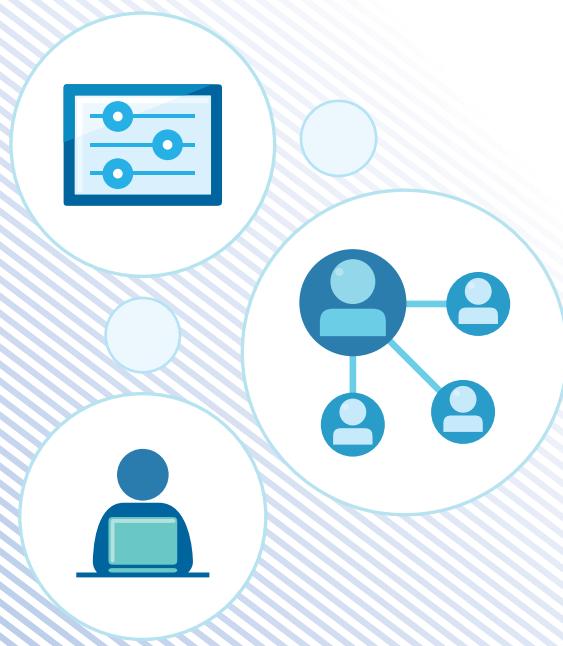


Course Exercises Guide

Process Implementing with IBM Business Process Manager V8.6 - I

Course code WB827 / ZB827 ERC 1.1



March 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.

© Copyright International Business Machines Corporation 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

Trademarks	v
Exercises description	vi
Exercise 1. Playback 0: Creating a process with ad hoc activities	1-1
How to use the course exercise instructions	3
User IDs and passwords	4
Part 1: Start IBM Business Process Manager and IBM Process Designer	1-6
Part 2: Create a process application	1-16
Part 3: Create a process	1-20
Part 4: Create process activities	1-23
Part 5: Create process variables	1-37
Part 6: Create process folders	1-42
Part 7: Test the process activities in Process Portal	1-45
Exercise 2. Playback 0: Creating a structured process	2-1
Part 1: Open the business process in Process Center	2-5
Part 2: Add lanes to the business process	2-7
Part 3: Model the activities in the process	2-11
Part 4: Create a nested process	2-18
Part 5: Attach the linked process	2-29
Exercise 3. Playback 0: Controlling process flow	3-1
Part 1: Create gateways for parent process	3-2
Part 2: Create gateways for the nested process	3-15
Part 3: Modeling timer intermediate events	3-26
Exercise 4. Validating the process model	4-1
Part 1: Verify the process model	4-2
Part 2: Add new process requirements to the process	4-4
Exercise 5. Playback 1: Controlling process flow with business data	5-1
Part 1: Implement the intermediate timer event	5-3
Part 2: Create the process flow variables for the Hiring Request Process	5-4
Part 3: Create a process flow variable for the Approve Hire Request linked process	5-10
Part 4: Implement gateways	5-13
Part 5: Implement routing for an activity	5-22
Part 6: Mapping variables	5-28
Exercise 6. Playback 1: Business data, services, and coaches	6-1
Part 1: Build business objects and variables	6-3
Part 2: Create a reusable client-side human service and define variables	6-15
Part 3: Create human services	6-41
Exercise 7. Playback 1: User interface design and implementation	7-1
Part 1: Group controls into tabs on a coach	7-2
Part 2: Change the appearance of the coach by applying a custom theme	7-17
Part 3: Apply the Spark UI theme	7-27
Part 4: Modify the mobile format presentation	7-31

Exercise 8. Playback 1: Conducting the playback session	8-1
Part 1: Demonstrate the Review Needed path	8-3
Part 2: Demonstrate the Review Not Needed path	8-17
Part 3: Create the Hiring Requisition Toolkit	8-24
Part 4: Create or update dependency on a toolkit	8-31
Exercise 9. Playback 2: Integrations	9-1
Part 1: Create a decision service	9-2
Part 2: Implement a message start event	9-21
Part 3: Apply asset tags	9-34
Part 4: Create a service to query a database and populate a list	9-39
Part 5: Change an input to a single select on a coach	9-61
Exercise 10. Playback 3: Creating error handling for a service	10-1
Part 1: Implement exception handling in a service	10-2
Exercise 11. Playback 3: Creating a snapshot for deployment	11-1
Part 1: Prepare for final snapshot	11-2
Part 2: Check the history of the process application	11-5
Part 3: Create a snapshot of the process application	11-7
Appendix A. Conducting playbacks	A-1
Overview	1
Objectives	1
Introduction	1
Requirements	1
Part 1: Playback 0: Capture, describe, and model the process	A-4
Part 2: Playback 1: Process flow data implementation, User interface design and implementation	A-5
Part 3: Conduct Playback 2: Integration	A-6
Part 4: Conduct Playback 3: Hardening processes and services	A-7
Appendix B. Modeling and implementation artifacts	B-1
Appendix C. Challenge exercises	C-1
Consolidation exercise	1
Appendix D. Data dictionary	D-1
Training database	1
Part 1: Departments	D-2
Part 2: Divisions	D-2
Part 3: JobLevels	D-2
Part 4: Positions	D-4
Part 5: IncidentCategory	D-4
Part 6: IncidentType	D-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:

Blueworks Live™

DB2®

FileNet®

PartnerWorld®

Redbooks®

CICS®

developerWorks®

IBM Connections™

PureApplication®

WebSphere®

DB™

Express®

IBM MobileFirst™

Rational®

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 is a registered trademark or trademark of VMware, Inc. or its subsidiaries in the United States and/or other jurisdictions.

Social® is a trademark or registered trademark of TWC Product and Technology, LLC, an IBM Company.

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

Exercises description

This course includes the following exercises:

- Exercise 1. Playback 0: Creating a process with ad hoc activities
- Exercise 2. Playback 0: Creating a process application
- Exercise 3. Playback 0: Controlling process flow
- Exercise 4. Validating the process model
- Exercise 5. Playback 1: Controlling process flow
- Exercise 6. Playback 1: Business data, services, and coaches
- Exercise 7. Playback 1: User interface design and implementation
- Exercise 8. Playback 1: Conducting the playback session
- Exercise 9. Playback 2: Integrations
- Exercise 10. Playback 3: Creating error handling for a service
- Exercise 11. Playback 3: Creating a snapshot for deployment

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. Some exercises also include optional sections that you might want to complete when you have sufficient time and want an extra challenge.



Important

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

Exercise 1. Playback 0: Creating a process with ad hoc activities

Estimated time

01:30

Overview

This exercise covers how to create ad hoc activities in IBM BPM Process Designer.

Objectives

After completing this exercise, you should be able to:

- Start IBM Business Process Manager
- Create a process application by using Process Designer
- Create the foundation for a structured process by adding the appropriate lanes to the default pool
- Create a process
- Add ad hoc activities to the team lanes

Introduction

A process is a project that starts and finishes over time to resolve a problem. The problem can involve a claim or a request or a proposal. Many documents and records relevant to the process are supplementary to the problem. A process usually involves multiple people from inside and outside an organization. These people often have a relationship to each other.

IBM Business Process Manager simplifies the job of designing and building processes with ad hoc activities and provides a graphical user interface for knowledge workers to manage processes. With IBM Business Process Manager, you design an application that is based on closely related processes and then deploy that solution into a production environment. Knowledge workers can then complete work items that are associated with processes.

For example, to design an application for resolving credit card disputes, IBM Business Process Manager provides tools to design and create an application that knowledge workers can then use to work with the processes that are created.

To create an application, you identify the user activities that are needed to complete the process. You also identify the documents that are used and the teams of users that work on the activities, and the conditions that are required to start and complete the process and activities. Finally, you create the user interface that knowledge workers see in Process Portal.

Requirements

None



Important

The exercises in this course use a set of lab files that might include scripts, applications, files, solution files, project interchange 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.



Cloud

For IBM BPM 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.

Exercise instructions

Each exercise in this course contains one or more of the following parts:

- An introduction
- A set of instructions
- A challenge exercise

How to use the course exercise instructions

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

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.

Tracking your progress

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

If you face any trouble or encounter errors while working on the exercise, you should go back and review your steps to find out whether you correctly completed the steps and did not miss any steps.

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

An underscore precedes each numbered step and lettered 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 when you experience interruptions during a lengthy exercise.

User IDs and passwords

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

Entry point	User ID	Password
VMware image	administrator	passw0rd
Windows 2012 R2	administrator	passw0rd
IBM Process Designer	author1	author01
IBM Business Process Manager WebSphere Application Server cell	administrator	passw0rd
administrator		



Cloud

For IBM BPM 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 the Appendix D, “Data dictionary” on page D-1 in this exercise guide. Contact your cloud administrator for assistance in setting up your environment.



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
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.

Exercise instructions

Hiring Requisition process

A company is experiencing rapid growth and must hire many people in a short amount of time. The process that you are going to examine and model is called the Hiring Requisition Process. This process covers a new job position through submission, approval, and completion so applicants can apply for the job position.

In this exercise, you create a process. You add ad hoc activities to this process. A Hiring Manager submits a hiring requisition to the HR Department. The request contains the following information:

- Requisition number
- Job title
- Department
- Salary to offer

Part 1: Start IBM Business Process Manager and IBM Process Designer

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.



Important

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 this task, IBM Business Process Manager provides a Quick Start routine to run the server start in order.

-
1. Start the **Process Center** server.



Cloud

For IBM BPM on Cloud users: You do not need to start the Process Center. It is already running in your cloud environment.

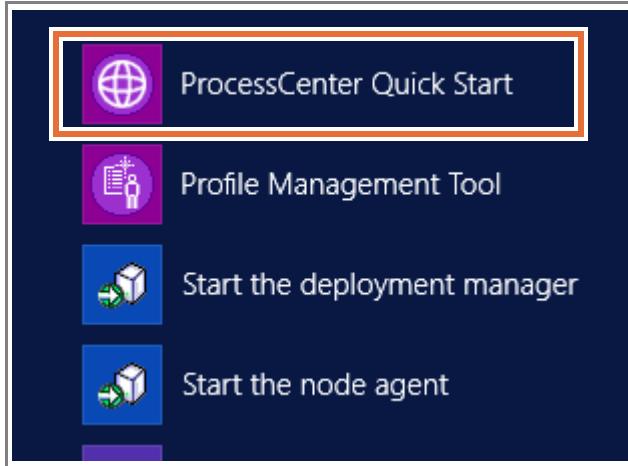
- __ a. Click the **Windows Icon**.



- __ b. Next, click the **Down Arrow**.



- __ c. Then, scroll right to the IBM group and click **ProcessCenter Quick Start**.



Important

You can verify the server version with the System Data Toolkit version that is installed on the Process Center later in the course.

- ___ d. In the IBM Business Process Manager Quick Start window, click the **Start the Process Center Deployment Environment** link.

IBM Business Process Manager Standard Quick Start for ProcessCenter

Deployment environment administration

Start the Process Center Deployment Environment. Start all configured clusters, node agents, and servers.

[Start the Process Center Deployment Environment](#)

It takes a while to start the deployment environment.



Important

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 about 15 – 20 minutes, so make sure that you provide ample time for this initial start.

