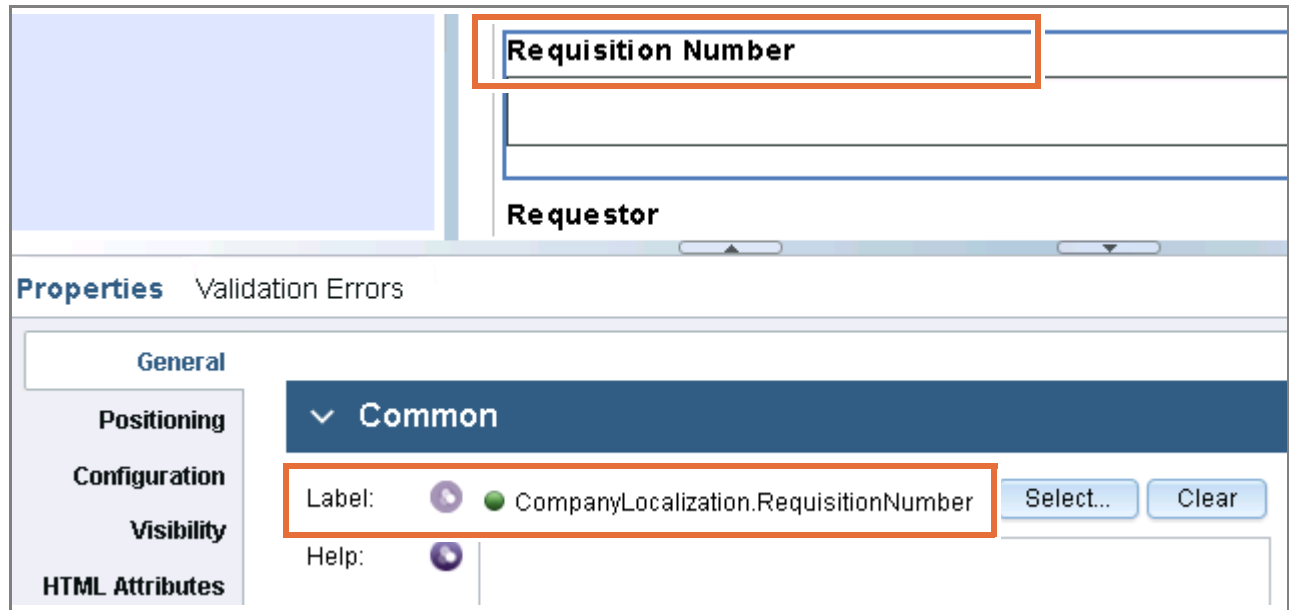


- \_\_\_ d. Click **Select** next to the icon, and expand the **Localization Resources** and **CompanyLocalization**. Select the **RequisitionNumber** to assign the key to this label.



**Note**

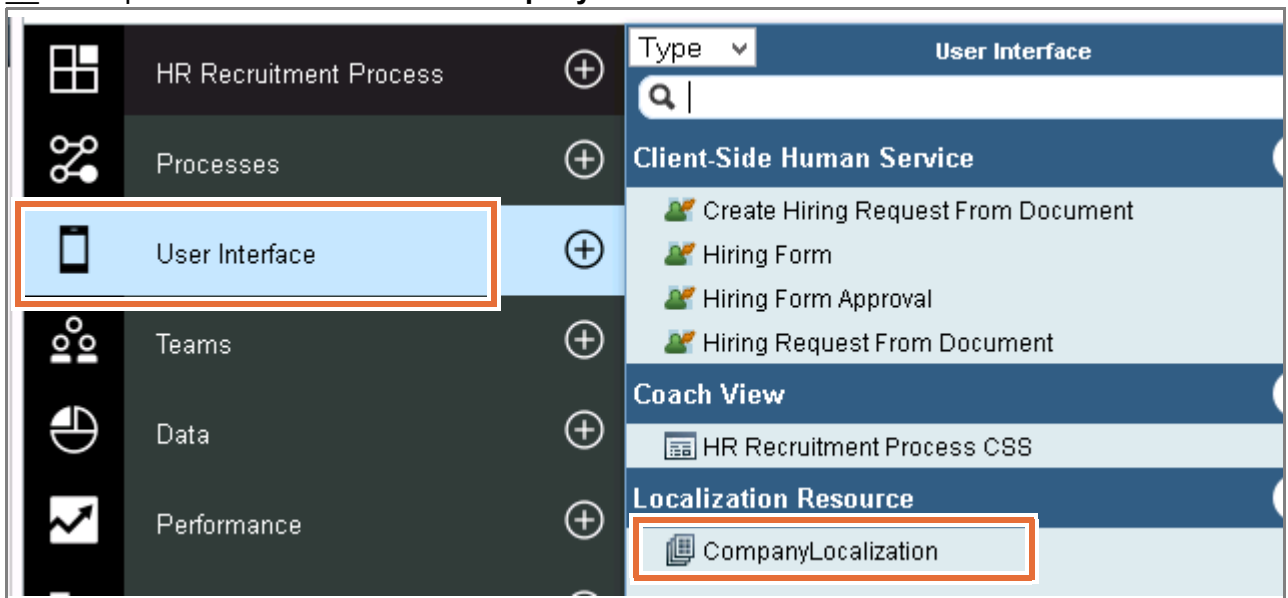
The control label on the canvas shows the value of the variable, but the label is bound to the localization resource.



\_\_ e. Click **Save**.

## 1.5. Import resource keys

- \_\_ 1. Switch back to the **web Process Designer**.
- \_\_ 2. Open the **User Interface > CompanyLocalization** localization resource.



**Hint**

You can create the key-value pairs by using the web-based company localization resource interface. You can also import and export the localization bundles. You must designate new translation languages/locales by using the Process Designer client application because the web Process Designer does not support adding new locales.

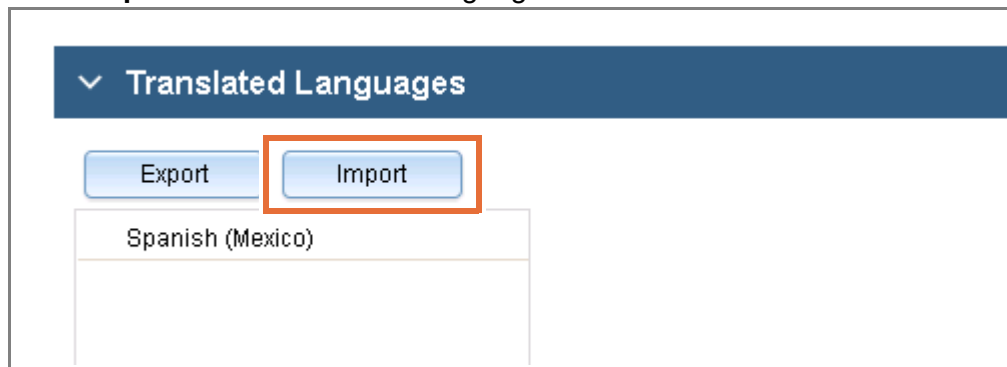
\_\_ 3. Create the localization keys and values for all the fields on the coach.

**Optional**

You can continue to manually create the key-value pairs by following steps 1.3.1.a - 1.3.1.d. After you complete the CompanyLocalization localization resource data, skip to step 1.5.3. If you don't want to manually create all the key-value pairs, the next steps take you through importing a localization bundle, which loads all the key-value pairs for this coach.

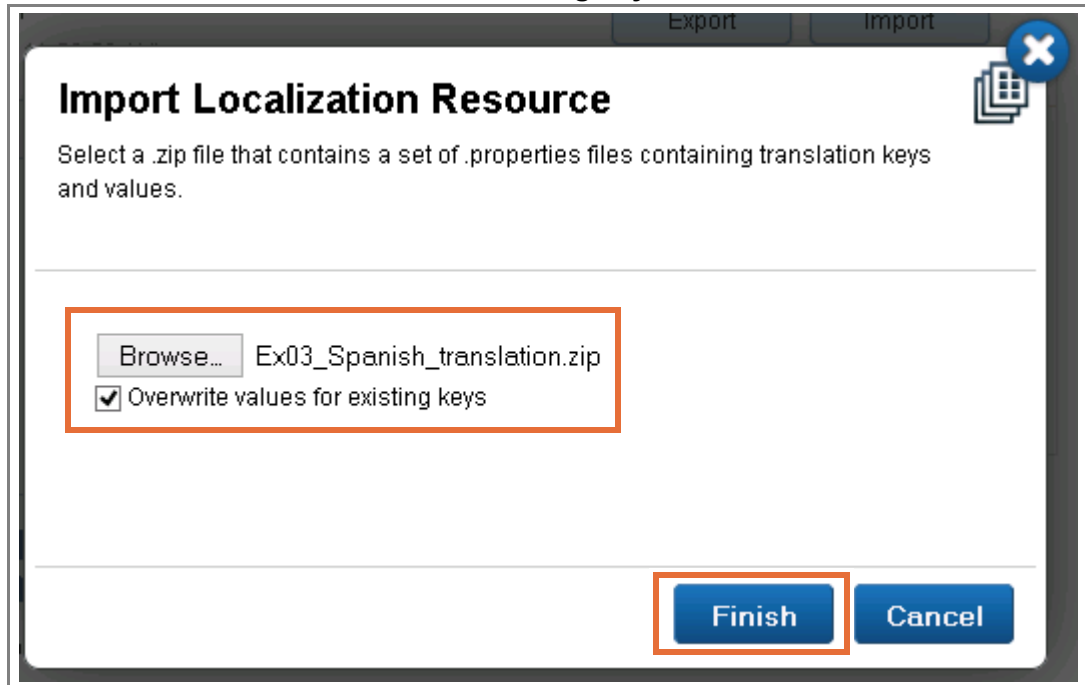
A file is provided for you that contains all of the Spanish translation for this coach. Examine the C:\labfiles\Lab and Exercise Assets\Exercise Code\Exercise 3\Ex03\_Spanish\_translation.zip file. This compressed file contains two files: CompanyLocalization.properties and CompanyLocalization\_es\_MX.properties. Open the files with Notepad, and examine the contents.

\_\_ a. Click **Import** in the Translated Languages section.

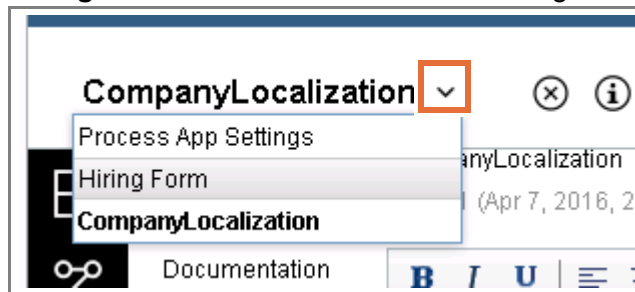


\_\_ b. Click **Browse** to select C:\labfiles\Lab and Exercise Assets\Exercise Code\Exercise 3\Ex03\_Spanish\_translation.zip and click **Open**.

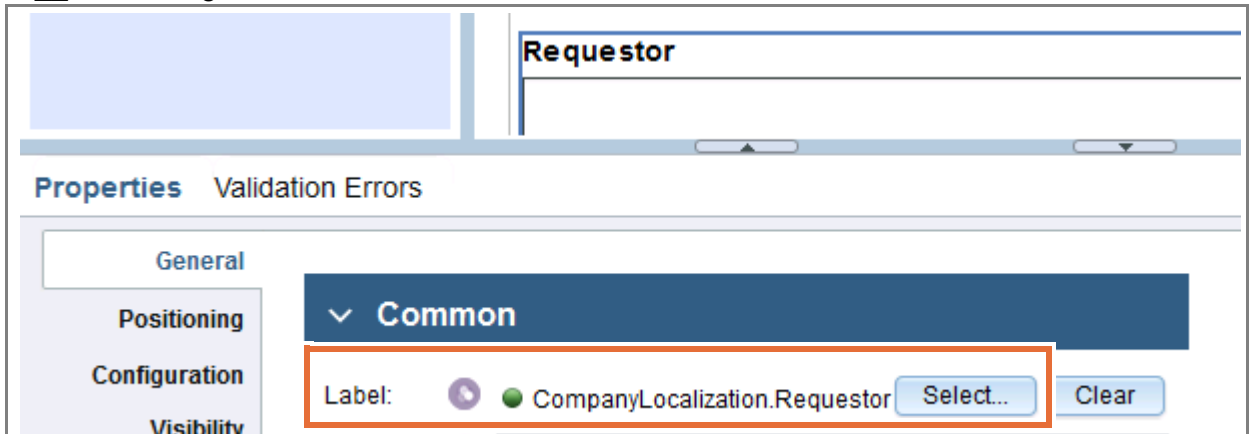
- c. Select the **Overwrite values for existing keys** check box.



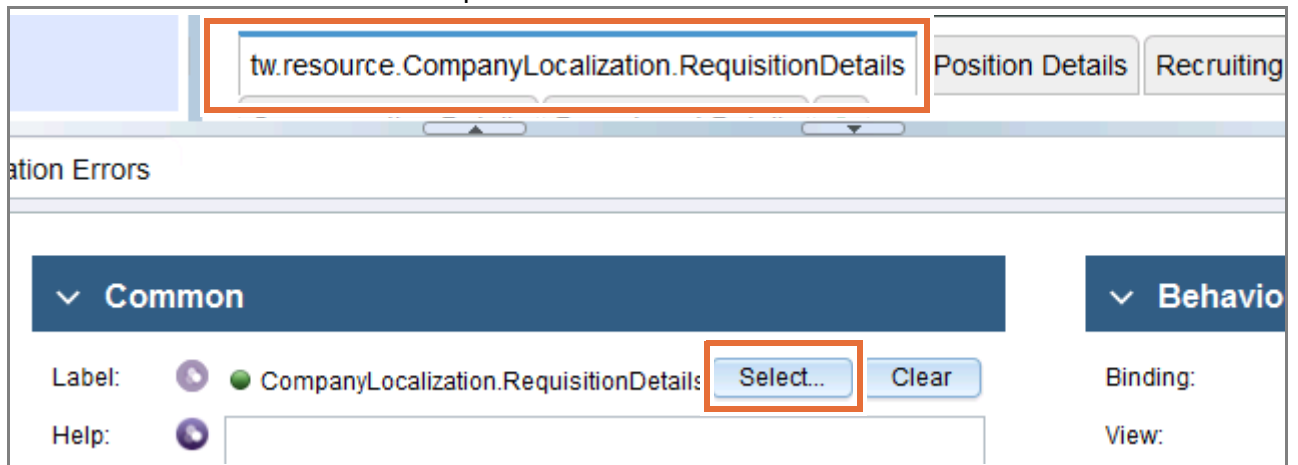
- d. Click **Finish**, and the import is completed successfully when the keys are shown in the section that is labeled Translation Keys and Default Values.
- e. Save your work.
4. Localize the remaining elements on the coach.
- a. Return to the **Hiring Form** inside the web Process Designer.



- \_\_\_ b. Change the label for all the tabs and the coach elements in the tabs.



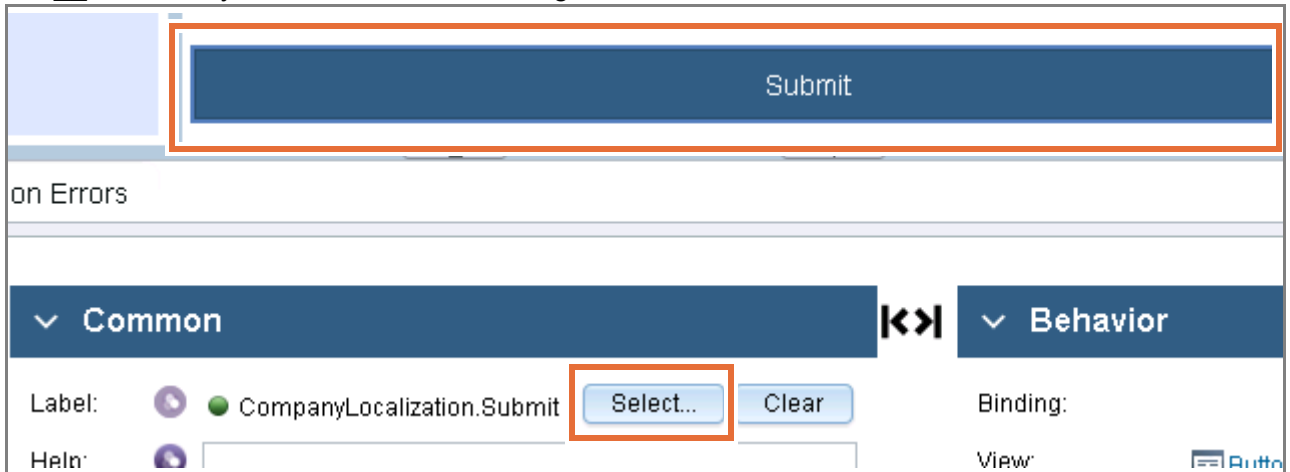
- \_\_\_ c. Localize all tab labels. You can change the tab label by clicking the tab and changing the label to the localization key. The tab name changes to the localization variable name, and the header at the top of the tab shows the localized label for the tab.



- \_\_\_ d. Also localize the table headers, which include the Date Details table header label on the first tab.



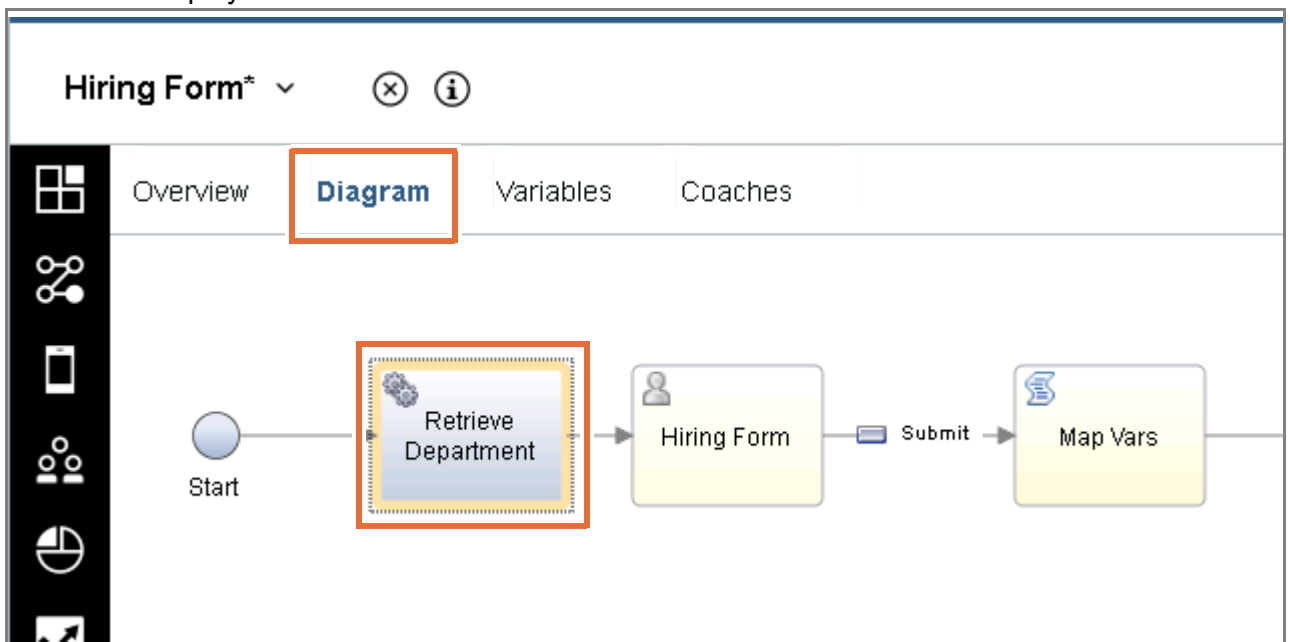
- \_\_\_ e. Finally, click **Submit**. And change the label for this button to the localization resource.



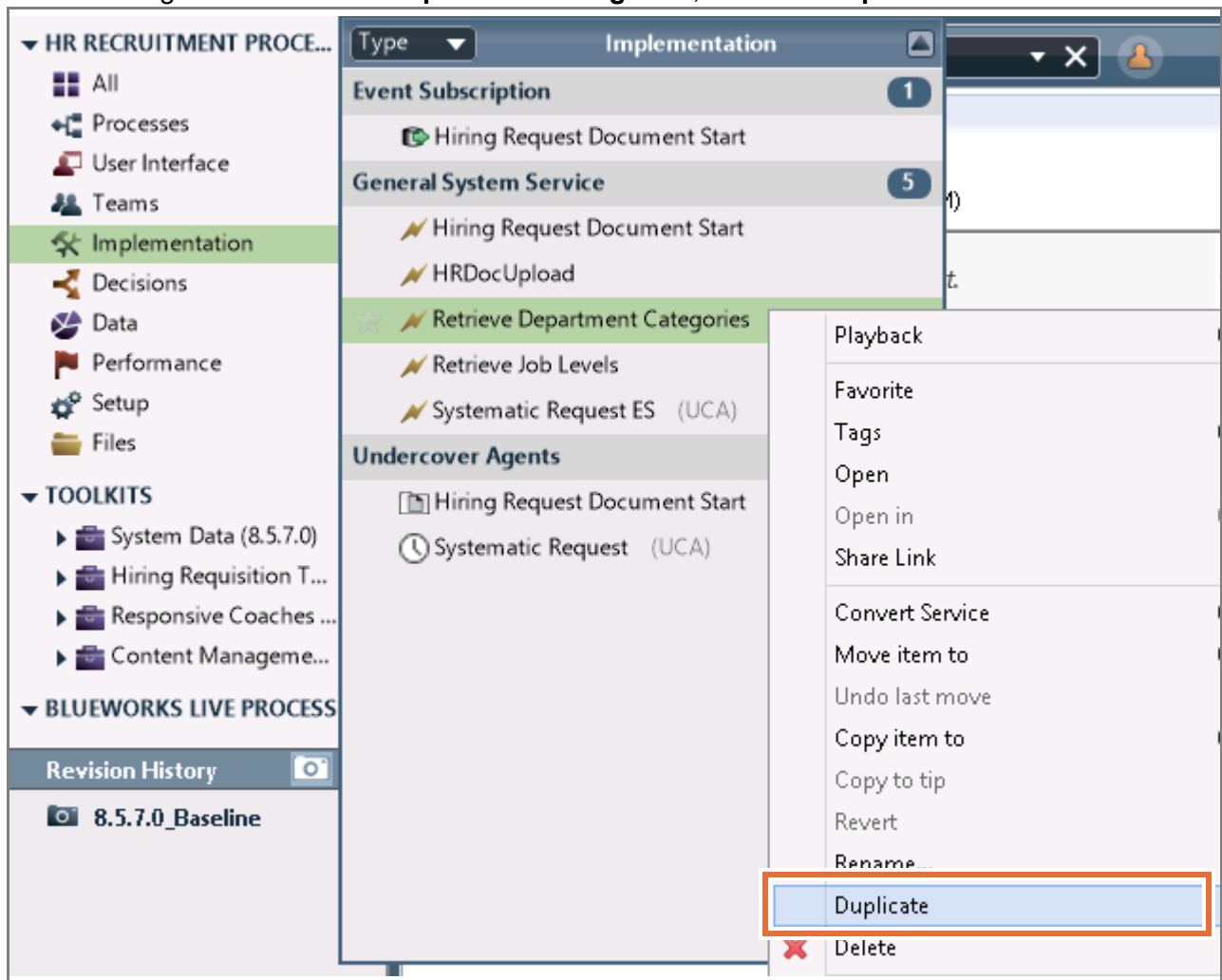
- \_\_\_ f. Most of the coach is now localized. The labels are localized, but on the **Department Details** tab, the department list is a service that queries a database and returns a list of strings. Those items must be localized as well. The following section tests the localization and adds the ability to localize the select menu. Save your work.

## 1.6. Localize the department menu

- \_\_\_ 1. Use a script to traverse through the SQL result and localize the options on the select menu.
- \_\_\_ a. Open the **Diagram** tab for the Hiring Form client-side human service. The Retrieve Department service queries an SQL database and returns a list of departments to be displayed on the coach.

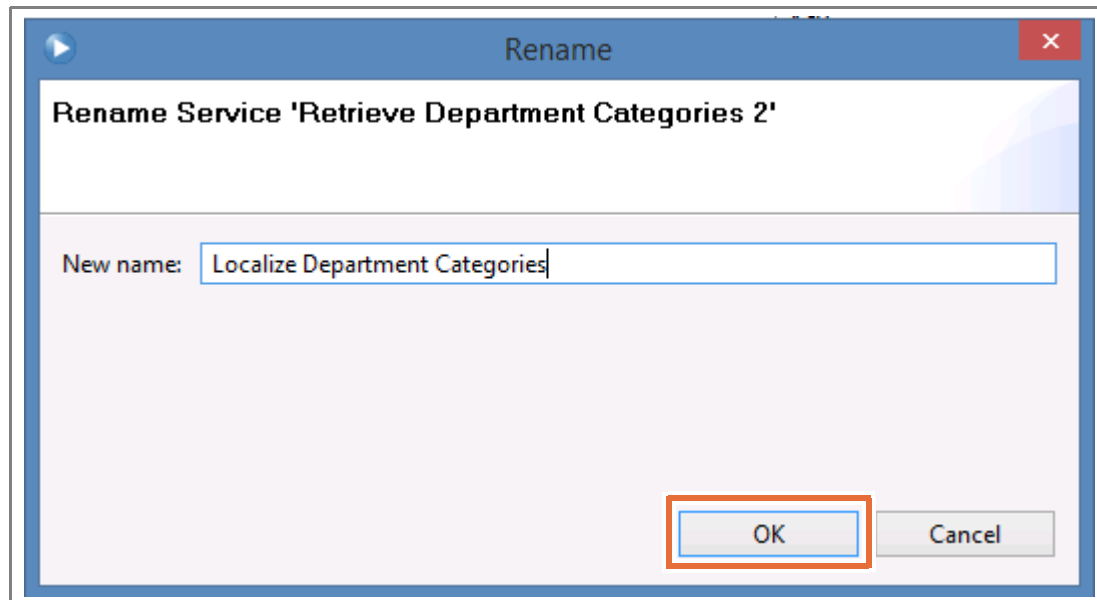


- \_\_\_ b. Return to the Process Designer client application. Click the **Implementation** menu, right-click **Retrieve Department Categories**, and click **Duplicate**.

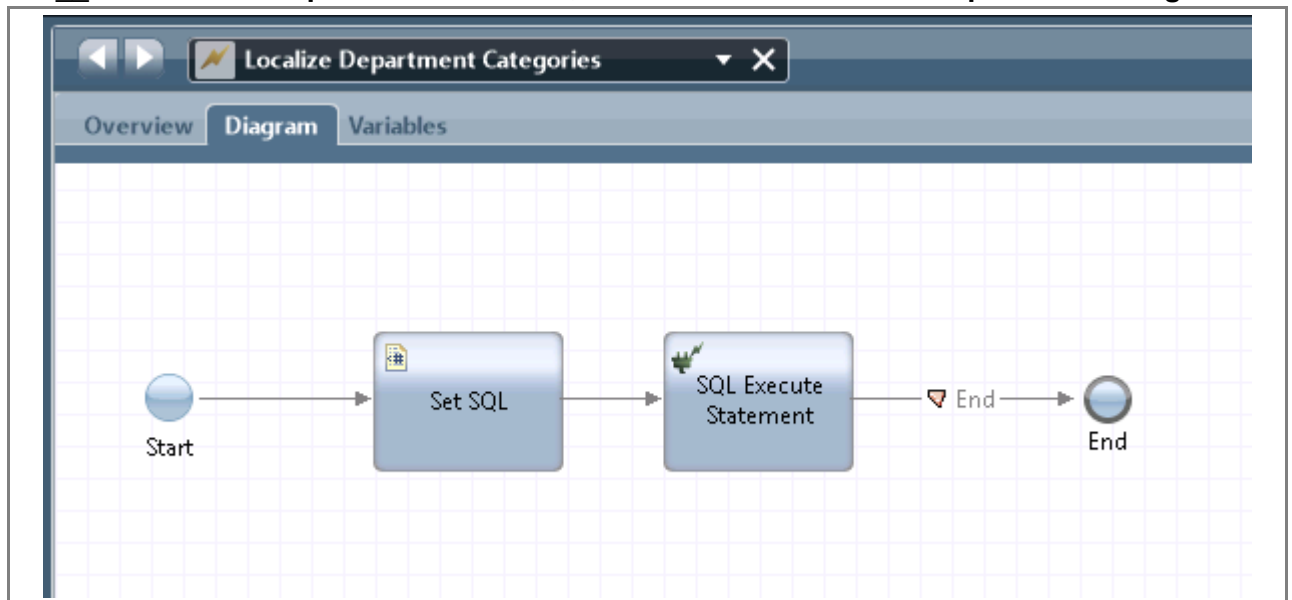


- \_\_\_ c. Right-click the cloned asset in the **Implementation** category that is named **Retrieve Department Categories 2** and click **Rename**.

- \_\_\_ d. Enter `Localize Department Categories` as the new name.
- \_\_\_ e. Click **OK**.



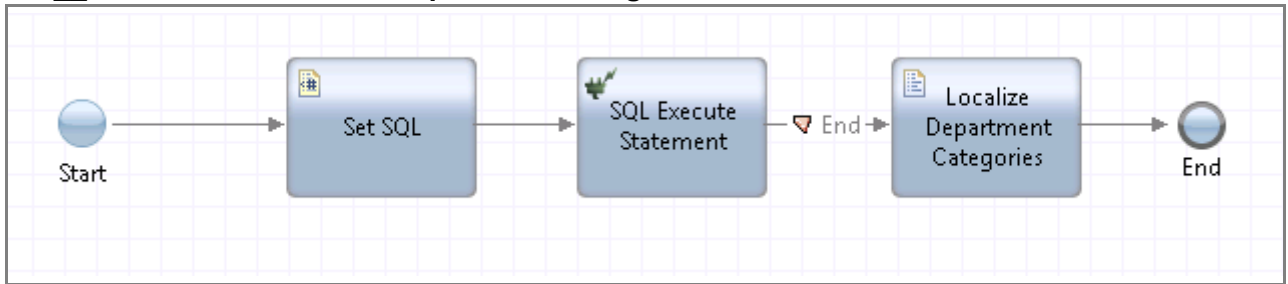
- \_\_\_ f. Click the **Implementation** menu, and double-click **Localize Department Categories**.



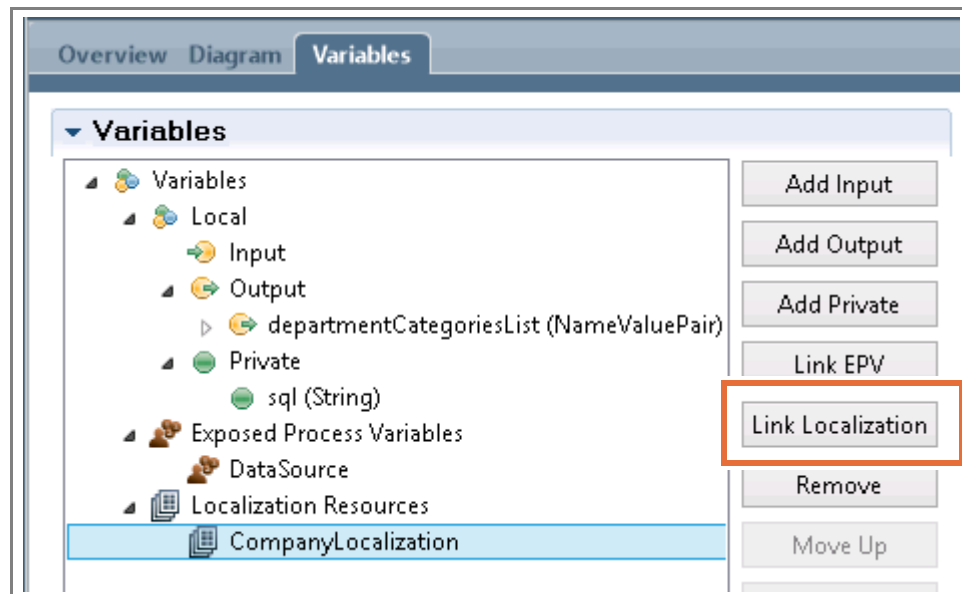
- \_\_\_ g. Drag a **Server Script** from the palette to the right of **SQL Execute Statement** on the canvas, and name it: `Localize Department Categories`
- \_\_\_ h. Connect the flow **SQL Execute Statement** to **Localize Department Categories**.