- ___ e. A command window opens. Press any key when it prompts **Press any key to continue.**

```
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 3716
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 4448
Starting cluster SingleCluster.
When the BPMConfig command is used to start a deployment environment, it invoke
the processes that are used to start the associated clusters. If the command i
successful in invoking the processes, it returns a message to report that the
command completed successfully. However, to determine whether the cluster member
were all started successfully, you need to check the log files of the cluster
members. The log files are located in <profile_root>/logs.
The 'BPMConfig.bat -start -profile DmgrProfile -de ProcessCenter' command compl
eted successfully.
Press any key to continue . . . -
```

- ___ f. The Quick Start browser window populates. It might take up to 10 – 20 minutes for the servers to fully engage depending on the hardware configuration. When the four sections are populated in the IBM Business Process Manager 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 Quick Start

IBM Business Process Manager Standard Quick Start for Process 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

[WebSphere Application Server Administrative Console](#)

Administer the process servers in your environment, including the users and installed snapshots for view and manage process instances for process applications.

[Process Admin Console](#)

Work with Performance Data Warehouse queues, manage data transfer errors, and monitor overall

[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](#)



Cloud

For IBM BPM 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 that are shown in the Quick Start menu.

- 2. Start the **IBM Process Designer**.
 - a. Click **Process Center Console** in the IBM Business Process Manager Quick Start browser window.

IBM Business Process Manager Quick Start

IBM Business Process Manager Standard Quick Start for Process 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 [WebSphere Application Server Administrative Console](#)

Administer the process servers in your environment, including the users and installed snapshots for view and manage process instances for process applications.

[Process Admin Console](#)

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

Process application consoles & tools

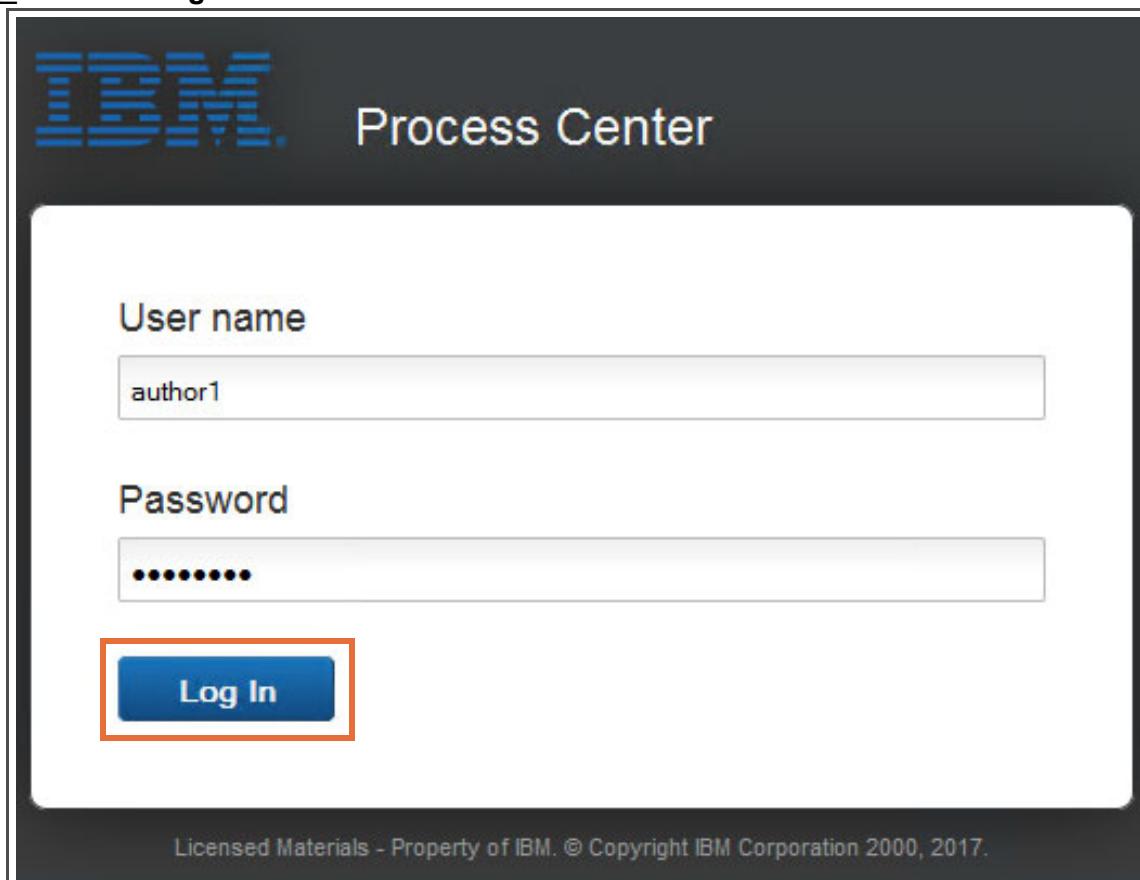
Create, manage, share, and test high-level containers such as process applications and toolkits.

[Process Center Console](#)

Test and administer the business user interface for completing tasks.

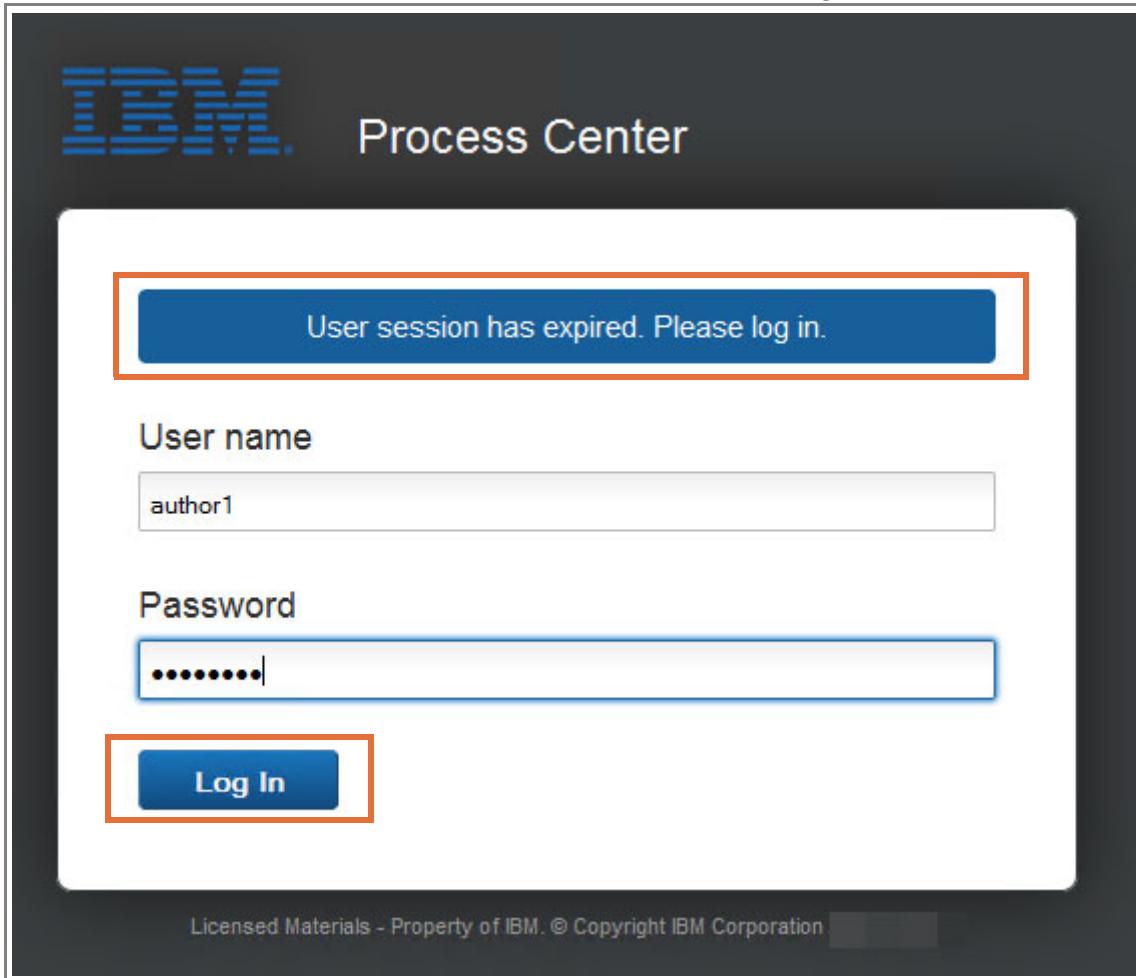
[Process Portal](#)

- __ b. Minimize the IBM Business Process Manager Quick Start browser window and the ProcessCenter Quick Start window so that you can use the page to access the Process Portal later in the exercises.
- __ c. A loading screen and then a login screen are displayed for the Process Center Console.
- __ d. Enter **author1** for the **User name** and **author01** for the **Password**.
- __ e. Click **Log In**.

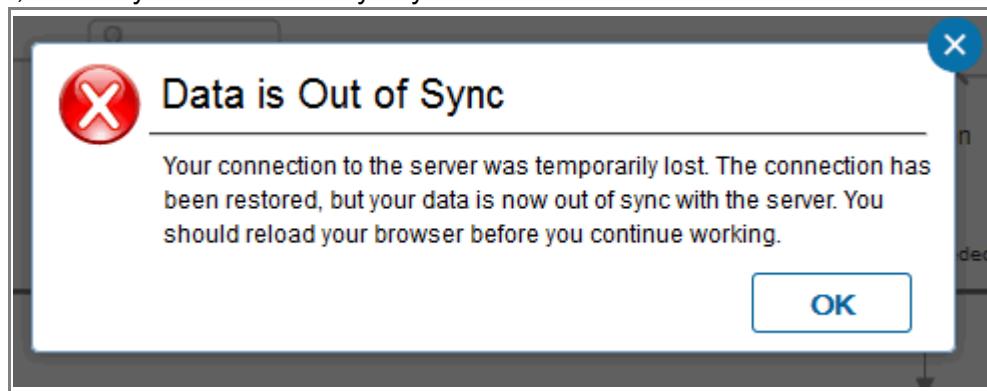


**Important**

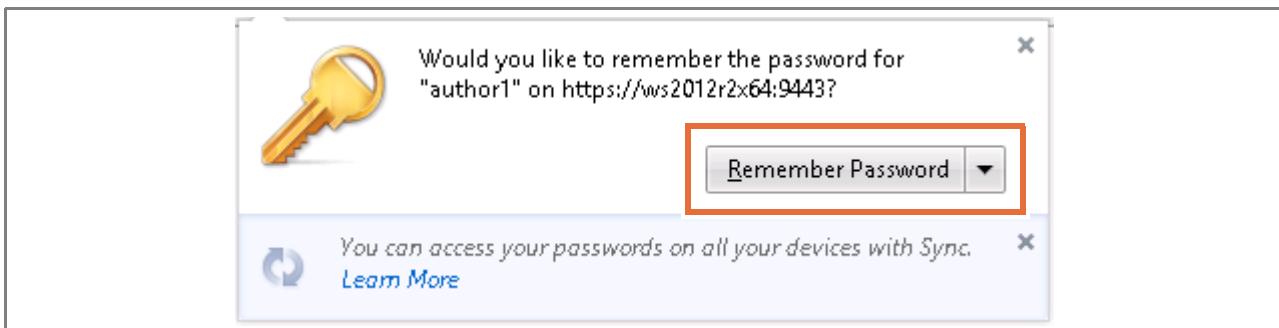
At certain times when working inside the web-based Process Designer, the session times out due to inactivity. If at any time during this course you see an expired session error, insert the same credentials that are shown (author1 and author01) and click **Log In**.



You might also experience a “Data is Out of Sync” error during this course. If this error is shown on your screen, refresh your browser to sync your data with the server.



- ___ f. For the Remember Password message window that appears, click **Remember Password**.



- ___ g. Close the Getting Started window that appears the first time you open the Process Center console.



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

Application	Details	Action
Performance (SYSPERFDB)	Last updated on [redacted] by author1	Open in Designer
Process Portal (SYSRP)	Last updated on [redacted] by author1	Open in Designer
Heritage Process Portal (deprecated) (TWP)	Last updated on [redacted] by author1	Open in Designer
Discover BPM UI Sample (DBPMUIS)	Last updated on [redacted] by author1	Open in Designer

Now that everything is started and you are in the Process Center, you are ready to start your exercises. Each exercise shows step-by-step instructions that you can follow to complete the tasks. In the IBM Process Designer, you find many different ways to complete modeling tasks. The step-by-step instructions show one way to accomplish these tasks in the exercises.



Cloud

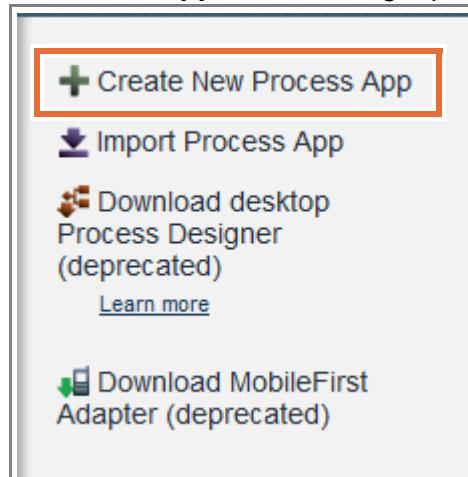
For IBM BPM on Cloud users: The Process Center perspective is the same when the Process Center is on the cloud.

Part 2: Create a process application

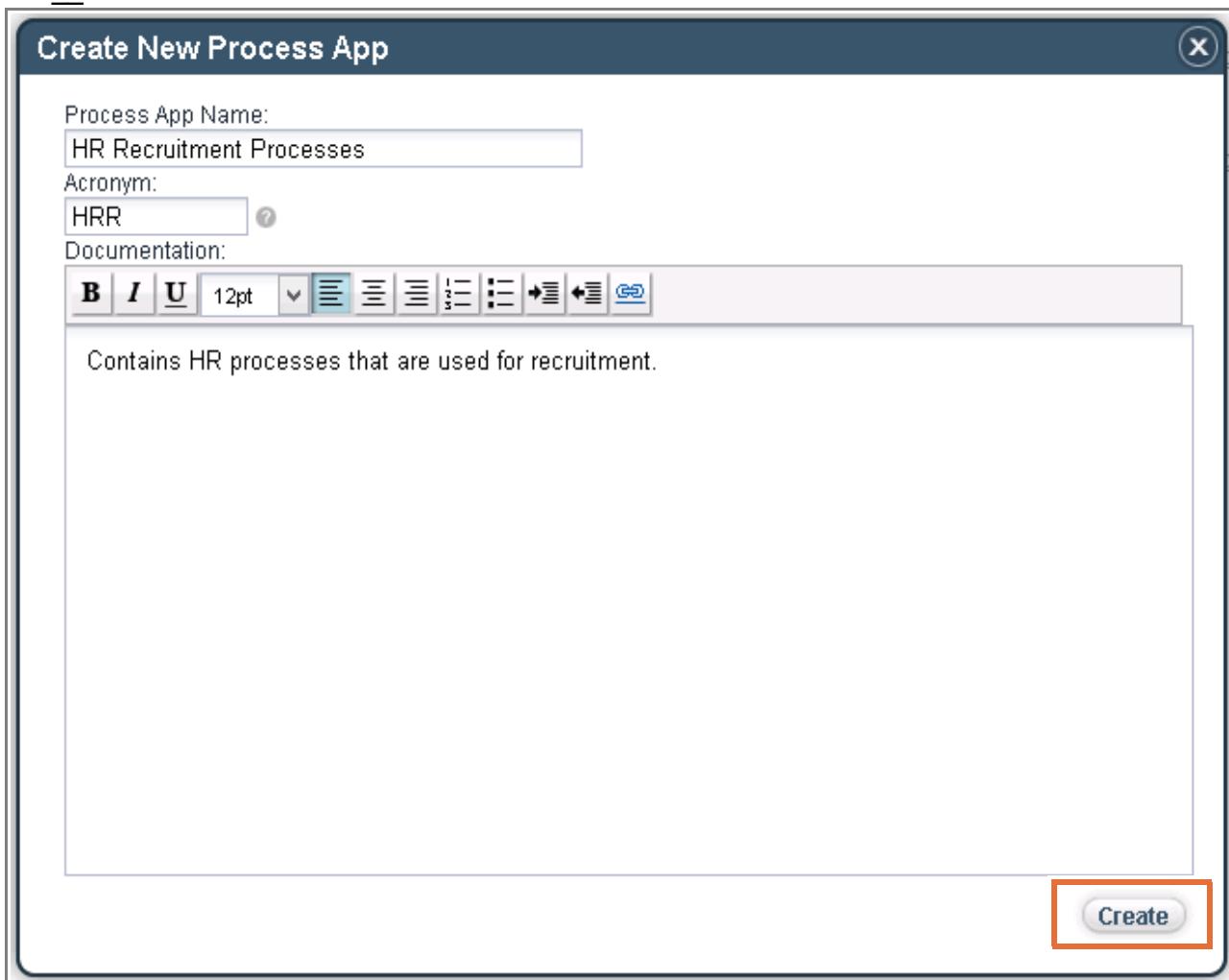
In this exercise, you create a hiring request process with ad hoc activities. In the subsequent lab exercise, you refine and extend your process, specifying more details as you progress.

To accomplish the task of creating a process in the IBM Process Designer, you must have a process application to contain the process. Using the Process Center, an author first creates a process application with all the appropriate information for the creation of a process.

- __ 1. Create a process application.
 - __ a. Click the **Create New Process App** link on the right pane of IBM Process Designer.



- __ b. When the **Create New Process App** dialog box is displayed, enter following values:
- Set **Process App Name** to: HR Recruitment Processes
 - Set **Acronym** to: HRR
 - Set **Documentation** to: Contains HR processes that are used for recruitment.
- __ c. Click **Create**.



**Note**

The process application is created and is now in your list of process apps.

The screenshot shows the 'Process Apps' interface. At the top, there are tabs for 'Process Apps', 'Toolkits', 'Servers', 'Admin', and links for 'Preferences' and 'Logout'. Below the tabs, there is a search bar with the placeholder 'Search' and a dropdown menu set to 'Recently Updated'. To the right of the search bar are buttons for 'All', 'Favorites' (with a star icon), and 'Archive'. A list of process applications is displayed below. The first item, 'HR Recruitment Processes(HRR)', is highlighted with a red box around its row. It has a star icon and a question mark icon next to it. To the right of the application name is a 'Last updated on' timestamp and the author 'by author1'. On the far right of this row is a blue 'Open in Designer' button with a small icon. The other three items in the list are 'Performance (SYSPERFDB)', 'Process Portal (SYSRP)', and 'Heritage Process Portal (deprecated) (TWP)', each with their own 'Open in Designer' button.

Process Application	Last updated on	Author	Action
HR Recruitment Processes(HRR)	Just now	by author1	Open in Designer
Performance (SYSPERFDB)	Just now	by author1	Open in Designer
Process Portal (SYSRP)	Just now	by author1	Open in Designer
Heritage Process Portal (deprecated) (TWP)	Just now	by author1	Open in Designer

- __ 2. Open the HR Recruitment Processes process application.
- __ a. In the Process Center, find the HR Recruitment Process (HRR) process application and click **Open in Designer**.

This screenshot is identical to the one above, showing the 'Process Apps' interface with the 'HR Recruitment Processes(HRR)' application selected. The 'Open in Designer' button for this application is highlighted with a red box.

After opening the process application, the initial view of the Designer is as follows:



Important

The highlighted area on the left shows the process application that you are editing. Notice that the initial screen is also an opportunity to edit the process application settings that are highlighted on the right. Here you can change the name, acronym, description, and authorization for the process application. For this course, leave the settings as they are and continue with your process modeling tasks.

The process owner provides detailed information about the process and its current state to the IBM BPM analyst, who in turn documents the information and analyzes the process for improvement. For this scenario, the process discovery and initial analysis are already completed, and now the process model can be created.

To begin the task of creating the initial process, create a process with ad hoc activities in the IBM Business Process Manager Process Designer.

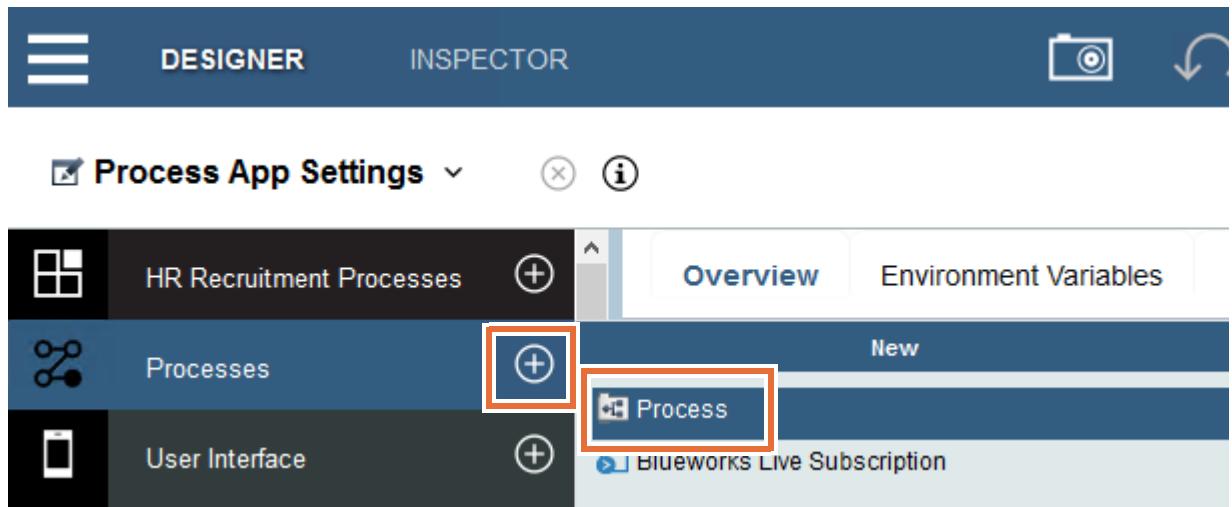
A process is a project that starts and finishes over time to resolve a problem. The problem can involve a claim, a request, or a proposal, and the supplement of many documents and records relevant to the process. A process usually involves multiple people from inside and outside of an organization. These people often have a relationship to each other. In this exercise, the Hiring Manager submits a hiring requisition to the HR Department.

To create a process, you identify the user activities that are needed to complete the process and then group those activities into a process. You identify the documents that are used and the teams of users that work on the activities. You also identify the conditions that are required to start and complete the ad hoc activities. Finally, you create the user interface that process workers see in the Process Portal.

Part 3: Create a process

A process defines the activities that are needed to resolve a specific business problem. A process also defines who works on these activities and the steps they take to resolve the problem. At run time, a business user works with an instance of a process.

- 1. Create a process.
- a. From the process library, click the (+) plus sign next to Processes and select Process.



- b. Enter Hiring Request Process in the Name field. Click Finish.