- \_\_\_ i. Connect **Localize Department Categories** to the **End**.



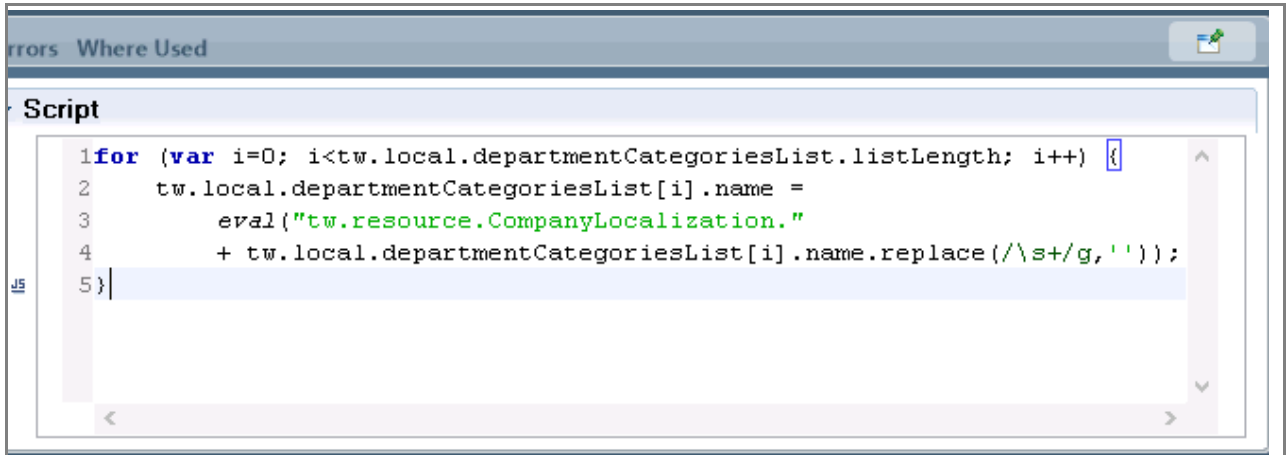
- \_\_\_ j. Click the **Variables** tab.
- \_\_\_ k. Click **Link Localization** and select **CompanyLocalization** to link to the service variables as a resource bundle.



- \_\_\_ l. Return to the **Diagram** tab. Select **Localize Department Categories**, and click the **Properties > Implementation** menu.

- \_\_\_ m. Insert the following code into the code block. The following script produces an error in the script block, but the code is valid, so ignore the error. The example that is shown is saved to the C:\labfiles\Lab and Exercise Assets\Exercise code\Exercise 3\Localize Department Categories.txt file:

```
for (var i=0; i<tw.local.departmentCategoriesList.length; i++) {
    tw.local.departmentCategoriesList[i].name =
        eval("tw.resource.CompanyLocalization."
            + tw.local.departmentCategoriesList[i].name.replace(/\s+/g, ''));
}
```

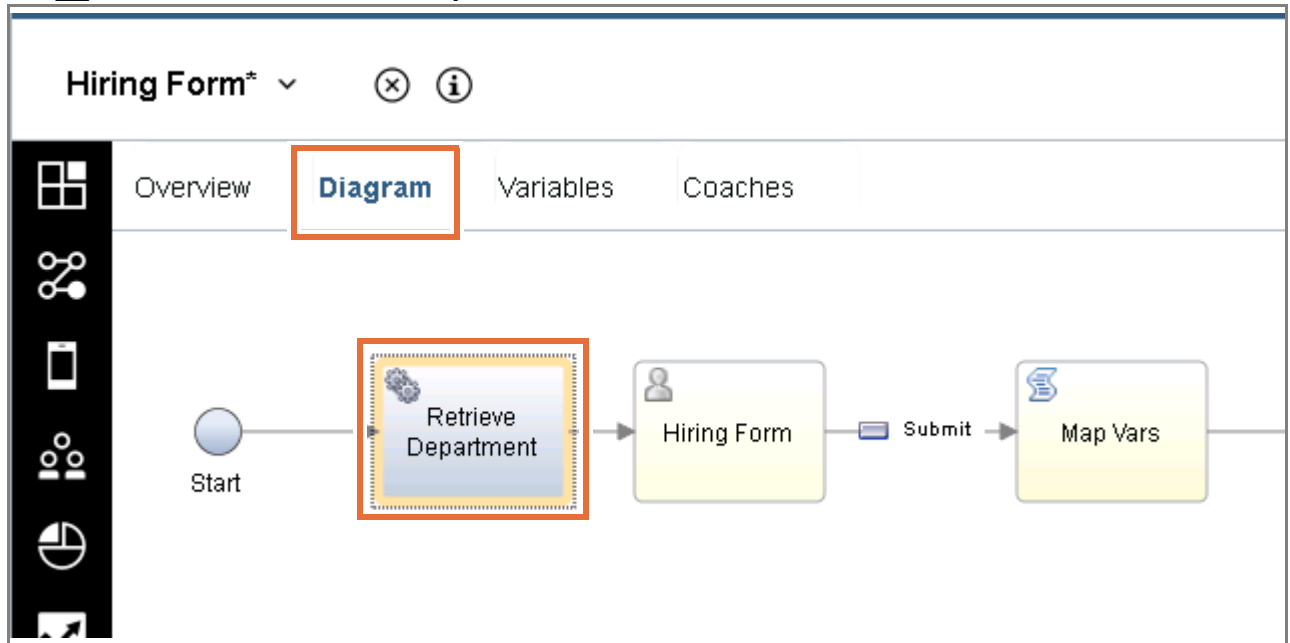


### Information

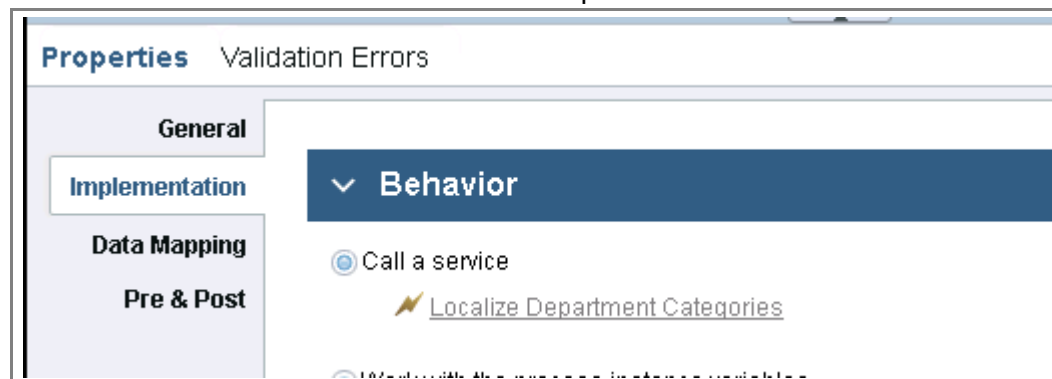
The replace function removes the space in the *Professional Services* selection field on the coach. The eval() function evaluates the value of the variable name that is built by concatenating the strings. The script steps through all of the items in the departmentCategoriesList and replaces the name with the localized version of it. In your own code for your organization, you can expand the regular expression that the replace function uses to format all the entries in the database to be transformed into the localization keys used.

- \_\_\_ n. Save your work.
- \_\_\_ 2. Localize a menu.
- \_\_\_ a. Return to the web Process Designer.
- \_\_\_ b. Open the **User Interface > Localization Resource > CompanyLocalization** localization resource from the library or from the history menu.
- \_\_\_ c. The translation keys and values that you now see were created when you imported the localization file earlier in this exercise. All of the items that are retrieved in the database tables are part of the localization bundle keys you imported. The values in the database populate the select menus on the coach. Return to the **Hiring Form > Diagram** tab in the web coach Web Designer.

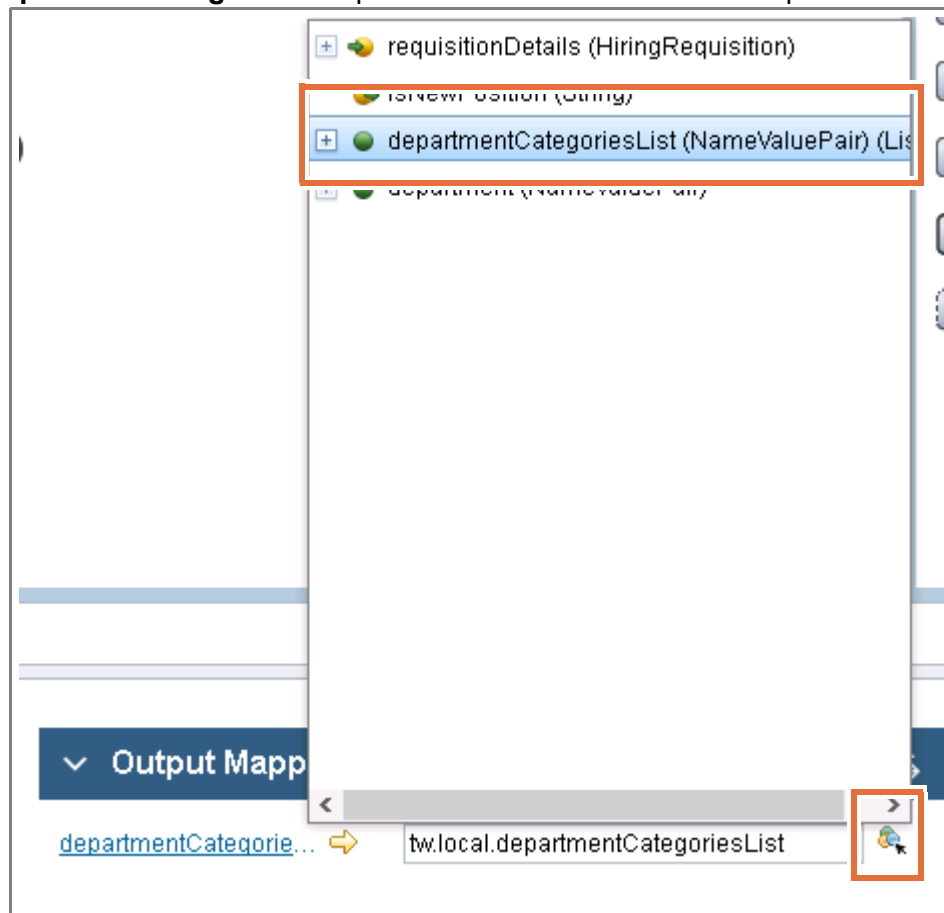
- \_\_\_ d. Select the **Retrieve Department** service.



- \_\_\_ e. Click the **Properties > Implementation** menu.
- \_\_\_ f. In the Behavior section, click **Select** next to Call a service, and select **Localize Department Categories** under the General System Service category. The service now uses the new service that localizes the department names.



- \_\_\_ g. Click the **Data Mapping** menu, and in the Output Mapping section, map the **departmentCategoriesList** private variable to the service output.



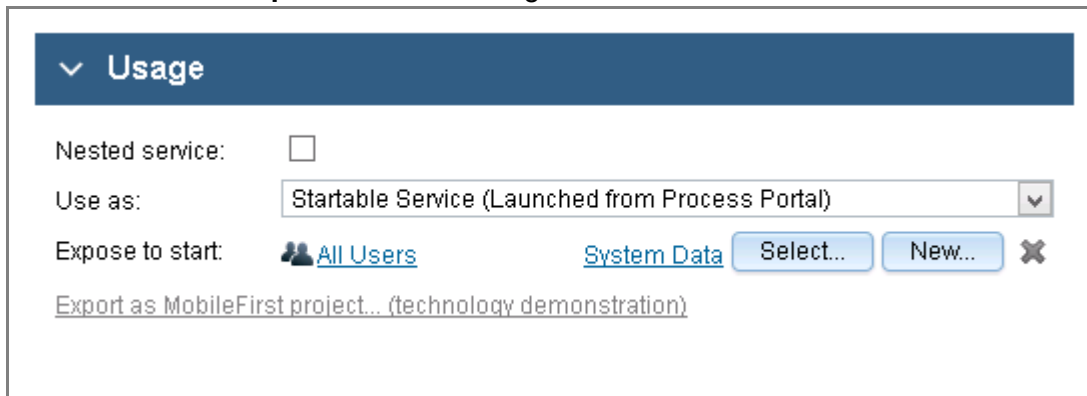
- \_\_\_ h. Save your work.

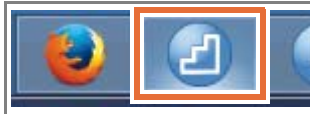
## Section 2. Test the localization

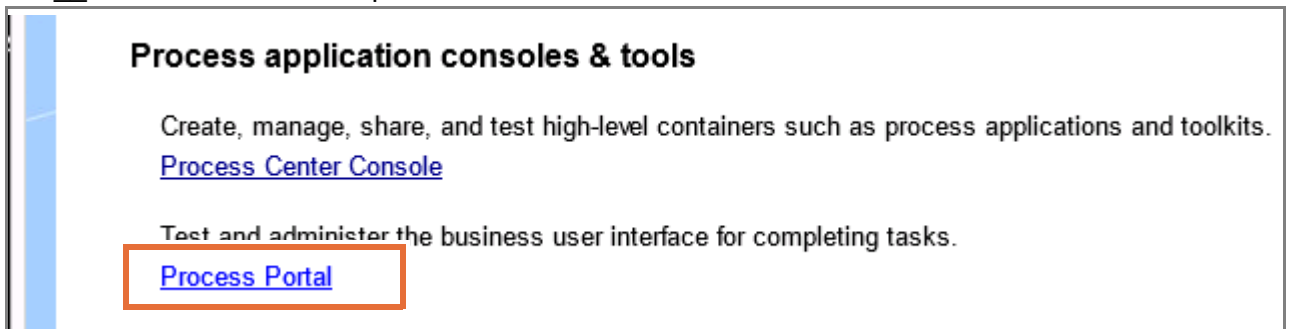
Now that the coach is localized, you need a way to test the results.

### 2.1. Change the localization preference in the portal

- \_\_\_ 1. After the localization menu is complete and tested, you can simulate a user's experience by changing the preference in the portal.
  - \_\_\_ a. If it is not already open, open the **Hiring Form** client-side human service in the web Process Designer.
- \_\_\_ 2. For testing purposes, expose the Hiring Form to all users.
  - \_\_\_ a. Click the **Overview** tab.
  - \_\_\_ b. In the Usage section, select **Startable Service** as the **Use as** setting. Verify that **All Users** for the **Expose to start** setting is selected.

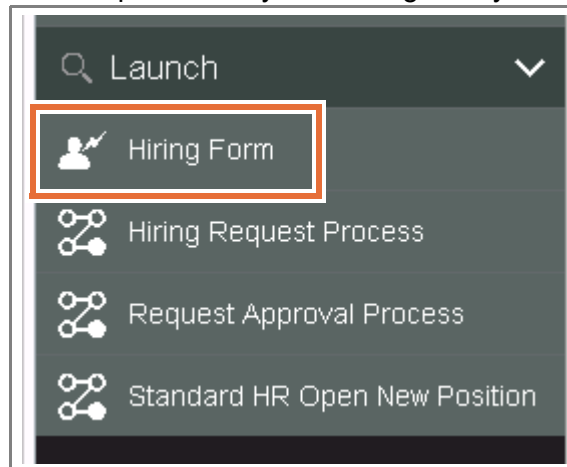


- \_\_\_ c. Save your work.
- \_\_\_ 3. Test the localization inside the portal.
  - \_\_\_ a. Open the **Quick Start** menu.
 
  - \_\_\_ b. Click the link to open the **Process Portal**.



- \_\_\_ c. If prompted, log in to the Process Portal with `author1` as the user ID and `author01` as the password.

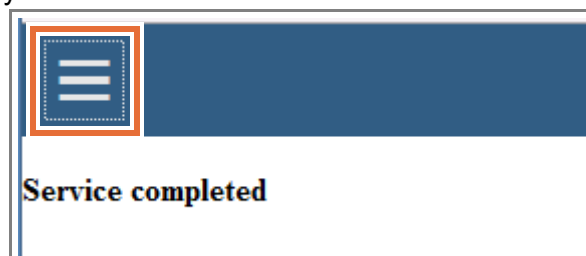
- \_\_\_ d. Click the **Hiring Form** link on the left under the Launch category to start the service. The interface and available options that you see might vary from this screen capture.



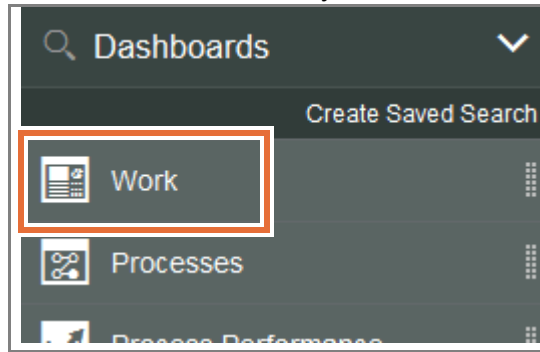
- \_\_\_ e. The coach is shown in the default locale language.

Requisition Details	Position Details	Recruiting Details	Compensation Details	Department Details
<b>Requisition number</b> <input type="text"/>				
<b>Requestor</b> <input type="text"/>				
<b>Date Details</b>				
<b>Date position becomes available</b> <b>Date of request</b> <div> <input type="text"/> <input type="text"/> </div>				
<b>Hiring manager comments</b> <input type="text"/>				
<div>Submit</div>				

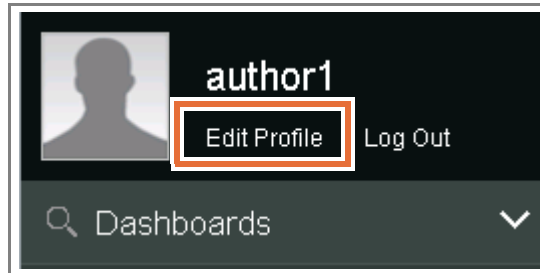
- \_\_\_ f. Click the **Submit** button to complete the service.
- \_\_\_ g. Open the library on the left.



- \_\_\_ h. Click **Work** to return to the list of tasks in your inbox.

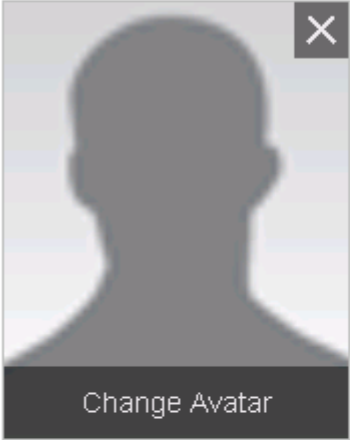


- \_\_\_ i. In the upper-left corner, click the **Edit Profile** link under the user name.



- \_\_\_ j. Select the language of choice. If you followed these instructions exclusively, select **español**.

author1



**Job Title**

**Phone Number**

**Email Address**

**Portal Preferences**

**Language**

English

▼

**Calendar**

Français  
Deutsch  
Italiano  
中文  
中文 (台灣)  
**español**  
português  
Português (Brasil)  
日本語  
한국어  
...

**Open Task in a New**

☒ **Show confirmation**  
☐ **Send me an email**

- \_\_\_ k. Scroll down and click **Save**.
- \_\_\_ l. Click the **Hiring Form** link on the left.



\_\_\_ m. The hiring form is now localized to the selected language.

<b>Requisición número</b>	
<input type="text"/>	
<b>Solicitante</b>	
<input type="text"/>	
<b>Fecha Detalles</b>	
<b>Fecha posición esté disponible</b>	<b>Fecha de la petición</b>
<input type="text"/>	<input type="text"/>
	
<b>La contratación de los comentarios del gestor</b>	
<input type="text"/>	
<input type="text"/>	
<b>Siguiente</b>	

\_\_\_ n. Click the **Department** tab (translated). The select menu is also localized.

<b>Departamento</b>
<input type="text"/>
Financiar
Ingeniería
Mercadeo
Recursos Humanos
Servicios Profesionales

- \_\_\_ o. Select the **first option** and click the **Submit** button (translated). Click the **Work** link to return to the inbox.
- \_\_\_ p. Change the portal language back to your preferred language (**English** if you chose Spanish as the locale).
- \_\_\_ q. Close the Process Portal tab.

## End of exercise

## Exercise review and wrap-up

In the first part of the exercise, you created a localization resource, created localization keys and values, and replaced all of the labels on the coach. You then tested the locale through code and through the portal.

---

# Exercise 4. Implementing the "four eyes" policy

## Estimated time

01:00

## Overview

In this exercise, you learn how to implement a “four eyes” policy in a process by using a team filter.

## Objectives

After completing this exercise, you should be able to:

- Implement the "four eyes" policy by using a team filter

## Introduction

The company chief executive officer (CEO) is concerned about the activity in the HR Administrator lane during the approval linked process. The CEO wants to make sure that if an override is required, two people must look at the request, enabling a “four eyes” policy for activities in that lane. You must create a second activity in the lane, and use a routing policy to ensure that the user who completed the first activity in a lane does not claim the second activity in the lane.

## Requirements

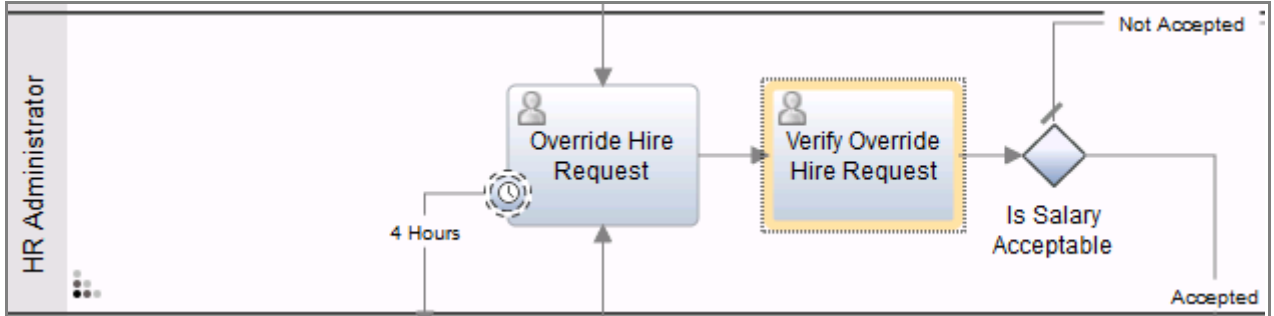
Completion of the previous exercise in this book.

## Section 1. Create the four eyes policy

The first steps of this exercise are to change the process to meet the new requirements of the organization. After the process is changed, you will then implement the four eyes policy.

### 1.1. Modify the process

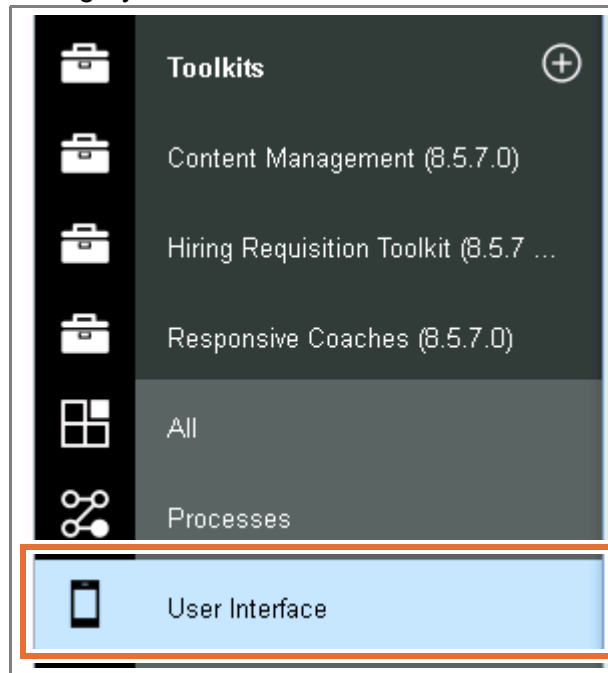
- \_\_\_ 1. Create the verify activity to make sure that if an override is required, two people must look at the request, enabling a "four eyes" policy for activities in that lane.
  - \_\_\_ a. Open the **Processes > Approve Hire Request** linked process in the web Process Designer.
  - \_\_\_ b. Drag an activity to the right of the Override Hire Request activity. Name the second activity: Verify Override Hire Request
  - \_\_\_ c. Connect **Override Hire Request** to **Verify Override Hire Request**.
  - \_\_\_ d. Connect **Verify Override Hire Request** to **Is Salary Acceptable**.



- \_\_\_ 2. Create a service to capture the user name during the Override Hire Request step.
  - \_\_\_ a. Open the **library** inside the web Process Designer.



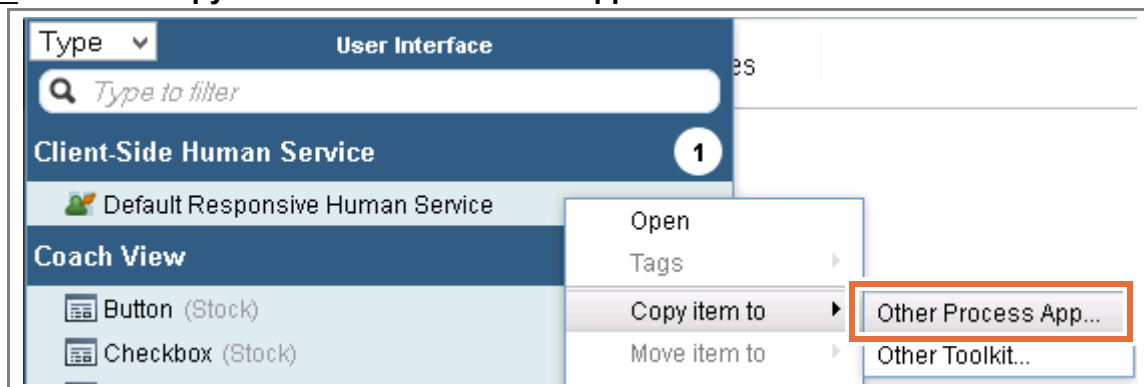
- \_\_\_ b. Expand the **Toolkits** category in the library, and then click the **Responsive Coaches > User Interface** category.



### Information

The Responsive Coaches toolkit contains all the coach views and supporting assets to enable responsive coaches inside the Coach Designer.

- \_\_\_ c. Under the Client-Side Human Service category, right-click the **Default Responsive Human Service**.
- \_\_\_ d. Click **Copy item to > Other Process App**.



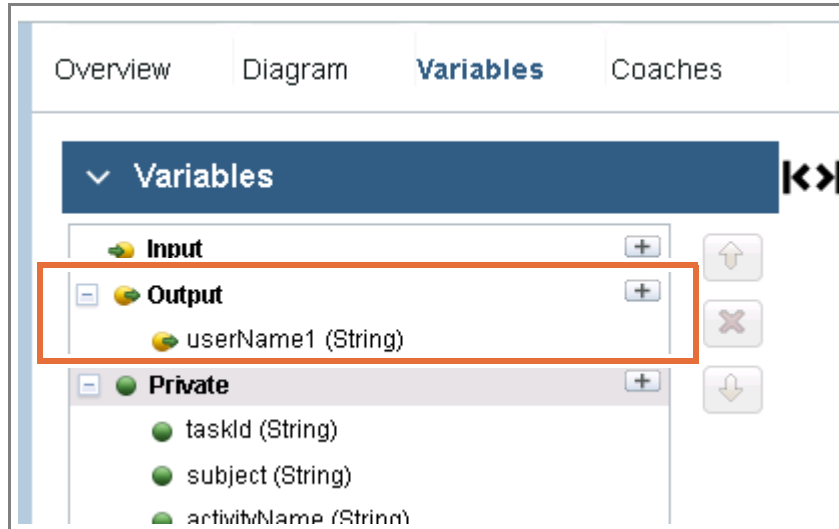
- \_\_\_ e. Click the **HR Recruitment Process** process application.



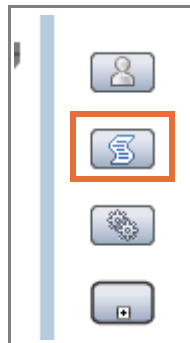
### Attention

If you imported a solution or turned on tracks, select the track that you are currently working on.

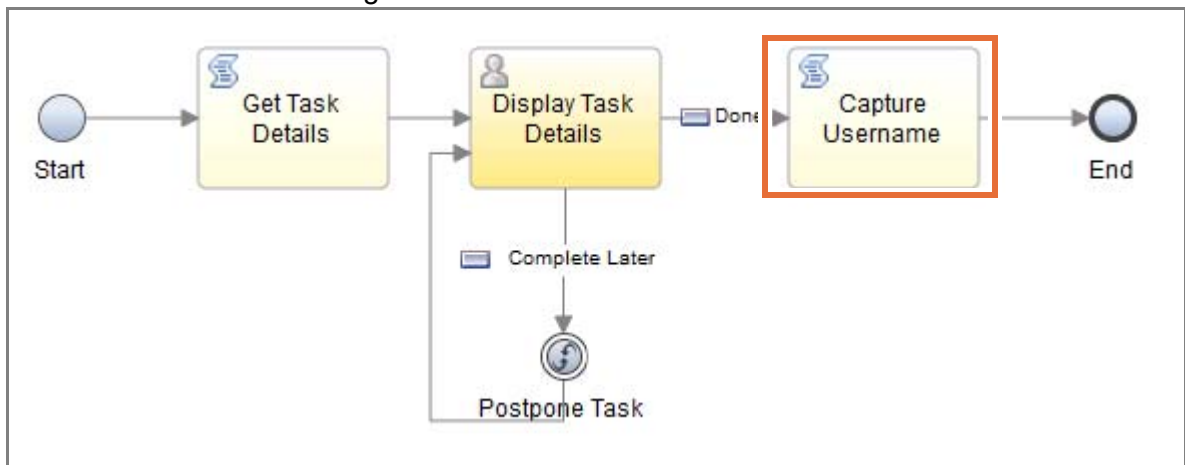
- \_\_\_ f. Click the **User Interface** menu in the library, and rename the **Default Responsive Human Service** to: **Override Hire Request**
- \_\_\_ g. Open the **Override Hire Request** human service.
- \_\_\_ h. On the **Variables** tab, add an output variable `userName1` (String) to the **Override Hire Request** service to capture the user name of the user.



- \_\_\_ i. On the **Diagram** tab, add a client-side script that is named **Capture Username** to the canvas after the **Display Task Details** coach.



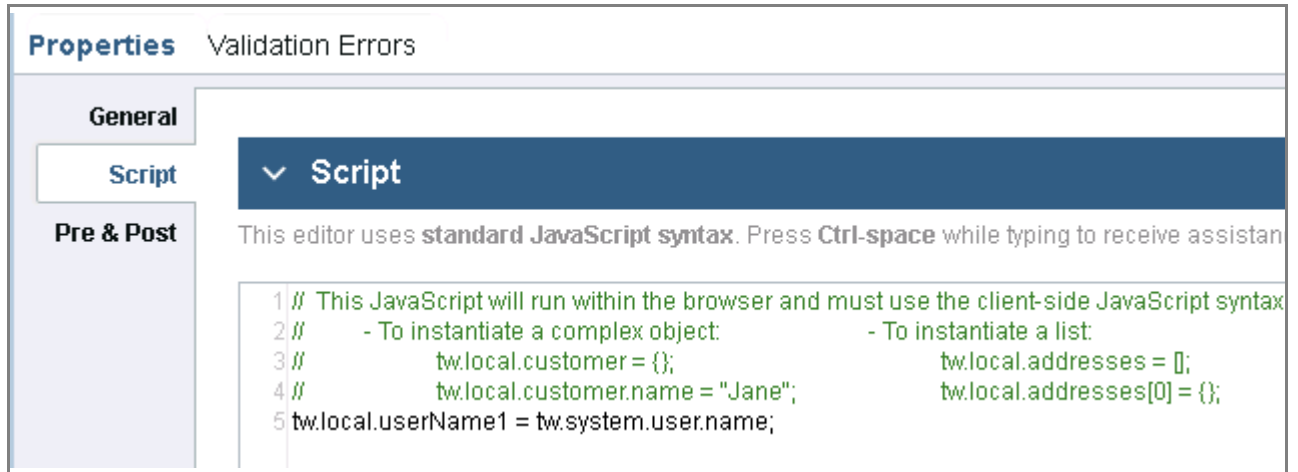
- \_\_\_ j. Connect **Display Task Details** to **Capture Username** and **Capture Username** to the **End** event in the diagram.