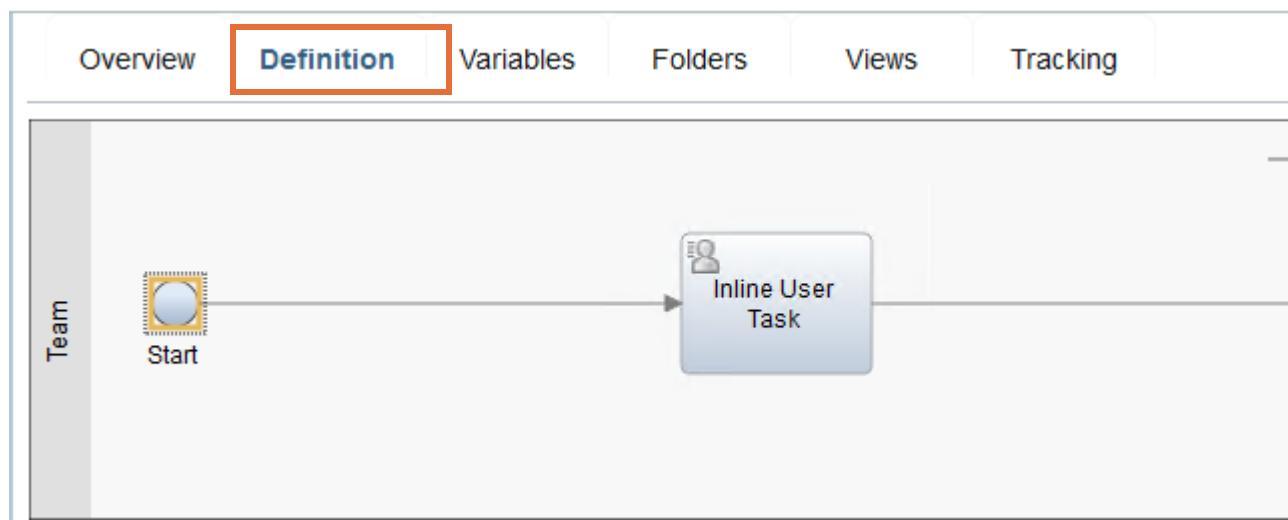
New Process

A process captures a set of activities and the data and content to support the activities. These activities can be part of a structured flow, ad-hoc activities that are not part of a structured flow, or a combination of the two.

Name

FINISH **CANCEL**

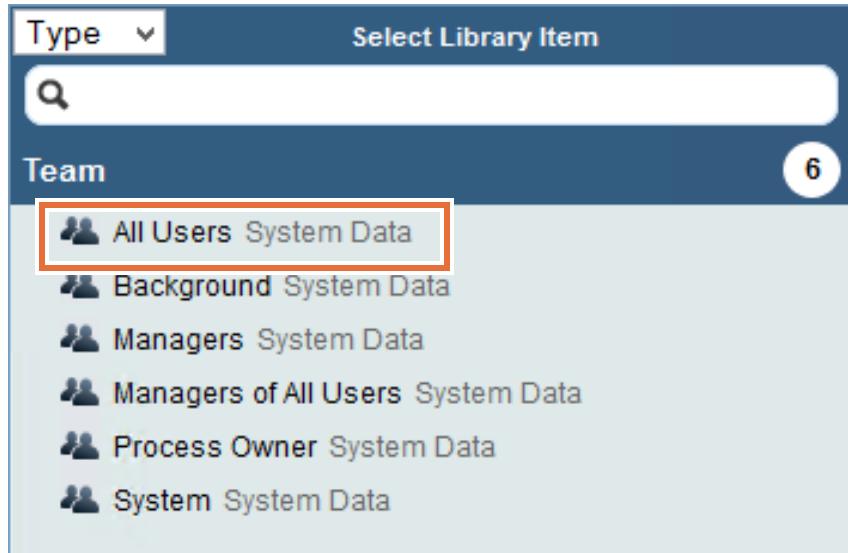
The process is created, and the initial view is shown under the **Definition** tab. By default you have a process with a team lane, a system lane, a start event, an inline user task, and an end event.



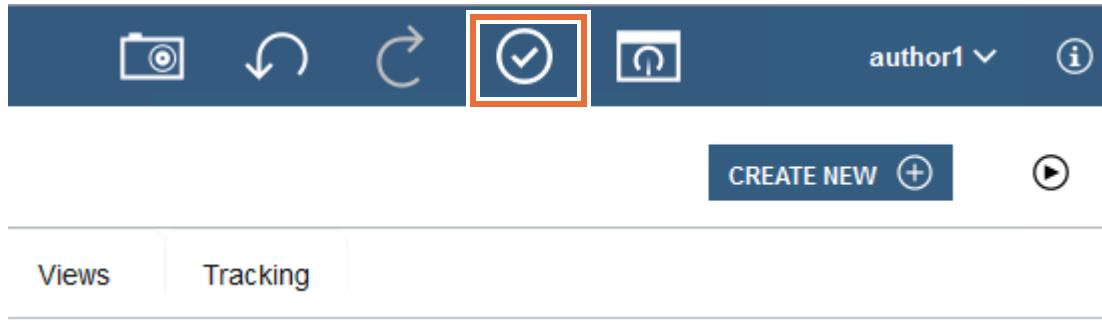
- 2. Expose the process to **All Users** so that all the users can start the process.
 - a. Click the **Overview** tab.

The screenshot shows the "Overview" tab selected. Below it, two sections are expanded: "Common" and "Details". Under the "Common" section, there is a sub-section titled "Work Schedule" containing fields for Time schedule, Timezone, and Holiday schedule, each with a "(use default)" option. Under the "Details" section, there is a sub-section titled "Exposing" containing fields for Expose to start, Expose business data, and Expose performance metrics. The "Expose to start" field has a dropdown menu showing "<none>" and a "Select..." button, which is highlighted with a red box. The "Expose business data" and "Expose performance metrics" fields also have dropdown menus showing "<none>" and "Select..." buttons.

__ c. Select All Users.



__ 3. Click **Finish Editing** to save your changes.



Hint

You can also use Ctrl+S to save your work. Even though the system saves every time you change your project, it is still a good idea to manually signal to the system to save your work at certain intervals.



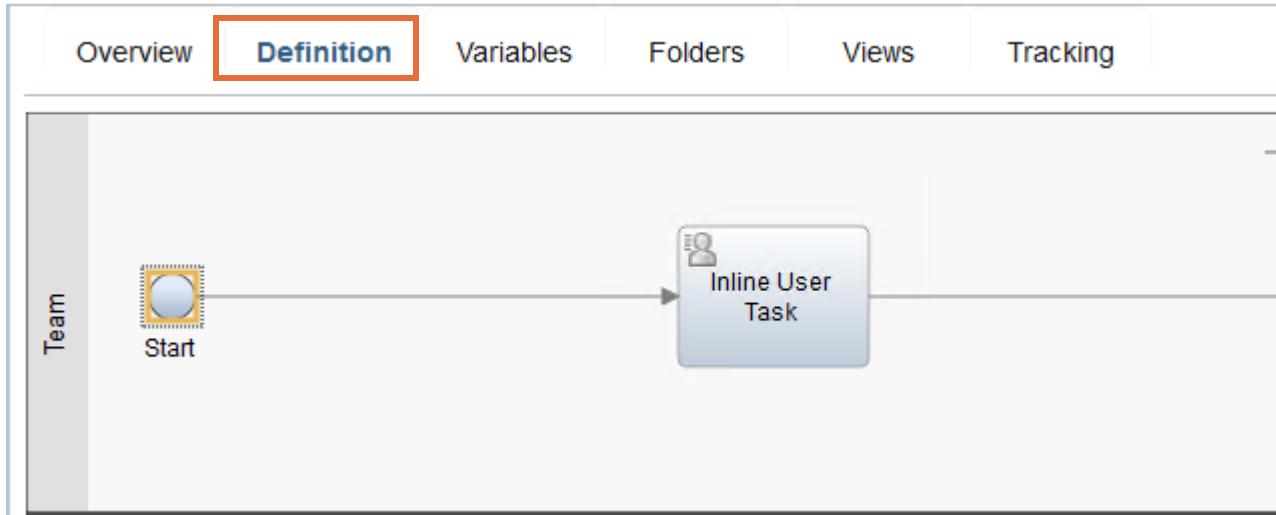
Note

You can assign a team whose members can start a process, or an instance owner's team whose members can work with the process at run time in the Process Portal. Currently, you assign all users to the HR Recruitment Processes process, and in a later unit and lab exercise in this course, you learn how to define and assign a team and users to your process.

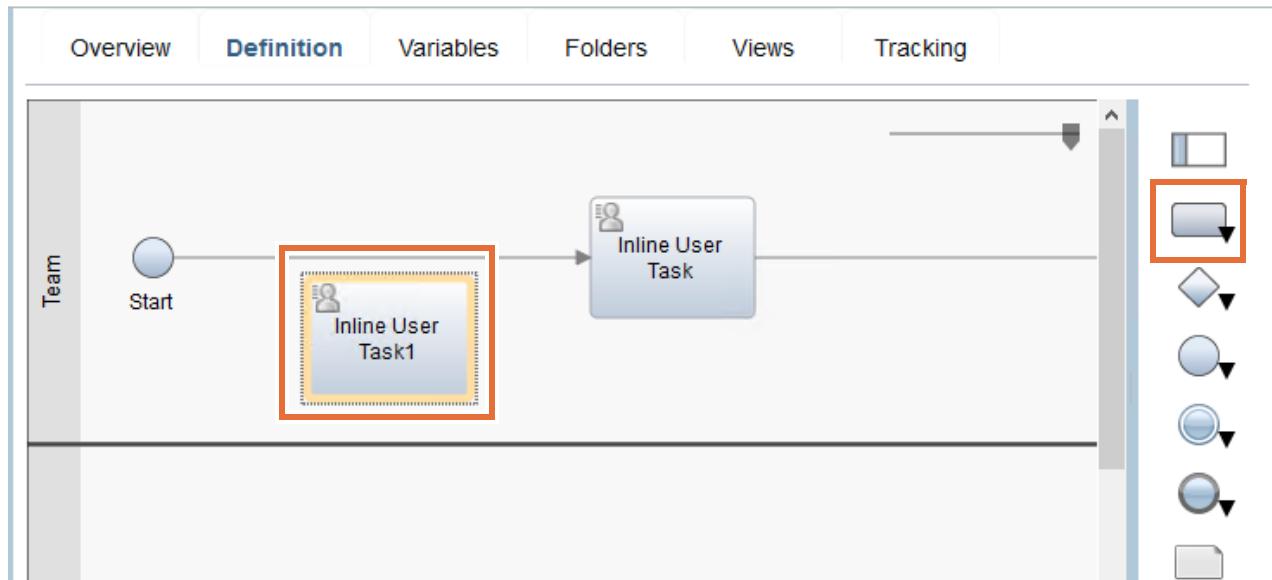
Part 4: Create process activities

An activity in a process is a discrete task that a person or a system completes as part of that process. Typically a process has a number of activities. Initially you add an activity in a process and classify it as required or optional. Next, you repeat the process to create three required activities and one optional activity.

- 1. Create a required process activity that is named **Submit Hiring Request**.
 - a. From the Hiring Request Process, click the **Definition** tab.



- b. Drag an activity from the palette on the right side of the canvas into the Team lane to the right of the start event. When you drag an object to the canvas, the cursor turns into a green plus sign while you drag the object.
- c. Release the left mouse button to place the activity on the canvas to the left of the Inline User Task.