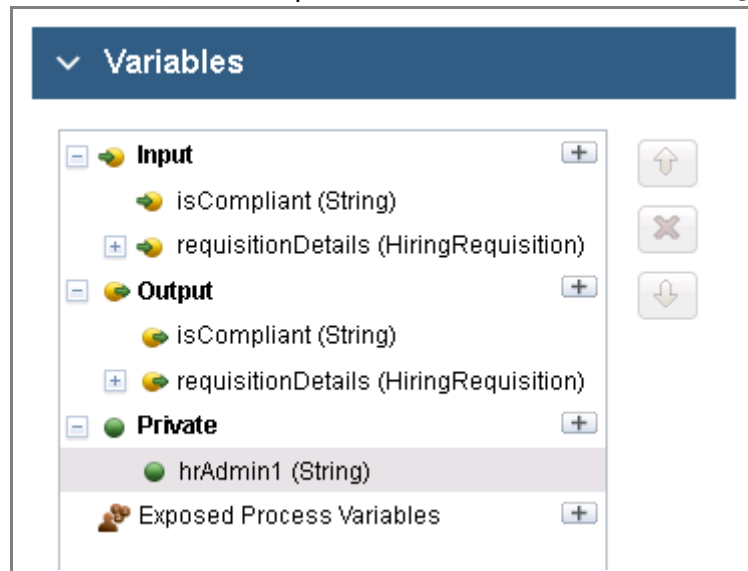
- \_\_\_ k. Click the **Capture Username** step.

- \_\_\_ l. Enter the code in the **Properties > Script** menu to capture the username in the `userName1` variable:

```
tw.local.userName1 = tw.system.user.name;
```



- \_\_\_ m. Save your work.
- \_\_\_ 3. Implement the new service on the process.
- \_\_\_ a. From the history menu, open the **Approve Hire Request** linked process in the web Process Designer.
- \_\_\_ b. On the **Variables** tab, create a private variable: `hrAdmin1 (String)`



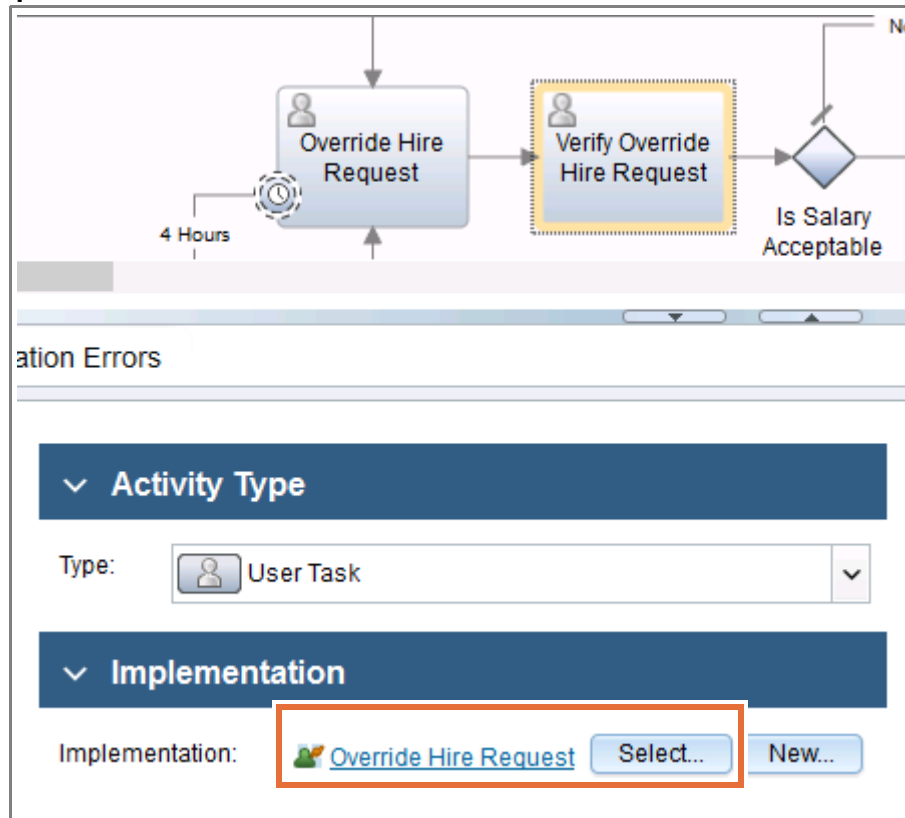
- \_\_\_ c. Click the **Definition** tab.

- \_\_\_ d. Click the **Override Hire Request** in the HR Administrator lane and click the **Properties > Implementation** menu. Implement the activity with the **Override Hire Request** client-side human service in the **Implementation** section.

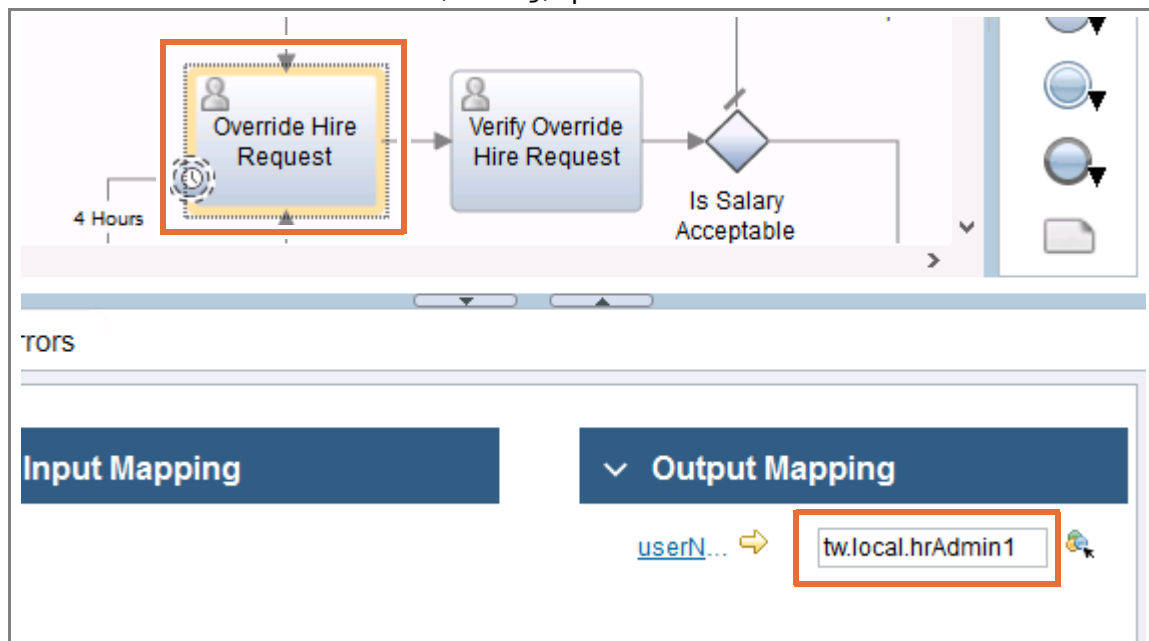
The screenshot displays the HR Administrator interface. At the top, a workflow diagram is visible, featuring a lane labeled "HR Administrator". Within this lane, there is a task named "Override Hire Request" (highlighted with a yellow border) followed by "Verify Override Hire Request". A decision diamond labeled "Is Salary Acceptable" follows, with a "Not." path leading away. A "4 Hours" timer icon is positioned near the "Override Hire Request" task. Below the diagram, the "Properties" tab is active, showing the "Implementation" section. The "Activity Type" is set to "User Task". The "Implementation" section shows a selection of "Override Hire Request" (highlighted with a red border), with "Select..." and "New..." buttons available.



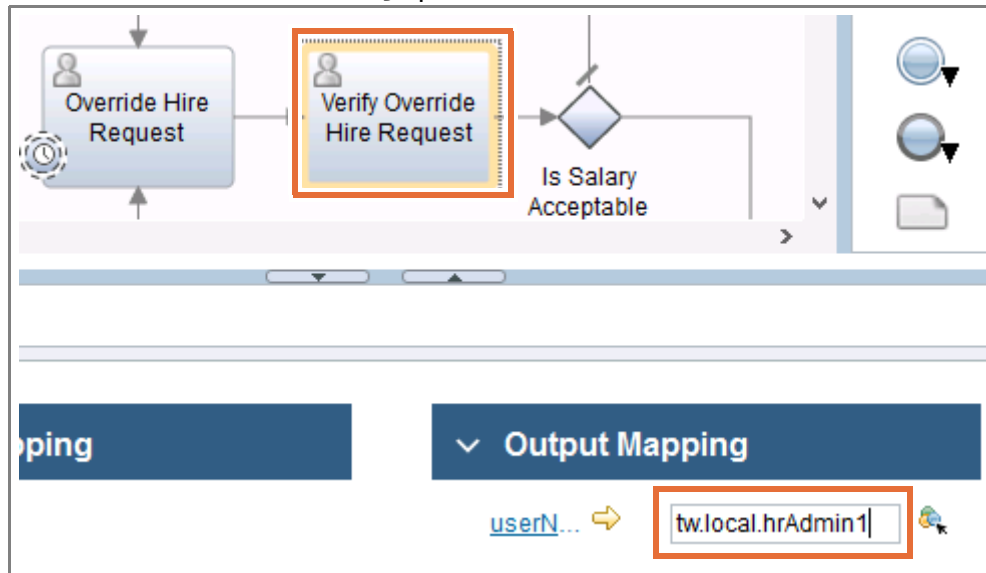
- \_\_\_ e. Implement the **Verify Override Hiring Request** activity with the same **Override Hire Request** client-side human service.



- \_\_\_ f. Click the **Override Hire Request** activity.
- \_\_\_ g. Open the **Properties > Data Mapping** menu and map the **userName1** output variable to the **tw.local.hrAdmin1 (String)** private variable.



- \_\_\_ h. Map the **userName1** variable of the **Verify Override Hire Request** activity to the `tw.local.hrAdmin1` (String) private variable.



- \_\_\_ i. Save your work.

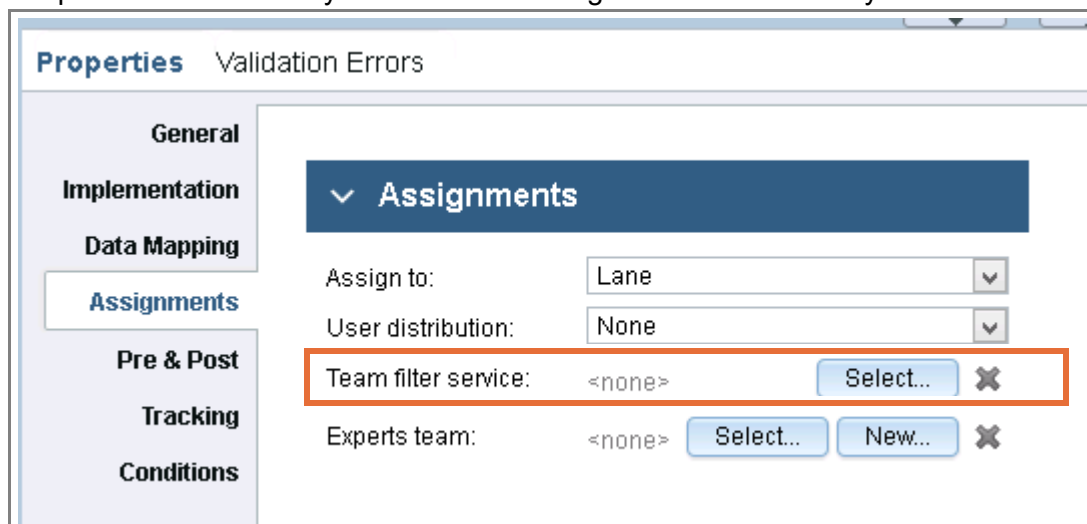
## 1.2. Create the team filter service

- \_\_\_ 1. The user of the Override Hire Request task can't verify their own action. You must ensure that the pool of users that can claim the verify activity does not include the user that completed the first activity. Create a team filter service to assign the second task to the correct pool of users.
- \_\_\_ a. Click the **Verify Override Hire Request** activity.

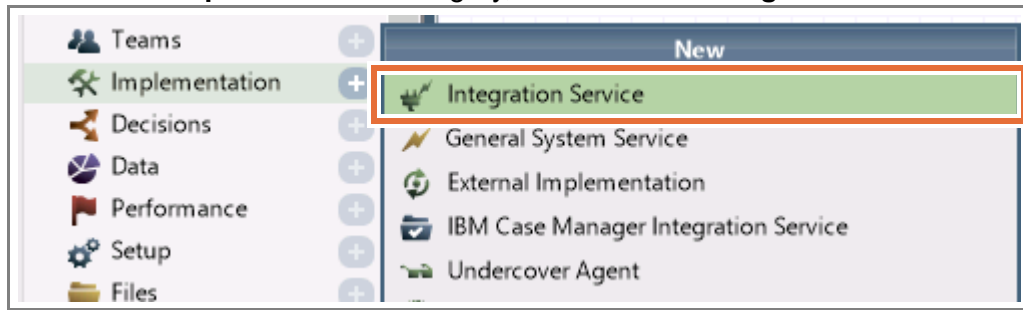


### Hint

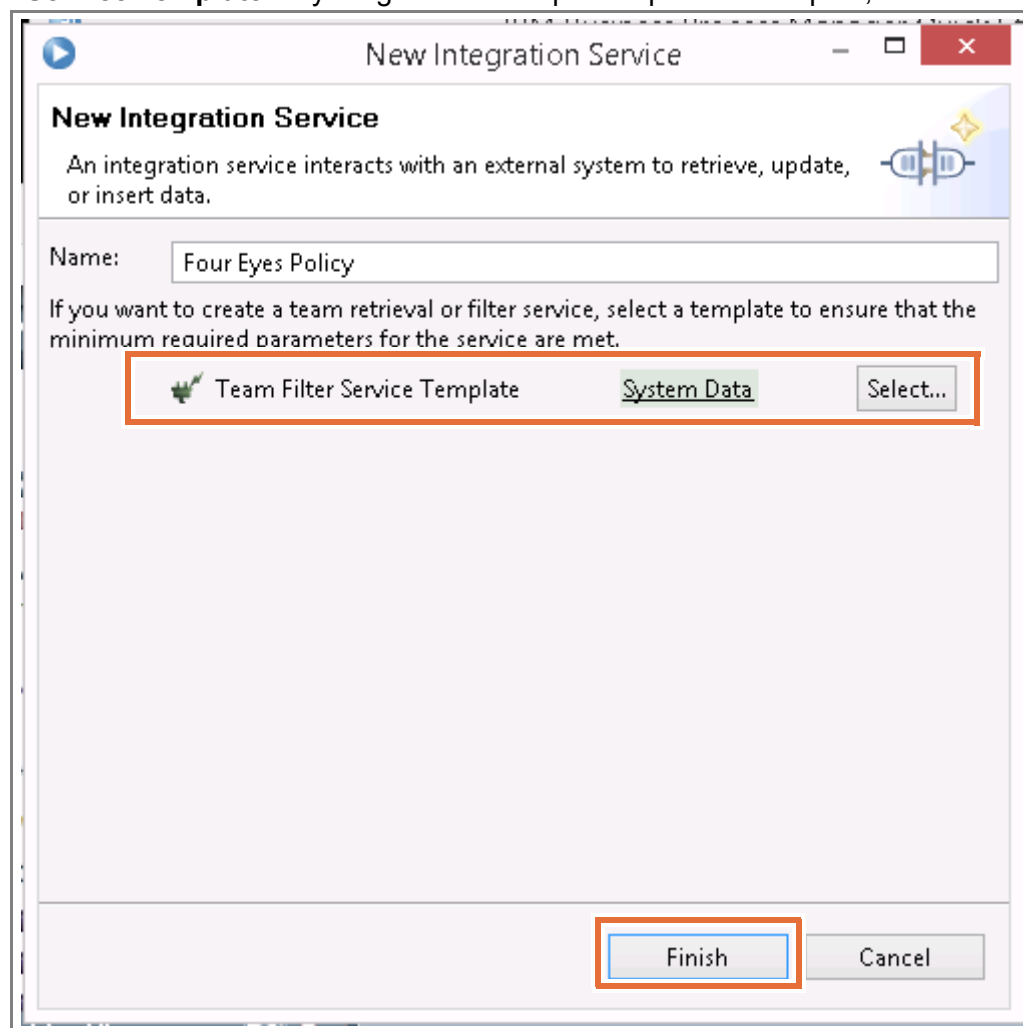
On the **Properties > Assignments** menu, you must create a Team Filter Service to filter the user that completes the first activity from the team assignment for this activity.



- \_\_\_ b. Return to the Process Designer client application. In the library, click the **plus sign (+)** next to the **Implementation** category, and then click **Integration Service**.



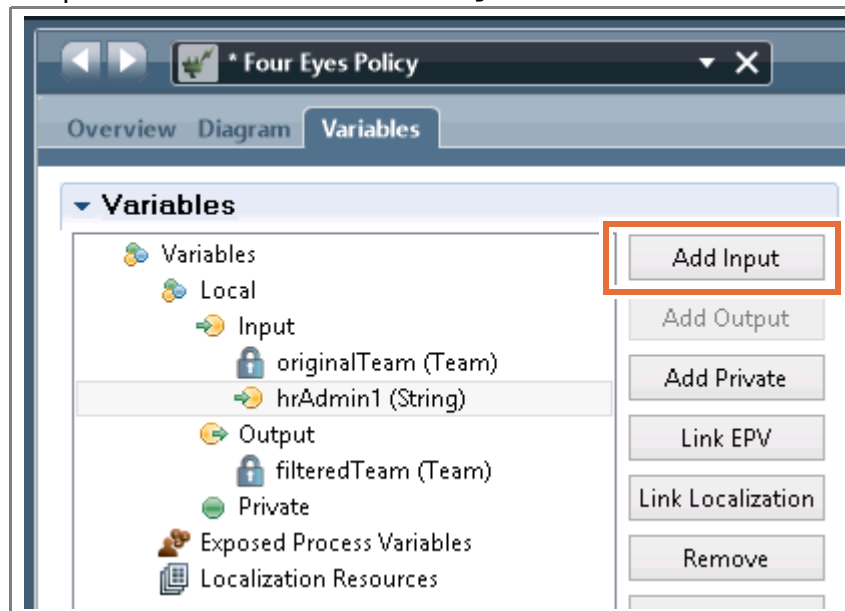
- \_\_\_ c. Name the Integration service: **Four Eyes Policy**
- \_\_\_ d. Click the **Select** button to choose a team service template. Select the **Team Filter Service Template** as your guide to set up the inputs and outputs, and click **Finish**.



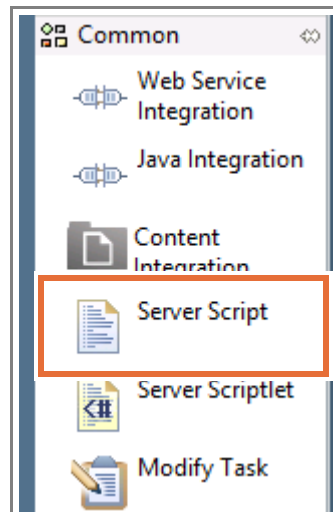
### Information

This template is part of the System Data toolkit. The template defines the inputs and output variables that are required for this type of service.

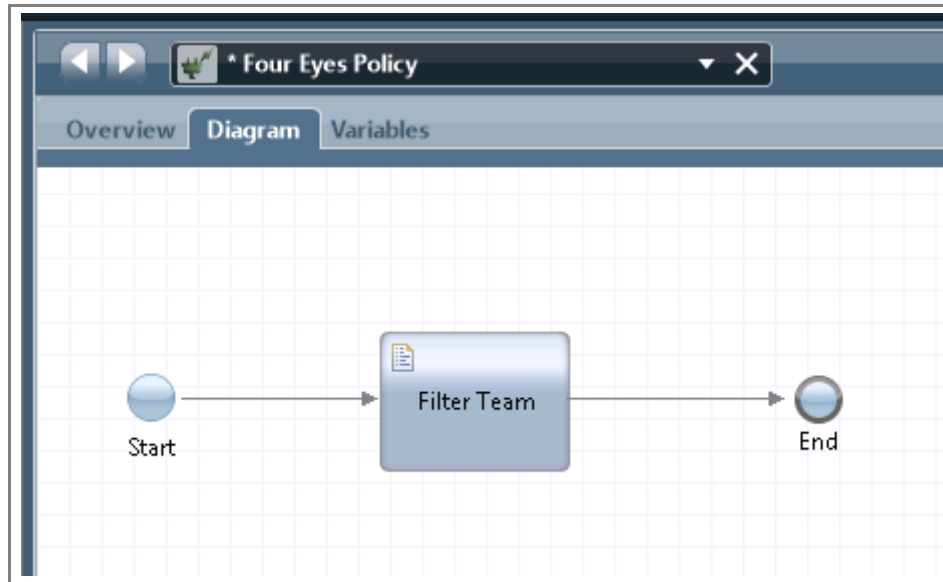
- \_\_\_ e. In the Four Eyes Policy integration service, click the **Variables** tab.
- \_\_\_ f. Add an input variable: hrAdmin1 (String)



- \_\_\_ g. Click the **Diagram** tab.
- \_\_\_ h. Do not use a Server **Scriptlet** for this task. Drag a Server **Script** onto the canvas and name the step: Filter Team



\_\_\_ i. Connect the flows.



\_\_\_ j. Click the **Filter Team** server script, and then click **Properties > Implementation**.

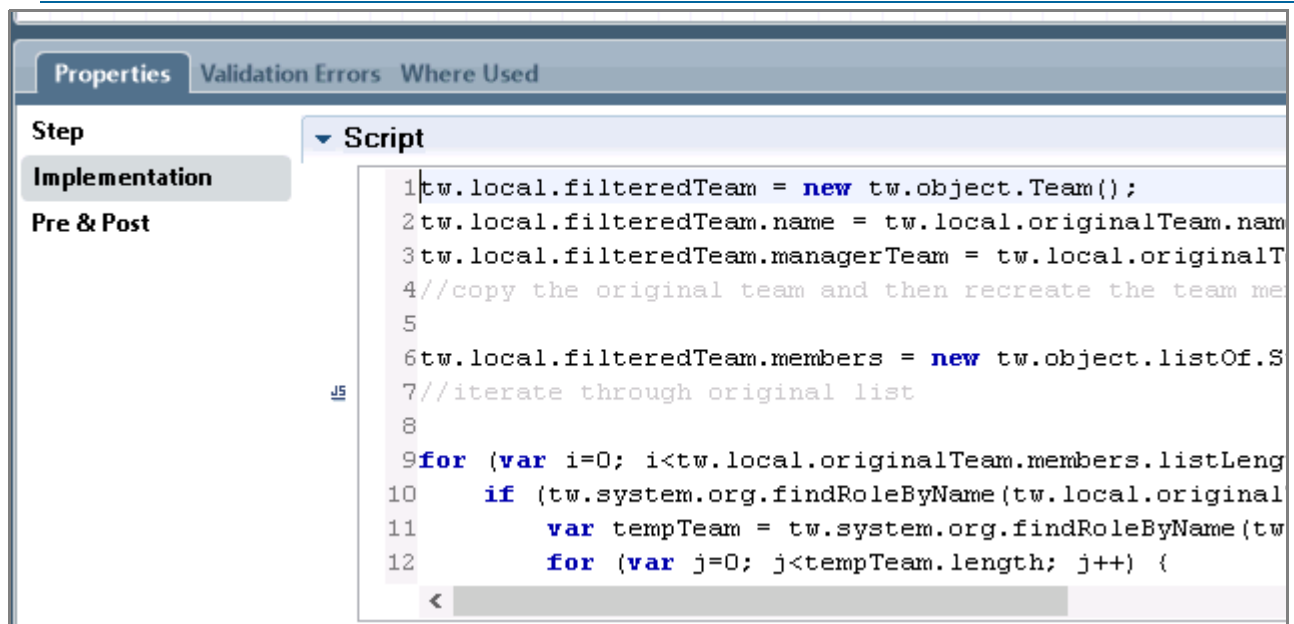
- \_\_\_ k. Filter the team by adding the following script. You can also copy this script from the C:\labfiles\Lab and Exercise Assets\Exercise code\Exercise 4\FilterJS.txt file:

```
tw.local.filteredTeam = new tw.object.Team();
tw.local.filteredTeam.name = tw.local.originalTeam.name;
tw.local.filteredTeam.managerTeam = tw.local.originalTeam.managerTeam;
//copy the original team and then recreate the team members

tw.local.filteredTeam.members = new tw.object.listOf.String();
//iterate through original list

for (var i=0; i<tw.local.originalTeam.members.listLength; i++) {
    if (tw.system.org.findRoleByName(tw.local.originalTeam.members[i])) { //team
        var tempTeam =
tw.system.org.findRoleByName(tw.local.originalTeam.members[i]).users;
        for (var j=0; j<tempTeam.length; j++) {
            if (tempTeam[j].name != tw.local.hrAdmin1)

tw.local.filteredTeam.members[tw.local.filteredTeam.members.listLength] =
tempTeam[j].name;
        }
    }
    else { //individual user
        if (tw.local.originalTeam.members[i] != tw.local.hrAdmin1)
            //if member is not the hrAdmin member, add member to filtered list
            tw.local.filteredTeam.members[tw.local.filteredTeam.members.listLength] =
tw.local.originalTeam.members[i].toString();
        }
    }
}
```





## Information

This code takes the original team that the task is assigned to, and filters out the user that is assigned to the `tw.local.hrAdmin1` variable.

- \_\_\_ l. Save and close the Four Eyes Policy integration service.

### 1.3. Apply the team filter service to the activity assignment

- \_\_\_ 1. Apply the team filter service.
  - \_\_\_ a. Return to the **Approve Hire Request** process application in the web Process Designer.
  - \_\_\_ b. Click the **Verify Override Hire Request** activity on the process.
  - \_\_\_ c. From the **Properties > Assignments** menu, select the **Four Eyes Policy** as the Team Filter service.

The screenshot shows the 'Properties' window with the 'Assignments' tab selected. The 'Team filter service' is set to 'Four Eyes Policy'. The 'Select...' button next to it is highlighted with a red rectangle.

- \_\_\_ d. In the input mapping section for the team filter service on the right, map the `tw.local.hrAdmin1` variable.

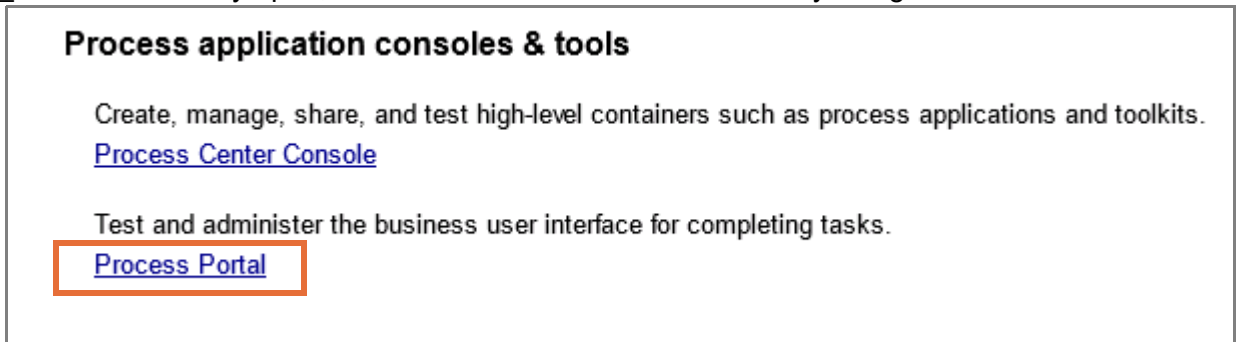
The screenshot shows the 'Input Mapping' section. It displays a mapping from the variable `tw.local.hrAdmin1` to the service input `hrAdmin1 (S...`.

- \_\_\_ e. Save the process application.

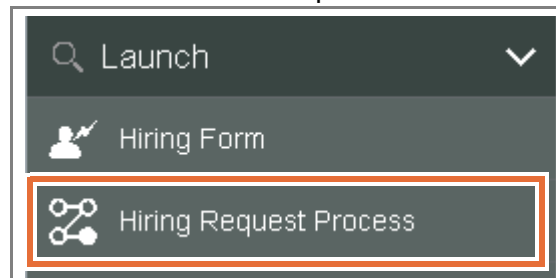
## Section 2. Test the four eyes policy

### 2.1. Test the four eyes policy in the process portal

- \_\_\_ 1. If not already open, click the link to the **Process Portal** by using the Quick Start menu.



- \_\_\_ 2. Create an instance of the process by clicking **Hiring Request Process** in the Launch section on the left. Wait for the first task to open in the main window.



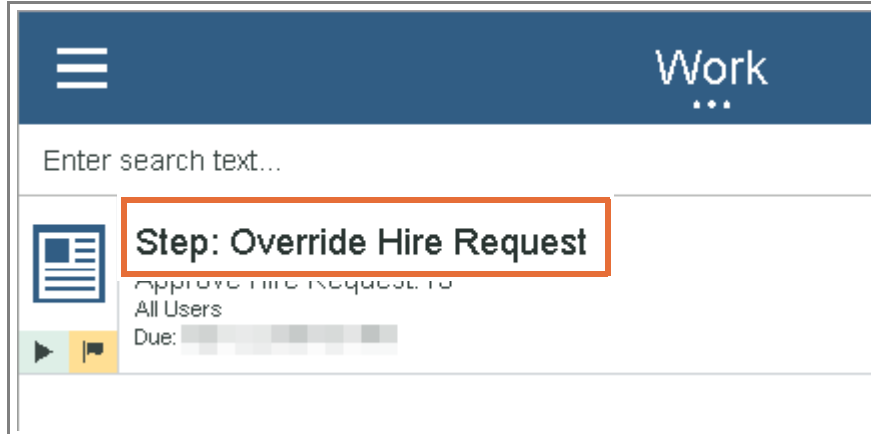
- \_\_\_ 3. Complete the first task by creating a non-compliant requisition. Use the following values for the variables on the form (settings inside the browser in parentheses):

- `tw.local.requisitionDetails.position.jobLevel` (**Position Details > Job Level**) set to: Manager
- `tw.local.requisitionDetails.recruitingDetails.newPosition` (**Recruiting Details > New Position**) set to: false (not selected)
- `tw.local.requisitionDetails.compensationDetails.salaryToOffer` (**Compensation Details > Salary to Offer**) set to: 100,000

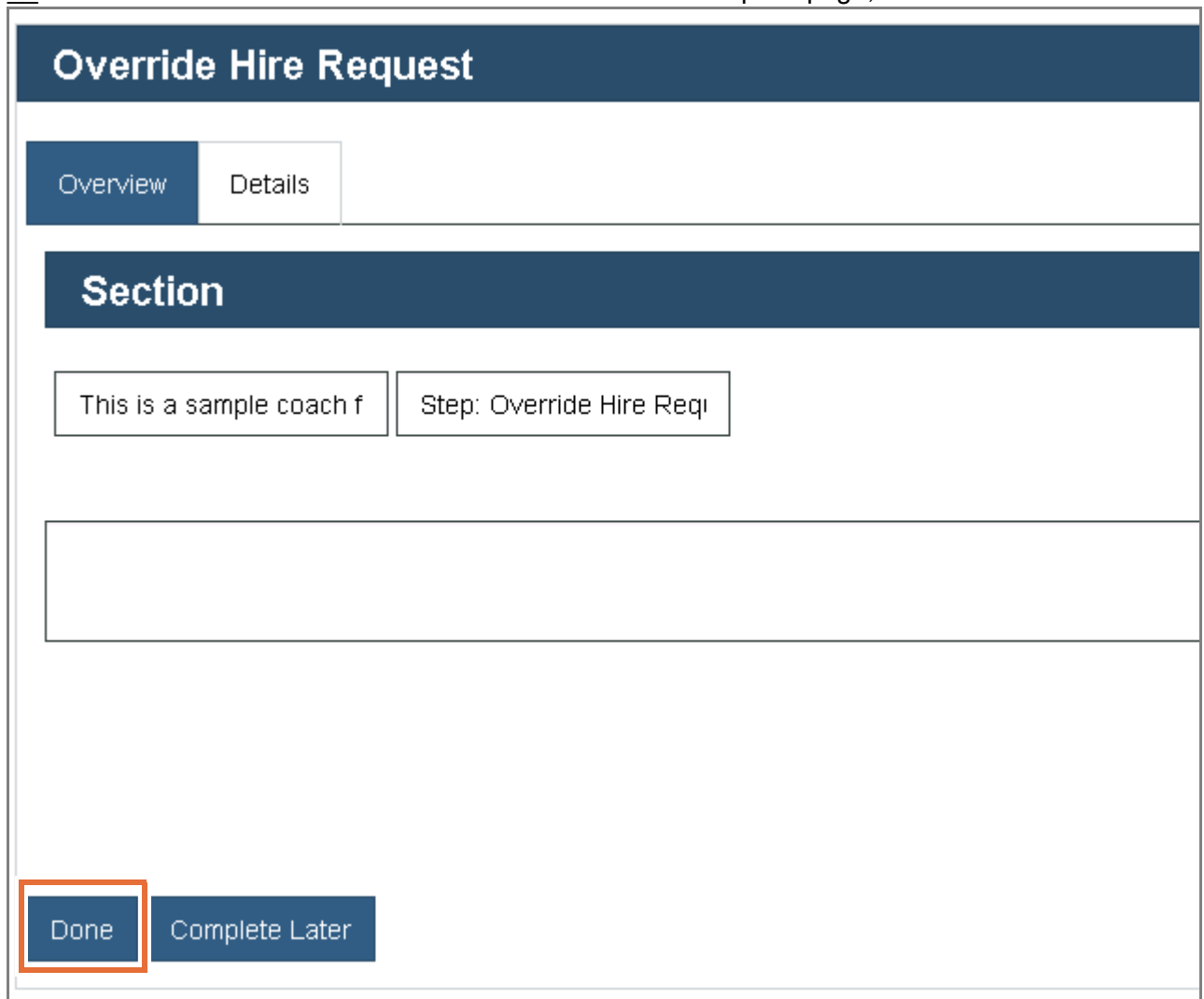
- \_\_\_ 4. Click **Submit**.



5. When the portal refreshes, the new task Override Hire Request activity is shown in the work dashboard. The next task inside the process is shown. This task is the first task in the four eyes requirement. Click the **Step: Override Hire Request** link to run the task.



6. Click the **Claim Task** button. In the Override Hire Request page, click **Done**.





## Information

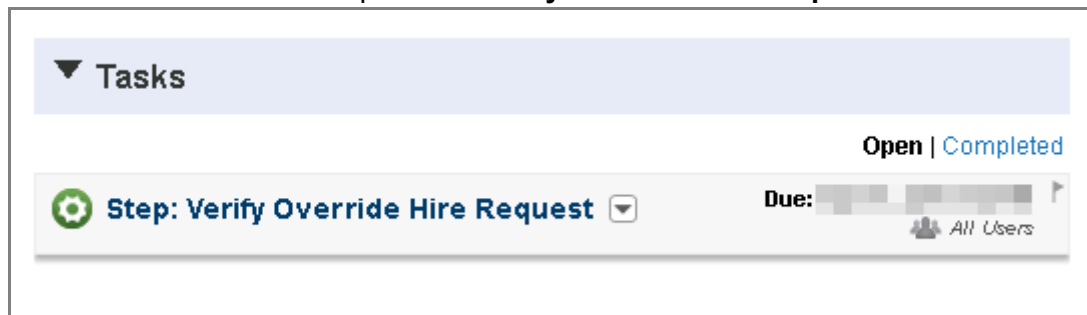
This coach is just a placeholder, and would be replaced with an approval form. You are trying to test the assignment, so ignore the placeholder.

Wait for the work dashboard to refresh. No other tasks are shown in the dashboard even though a second task is created and assigned to the All Users team, but the current user `author1` is filtered from the list. If you see any tasks, click the **Work** link on the left a second time to refresh the inbox.

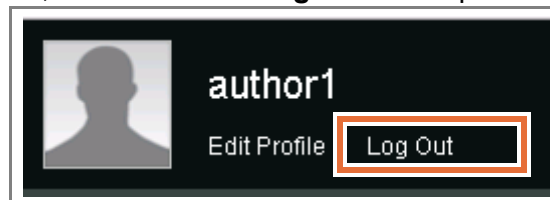
- \_\_\_ 7. Click the **Processes** link on the left, then click the **Approve Hire Request** link to view the information about the instance.



- \_\_\_ 8. The list of tasks is shown on the right. The Verify Override Hire Request task is assigned to "All Users," which is misleading. Although the role might seem to contain all users, it is a filtered list. The original approver (`author1`) is filtered from the All Users list. The filter is the reason why the task is not listed in the `author1` user's work dashboard. The task is assigned to a team that filtered out the user who completed the override hire request task. In this example, user `author1` completed the task of **Override Hire Request**, so `author1` is filtered out and cannot complete the **Verify Override Hire Request**.



- \_\_\_ 9. In the library on the left, click the link to **Log Out** of the process portal.



- \_\_\_ 10. Log in to the portal with the user name `user1` and the password `user01`. Notice that the Verify Override Hire Request task is listed on the Work dashboard.
- \_\_\_ 11. Complete the next activity **Verify Override Hire Request** to complete the instance.
- \_\_\_ 12. Close the Process Portal.

## End of exercise

## Exercise review and wrap-up

In the first part of the exercise, you modified the process and service to capture the first reviewer user ID. Next, you modified the activity on the process and created a routing policy. You tested the routing policy and verified that it filtered out the user from the list of participants available to claim the task.

---

# Exercise 5. Building a cancellation pattern

## Estimated time

01:00

## Overview

In this exercise, you learn how to implement a cancellation pattern.

## Objectives

After completing this exercise, you should be able to:

- Implement an undercover agent (UCA) to cancel the hiring request
- Implement a cancellation pattern in a process application

## Introduction

A general manager requested the ability to cancel a hiring request while waiting for approval from the approvers. Create a cancellation message listener inside IBM Business Process Manager.

## Requirements

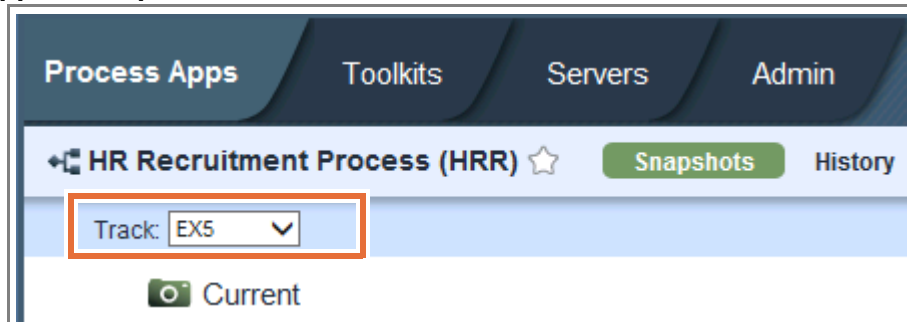
Completion of the previous exercise in this book.

## Section 1. Implement a cancel event

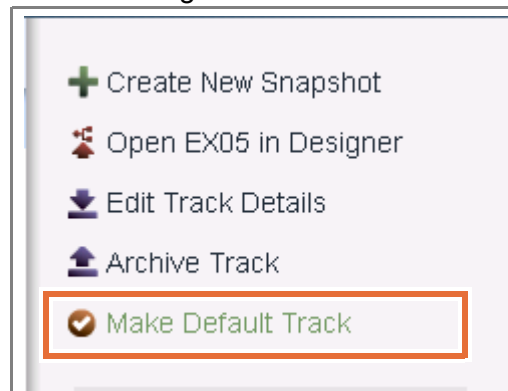


### Important

If you previously imported one of the solution files or are using tracks, select the track from the **Process Apps > Snapshots > Track** menu.



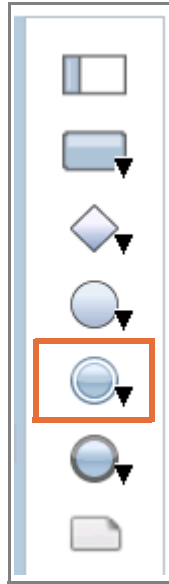
Click the **Make Default Track** link on the right to make this track the default track.



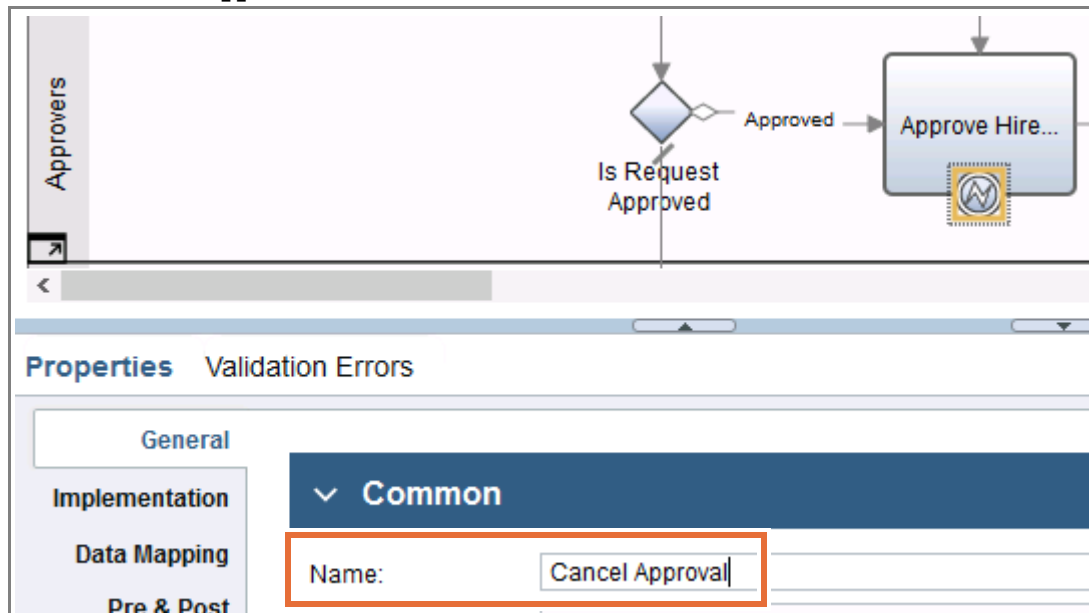
Open the track in the web Process Designer.

## 1.1. Create the event

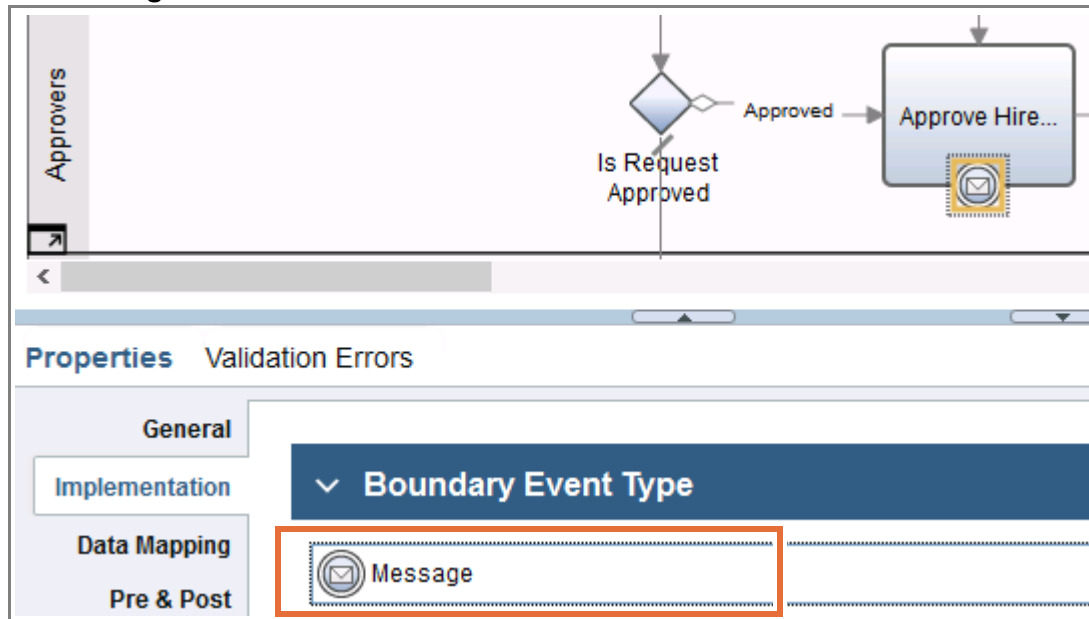
- \_\_\_ 1. A cancellation event must be attached to the Approve Hire Request linked process.
  - \_\_\_ a. Open the **Hiring Request Process** in the web Process Designer.
  - \_\_\_ b. Drag an **intermediate event** from the palette and place it on one of the attachment points on the **Approve Hire Request** linked process. Make sure that you attach the intermediate event to the **Approve Hire Request** linked process and not the **Approve New Hire Request** activity.



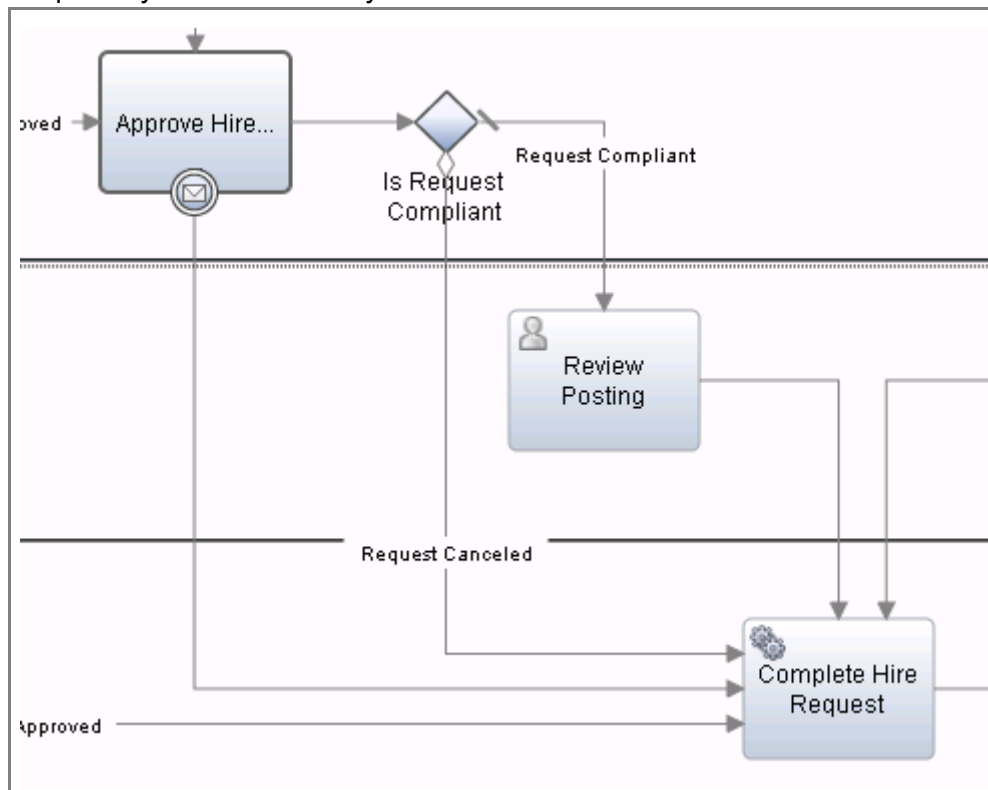
- \_\_\_ c. Verify that the attached intermediate event is selected. Change the name of the event to: Cancel Approval.



- \_\_\_ d. From the **Properties > Implementation** menu, change the Boundary Event Type to **Message**.



- \_\_\_ e. Create a flow from the attached message intermediate event to the Complete Hire Request system lane activity.



- \_\_\_ f. Save your changes.



## Important

If an event changes the flow of a process similar to a cancellation event, you should also include some cleanup steps. This approach ensures that the process participants know about the event, and participants can react to the event as part of the process. Merely stopping or terminating an instance is not an ideal approach.

## 1.2. Create an undercover agent

- \_\_\_ 1. Add an undercover agent in the library.
  - \_\_\_ a. Switch to the Process Designer client application.
  - \_\_\_ b. In the library, click the **plus sign (+)** next to the **Implementation** category, and click **Undercover Agent**.
  - \_\_\_ c. Name the undercover agent: `Cancel Hiring Request`
  - \_\_\_ d. Set the Schedule Type to: **On Event**.
  - \_\_\_ e. Click **Finish**.

**New Undercover Agent**

Undercover agents (UCAs) are started by an event and invoke a BPD. UCAs pass data to the BPD through an associated service or variable.

Name:

Schedule Type:



- \_\_\_ 2. Configure the undercover agent.
  - \_\_\_ a. Under the **Details** section, keep all the default settings.
  - \_\_\_ b. Verify that the **Variable** Implementation type is enabled.
  - \_\_\_ c. Leave the default Variable Type as **NameValuePair**.
  - \_\_\_ d. Verify that the **Enabled** check box is selected.

**Scheduler**

Schedule Type: On Event

Event Marker: ☒ Message Select...

Run now Add Event Subscription...

**Details**

Queue Name: Async Queue

Implementation: ☒ Variable ☐ Service

Variable Type: NameValuePair System Data Select... New...

Enabled: ☒

**Parameter Mapping**

☐ Use default Input (NameValuePair)

- \_\_\_ e. Change the Event Message to: CancelHiringRequest.

**Event**

Event Message: CancelHiringRequest

- \_\_\_ f. Save the undercover agent.

### 1.3. Implement the boundary event with the UCA

- \_\_\_ 1. Create a variable to correlate the UCA.
  - \_\_\_ a. Return to the **Hiring Request Process** in the web Process Designer.







- b. On the **Variables** tab, create a private variable: `cancellationId` (String)



▼ Details

Name:


cancellationId

Documentation:

**B** *I* U |      

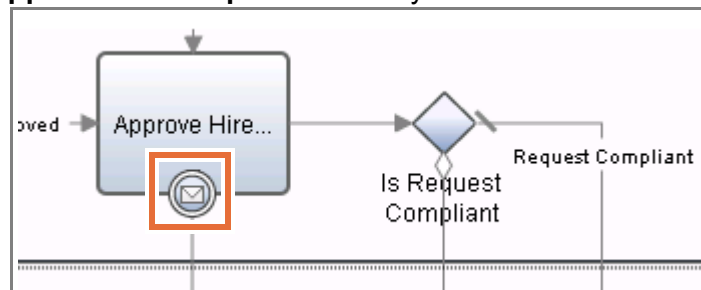
Variable type:

 [String](#) [System Data](#) Select... New...

List:

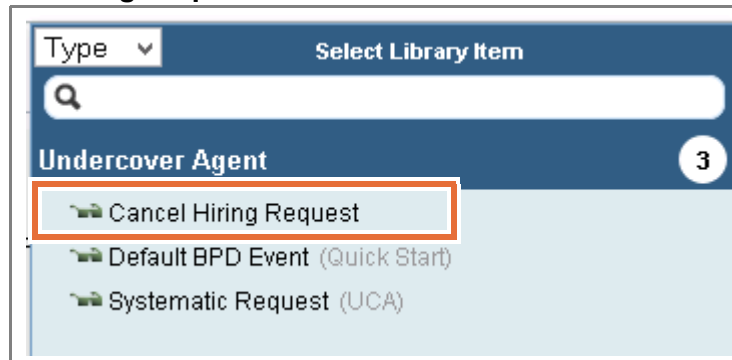
☐

- \_\_\_ 2. Configure the boundary event.
  - \_\_\_ a. Click the **Definition** tab.
  - \_\_\_ b. Click the **Approve Hire Request** boundary event.

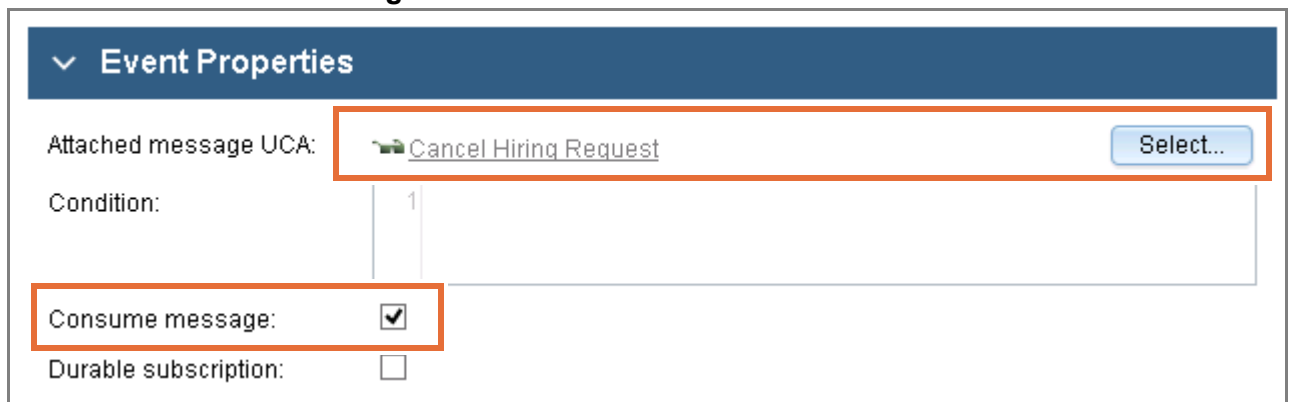


- \_\_\_ c. Click the **Properties > Implementation** menu.
- \_\_\_ d. Verify that the **Interrupt activity** check box is selected. The interrupt activity option consumes the token of the attached activity when the event is triggered. Because it is a linked process, even if numerous tokens exist inside the linked process, all the tokens are closed when the event is triggered.

- \_\_\_ e. In the Event Properties section, click **Select** next to the Attached message UCA and click **Cancel Hiring Request**.



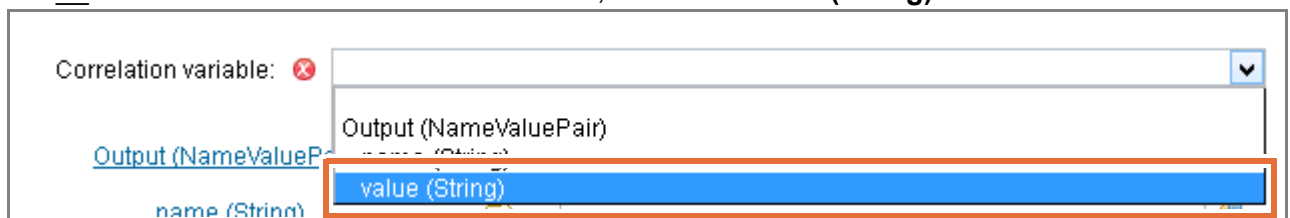
- \_\_\_ f. In the Event Properties section, select the **Consume message** check box. Consume message allows correlation with only a single process instance. If you configure multiple event listeners to correlate on the same message, then all events trigger if the **Consume message** check box is not selected.



- \_\_\_ g. Save your changes.

- \_\_\_ 3. Map the elements of the Output (NameValuePair) to simple variables in the process.

- \_\_\_ a. Click the **Properties > Data Mapping** menu.
- \_\_\_ b. From the **Correlation variable** list, select the **value (String)** variable.



- \_\_\_ c. Map the Output.name (String) value to the `tw.local.cancellationId` variable.

- \_\_\_ d. Map the Output.value (String) to the `tw.system.process.instanceId` variable.

Correlation variable:

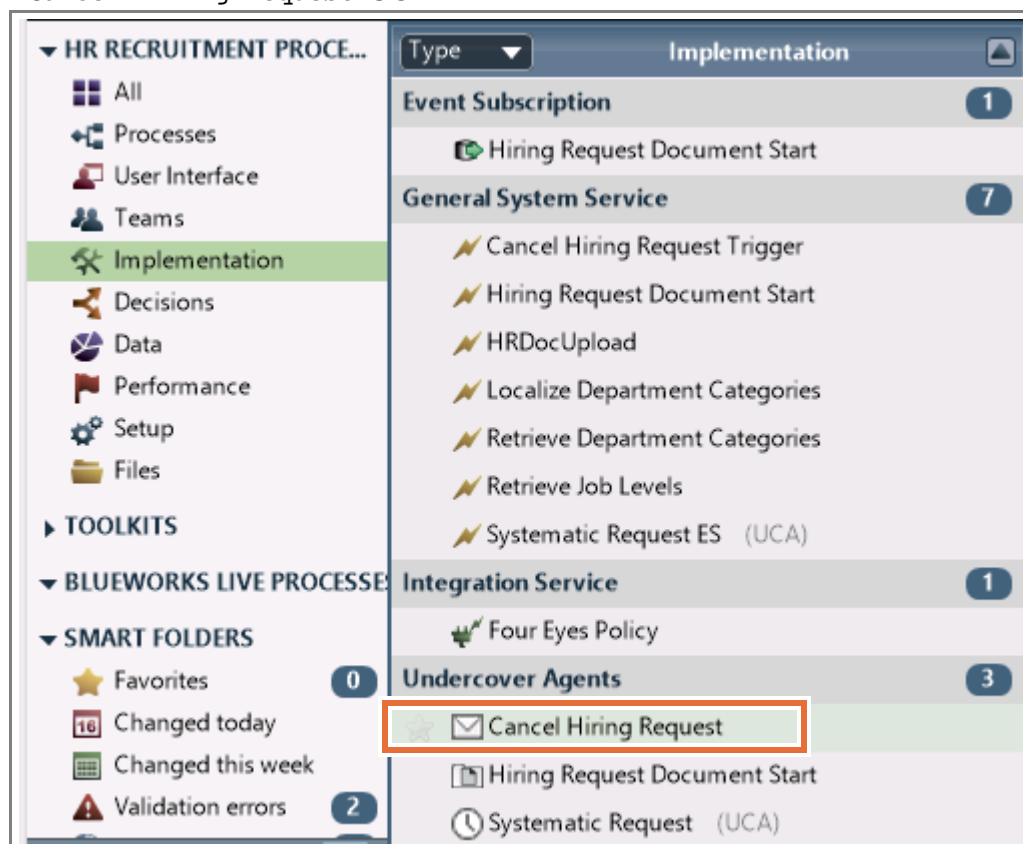
<a href="#">Output (NameValuePair)</a>	⇒	<input type="text"/>	
<a href="#">name (String)</a>	⇒	<input type="text" value="tw.local.cancellationId"/>	
<a href="#">value (String)</a>	⇒	<input type="text" value="tw.system.process.instanceId"/>	

- \_\_\_ e. The instance must know which message is targeting it. The messages sent to the UCA must contain a value that matches the instance ID of this process instance in order for this instance to receive the message. When the Output.value and the instance ID correlate, the intermediate event processes the message, and a token flows down the path. The user ID that canceled the process is also captured and mapped to a local variable. Save your work.

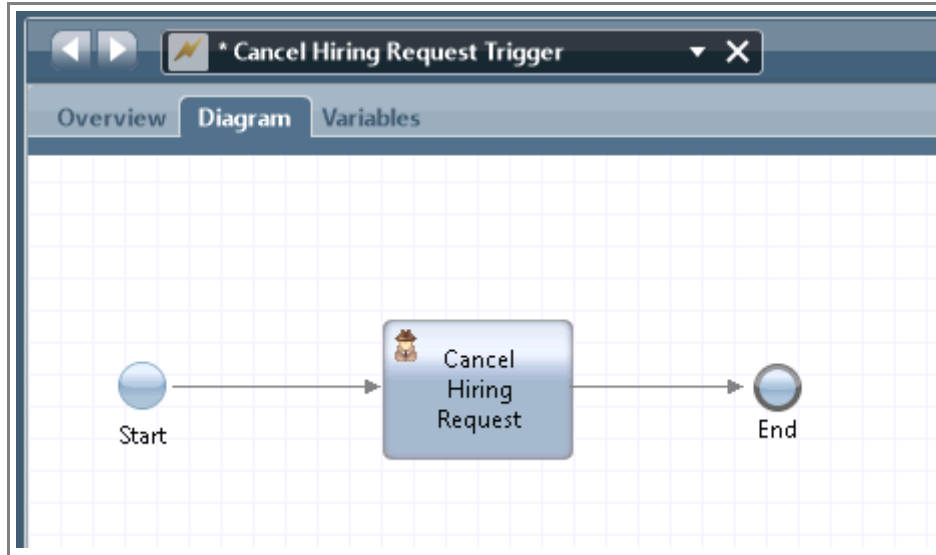
## Section 2. Create a service to trigger the UCA

### 2.1. Create the wrapper service

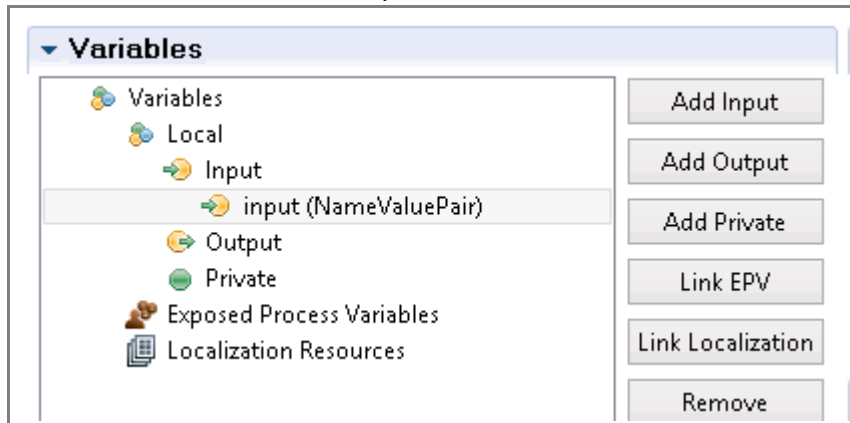
- \_\_\_ 1. Create a service to trigger the UCA.
  - \_\_\_ a. Return to the Process Designer client application.
  - \_\_\_ b. In the library, click the **plus sign (+)** next to the **Implementation** menu and create a **General System Service**.
  - \_\_\_ c. Name the service: Cancel Hiring Request Trigger
  - \_\_\_ d. Click **Finish**.
  - \_\_\_ e. Open the **Library > Implementation > Undercover Agents** category and drag the Cancel Hiring Request UCA onto the canvas.



\_\_\_ f. Connect the flows.



\_\_\_ g. On the **Variables** tab, create an input variable: `input` (NameValuePair)



\_\_\_ h. Save your changes.