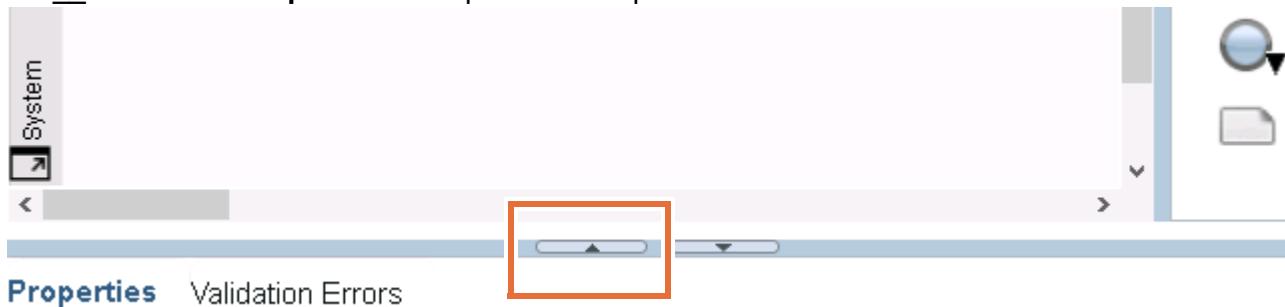


**Hint**

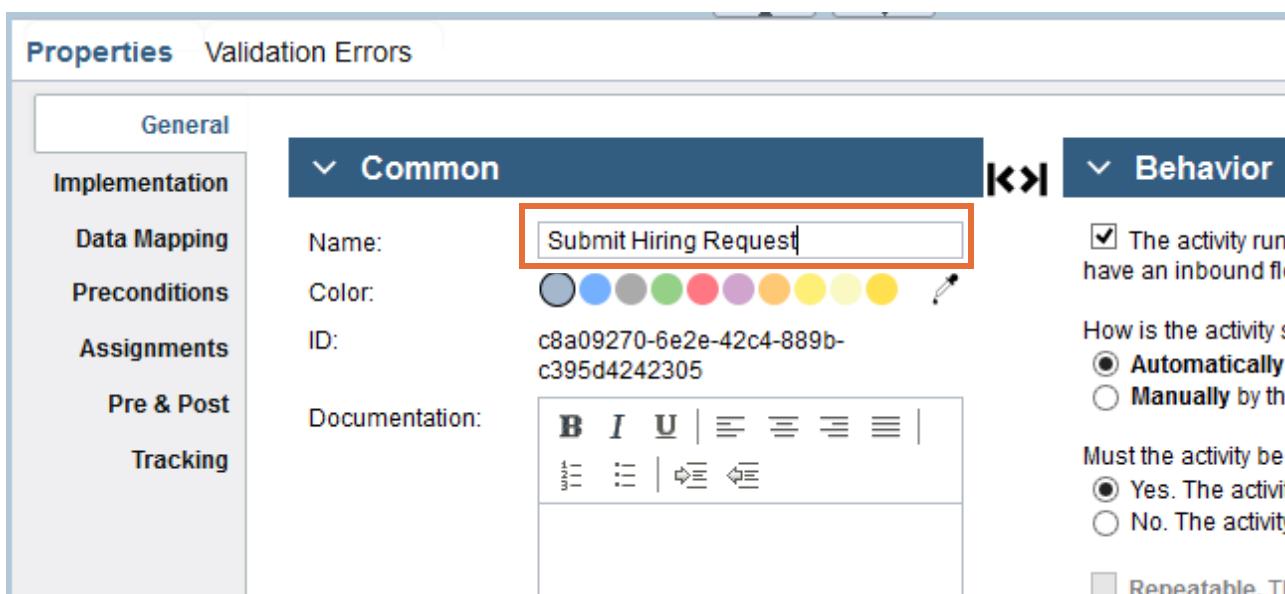
Click the element in the palette and hold for 1 second. When you drag to the canvas, a green plus sign verifies that you are dragging an element to the canvas. If you want to place an item above another item on the canvas, you can drag it to the item that must appear below it. Then, release the element on top of it when the element highlights on the canvas.

If you need to rearrange any elements on the canvas, use the same technique of clicking and holding the mouse button for 1 second before dragging to move the objects after they are placed.

- ___ d. Click the **up arrow** to expand the Properties section in the editor.

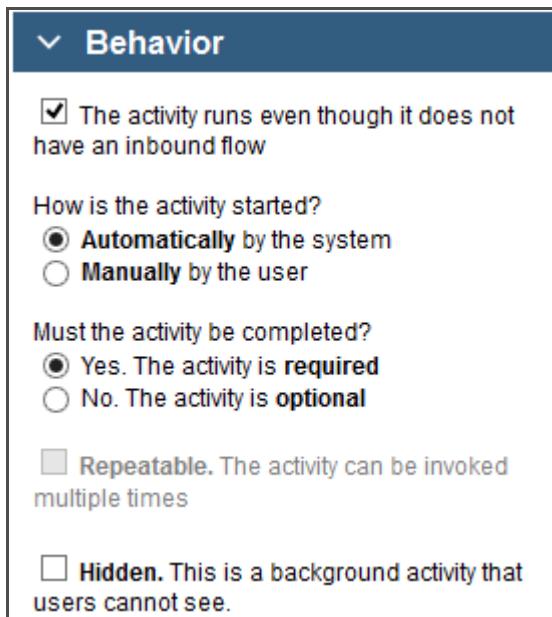


- ___ e. In the **Properties > General > Common** section, enter Submit Hiring Request as the name.

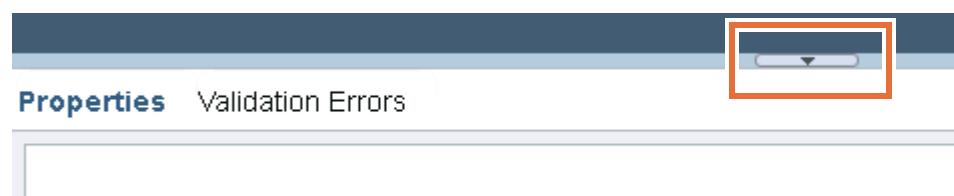


___ f. In the **Properties > General > Behavior** section, verify that the following options are selected by default:

- **This activity runs even though it does not have an inbound flow:** Selected.
- **How is the activity started?** Automatically.
- **Does the activity have to be completed?** Yes. The activity is required.
- **Repeatable:** Disabled.
- **Hidden:** Cleared.

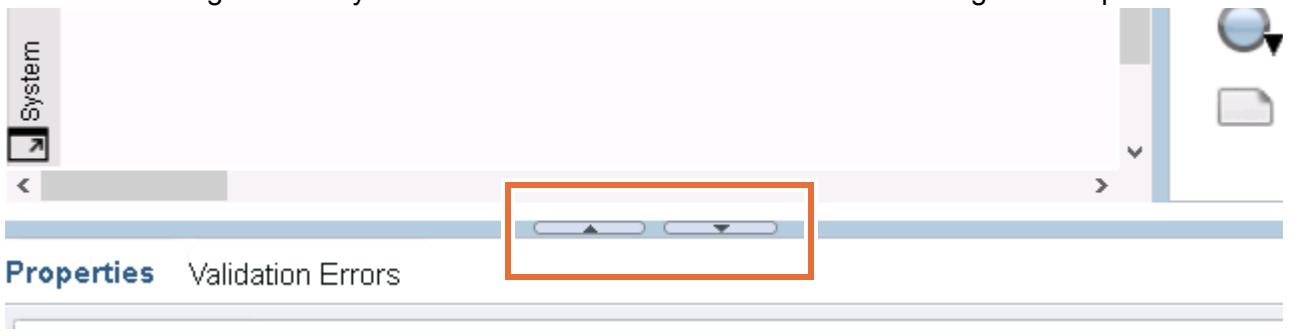


___ g. Click the **down arrow** to collapse the Properties section.

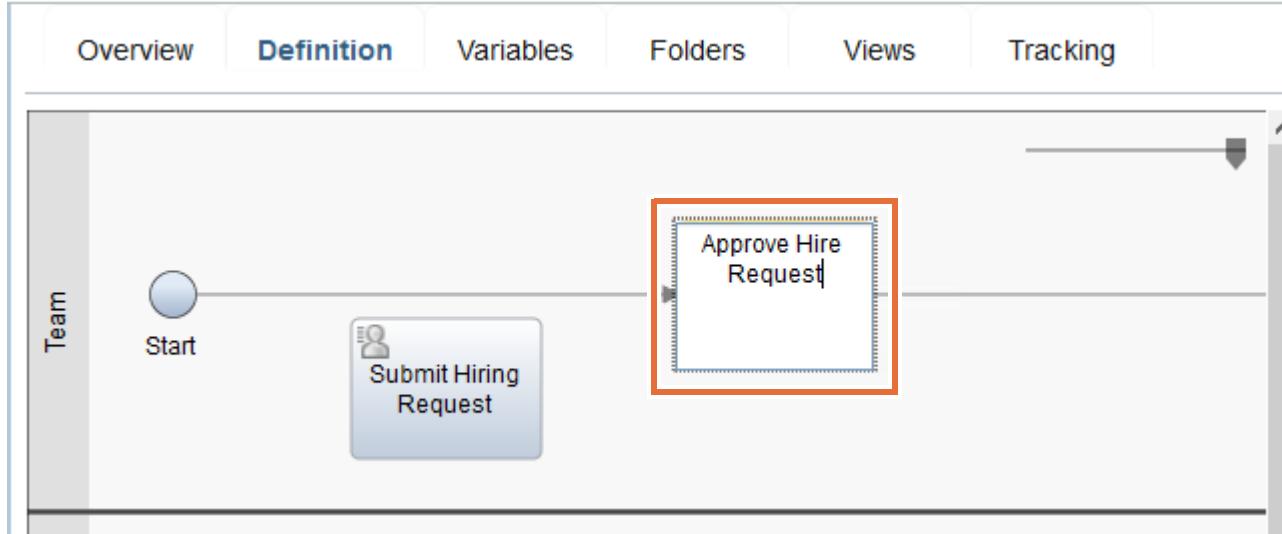


Reminder

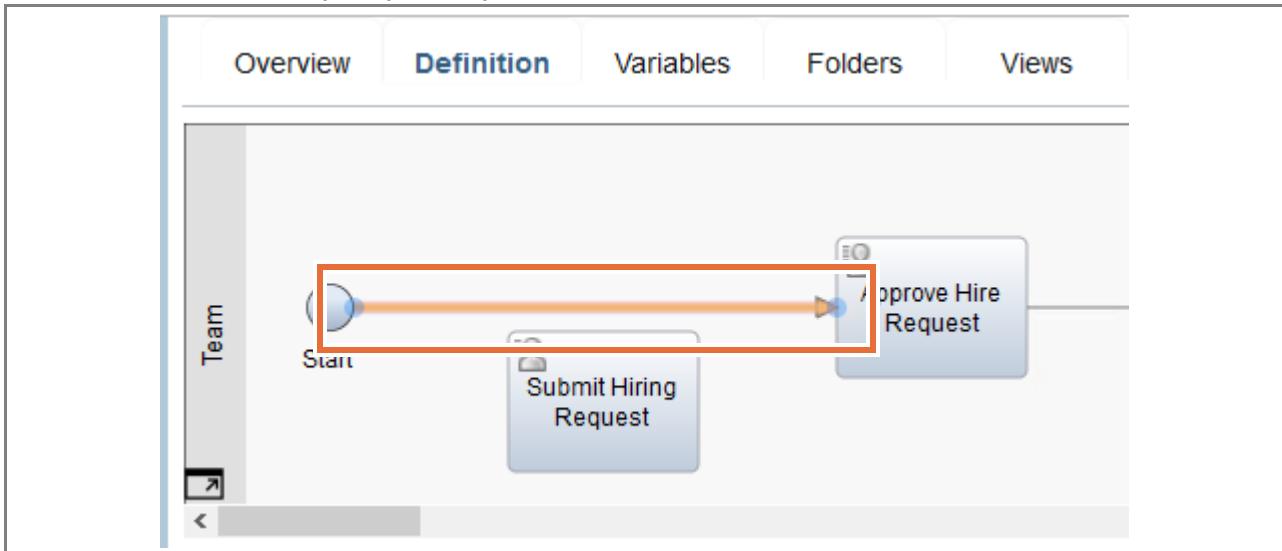
Use the down arrow or up arrow to minimize or maximize the properties section for a process in the Process Designer when you need to see more of the canvas or the configuration options.