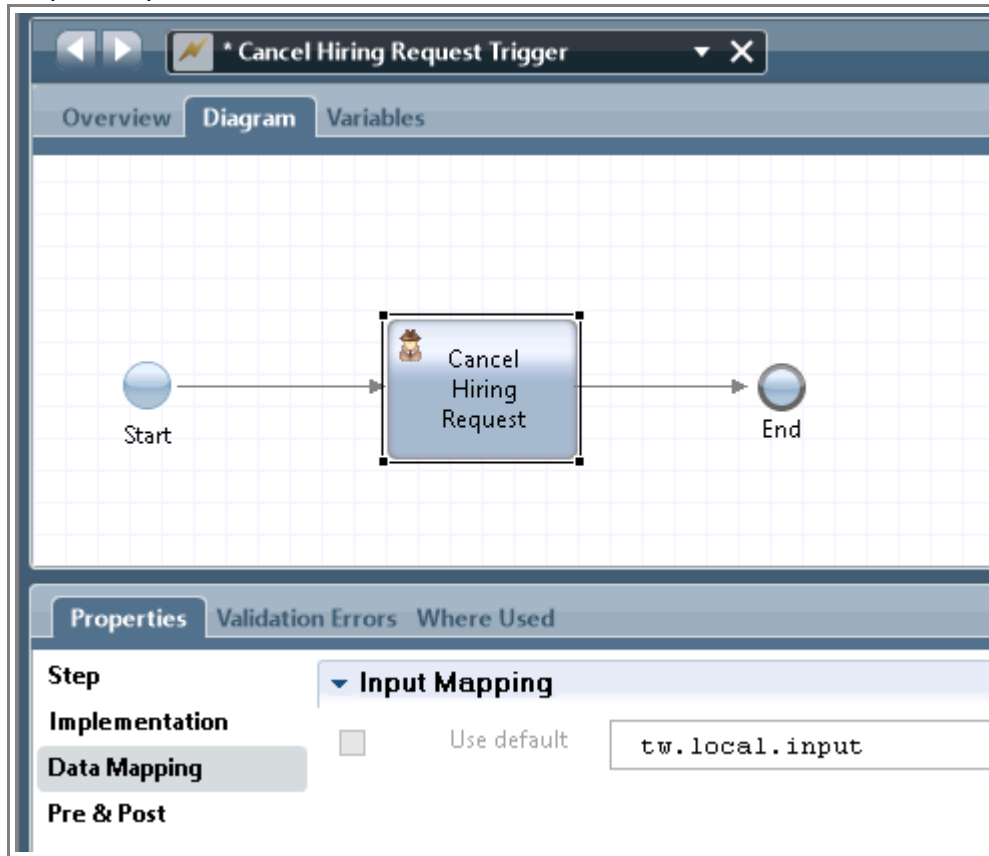
\_\_\_ 2. Map the variable.

\_\_\_ a. Click the **Diagram** tab.

\_\_\_ b. Click the **Cancel Hiring Request** UCA step.

\_\_\_ c. Click the **Properties > Data Mapping** menu.

- \_\_\_ d. Map the input of the UCA to the `tw.local.input` variable.



- \_\_\_ e. Save your work.

## 2.2. Create a test harness

- \_\_\_ 1. Add a client-side human service to test the UCA.
- \_\_\_ a. Return to the web Process Designer. In the library, click the **plus sign (+)** next to **User Interface**. Click **Client-Side Human Service**.
  - \_\_\_ b. Name the service: Cancel Hiring Request Test. Select the **Intended for use on multiple devices** check box. Click **Finish**.
  - \_\_\_ c. Drag a Client-side script from the palette and place it next to the Start event on the canvas. Name it: Create NVP

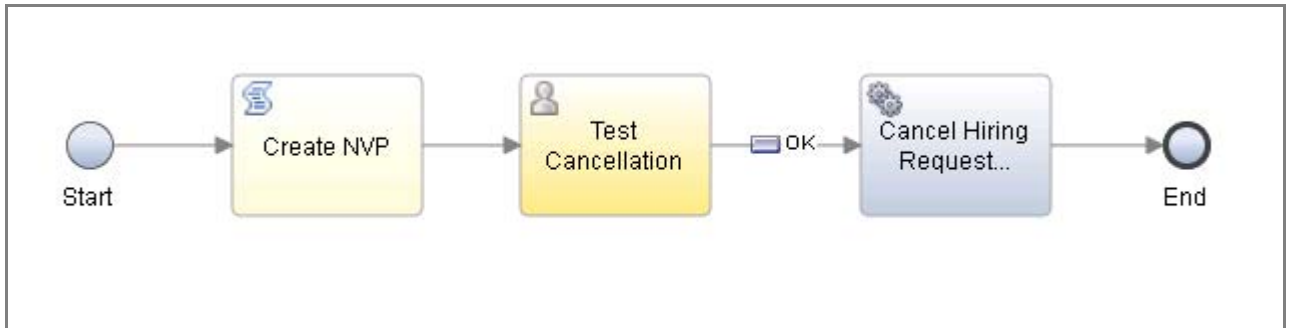




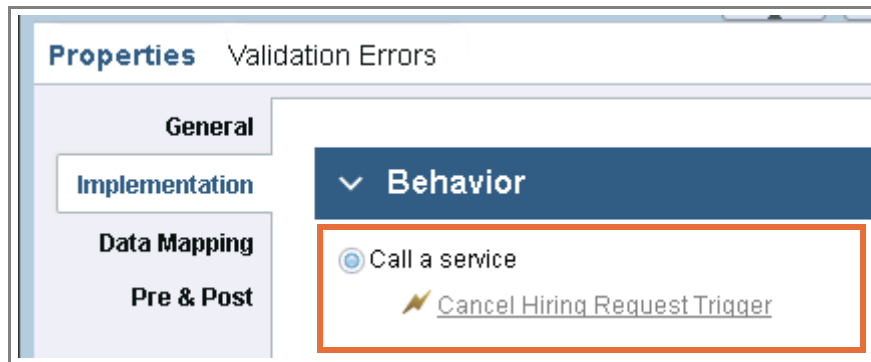
## Information

The abbreviation for name-value pair is NVP for this step.

- \_\_\_ d. Select the coach on the canvas and name it: **Test Cancellation**
- \_\_\_ e. Drag a service from the palette onto the canvas to the right of the coach and name it: **Cancel Hiring Request Trigger**
- \_\_\_ f. Connect the flows.



- \_\_\_ g. Click the **Cancel Hiring Request Trigger** service. Click the **Properties > Implementation** menu.
- \_\_\_ h. For the **Call a service**, click **Select**, and choose the **Cancel Hiring Request Trigger** service.

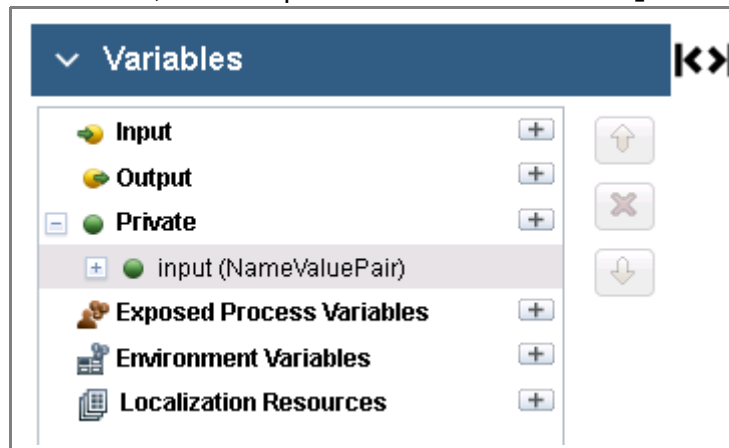


- \_\_\_ i. Save your changes.

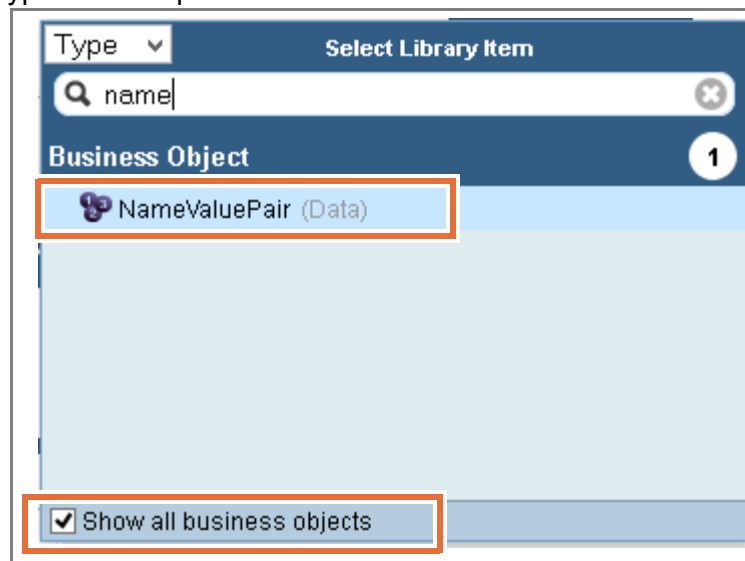


\_\_\_ 2. Implement the Create NVP step on the canvas.

\_\_\_ a. On the **Variables** tab, create a private variable named: `input`.



\_\_\_ b. Select the **Show all business objects** check box and select the **NameValuePair** variable type for the input variable.

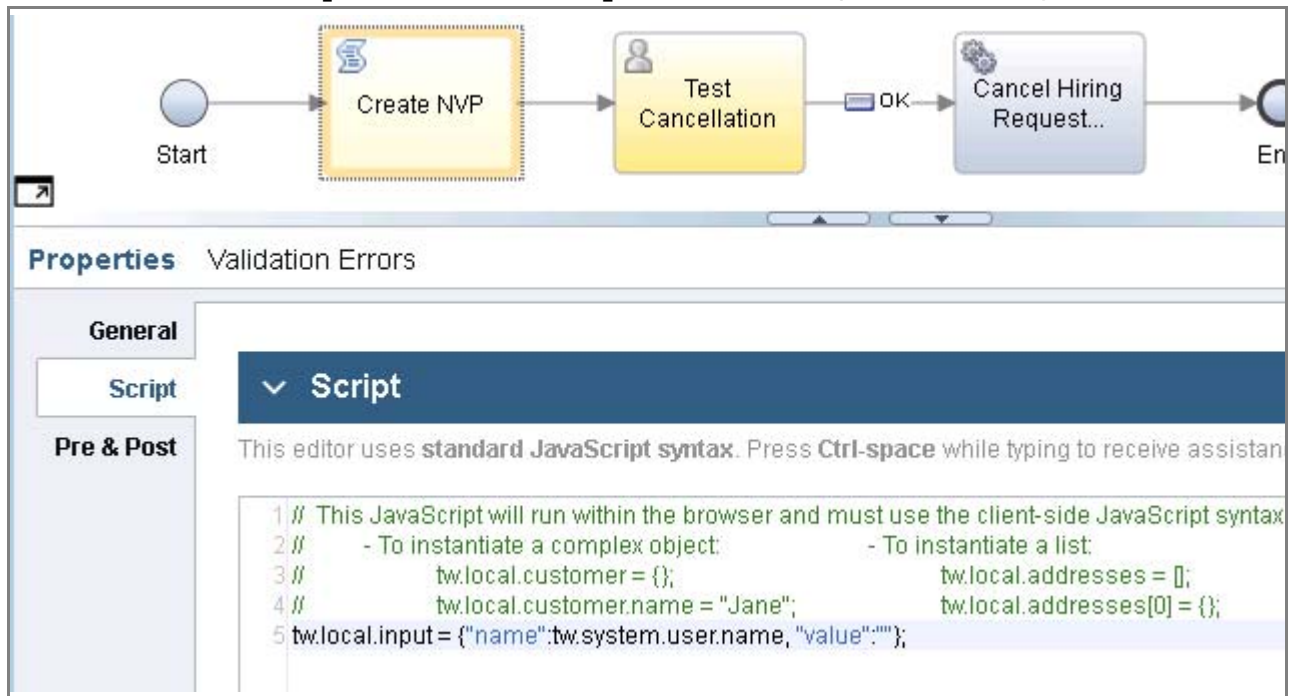


\_\_\_ c. Click the **Diagram** tab.

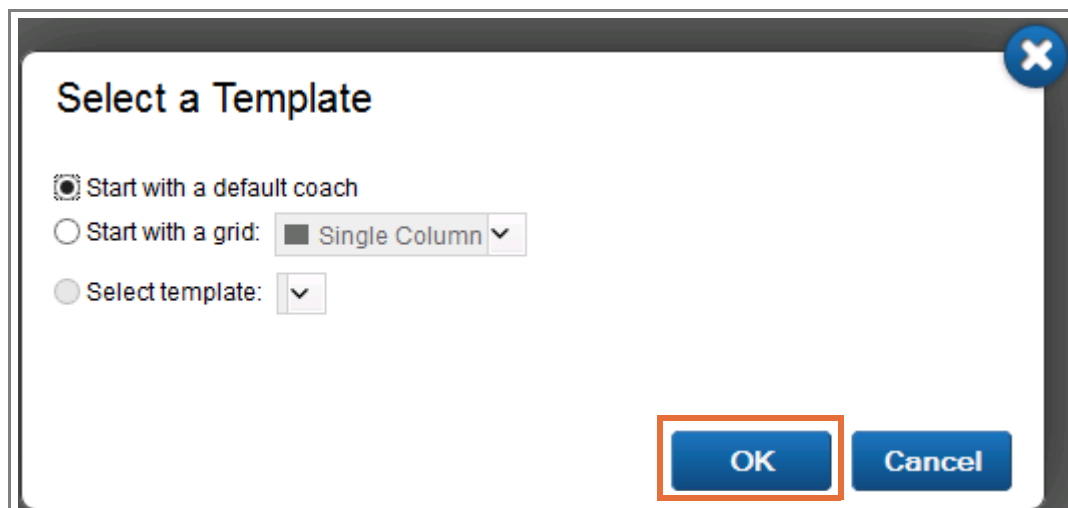
\_\_\_ d. Select the **Create NVP** script, and click the **Properties > Script** menu.

- \_\_\_ e. Copy the following code and paste it into the code block. The example that is shown is saved to the C:\labfiles\Lab and Exercise Assets\Exercise code\Exercise 5\Create NVP.txt file:

```
tw.local.input = {"name":tw.system.user.name, "value":""};
```

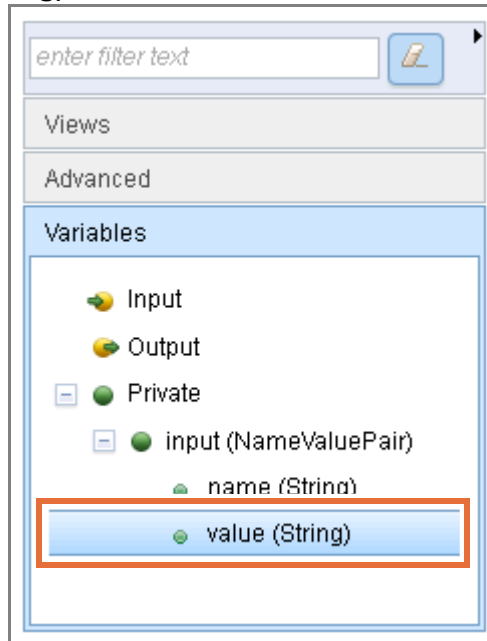


- \_\_\_ f. Save your changes.
- \_\_\_ 3. Build the Test Cancellation coach.
- \_\_\_ a. On the **Coaches** tab, click the **Test Cancellation** coach.
- \_\_\_ b. Select **Start with a default coach** and click **OK**.

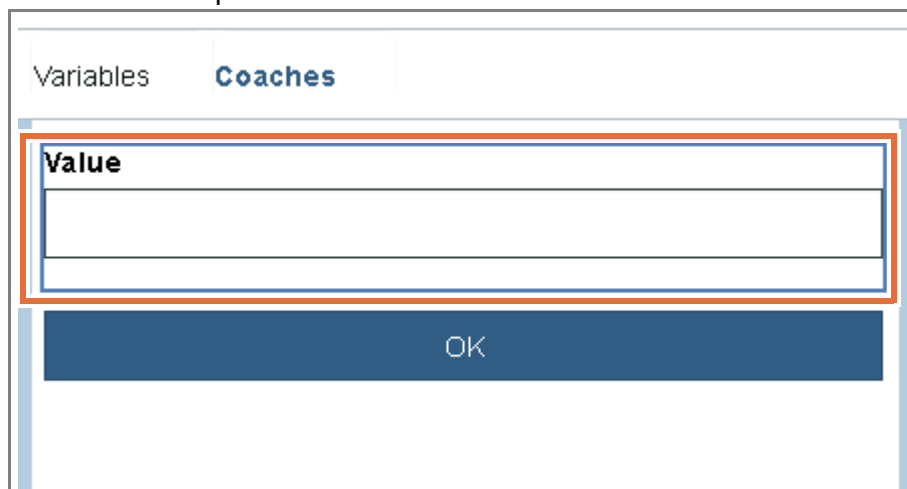


- \_\_\_ c. In the palette on the right, expand the **Variables** section.
- \_\_\_ d. Click the **plus sign (+)** next to **Private > input** to expand it.

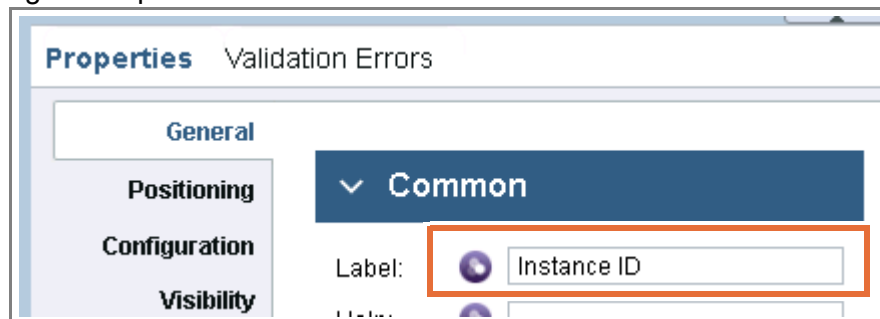
- \_\_\_ e. Drag the **value (String)** variable onto the canvas above the OK button.



- \_\_\_ f. Select the **Value** input control on the canvas.

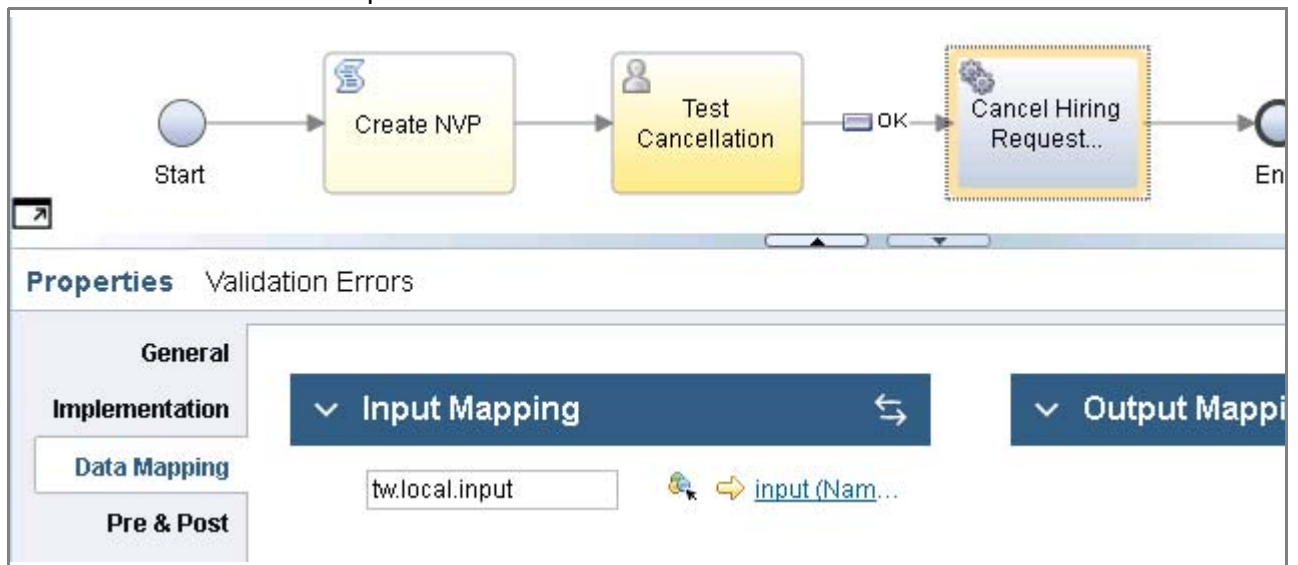


- \_\_\_ g. Change the input field label to: Instance ID



- \_\_\_ 4. Map the input variable for the Cancel Hiring Request Trigger step on the canvas.
- \_\_\_ a. Open the **Diagram** tab.
- \_\_\_ b. Click the **Cancel Hiring Request Trigger** service.

- \_\_\_ c. Click the **Properties > Data Mapping** menu, and map the `tw.local.input` private variable to the input.

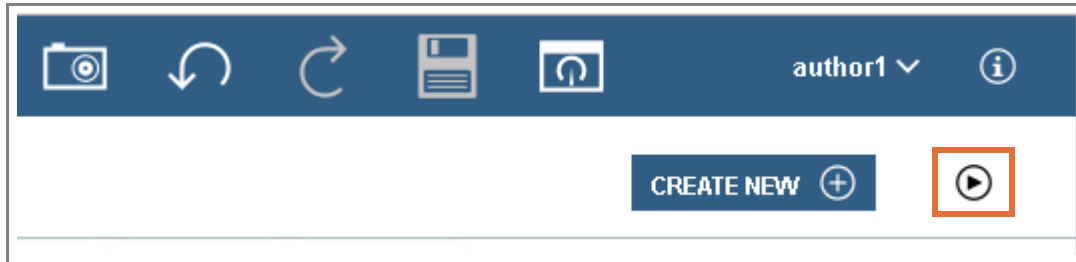


- \_\_\_ d. Save your work.

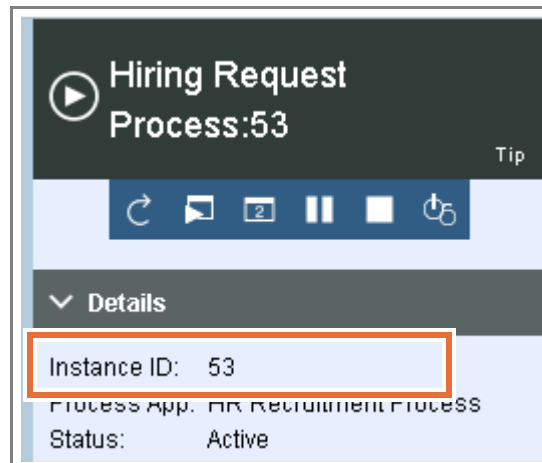
## Section 3. Cancel an instance

### 3.1. Test the cancellation UCA

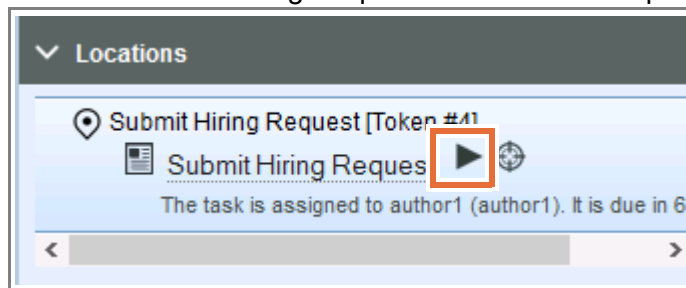
- \_\_\_ 1. Create an instance of the Hiring Request Process.
  - \_\_\_ a. Open the **Hiring Request Process** inside the web Process Designer.
  - \_\_\_ b. Click **Run** inside the process window to create an instance of the process.



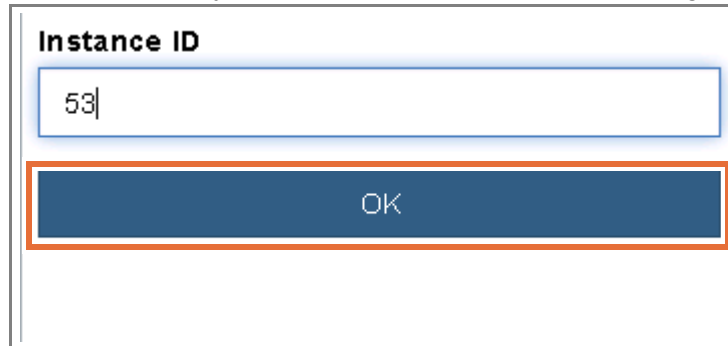
- \_\_\_ c. The perspective changes to the **Inspector** tab, and the information bar opens on the right.
- \_\_\_ d. The Instance ID in the Process Inspector is listed on the right. You use this number in the next step to cancel the instance.



- \_\_\_ 2. Complete the first activity in the process.
  - \_\_\_ a. To complete the first activity and get to the linked process, expand the **Locations** section.
  - \_\_\_ b. Click **Run** to run the Submit Hiring Request task from the Inspector information bar.



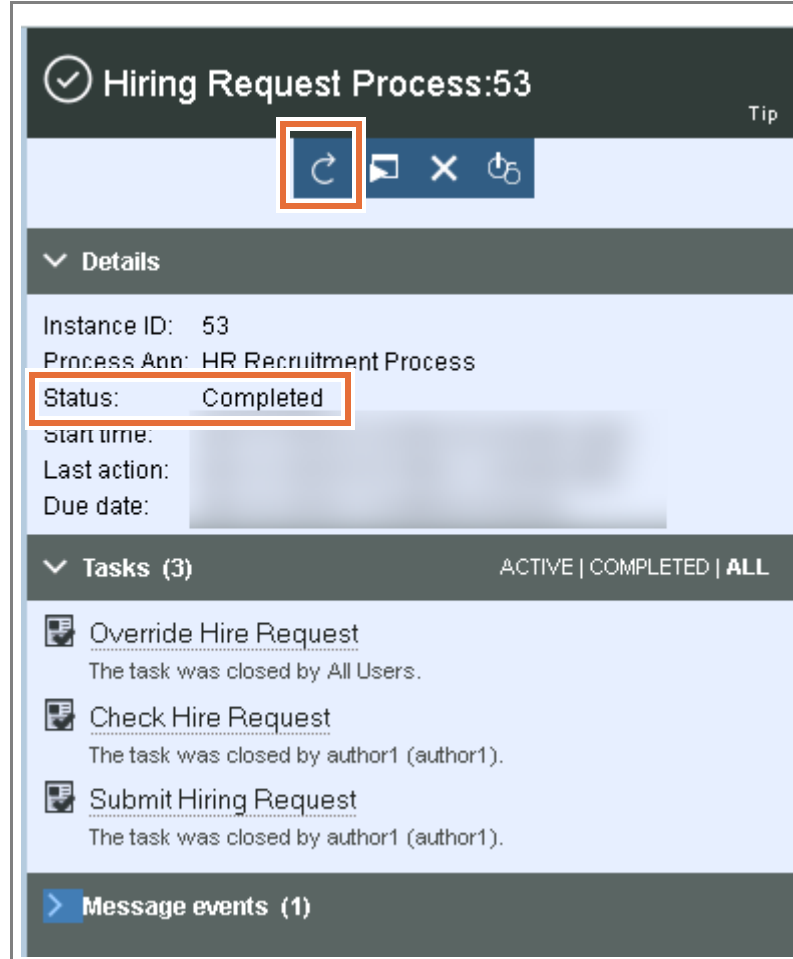
- \_\_\_ c. Use the following values for the variables on the form:
  - Position Details > Job Level set to: *Manager*
  - Recruiting Details > New position set to **false** (not selected)
  - Compensation Details > Salary to offer set to 100,000
- \_\_\_ d. Click **Submit**.
- \_\_\_ e. When the service completes, close the coach browser window.
- \_\_\_ f. When the Inspector refreshes, a token is associated with the Approve Hire Request linked process. The next thing that you do is to cancel the linked process and close all the tokens that are associated with the linked process by sending a message to the attached intermediate message event.
- \_\_\_ 3. Trigger the cancellation event on the linked process.
  - \_\_\_ a. Open the **User Interface > Cancel Hiring Request Test** client-side human service.
  - \_\_\_ b. Click **Run**.
  - \_\_\_ c. Input the instance ID that you noted from the Note block on page 5-18. Click **OK**.



The screenshot shows a dialog box with the title "Instance ID". Inside the dialog, there is a text input field containing the number "53". Below the input field is a blue button labeled "OK". The "OK" button is highlighted with a thick orange border.

- \_\_\_ d. When the service completes, close the window.
- \_\_\_ 4. Verify that the linked process is canceled.
  - \_\_\_ a. Open to the **Hiring Request Process** in the Inspector view.

- \_\_\_ b. Refresh the Process Inspector, and verify that the message consumed the token for the Verify Override Hire Request linked process and all tokens that are associated with the linked process are closed.



- \_\_\_ c. Return to the **Designer** tab.



### Information

All the tasks complete and the process consumes all the tokens in the linked process. The process flow travels out of the attached message event to the system lane activity. After the Complete Hire Request activity completes, the token moves to the End event and the process is complete. The instance status changes to **Completed**.

## End of exercise

## Exercise review and wrap-up

In the first part of the exercise, you created an attached intermediate event on the process. You implemented it as an intermediate message event. You created the enabling service, created the on-demand UCA, and then implemented the message service. You then created a service to wrap the UCA and created a test harness to trigger the UCA on the coach.



---

# Exercise 6. Implementing a multi-instance loop

## Estimated time

01:00

## Overview

Multi-instance loops in IBM Business Process Manager allow process applications to have multiple tasks that must be completed to move to the next activity. This exercise teaches you how to manage multi-instance loops to function properly.

## Objectives

After completing this exercise, you should be able to:

- Set up the inputs and outputs to a multi-instance loop
- Determine the behavior and start quantity of a multi-instance loop
- Apply the correct exit criteria for a multi-instance loop

## Introduction

The four eyes policy caused the overall process instance duration to increase by 25%. This performance decrease is unacceptable. Furthermore, the process does not accommodate the logic where one denial causes the denial of the override, and the process should not wait for any other reviews to occur before the process continues down the path. Management would like to turn the serial activities into a parallel process. Management would also like the ability to choose the number of HR Administrators that it takes to override the hiring request during the review.

## Requirements

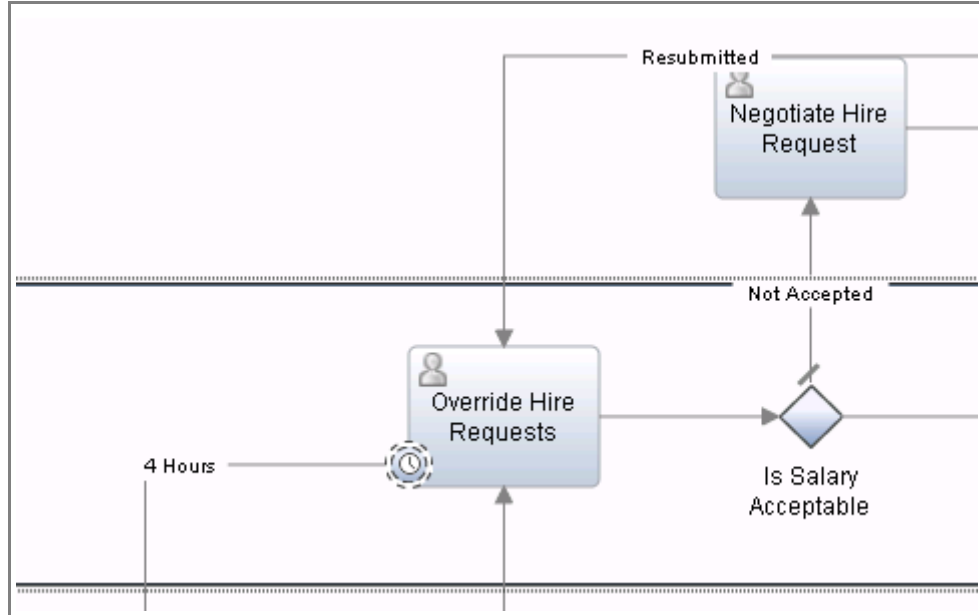
Completion of the previous exercise in this book.