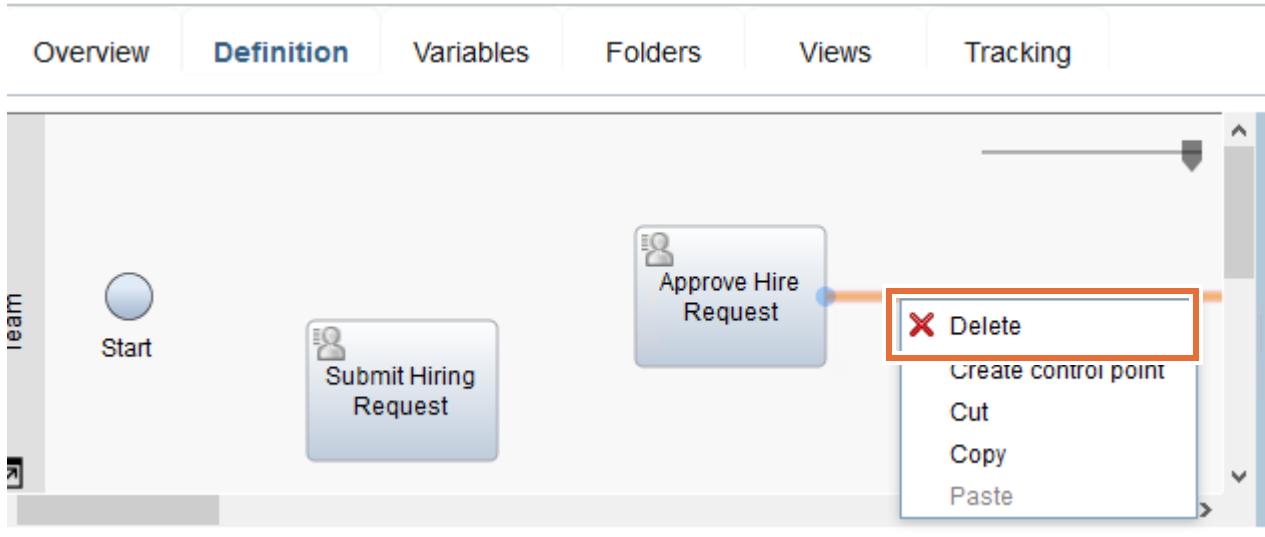
2. Create a manually started, required activity that is named **Approve Hire Request**.
- Click the **Inline User Task** to the right of the Submit Hiring Request activity.
 - When highlighted, click the text on the activity to rename it as **Approve Hire Request** and press the Enter key to apply the name change.



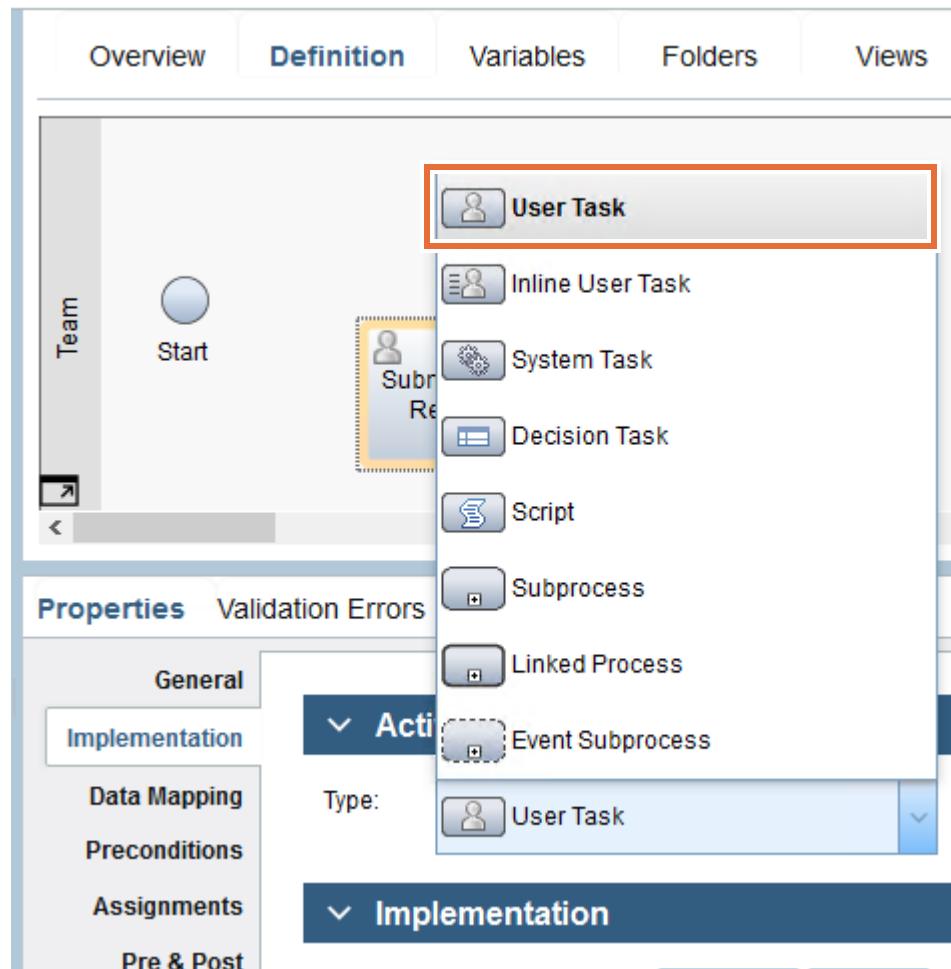
- Click the **flow line** between the Start event and the Approve Hire Request activity. Press the Delete key on your keyboard to delete the flow line.



- __ d. Click the **flow line** between the Approve Hire Request and the End Event. This time, right-click the flow line and click **Delete**.



- __ e. Later in the course, you build custom front ends for these tasks. Click the **Submit Hiring Request** activity. In the **Properties > Implementation > Activity Type** menu, expand **Type** and change the setting to **User Task**.



- ___ f. Under the **Implementation** section, click the **Select** button and click **Default UI Human Service**.
- ___ g. Verify that the Default UI Human Service implements your User Task.

Properties Validation Errors

General

Implementation

Data Mapping

Preconditions

Assignments

Pre & Post

Tracking

Activity Type

Type: User Task

Implementation

Implementation: Default UI Human Service BPM UI Select... New...



Hint

The Default UI Human Service is used to run an activity that uses a default coach. This service is useful when creating your processes and demonstrating process flow and functionality in Playback 0. Later in a course playback, you change this default coach into a custom coach.

- ___ h. Click the **Approve Hire Request** activity. In the **Properties > Implementation > Activity Type** menu, change this activity to **User Task**. Both activities now have the same icon.

Overview Definition Variables Folders Views Tracking

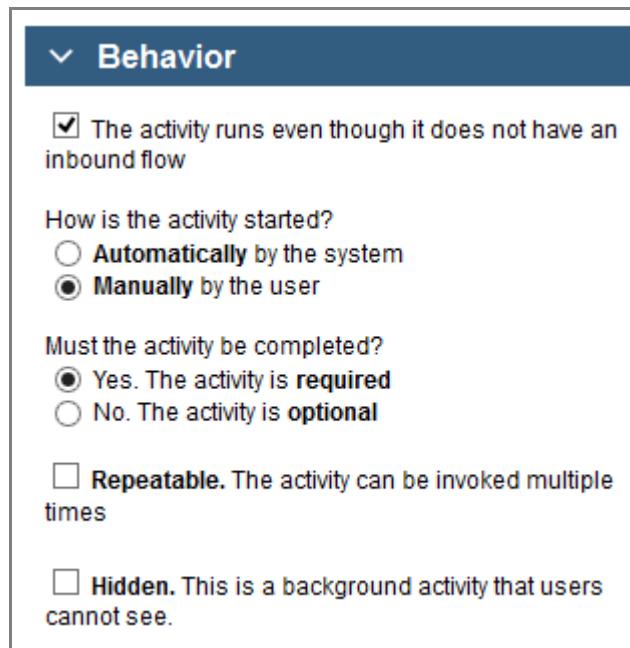
Team

Start

Submit Hiring Request

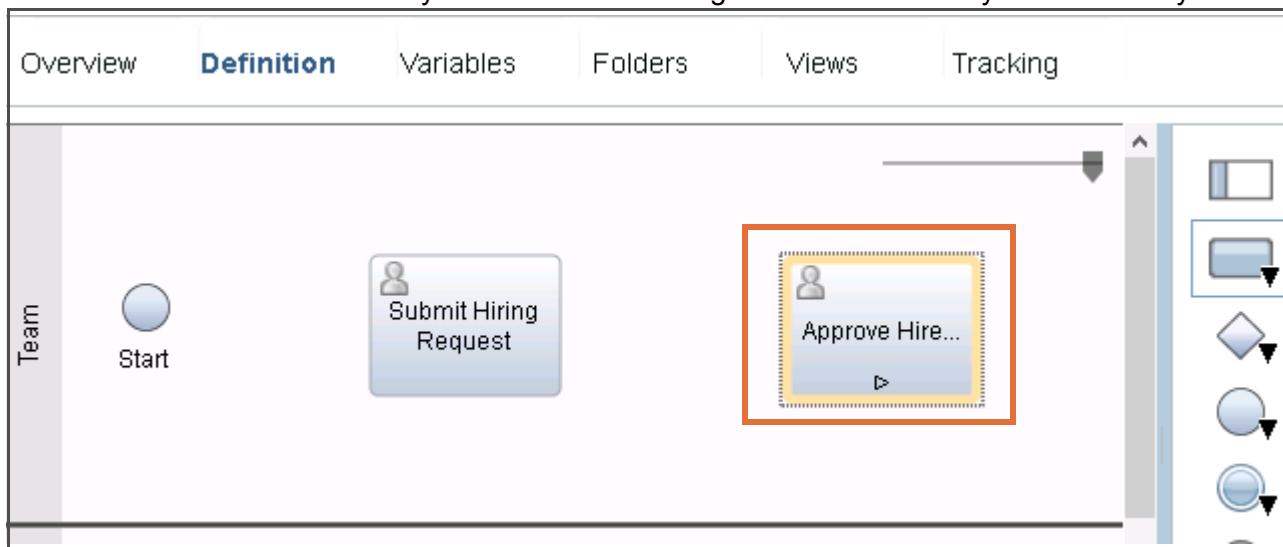
Approve Hire Request

- i. With the **Approve Hire Request** selected, In the **Properties > General > Behavior** section, set the following options:
- **This activity runs even though it does not have an inbound flow:** Selected.
 - **How is the activity started?** Manually by the user.
 - **Must the activity be completed?** Yes. The activity is required.
 - **Repeatable:** Cleared.
 - **Hidden:** Cleared.



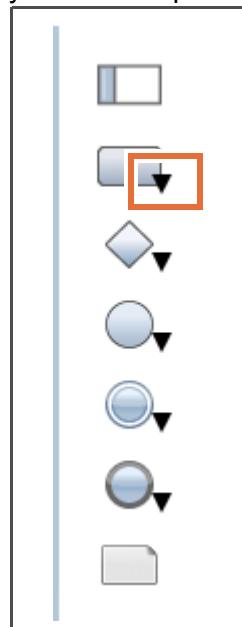
Information

An icon is added to the activity in the canvas to designate it as a manually started activity.

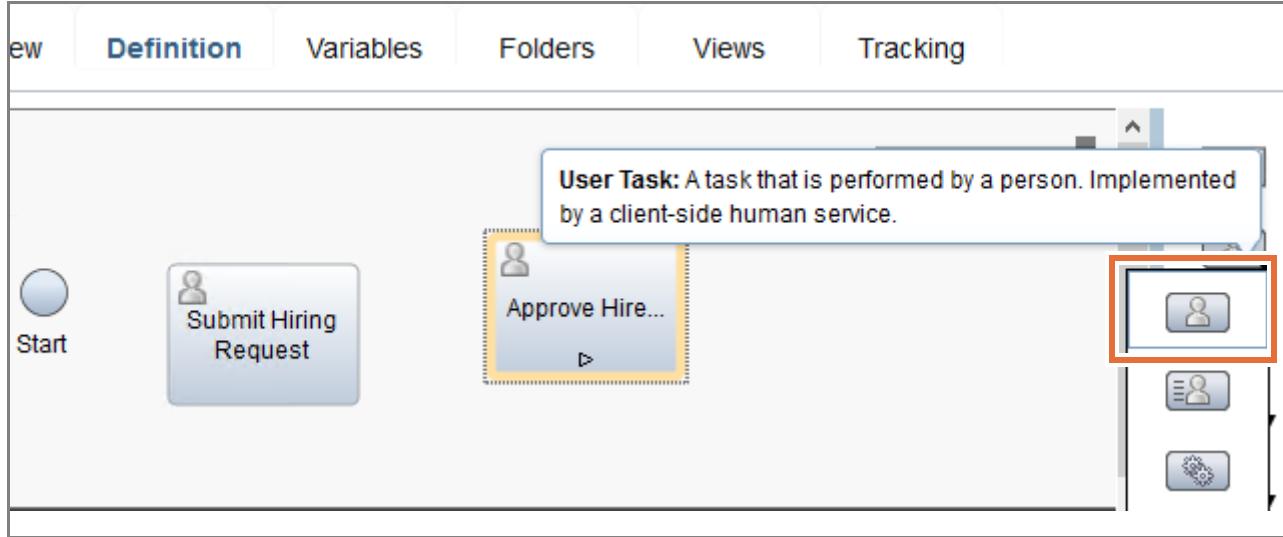


- j. In the **Properties > Implementation > Implementation** section, click **Select**, and then select **Default UI Human Service**.

3. Create a manually started, required activity that is named **Complete Hire Request**.
 a. Click the **arrow** on the activity icon in the palette.



- b. Drag a **User Task** from the palette to the right of the Approve Hire Request activity.



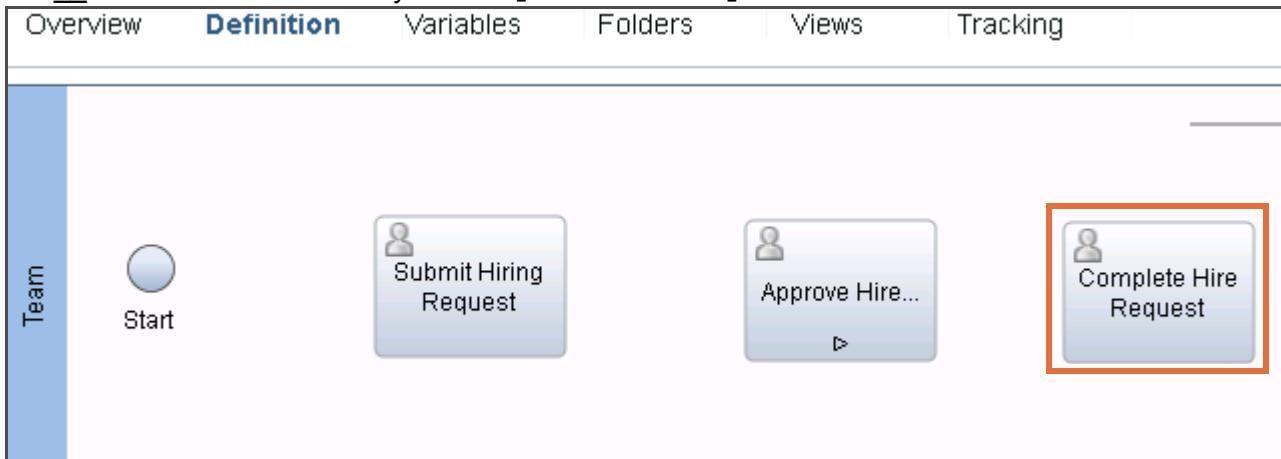
Note

To align the activities for the better presentation, you can try the following two different ways:

- Click the activity and then drag it in the position that you want.
- Click the activity and then use the arrow keys on the keyboard to move it in the position that you want.

Later in this unit, you learn other ways to align multiple activities on the canvas.

___ c. Rename the activity as: Complete Hire Request



___ d. In the **Properties > General > Behavior** section, select the following options:

- **This activity runs even though it does not have an inbound flow:** Selected.
- **How is the activity started?** Manually by the user.
- **Must the activity be completed?** Yes. The activity is required.
- **Repeatable:** Cleared.
- **Hidden:** Cleared.

The screenshot shows the 'Behavior' properties section for an activity. At the top, a dark blue header bar has a dropdown arrow and the text 'Behavior'. Below the header, there are several configuration options:

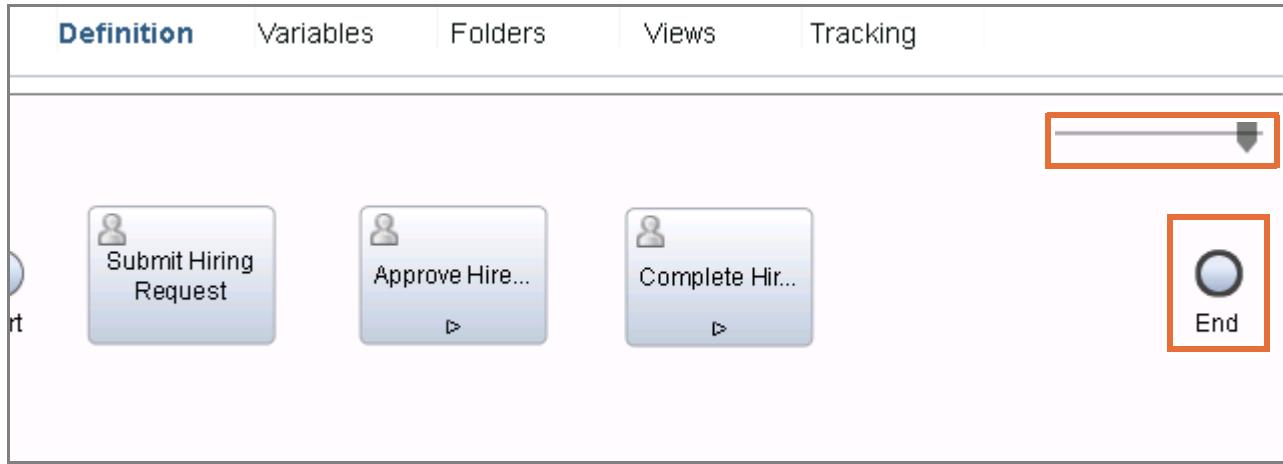
- The activity runs even though it does not have an inbound flow:** A checkbox is checked.
- How is the activity started?** A radio button for 'Manually by the user' is selected.
- Must the activity be completed?** A radio button for 'Yes. The activity is required' is selected.
- Repeatable:** A checkbox for 'Repeatable. The activity can be invoked multiple times' is cleared.
- Hidden:** A checkbox for 'Hidden. This is a background activity that users cannot see.' is cleared.

___ e. Save your changes.

**Note**

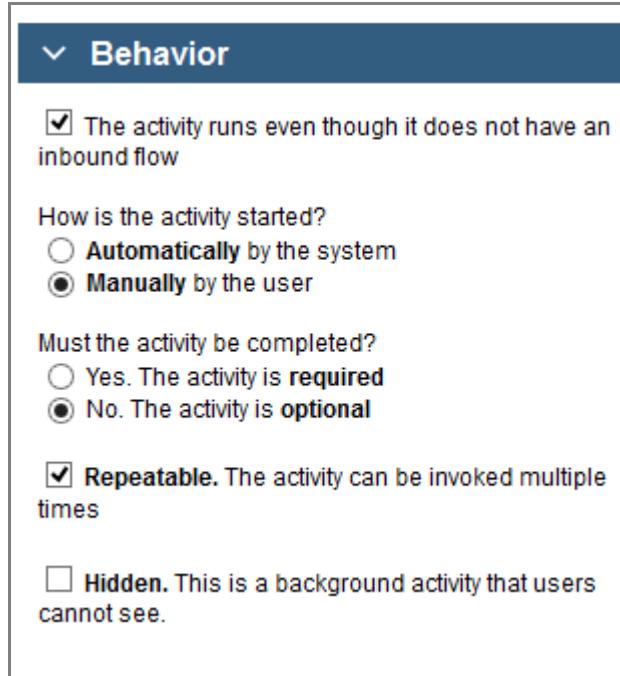
To create space for other activities in a process, select the **End** event and drag to the right side of the canvas.

You can also use the zoom bar to zoom out of the process to view more of the process in the canvas.



- ___ 4. Create an optional, manually started, repeatable User Task named **Review Salary**.
 - ___ a. Drag a **User Task** from the palette to the right of Complete Hire Request activity.
 - ___ b. Rename the activity as: **Review Salary**

- ___ c. In the **Properties > General > Behavior** section, select the following options:
- **This activity runs even though it does not have an inbound flow:** Selected.
 - **How is the activity started?** Manually by the user.
 - **Must the activity be completed?** No. The activity is optional.
 - **Repeatable:** Selected.
 - **Hidden:** Cleared.



- ___ d. Save your changes.

- ___ 5. Create a flow in the process.