## Section 1. Create a multi-instance loop

When the number of times an activity is going to be created is unknown at design time, a powerful tool you can use to implement the activity is called the multi-instance loop. You can configure the loop in numerous ways to provide the developer the ability to meet complex process requirements.

### 1.1. Delete the unneeded activity

- \_\_\_ 1. Remove the **Verify Override Hire Request** activity.
  - \_\_\_ a. Open the **Approve Hire Request** linked process in the Designer view in the web Process Designer.
  - \_\_\_ b. Delete the **Verify Override Hire Request** activity so the only activity in the HR Administrator lane is the **Override Hire Request** activity.
  - \_\_\_ c. Reconnect the flows from the **Override Hire Request** activity to **Is Salary Acceptable**.
  - \_\_\_ d. Rename the activity to: **Override Hire Requests**



## 1.2. Add the selection list to the approval task

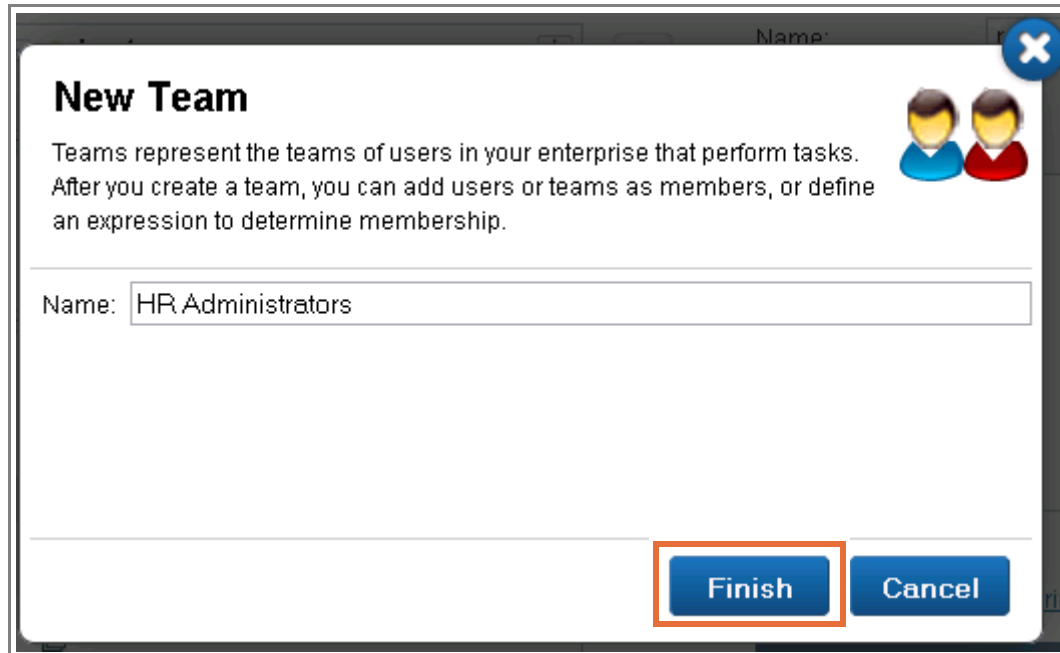
Provide the general manager the ability to choose the users who can override a rejection. The managers believe that they are able to target the overview to those reviewers who are familiar with the record.

- \_\_\_ 1. Create a coach for Approve Request.
  - \_\_\_ a. In the library, right-click the **User Interface > Hiring Form Approval** service.
  - \_\_\_ b. Click **Duplicate**.
  - \_\_\_ c. As soon as the **Hiring Form Approval 2** is created, rename it to: `Approve Request`
  - \_\_\_ d. Click **Finish**.
- \_\_\_ 2. Update the service.
  - \_\_\_ a. Open the **Approve Request** client-side human service.
  - \_\_\_ b. Add the output variable `hrAdminList (String) (List)` to hold the list of selected reviewers.



- \_\_\_ c. Save your work.
- \_\_\_ 3. Create a team to store members of the HR Administrators group.
  - \_\_\_ a. Click the **plus sign (+)** next to the **Teams** category in the library, and click **Team**.

- \_\_\_ b. Name the team **HR Administrators** and click **Finish**.



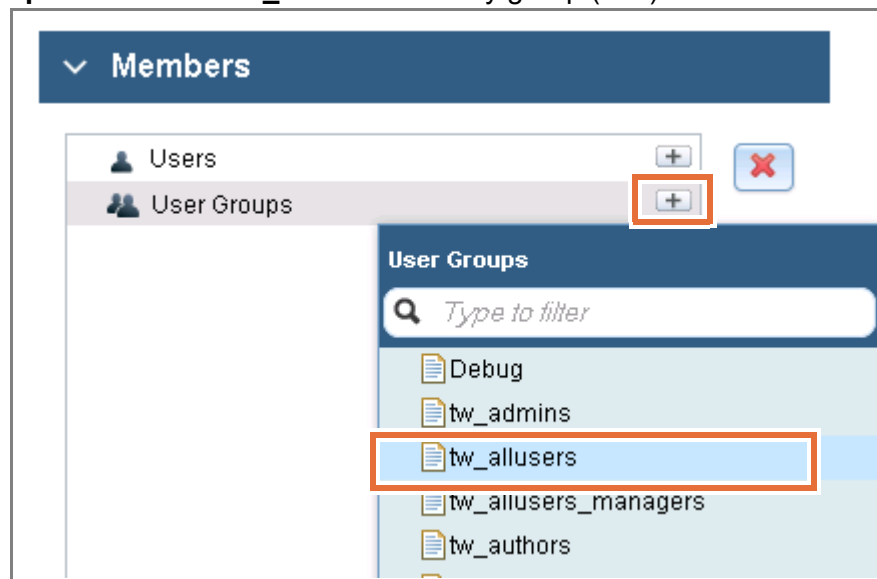
**New Team**

Teams represent the teams of users in your enterprise that perform tasks. After you create a team, you can add users or teams as members, or define an expression to determine membership.

Name:

**Finish** **Cancel**

- \_\_\_ c. For testing purposes, under the Members section, click the **plus sign (+)** next to **User Groups** and add the **tw\_allusers** security group (role) to add all the users to the team.



**Members**

Users **+** **×**

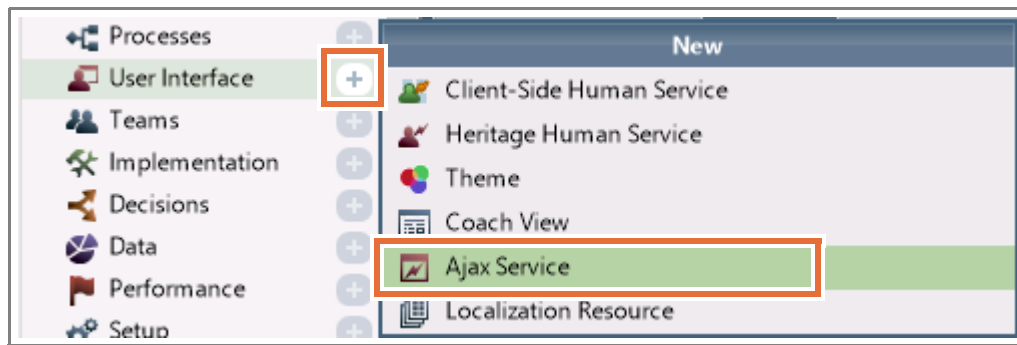
User Groups **+**

**User Groups**

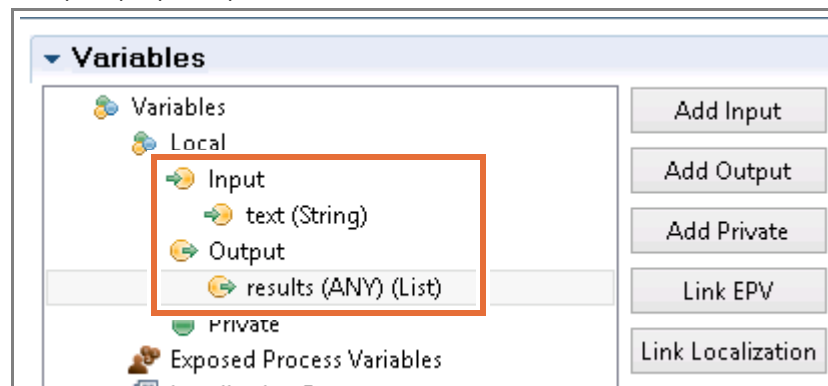
- Debug
- tw\_admins
- tw\_allusers**
- tw\_allusers\_managers
- tw\_authors

- \_\_\_ d. Save your work.
- \_\_\_ 4. Create an Ajax service to retrieve users.
- \_\_\_ a. Switch to the Process Designer client application.

- \_\_\_ b. In the library, click the **plus sign (+)** next to the **User Interface** category. Click **Ajax Service**.



- \_\_\_ c. Name the new Ajax service: Retrieve Users
- \_\_\_ d. Click **Finish**.
- \_\_\_ e. On the **Variables** tab, create an input variable text (String) and an output variable: results (ANY) (List)

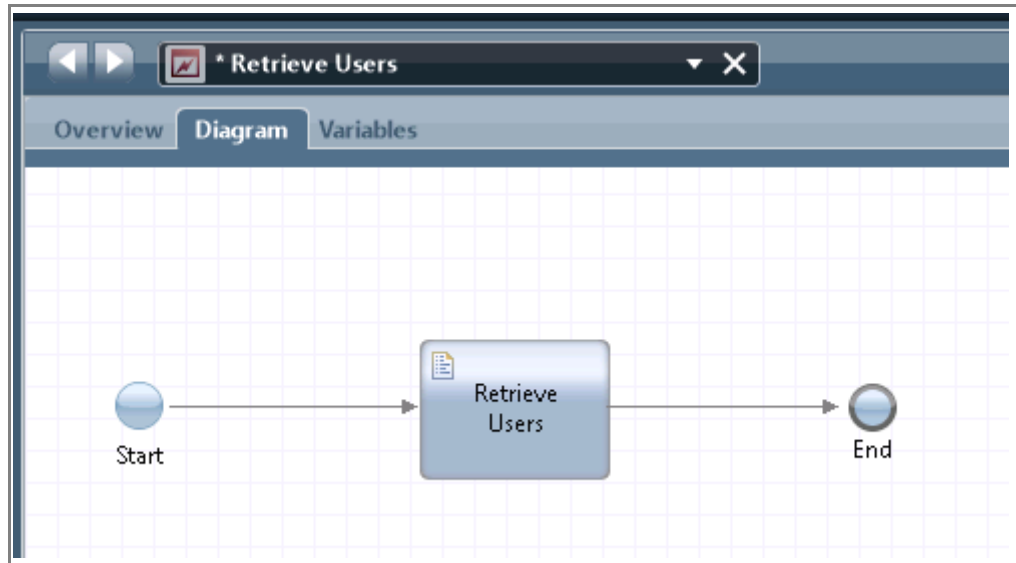


### Important

For the Ajax calls to work with the stock responsive coach view controls, the system requires specific inputs and outputs from the service. The variables must match the variable names that are used in the Default Selection Service Ajax Service found in the Responsive Coaches Toolkit. You can also copy the default service to your library and change the name so the default variables are the correct name and type.

- \_\_\_ f. Click the **Diagram** tab.
- \_\_\_ g. Drag a server script onto the canvas and name it: Retrieve Users

\_\_\_ h. Connect the flows.



\_\_\_ i. Click the **Retrieve Users** server script.

\_\_\_ j. In the **Properties > Implementation** menu, use the JavaScript API to assign all the members of the HR Administrators participant group to the `reviewersList` object. The following code is provided for you in the `C:\labfiles\Lab and Exercise Assets\Exercise code\Exercise 6\Assign All Users.txt` file:

```

tw.local.results = new tw.object.listOf.String();
var users =
tw.system.org.findTeamByName('HR Administrators').allUsers;
for (var i=0; i<users.length; i++) {
    tw.local.results[i] = users[i].name;
}
  
```

▼ Script

```

1 tw.local.results = new tw.object.listOf.String();
2 var users =
3 tw.system.org.findTeamByName('HR Administrators').allUsers;
4 for (var i=0; i<users.length; i++) {
5     tw.local.results[i] = users[i].name;
6 }
      
```

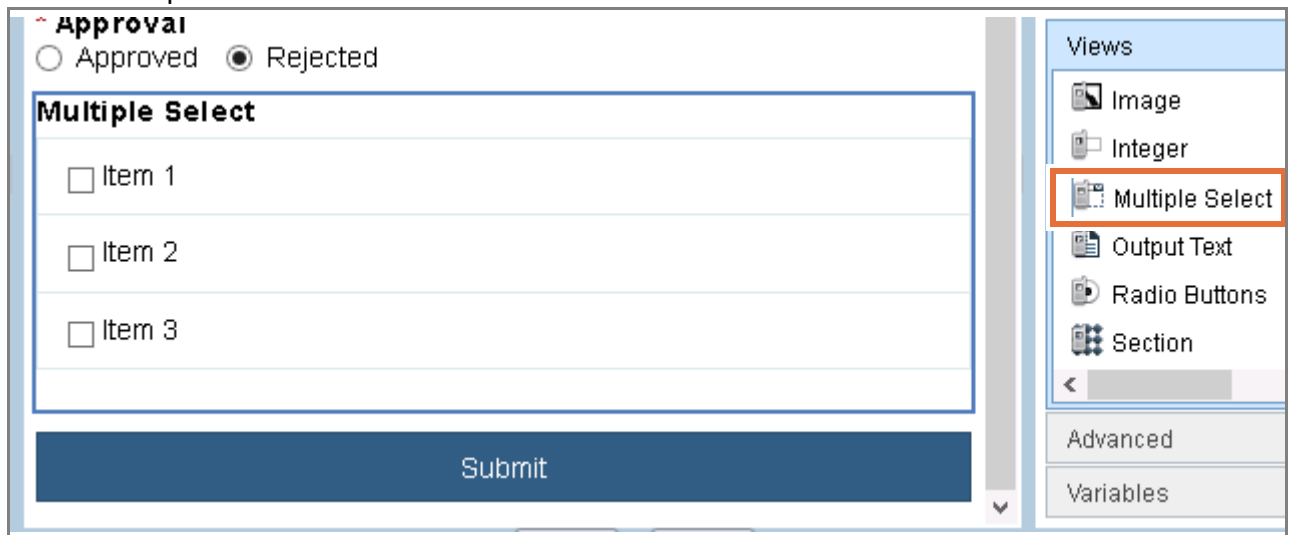
\_\_\_ k. Save your work.

\_\_\_ 5. Update the coaches.

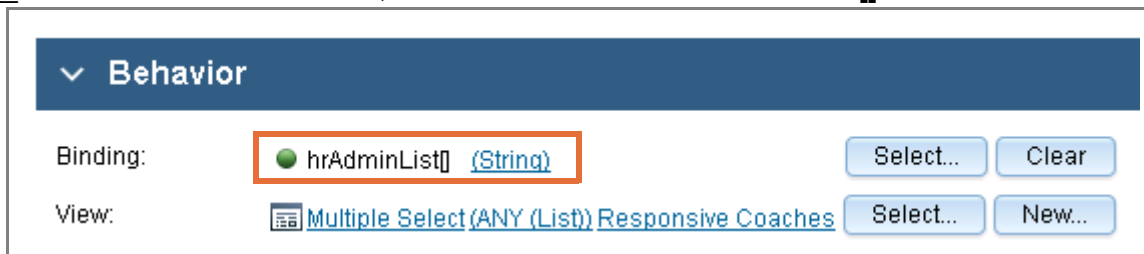
\_\_\_ a. Switch to the web Process Designer and open the **Approve Request** client-side human service.

\_\_\_ b. Click the **Coaches** tab, and then click **Hiring Form**.

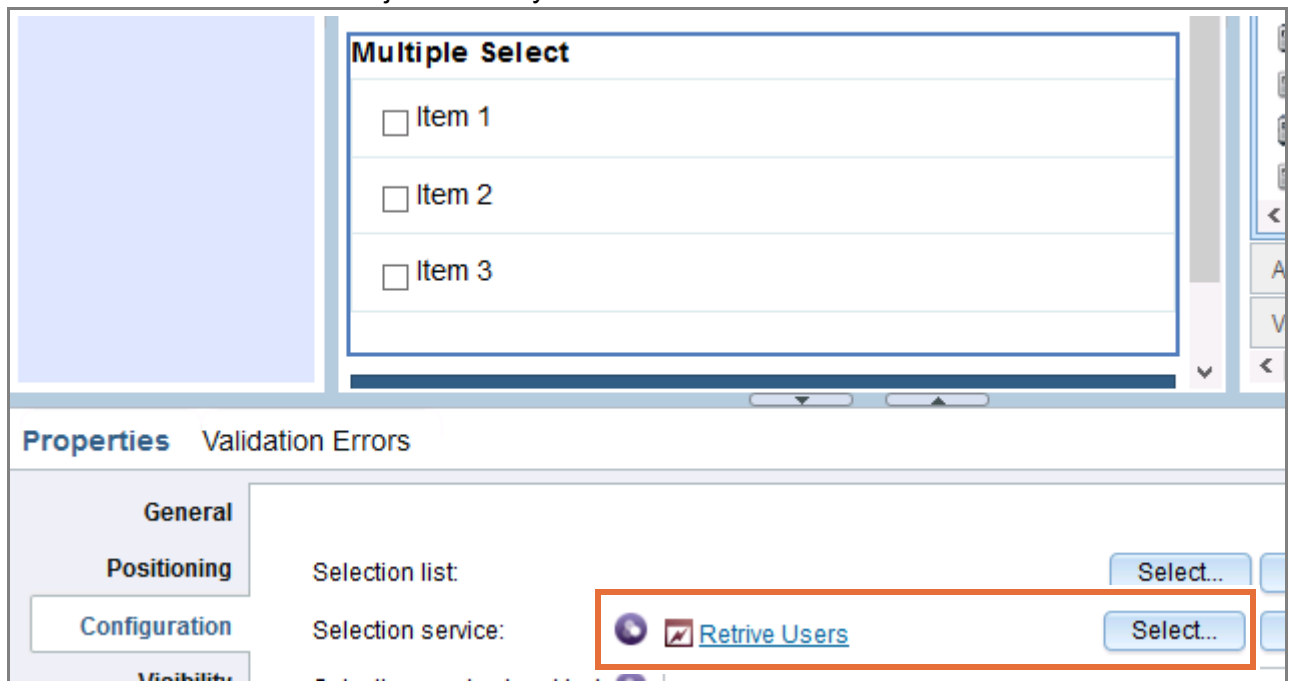
- \_\_\_ c. Drag the **Multiple Select** control from the **Views > Stock** section in the palette to the space above the **Submit** button.



- \_\_\_ d. Click the **Multiple Select** control on the canvas. From the **Properties > General** menu, change the label to: **Override Reviewers List**
- \_\_\_ e. In the Behavior section, bind the control to the **hrAdminList[]** variable.



- \_\_\_ f. From the **Properties > Configuration** menu, change the Selection service to the **Retrieve Users** Ajax service you created.



- \_\_\_ g. Save your work.
- \_\_\_ h. Click **Run** to run the service to test the Ajax call.



### Information

When the Hiring Form is rendered, the Retrieve Users Ajax call is run. The members of the HR Administrators team you created are retrieved and the select list is populated with the members of the team.

**Override Reviewers List**

<input type="checkbox"/> administrator
<input type="checkbox"/> author1
<input type="checkbox"/> user1
<input type="checkbox"/> user2

- \_\_\_ i. Click **Submit** and close the browser.
- \_\_\_ 6. In the **Hiring Request Process**, map the variables at the service level.
  - \_\_\_ a. Return to the **Hiring Request Process** in the web Process Designer.
  - \_\_\_ b. Add a private variable `hrAdminList (String) (List)`

▾ Variables

☐ **Input** +

☐ **Output** +

☒ **Private** +

☐ isNewPosition (String)

☐ isApproved (String)

☐ isCompliant (String)

☐ requisitionDetails (HiringRequisition)

☐ content (ECMDocumentInfo) (List)

☐ cancellationId (String)

☒ **hrAdminList (String) (List)**

☐ Exposed Process variables +

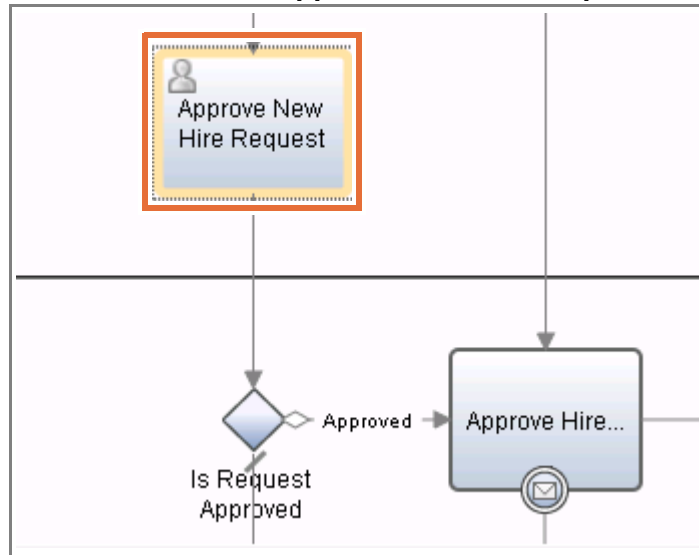
↑

×

↓



- \_\_\_ c. On the **Definition** tab, click the **Approve New Hire Request** activity.



- \_\_\_ d. From the **Properties > Implementation** menu, click **Select** to change the Implementation to: *Approve Request*

- \_\_\_ e. Click the **Properties > Data Mapping** menu.
- \_\_\_ f. In the for Input Mapping section, map the `requisitionDetails` variable to `tw.local.requisitionDetails`.

- \_\_\_ g. In the Output Mapping section, map the output variables:
- requisitionDetails variable to `tw.local.requisitionDetails`
  - isApproved variable to `tw.local.isApproved`
  - hrAdminList variable to `tw.local.hrAdminList`

Output Mapping	
<a href="#">requisitionDetails (HiringRequisi...</a>	tw.local.requisitionDetails
<a href="#">isApproved (String)</a>	tw.local.isApproved
<a href="#">hrAdminList (List of String)</a>	tw.local.hrAdminList

\_\_\_ h. Save your work.

\_\_\_ 7. Pass the `hrAdminList` list variable to the linked process.

- \_\_\_ a. Open the **Approve Hire Request** linked process.
- \_\_\_ b. Add an input variable: `hrAdminList (String) (List)`

Variables

Input

isCompliant (String)

requisitionDetails (HiringRequisition)

hrAdminList (String) (List)

Output

isCompliant (String)

requisitionDetails (HiringRequisition)

Private

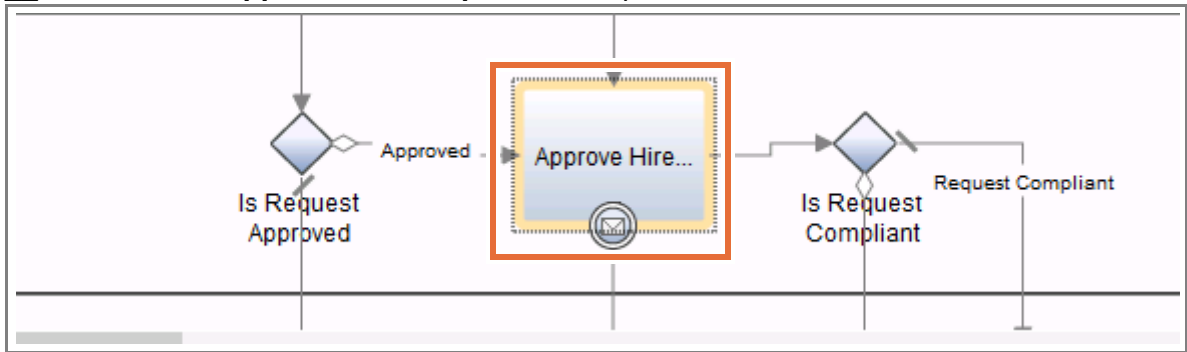
hrAdmin1 (String)

Exposed Process Variables

\_\_\_ c. Save your work.

\_\_\_ d. Return to the **Hiring Request Process**.

- e. Click the **Approve Hire Request** linked process.



- f. Map the `tw.local.hrAdminList` variable in the Hiring Request process to the `hrAdminList` (List of String) input variable.

Input Mapping

tw.local.isCompliant

tw.local.requisitionDetails

tw.local.hrAdminList

isCompliant (String)

requisitionDetails (HiringRequ

hrAdminList (List of String)

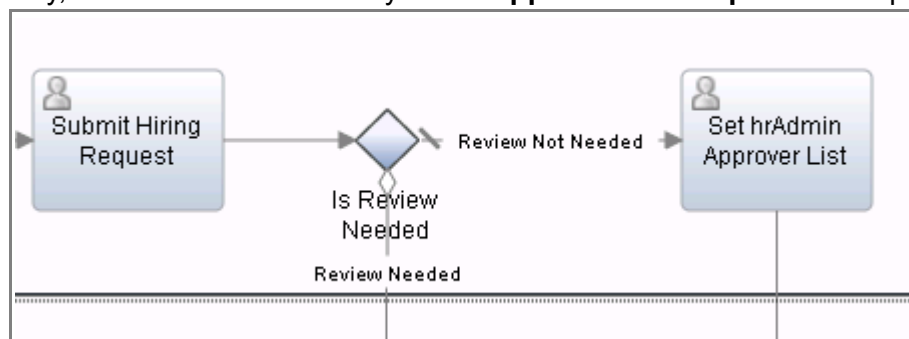
- g. Save your work.
8. Set up a default list of users for the approver list.



### Important

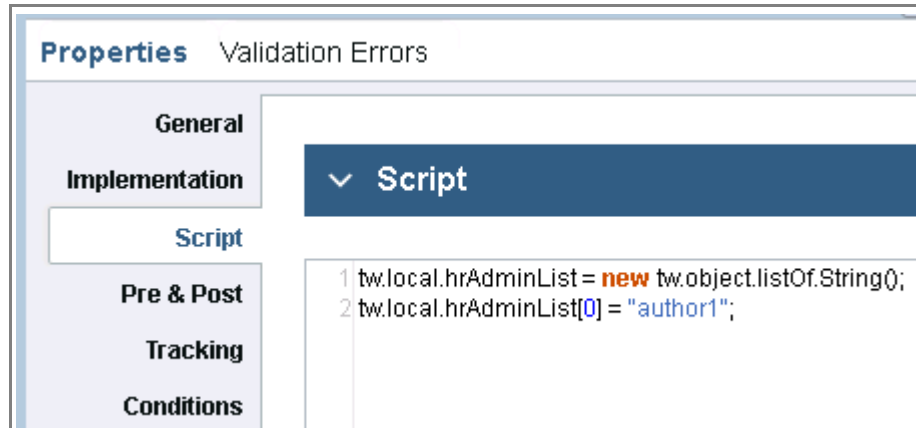
If the task goes to the HR approvers, when the review is not needed, the multi-instance loop must have a default list of reviewers.

- a. In the **Hiring Request Process**, drag an activity onto the canvas in the **Hiring Manager** lane to the right of the **Is Review Needed** decision gateway on the flow that is labeled **Review Not Needed**.
- b. Name the activity: `Set hrAdmin Approver List`
- c. Connect the flow that is labeled **Review Not Needed** from the gateway to the new activity, and from the new activity to the **Approve Hire Request** linked process.



- \_\_\_ d. Click the new activity and change the **Properties > Implementation > Activity Type > Type** to **Script**.
- \_\_\_ e. To add a default user to the approvers when the salary is not compliant, copy the following code and paste it into the **Properties > Script** text area. The example that is shown is saved to the C:\labfiles\Lab and Exercise Assets\Exercise Code\Exercise 6\Add Default user.txt file:

```
tw.local.hrAdminList = new tw.object.listOf.String();
tw.local.hrAdminList[0] = "author1";
```

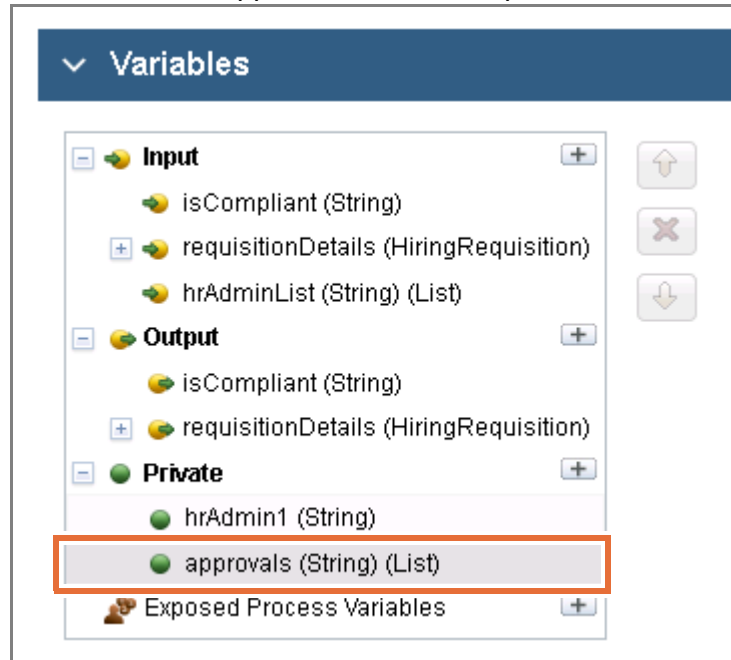


- \_\_\_ f. Save your work.

### 1.3. Convert the approval step to a multi-instance loop and map the output variables

- \_\_\_ 1. Set up the variables.
  - \_\_\_ a. Open the **Approve Hire Request** linked process.
  - \_\_\_ b. Create a private variable: `approvals (String) (List)`

This variable holds all the approvals that are output from the multi-instance loop.



- \_\_\_ c. Save your work.



#### Important

If you map the output of the Override Hire Requests activity to a simple approval object, when the task completes, the approval output variable overwrites the previous approval variable.

- \_\_\_ 2. Create the multi-instance loop.
  - \_\_\_ a. On the **Definition** tab, click the **Override Hire Requests** activity on the process.
  - \_\_\_ b. From the **Properties > General** menu, change the **Behavior > Loop Type** to **Multi-instance Loop**.
  - \_\_\_ c. The Multi-instance loop section appears below the Behavior section. For Start Quantity, enter: `tw.local.hrAdminList.listLength`
  - \_\_\_ d. Verify that the Ordering is **Run in parallel**.
  - \_\_\_ e. Change the Flow condition to **Conditional wait (Complex)**.
  - \_\_\_ f. Select the **Cancel remaining instances** check box.
  - \_\_\_ g. Finally, provide the following **Complex Flow Condition**. Copy the following code and paste it into the code block. The example that is shown is saved to the `C:\labfiles\Lab and Exercise Assets\Exercise Code\Exercise 6\Complex Flow Condition.txt` file:

```
tw.local.approvals[tw.local.approvals.listLength-1]=="0"
```

**Behavior**

Loop type: Multi-instance loop

**Multi-instance loop**

Start quantity: `tw.local.hrAdminList.listLength`

Ordering: Run in parallel

Flow condition: Conditional wait (Complex)

Complex flow condition: `1 tw.local.approvals[tw.local.approvals.listLength-1]=="0"`

Cancel remaining instances: ☒



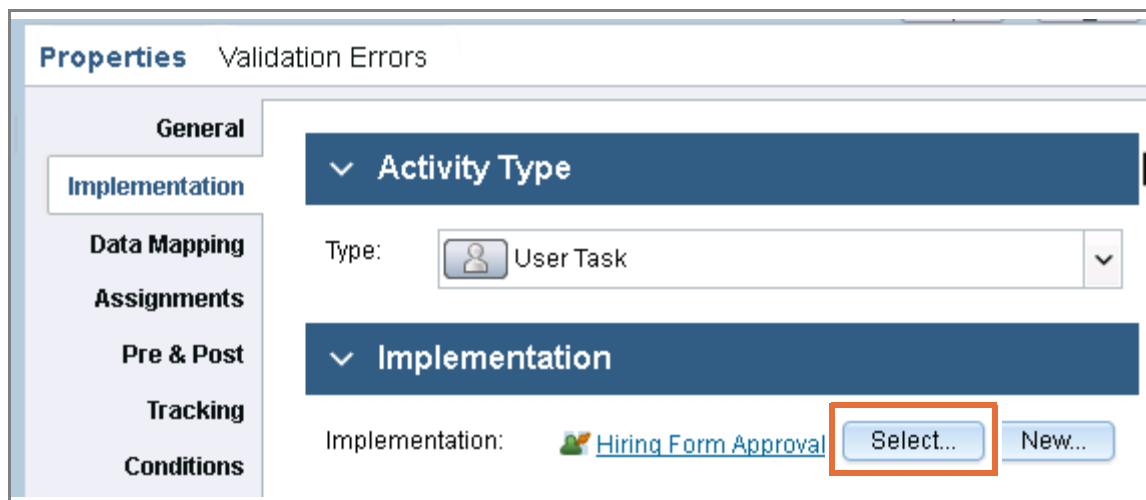
## Information

The start quantity is set to create an instance of the loop for every user who is chosen during the review step. The ordering allows all reviewers to work on their task at the same time instead of waiting for the previous approval to be returned.

The conditional wait complex flow condition tells the system whether to cancel all the remaining instances and move the flow along the path. The added logic looks at the last approval that was completed, and if the approval was approved (true), then the code resolves to false, and canceling the remaining instances does not occur. If the approval was denied (false), then the code resolves to true, and the multi-instance loop cancels any remaining tasks and moves down the flow.

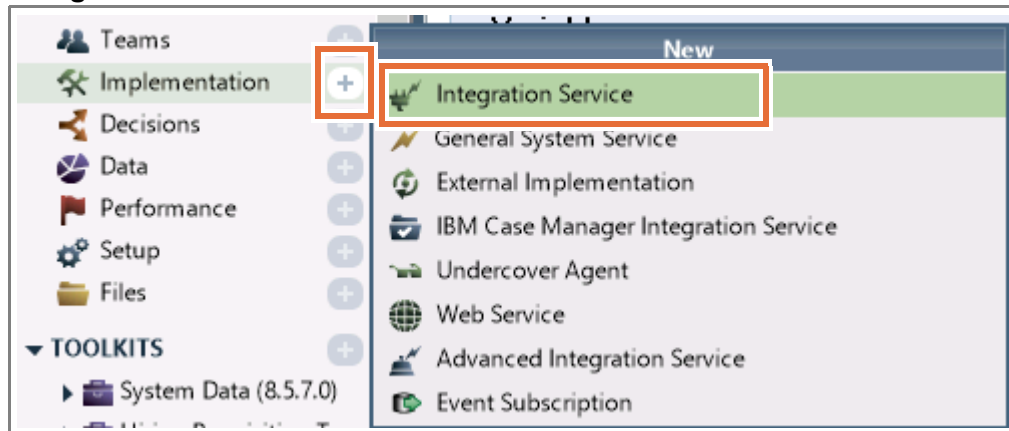
Therefore, any rejection means that the entire override step is rejected, and the process is not required to wait for the remaining reviewers to complete their task. The pattern is implemented where all approvers must approve. You can change the logic to accommodate any type of approval scheme.

- \_\_\_ h. Save your work.
- \_\_\_ 3. Next, you create a team filter, implement the routing for the Multi-instance loop (MIL) step, and then map the inputs and outputs of the Override Hire Requests activity.
- \_\_\_ a. Verify that the **Override Hire Requests** activity is selected, and click the **Properties > Implementation** menu.
- \_\_\_ b. Under the Implementation section, click **Select** to choose the **Hiring Form Approval** client-side human service.

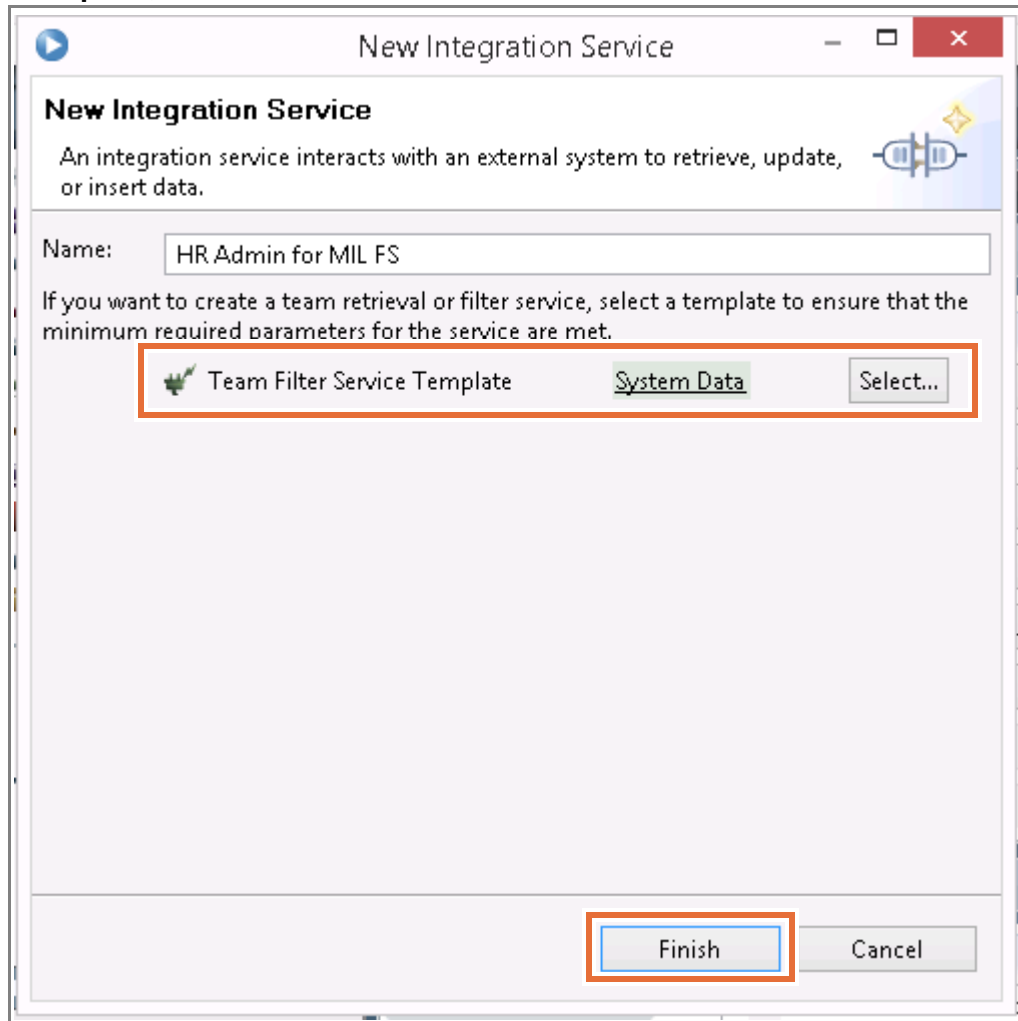


- \_\_\_ 4. Create a team filter service for the multi-instance loop routing.
- \_\_\_ a. Switch to the Process Designer client application.

- \_\_\_ b. In the library, click the **plus sign (+)** next to the **Implementation** category, and then click **Integration Service**.

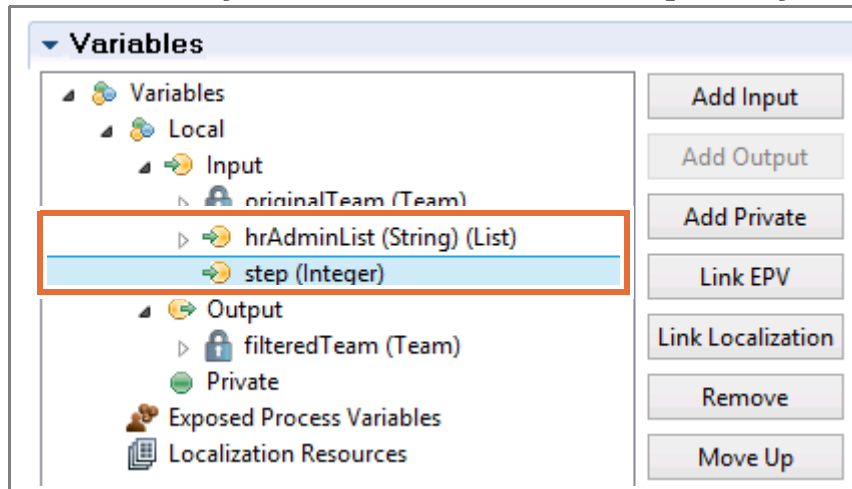


- \_\_\_ c. Name the service: **HR Admin for MIL FS**. Select the **Team Filter Service Template**, and click **Finish**.

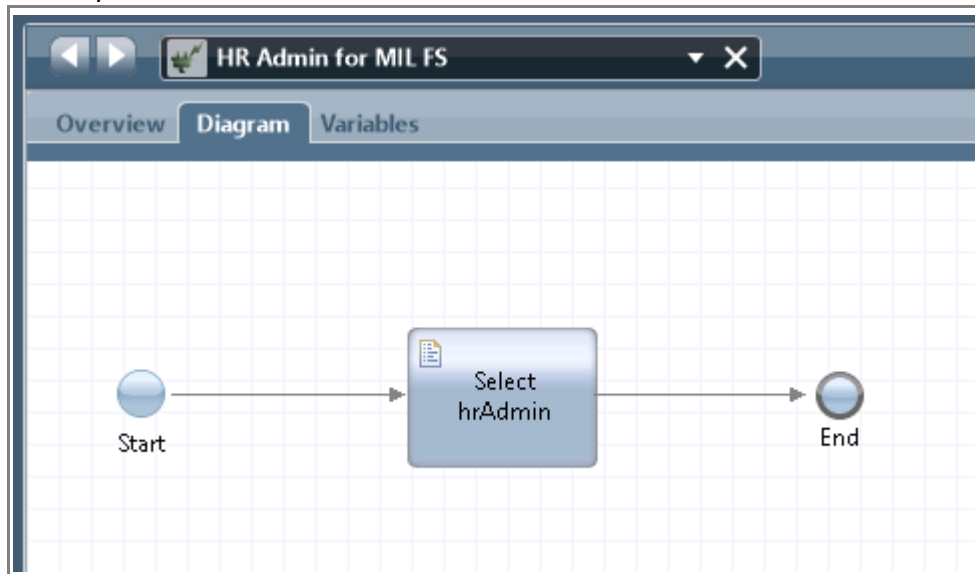




- \_\_\_ d. Click the **Variables** tab. Create two input variables for the filter service. The first is hrAdminList (String) (List) and the second is step (Integer)



- \_\_\_ e. Click the **Diagram** tab. Drag a **Server Script** from the palette onto the canvas. Name the script **Select hrAdmin** and connect the flows.



- \_\_\_ f. Click the **Select hrAdmin** step.
- \_\_\_ g. Click **Properties > Implementation**, copy the following code, and paste it into the code block. The example that is shown is saved to the C:\labfiles\Lab and Exercise Assets\Exercise Code\Exercise 6\Filtered Team.txt file:

```
tw.local.filteredTeam = tw.local.originalTeam;
tw.local.filteredTeam.members = new tw.object.listOf.String();
tw.local.filteredTeam.members[0] = tw.local.hrAdminList[tw.local.step];
```



## Information

This team filter service works with the multi-instance loop to assign a task to each user in the list. This script uses the list of each user name that was selected in the previous task. For each iteration of the loop (counted by using the `tw.local.step` variable), it saves the user name as the single member of the filtered team for that instance of the loop. Therefore, the script assigns a single user a task for each iteration of the loop.

- \_\_\_ h. Save your work.
- \_\_\_ 5. Implement the routing for the multi-instance loop step.
  - \_\_\_ a. Return to the **Approve Hire Request** linked process in the web Process Designer.
  - \_\_\_ b. Click the **Override Hire Requests** activity.
  - \_\_\_ c. Click the **Properties > Assignments** menu.
  - \_\_\_ d. Keep the **Assign To:** field as **Lane** and **User Distribution** as **None**.
  - \_\_\_ e. Select **HR Admin for MIL FS** for the Team filter service.
  - \_\_\_ f. Enter `tw.local.hrAdminList` to map the input to `hrAdminList (List of String)`.
  - \_\_\_ g. Enter `tw.system.step.counter` to map the input to `step (Integer)`.

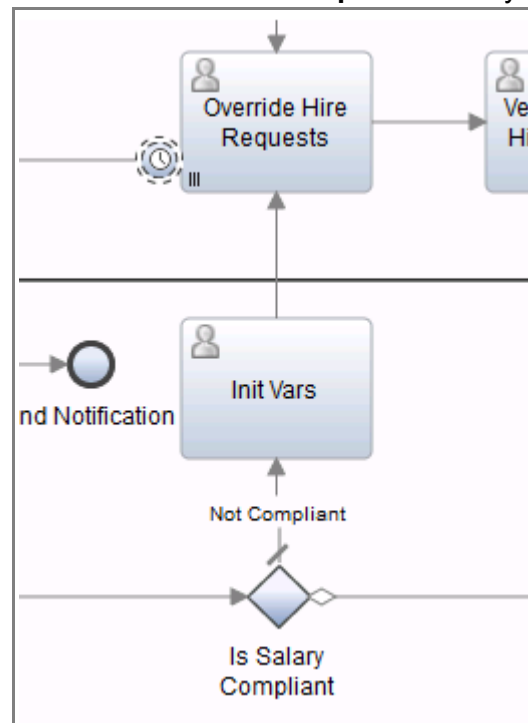
The screenshot displays two panels from a web process designer. The top panel, titled 'Assignments', contains four fields: 'Assign to:' with a dropdown set to 'Lane', 'User distribution:' with a dropdown set to 'None', 'Team filter service:' with a dropdown set to 'HR Admin for MIL FS' (highlighted with an orange box), and 'Experts team:' with a dropdown set to '<none>'. The bottom panel, titled 'Input Mapping', contains two input fields: 'tw.local.hrAdminList' and 'tw.system.step.counter', both highlighted with an orange box. To the right of these fields are two arrows pointing to the output variables 'hrAdminList (List of S...' and 'step (Integer)' respectively.

- \_\_\_ h. Because you defined a team filter service for the activity, the Assign to and User distribution settings are ignored. When the system creates one or more tasks, the system runs the team filter service. The output of the service is a list of user names, team names, or security groups. The list defines an ad hoc team that the task is assigned to. Save your work.

- \_\_\_ 6. Map the inputs and outputs of the Override Hire Requests activity.
  - \_\_\_ a. Click the **Properties > Data Mapping** menu.
  - \_\_\_ b. Map the requisitionDetails input and output to `tw.local.requisitionDetails`.
  - \_\_\_ c. To capture all of the approvals, enter `tw.local.approvals[tw.local.approvals.listLength]` to map the String output of the multi-instance loop to the String list.



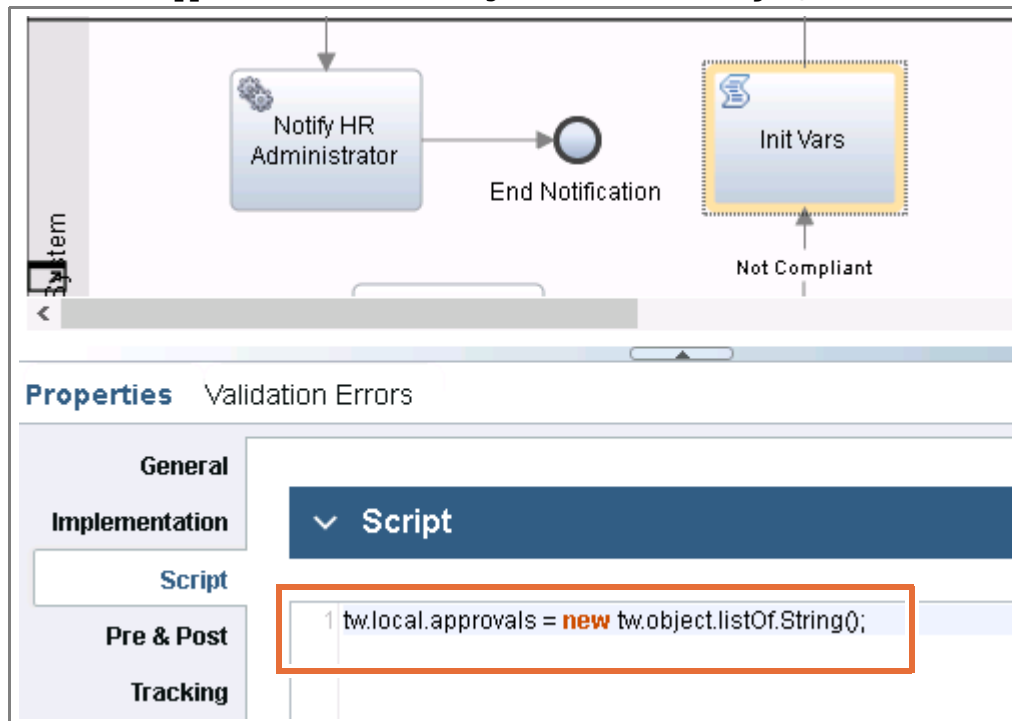
- \_\_\_ d. Mapping the variable causes a warning because the variable cannot be evaluated at design time. Ignore the error.
- \_\_\_ 7. Create a script to initialize the approvals String list variable.
  - \_\_\_ a. Drag an **Activity** from the palette and place it between the **Is Salary Compliant** gateway and the **Override Hire Requests** activity.
  - \_\_\_ b. Name the activity: `Init Vars`
  - \_\_\_ c. Connect the Not Compliant flow between **Is Salary Compliant** and **Init Vars**; then, connect **Init Vars** to the **Override Hire Requests** activity.



- \_\_\_ d. Click the **Init Vars** activity.
- \_\_\_ e. Change the **Properties > Implementation > Activity Type > Type** to **Script** for the **Init Vars** activity.

- \_\_\_ f. From the **Script** menu, use the following script to initialize the approvals object:

```
tw.local.approvals = new tw.object.listOf.String();
```



- \_\_\_ g. Save your work.

## 1.4. Test the multi-instance loop with the condition feature

- \_\_\_ 1. Create an instance of the process.
  - \_\_\_ a. Open the **Hiring Request Process**.
  - \_\_\_ b. Create an instance of the process by clicking **Run**.
  - \_\_\_ c. Click **Run** for the Submit Hiring Request task. Complete and submit the form with the following data:
    - Position Details – Job Level: *Manager*
    - Recruiting Details – New Position: *selected*
    - Compensation Details – Salary To Offer: *100,000*

- \_\_\_ d. Click **Run** for the Approve New Hire Request task. Complete and submit the form with the following data:

- Approval: **Approved**
- Override Reviewers List: Select **author1**, **user1**, and **user2**

**\* Approval**  
☒ Approved ☐ Rejected

**Override Reviewers List**

☐ administrator

☒ author1

☒ user1

☒ user2

Submit

- \_\_\_ e. Expand the **Tasks** section on the right. Verify that three tasks are created for the three reviewers that are selected in the Approve New Hire Request activity.

▼ **Tasks (6)** ACTIVE | COMPLETED | ALL

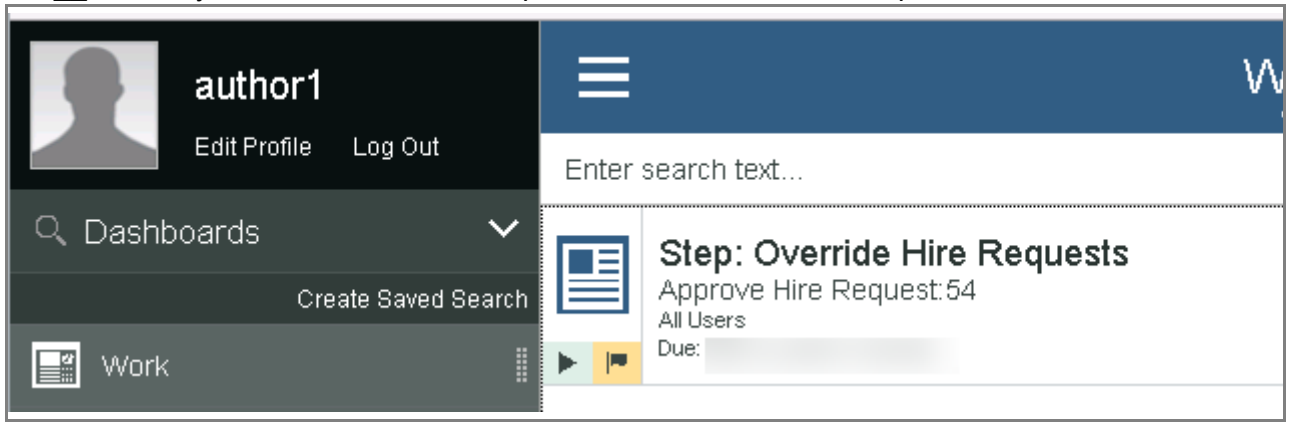
Override Hire Requests ▶   
 The task is assigned to All Users. It is due in 60 minutes.

Override Hire Requests ▶   
 The task is assigned to All Users. It is due in 60 minutes.

Override Hire Requests ▶   
 The task is assigned to All Users. It is due in 60 minutes.

- \_\_\_ f. The system shows that each task is assigned to All Users. Next, you verify that although the task is assigned to the All Users team, the task is assigned to only one user through the filtered team function.
- \_\_\_ 2. Test the conditional loop cancellation feature.
- \_\_\_ a. Open the **Process Portal** by using the Quick Start menu.
- \_\_\_ b. Verify that you are logged in with the **author1** user account.

- \_\_\_ c. Only one task is listed in the portal for this instance of the process.



- \_\_\_ d. Run the task, claim the task, and approve the Override Hire Requests task.  
 \_\_\_ e. Click **Submit**.

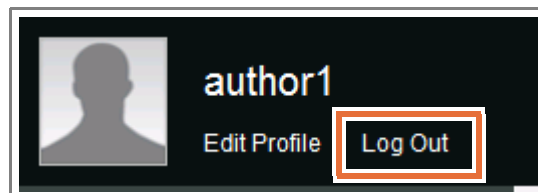
**Hiring Manager Comments**

I approve

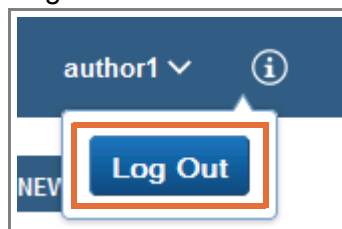
**\* Approval**  
☒ Approved ☐ Rejected

**Submit**

- \_\_\_ f. After the task completes, the inbox is empty. Log out of the Process Portal but leave the tab open.



- \_\_\_ g. Return to the web Process Designer.  
 \_\_\_ h. Refresh the instance. Verify that two active tasks remain in the Tasks section.  
 \_\_\_ i. Log out of the Process Designer.



- \_\_\_ j. Return to the **Process Portal** tab.
- \_\_\_ k. Log in to the portal by using `user1` as the user name and `user01` as the password. Only one task is shown in the inbox for this instance.
- \_\_\_ l. Run the **Override Hire Requests** task and reject the approval.

**Hiring Manager Comments**

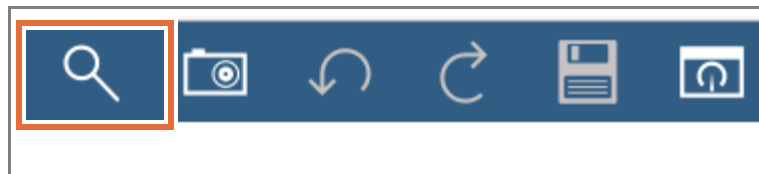
No way buddy.

\* **Approval**

☐ Approved ☒ Rejected

Submit

- \_\_\_ m. Log out of the process portal and close the Process Portal tab.
- \_\_\_ n. Log in to the web Process Designer by using the `author1` user name and `author01` password credentials.
- \_\_\_ o. Open the **HR Recruitment Process** in the web Process Designer.
- \_\_\_ p. Switch to the **Inspector** view.
- \_\_\_ q. Click the **Search** icon.



- \_\_\_ r. Click the **Refresh** button on the left.

**Last modified date**

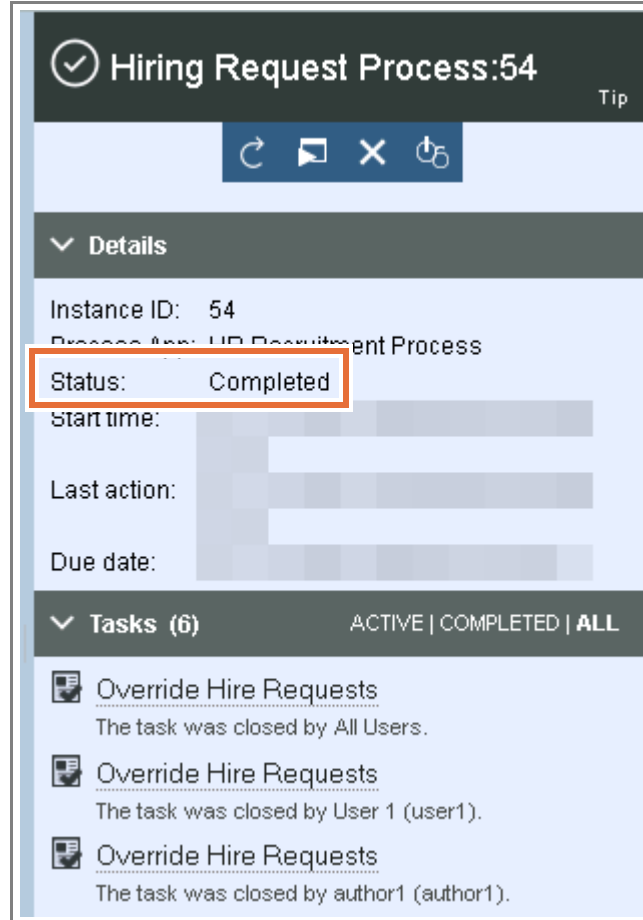
From  Date  Time

To  Date  Time

Refresh

- \_\_\_ s. Click the top instance. The information bar opens on the right. Verify that the multi-instance loop closed the third task and the status of the instance is listed as: **Completed**.

- \_\_\_ t. The Complex Flow Condition evaluated the denial, and it canceled the last approval task. Therefore, only one denial caused the rejection of the entire multi-instance loop, and the process continued without waiting for the remaining approval or denial.



- \_\_\_ u. Return to the **Designer** view.

## End of exercise



## Exercise review and wrap-up

In the first part of the exercise, you implemented services and activities on the process to prepare for the multi-instance loop. You then created a multi-instance loop that cancels remaining instances when a rejection is received. You ran the process and verified that the process meets the requirements.

---

# Exercise 7. Building web service connections

## Estimated time

01:00

## Overview

In this exercise, you learn how to build inbound and outbound web service connections.

## Objectives

After completing this exercise, you should be able to:

- Create an event-based undercover agent
- Build an inbound web service connection
- Build an outbound web service to message the inbound web service

## Introduction

The project sponsor identified a requirement to expose a cancellation service to an external system. The ability to cancel a request must be enabled internally, but also must be available to systems that integrate externally with IBM Business Process Manager.

You must create a cancellation message listener as an inbound web service inside IBM Business Process Manager. Then, create a web service to test or simulate an external system, which connects to the inbound web service. This activity gives you practice in creating web services and calling external SOAP services. Even though the message is targeting the localhost, the approach and concepts that are used in this activity are the same for an external system.

## Requirements

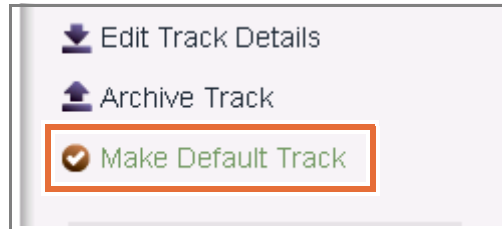
Completion of the previous exercise in this book.

## Section 1. Create an inbound integration



### Important

If you previously imported one of the solution files or are using tracks, select the track from the **Process Apps > Snapshots > Track** menu. Click the **Make Default Track** link on the right to make this track the default track.



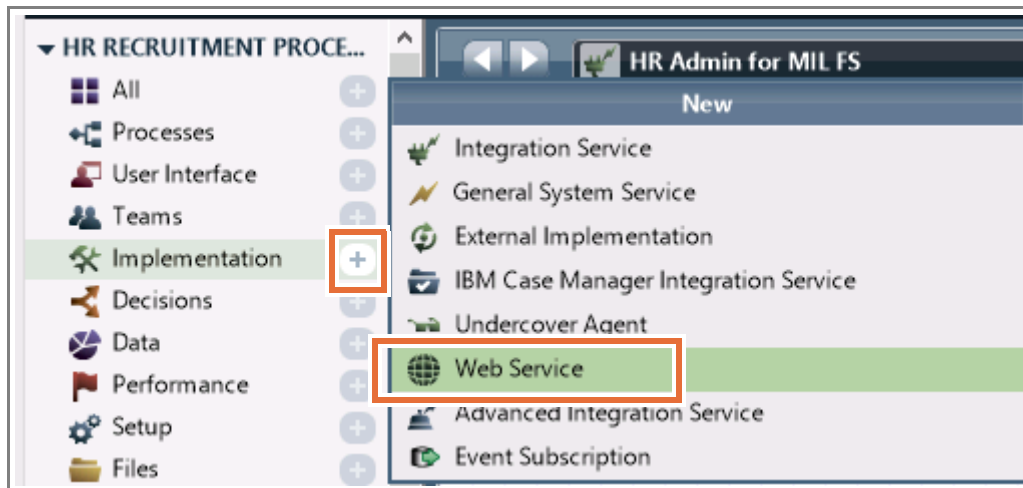
Open the track in the web Process Designer and the Process Designer client application.

### 1.1. Create an inbound web service to trigger the UCA

\_\_\_ 1. Create an inbound web service.

\_\_\_ a. In the Process Designer client application library, click the **plus sign (+)** next to the **Implementation** category.

\_\_\_ b. Click **Web Service**.



\_\_\_ c. Name the web service: `hiringRequestProcess` and click **Finish**.

\_\_\_ d. Click **Add** under the Operations section, and select the **Cancel Hiring Request Trigger** service.

- \_\_\_ e. Under the Operation Detail section, change the **Operation Name** to: cancel

**Operation Detail**

Operation Name:

Attached Service: [Cancel Hiring Request Trigger](#)

Documentation: Edit to add or edit text."/>

- \_\_\_ f. Under the Behavior section, verify that the **Protected** check box is not selected.

**Behavior**

Protected: ☐

WSDL URI: <https://ws2012r2x64:9443/tea...iringRequestProcess.tws?WSDL>

Target namespace scheme:

Target namespace:

SOAP Version: ☐ 1.1 ☒ 1.2

- \_\_\_ g. Save your work.
- \_\_\_ h. Click the **WSDL URI** link.

Protected: ☐

WSDL URI: <https://ws2012r2x64:9443/tea...iringRequestProcess.tws?WSDL>

Target namespace scheme:

- \_\_\_ i. The WSDL is loaded in your browser. The information about the web service you created is shown to include the methods available, the variable inputs and outputs, and metadata about the service call. Highlight the address in the browser address bar, right-click, and click **Copy**.

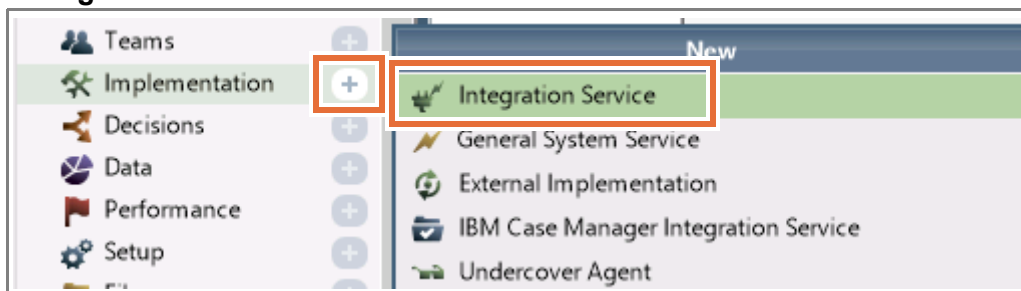


- \_\_\_ j. The URL is used to define the server setting for the **Web Service Integration** component in the outbound integration in the following steps.

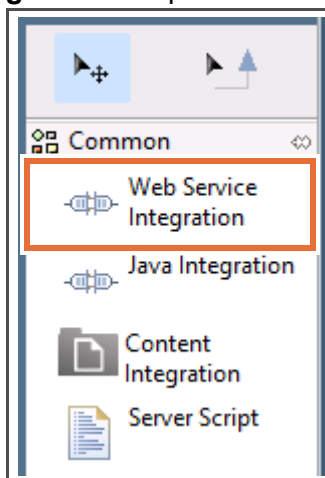
## Section 2. Create an outbound integration

### 2.1. Test the inbound integration by creating an outbound integration

- \_\_\_ 1. Create an outbound integration service.
  - \_\_\_ a. Return to the Process Designer client application.
  - \_\_\_ b. In the library, click the **plus sign (+)** next to the **Implementation** category, and select **Integration Service**.

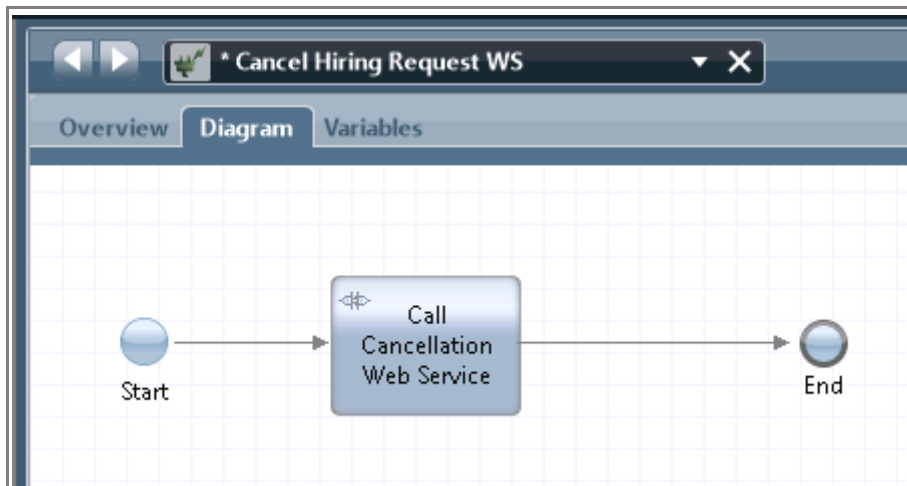


- \_\_\_ c. Name the service: Cancel Hiring Request WS
  - \_\_\_ d. Do not select a template, and click **Finish**.
- \_\_\_ 2. Add the outbound integration component to the service.
    - \_\_\_ a. Drag a **Web Service Integration** component from the palette onto the canvas.



- \_\_\_ b. Name the step: Call Cancellation Web Service

- \_\_\_ c. Connect the flows.



- \_\_\_ d. With the **Call Cancellation Web Service** step selected, click the **Properties > Implementation** menu.

- \_\_\_ e. Click the **Use the Process Application Settings** editor to add a server link.



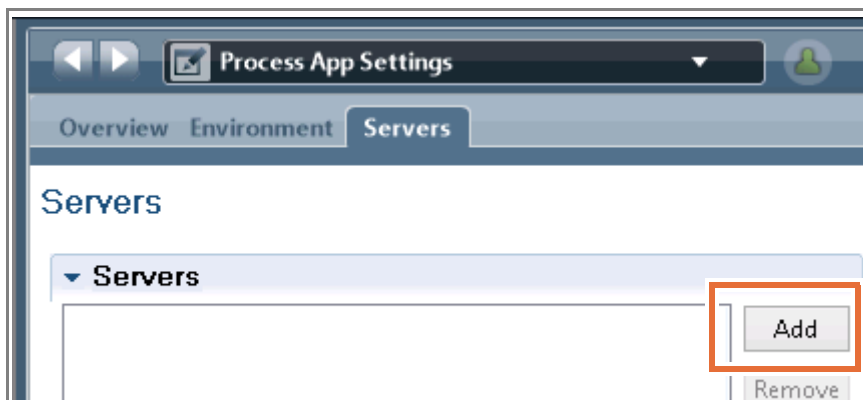
### Note

This link takes you to the Process App Settings page. You can define servers by using the client application or the web Process Designer.

- \_\_\_ 3. Add a server definition to the development environment.

- \_\_\_ a. Click the **Servers** tab.

- \_\_\_ b. Click **Add** to create a server definition.



- \_\_\_ c. Under the Server Details section, name the server: **IBMBPM**

- \_\_\_ d. Change the Type to **Web Service**.

 A screenshot of the 'Server Details' section. The 'Name' field is filled with 'IBMBPM'. The 'Type' dropdown menu is set to 'Web Service'. The 'Description' field has a placeholder text: 'Click [Edit](#) to add or edit text.' There is an '(Edit)' link below the description field.

- \_\_\_ e. In the Server Locations section, change the Environment Type to **Development**. Paste the URL that you copied into the **WSDL URL** field.

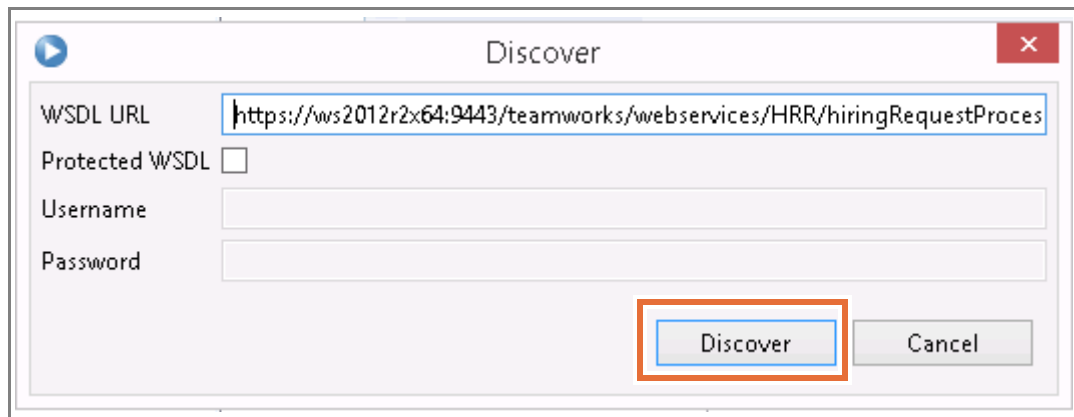
 A screenshot of the 'Server Locations' section. The 'Environment Type' dropdown is set to 'Development'. The 'WSDL URL' field contains the text '/hiringRequestProcess.tws?WSDL'. To the right of the URL field are three buttons: 'Browse...', 'View', and 'Discover'. Below the URL field is a 'Protected WSDL' checkbox, which is unchecked. There are also fields for 'Username' and 'Password'.


### Hint

If you need to copy the URL again, you can return to the **hiringRequestProcess** web service in the Process Designer client application. Return to the Process Apps Settings page to paste the URL.

- \_\_\_ f. Save your work.

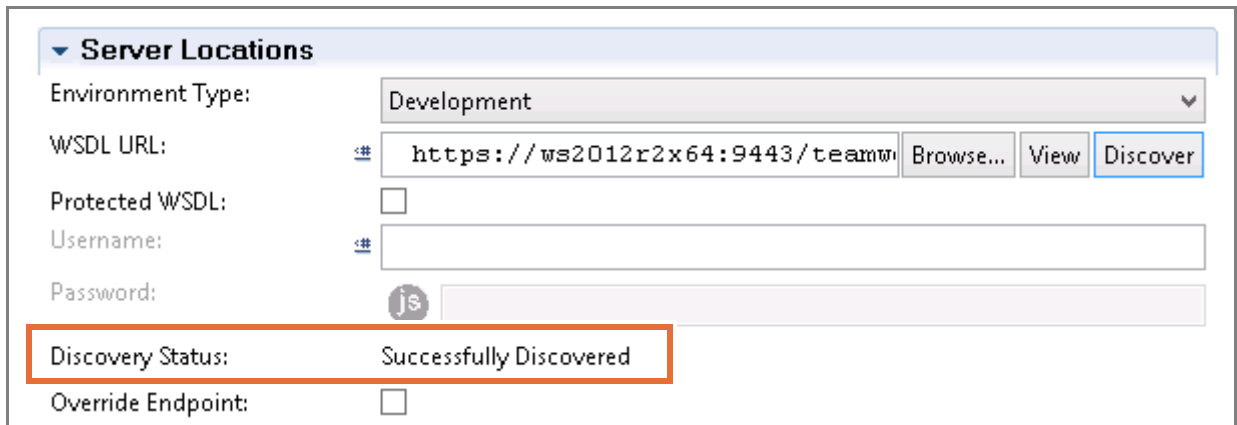
- \_\_\_ g. Click **Discover** next to the WSDL URL you pasted, and click **Discover** again on the menu.



The screenshot shows a 'Discover' dialog box with the following fields and buttons:

- WSDL URL:** `https://ws2012r2x64:9443/teamworks/webservices/HRR/hiringRequestProces`
- Protected WSDL:** ☐
- Username:** (empty text field)
- Password:** (empty text field)
- Buttons:** 'Discover' (highlighted with a red rectangle) and 'Cancel'.

- \_\_\_ h. If everything is correctly configured, the Discovery Status changes to **Successfully Discovered**.



The screenshot shows the 'Server Locations' section with the following details:

- Environment Type:** Development
- WSDL URL:** `https://ws2012r2x64:9443/teamw` (with 'Browse...', 'View', and 'Discover' buttons)
- Protected WSDL:** ☐
- Username:** (empty text field)
- Password:** (empty text field with a 'js' icon)
- Discovery Status:** **Successfully Discovered** (highlighted with a red rectangle)
- Override Endpoint:** ☐

- \_\_\_ 4. Implement the outbound integration component with the server environment variable you created.
- \_\_\_ a. Return to the **Cancel Hiring Request WS** integration service in the Process Designer client application.
  - \_\_\_ b. Click the **Call Cancellation Web Service** step.
  - \_\_\_ c. Click the **Properties > Implementation** menu.



### Important

If IBM Business Process Manager is not available in the list, click another object on the canvas. Then, click back on the **Call Cancellation Web Service** step, and repeat steps b, c, and d.

- \_\_\_ d. Select **IBMBPM** for the Web Service option.



- \_\_\_ e. From the **Operations** list, select the `cancel(NameValuePair)` operation. The server reads the WSDL and selects the SOAP version for you.

**Process Application Selection**

Web Service: IBMBPM

Operations: **cancel(NameValuePair)**

SOAP version: ☐ SOAP 1.1 ☒ SOAP 1.2

[Use the Process Application Settings editor to add](#)

- \_\_\_ f. On the **Variables** tab, add an input variable named: `requestId (NameValuePair)`

**Variables**

Variables

Local

Input

**requestId (NameValuePair)**

Output

Private

Exposed Process Variables

Localization Resources

- \_\_\_ g. Click the **Diagram** tab. Click the **Call Cancellation Web Service** component. Open the **Properties > Data Mapping** menu.
- \_\_\_ h. In the Input Mapping section, map the `tw.local.requestId` variable to the input of the Call Cancellation Web Service. Leave Input Header Mapping and Output Header Mapping blank.

**Properties** Validation Errors Where Used

Step

Implementation

Header

**Data Mapping**

Pre & Post

**Input Mapping**

tw.local.requestId input

**Input Header Mapping**

- \_\_\_ 5. Save your work.

## 2.2. Test the integrations

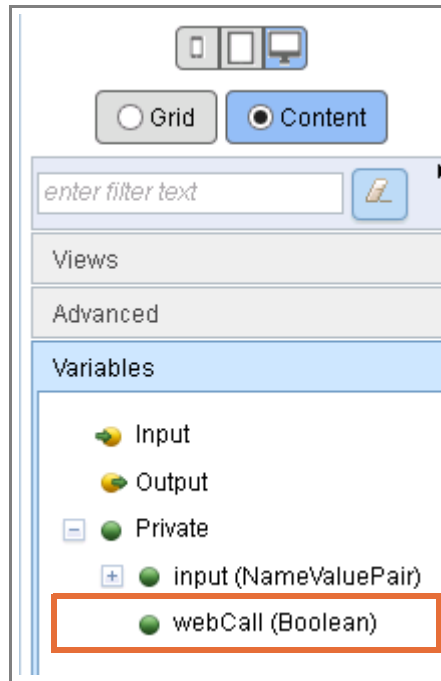
- \_\_\_ 1. Modify the Cancel Hiring Request Test service to cancel an instance with a UCA or a web service call.
- \_\_\_ a. Open the **User Interface > Cancel Hiring Request Test** client-side human service in the web Process Designer.

- \_\_\_ b. Add a private variable: webCall (Boolean)

The screenshot shows a 'Details' panel for a variable named 'webCall'. The panel has a dark blue header with a dropdown arrow and the word 'Details'. Below the header, there is a 'Name:' field containing 'webCall'. To the right of the 'Name:' field is a back arrow icon. Below the 'Name:' field is a 'Documentation:' label followed by a rich text editor. The rich text editor has a toolbar with icons for bold (B), italic (I), underline (U), bulleted list, numbered list, link, and unlink. Below the rich text editor is an 'Is list:' checkbox, which is currently unchecked. Below the 'Is list:' checkbox is a 'Variable type:' label. To the right of the 'Variable type:' label is a small icon of three overlapping circles, followed by the text 'Boolean'. To the right of 'Boolean' is a link labeled 'System Data'. To the right of 'System Data' are two buttons: 'Select...' and 'New...'.

- \_\_\_ c. Click the **Coaches** tab and click the **Test Cancellation** coach.

- \_\_\_ 2. Create a control on the page to allow users to select the type of test to perform.
  - \_\_\_ a. Expand the **Variables** tray inside the palette on the right.
  - \_\_\_ b. Drag the **Private > webCall** variable onto the canvas between the **Instance ID** input and the **OK** button.



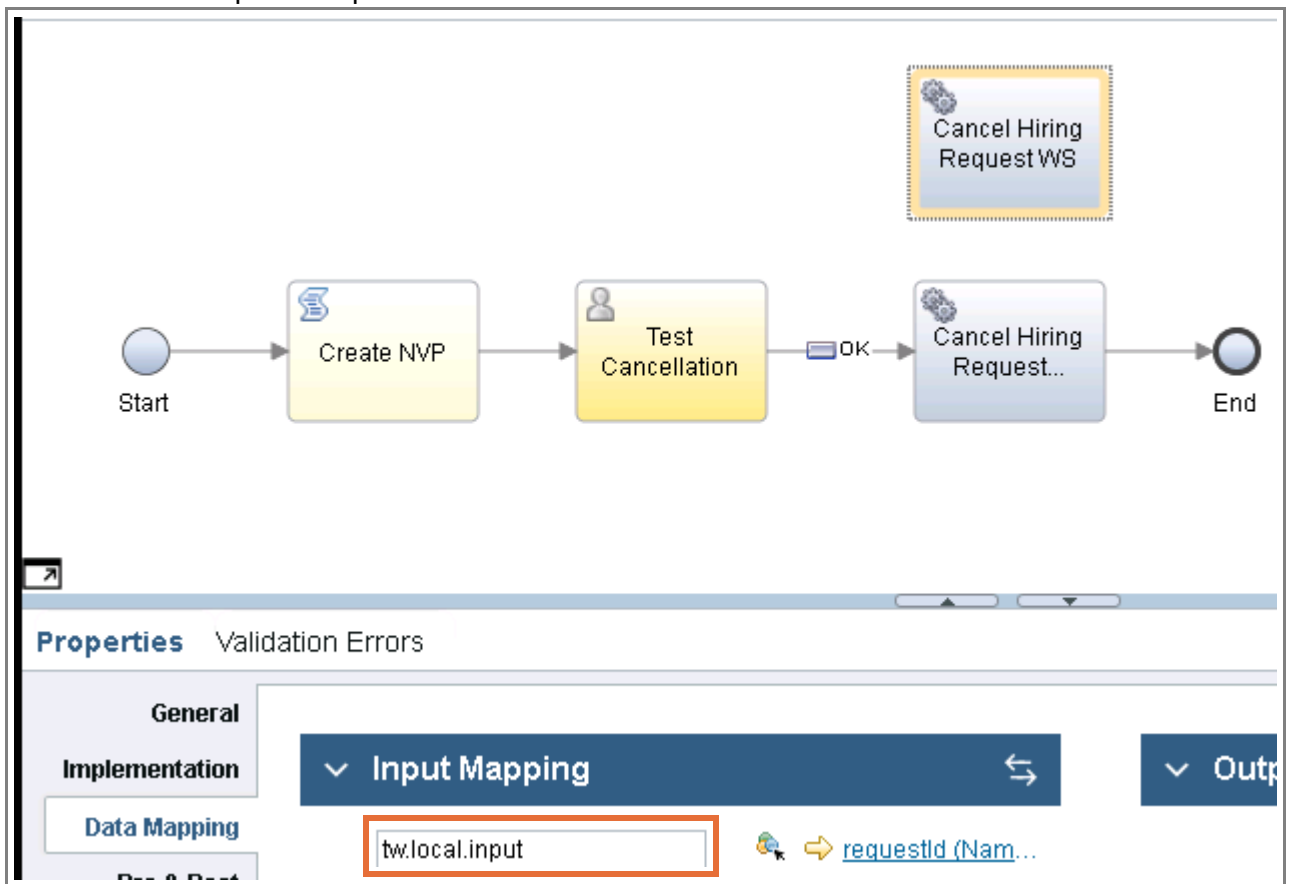
- \_\_\_ c. Change the WebCall control label to `Type of Test`.
- \_\_\_ d. Click the **Properties > Configuration** menu:

- \_\_\_ e. Set **Show as:** Switch
- \_\_\_ f. Set **True label:** Web Service
- \_\_\_ g. Set **False label:** UCA Message

The screenshot shows the IBM Design Studio interface. At the top, there are tabs: Overview, Diagram, Variables, and Coaches. The 'Coaches' tab is active, showing a 'Test Cancellation' service. Below the service name, there is a section for 'Instance ID' and a 'Type of Test' dropdown menu. The 'Type of Test' is currently set to 'UCA Message'. Below this, there is an 'OK' button. At the bottom, the 'Properties' panel is open, showing various tabs: General, Positioning, Configuration, Visibility, and HTML Attributes. The 'Configuration' tab is selected, and it contains a table with three rows: 'Show as', 'True label', and 'False label'. Each row has a radio button, a dropdown menu, and a text input field. The 'Show as' dropdown is set to 'Switch', 'True label' is 'Web Service', and 'False label' is 'UCA Message'. These three rows are highlighted with a red box. Below the table, there is a 'Style' button.

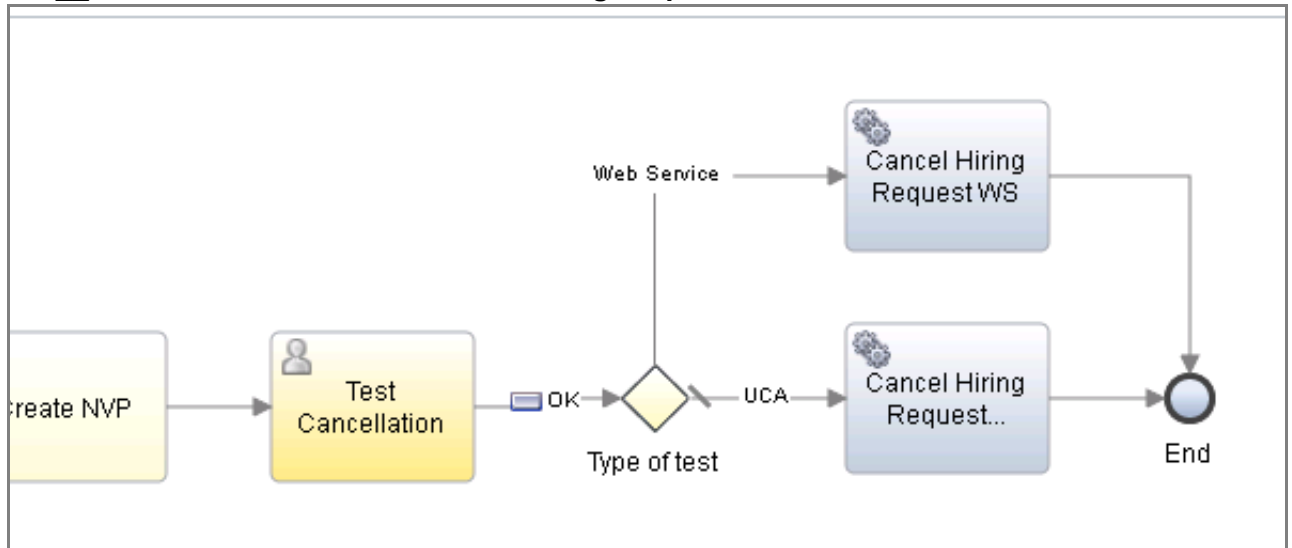
- \_\_\_ 3. Add the **Cancel Hiring Request WS** service to the service flow.
  - \_\_\_ a. Click the **Diagram** tab.
  - \_\_\_ b. Drag a **service** from the palette right above the Cancel Hiring Request service step on the canvas.
  - \_\_\_ c. Name it: Cancel Hiring Request WS
  - \_\_\_ d. Under **Properties > Implementation > Behavior**, ensure that **Call a service** is selected.
  - \_\_\_ e. Click **Select** and under the Integration Service category, select **Cancel Hiring Request WS**.

- \_\_\_ f. Open the **Properties > Data Mapping** menu, and map the `tw.local.input` variable to the `requestId` input variable.



- \_\_\_ 4. Route the flow dependent on which type of test the user selects.
- \_\_\_ a. Add an **Exclusive Gateway** to the right of the Test Cancellation coach, and name it: Type of test
  - \_\_\_ b. Create a flow from the **gateway** to **Cancel Hiring Request WS**, and name this flow: Web Service. Select the **Name visible** check box.
  - \_\_\_ c. Create a second flow from the **gateway** to the **Cancel Hiring Request Trigger** service, and name this flow: UCA. Select the **Name visible** check box.
  - \_\_\_ d. Connect the flow from the **Test Cancellation** coach to the **gateway**.

- \_\_\_ e. Connect a flow from **Cancel Hiring Request WS** to the **End** event.



- \_\_\_ 5. Configure the gateway.

- \_\_\_ a. Click the **Type of test** gateway. Click the **Properties > Implementation** menu.
- \_\_\_ b. Implement the decision gateway by making the **UCA** flow the Default Sequence Flow. This flow connects to the Cancel Hiring Request Trigger service.
- \_\_\_ c. Enter `tw.local.webCall` in the Web Service path, select `==` from the operands menu, and enter `true` in the last field.

**Properties** Validation Errors

**General**

**Implementation**

**Pre & Post**

**Decisions**

Web Service:  ==

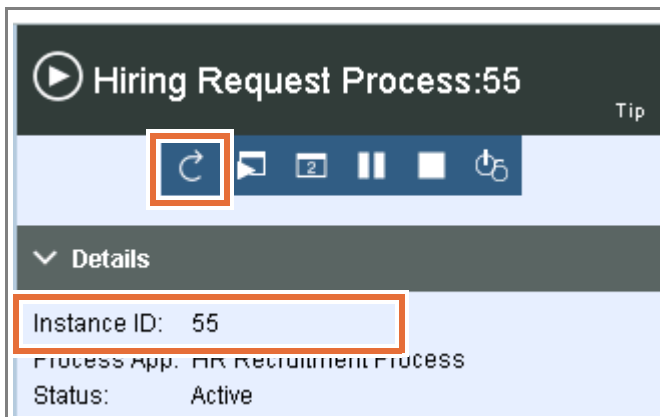
Default flow:

- \_\_\_ d. Save the service.

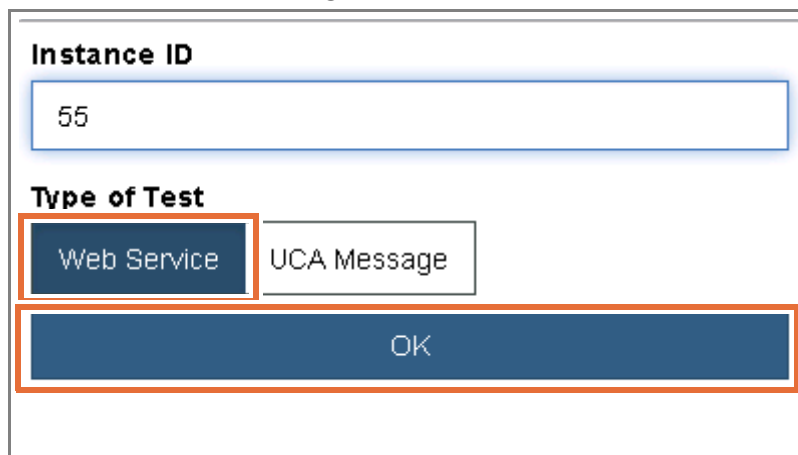
- \_\_\_ 6. Test the outbound web service call.

- \_\_\_ a. In the web Process Designer, open the **Hiring Request Process**.
- \_\_\_ b. Create an instance of the process by clicking **Run**.
- \_\_\_ c. Run the **Step: Submit Hiring Request** activity.
- \_\_\_ d. Use the following values for the variables on the form:
- **Position Details > Job level** set to *Manager*
  - **Recruiting Details > New Position** set to **false** (not selected)
  - **Compensation Details > Salary to offer** set to *100,000*

- \_\_\_ e. Refresh the instance details until you see that the **Override Hire Requests** task appears in the Locations section of the information window. Note the instance ID.

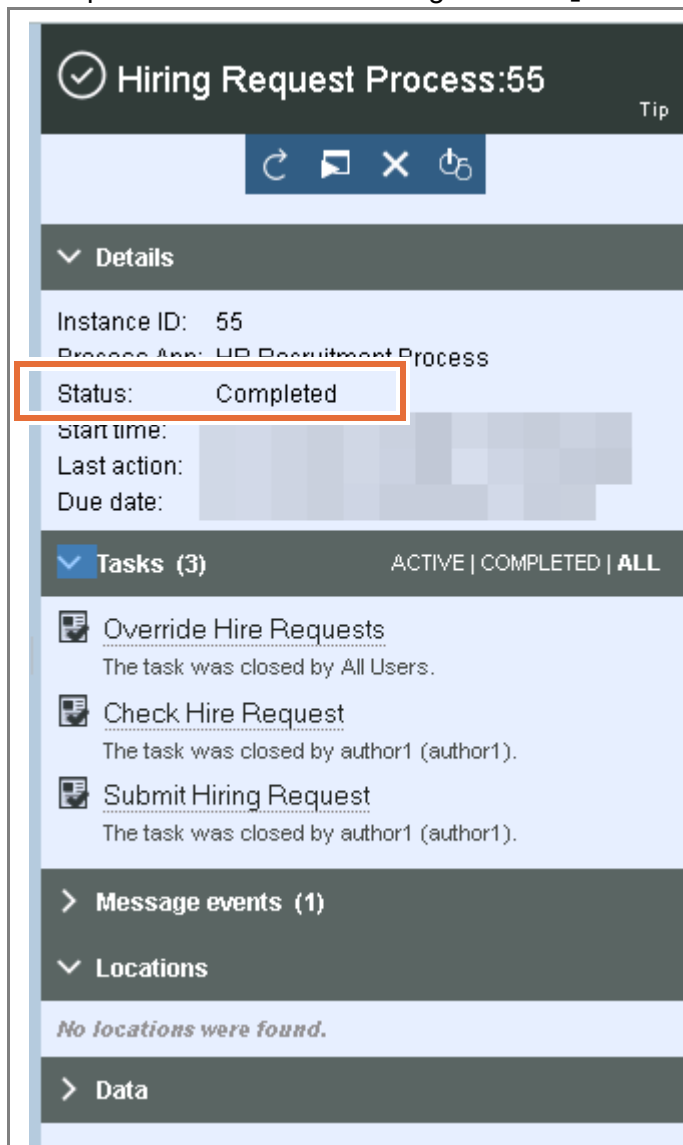


- \_\_\_ f. Return to the **Designer** view.
- \_\_\_ g. Open the **Cancel Hiring Request Test** client-side human service.
- \_\_\_ h. Click **Run**.
- \_\_\_ i. Select the **Web Service** option.
- \_\_\_ j. Enter the Instance ID and click **OK**.



- \_\_\_ k. Close the service window and return to the **Hiring Request Process**.
- \_\_\_ l. Click the **Inspector** tab.

- \_\_\_ m. Refresh the instance details window. Verify that the message event consumes all the tokens inside the linked process and moves the process down the message event flow. The instance completes and the status changes to **Completed**.



**End of exercise**



## Exercise review and wrap-up

In the first part of the exercise, you created an inbound integration by using an existing generic system service to implement the service. You viewed the WSDL in the browser. You then created an outbound web service to test the inbound web service integration. You configured the outbound web service, discovered the endpoint methods, and selected the method to call. Finally, you modified an existing test harness to call both the internal UCA and the outbound web service to test the ability to cancel an in-flight process.

---

# Appendix A. Data dictionary

## Training database

The JNDI is `jdbc/TrainingDB`.

The training database that is used in this course uses the following database tables.

## Part 1: Departments

### Structure

Column	Type	Null allowed
departmentCode	varchar (5)	No
divisionCode	varchar (5)	No
departmentName	varchar (50)	Yes

### Data

departmentCode	divisionCode	departmentName
101	201	Marketing
102	201	Finance
103	202	Engineering
104	202	Professional Services
105	203	HR

## Part 2: Divisions

### Structure

Column	Type	Null allowed
divisionCode	varchar (5)	No
divisionName	varchar (50)	Yes

### Data

divisionCode	divisionName
201	APAC
202	US
203	EMEA

## Part 3: JobLevels

### Structure

Column	Type	Null allowed
--------	------	--------------

jobLevelCode	varchar (5)	No
jobLevelName	varchar (50)	Yes

## Data

jobLevelCode	jobLevelName
5001	Jr Associate
5002	Associate
5003	Manager
5004	Sr Manager
5005	Director
5006	Vice President
5007	President

## Part 4: Positions

### Structure

Column	Type	Null allowed
id	integer	Yes
positionStatus	integer	Yes
jobTitle	varchar(50)	Yes
jobDescription	varchar(4000)	Yes
jobLevel	char(10)	Yes
numberOfDirectReports	integer	Yes
salaryToOffer	double	Yes
bonus	double	Yes
department	varchar(50)	Yes
departmentManager	varchar(50)	Yes
comments	varchar(4000)	Yes

## Part 5: IncidentCategory

### Structure

Column	Type	Null allowed
categoryID	varchar(5)	No
categoryName	varchar(50)	Yes

## Data

categoryID	categoryName
1001	Collision
1002	Theft
1003	Natural Event or Disaster
1004	Other

## Part 6: IncidentType

### Structure

Column	Type	Null allowed
typeID	varchar(5)	No
categoryID	varchar(5)	No
typeName	varchar(100)	Yes

## Data

typeID	categoryID	typeName
2001	1001	Collision with another vehicle
2002	1001	Collision with a stationary object
2003	1001	Collision with a cyclist
2004	1001	Collision with a pedestrian
2005	1001	Collision with an animal
2006	1002	Theft of entire vehicle
2007	1002	Theft of stereo
2008	1002	Theft of items in vehicle
2009	1002	Theft of part of vehicle, not listed above
2010	1003	Fire
2011	1003	Flood
2012	1003	Hail damage
2013	1003	Other storm damage
2014	1004	Glass damage
2015	1004	Pothole damage

2016	1004	Parking lot damage by shopping cart
2017	1004	Other



IBM Training