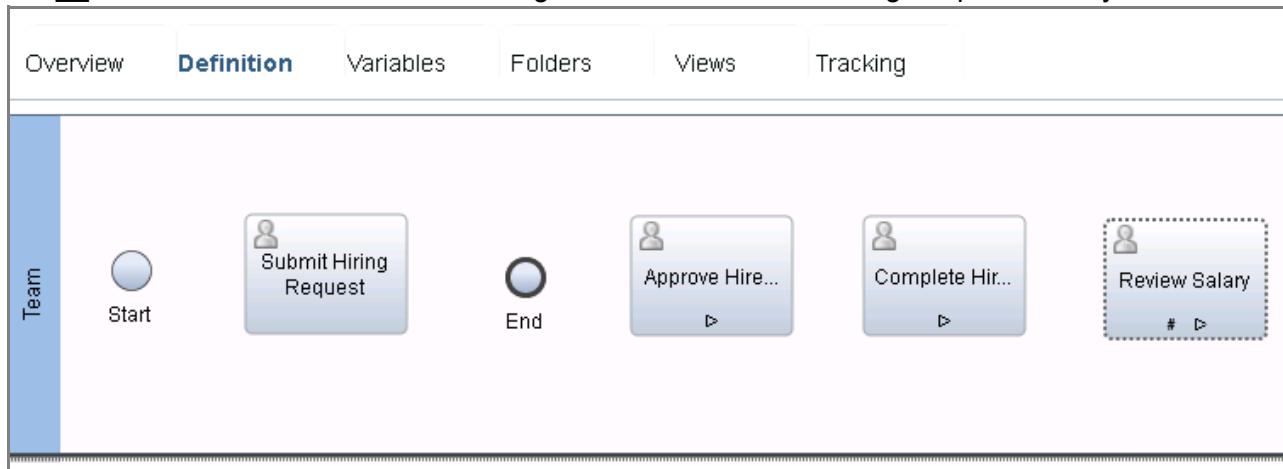


Note

In this exercise, you modeled an as-is process with ad hoc activities, and in the next exercises you work on creating a to-be process model. Because the activities in the current process can occur at any time, you modeled the process by using ad hoc activities.

In the later part of this exercise, you create an instance of the process, and for that you connect the flow from the Start event to the first activity and then to the End event. The Start event must always have an outgoing flow, and there must always be a flow to the End event.

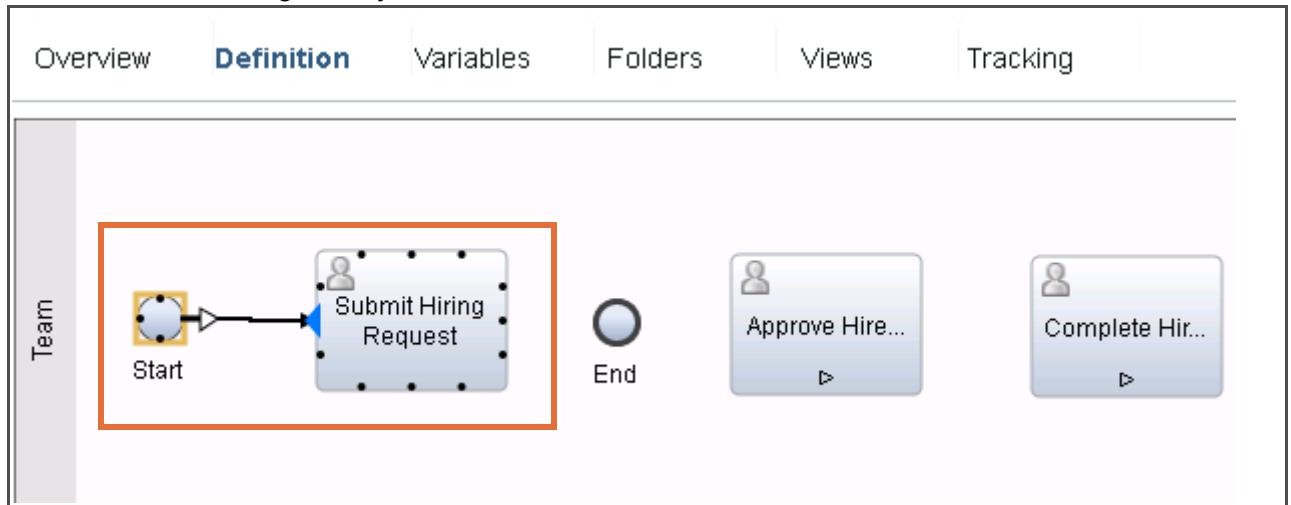
- ___ a. Select the **End** event and drag it next to the Submit Hiring Request activity.



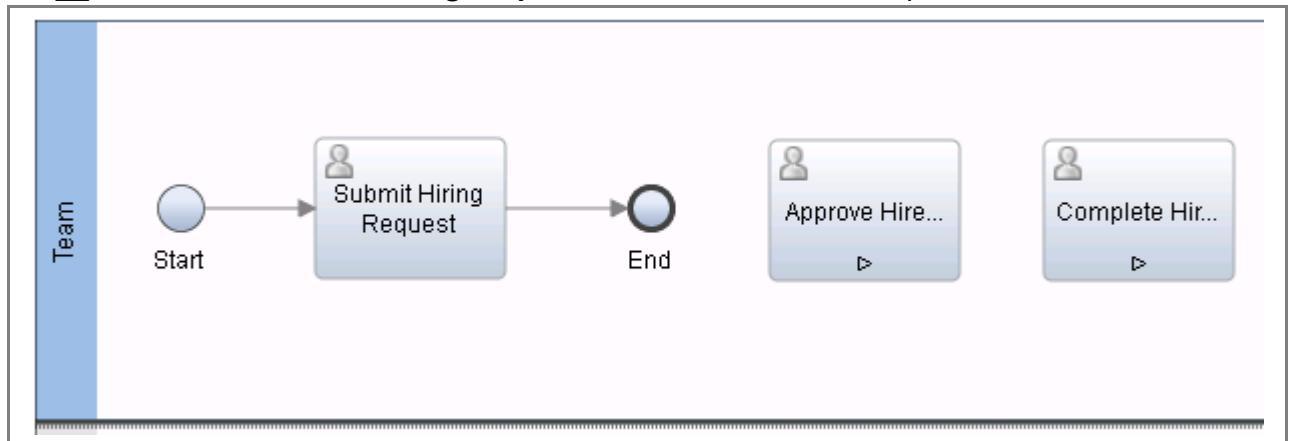
- ___ b. Hover over the **Start** event to see a flow that emerges out from the control point on the edge.



- c. Drag the sequence flow from the **Start** event to the **Submit Hiring Request** activity in the diagram and release the left mouse while you are on the anchor point on the connecting activity.

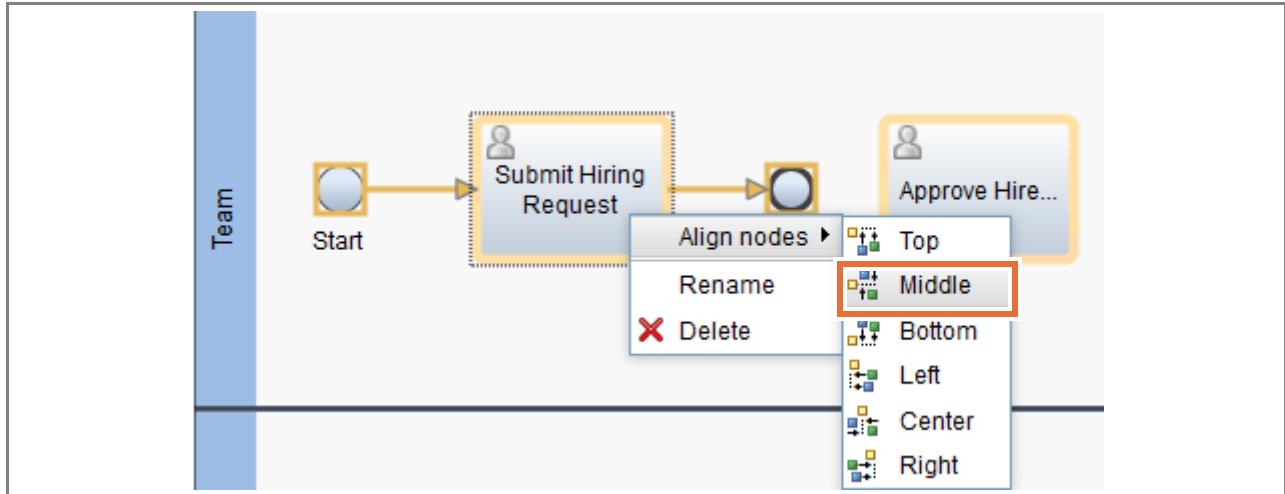


- d. Connect **Submit Hiring Request** to the **End** event to complete the flow.



**Hint**

To align all the items on the canvas, select the items to align or press Ctrl+A to select all. Right-click one of the items and click **Align nodes > Middle**.



- ___ e. Save your changes.

Part 5: Create process variables

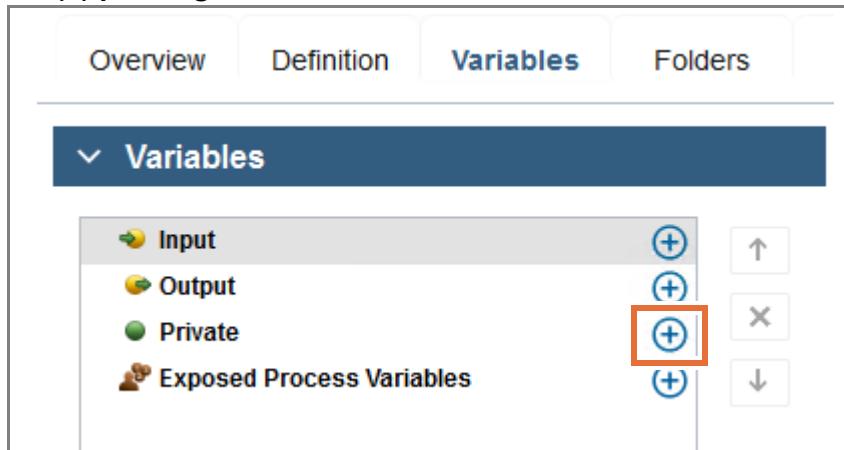
A variable contains information that can be shared between activities. A variable can have a primitive type such as string. A variable can also be a business object with either a complex or a simple type.

- 1. Create the variables.

- a. From the Hiring Request Process, click the **Variables** tab.



- b. Click the **(+)** plus sign next to Private to add a variable.



__ c. In the Details section on the right, enter the following values:

- **Name:** requisitionNumber
- **Variable Type:** String
- **List:** Cleared.
- **Visible in Process Portal:** Selected.
- **Alias:** requisitionNumber

The screenshot shows the 'Details' configuration screen for a variable. The 'Name' field contains 'requisitionNumber'. The 'Variable type' dropdown is set to 'String System Data'. The 'Visible in Process Portal' checkbox is checked. The 'Alias' field is also set to 'requisitionNumber'.



Information

For each variable whose runtime values you want to search or to make viewable in the Process Portal task list, select the **Visible in Process Portal** check box in the Business Data section. For complex variables, be sure to select the check box for each parameter that you want to make available.

- ___ d. The variable is added in the Private list under the Variables section.

The screenshot shows a user interface for managing variables in a process. At the top, there are tabs: Overview, Definition, Variables (which is selected and highlighted in blue), and Folders. Below these, a section titled 'Variables' is expanded. It contains three categories: Input (with a plus sign icon), Output (with a plus sign icon), and Private (with a plus sign icon). Under the Private category, there is a row for 'requisitionNumber (String)'. To the right of this row are icons for moving up, down, and deleting the variable. Below the Private category, there is a section for 'Exposed Process Variables' with a plus sign icon.

- ___ 2. Repeat the steps to create private variables for the rest of the properties as shown in the following table:

Name	Variable Type	List	Visible	Alias
jobTitle	String	No	Yes	jobTitle
salary	Integer	No	Yes	salary
department	String	No	Yes	department

- ___ a. Click the **(+)** plus sign next to Private to add another private variable.

This screenshot is similar to the one above, showing the 'Variables' section of a process editor. The 'Private' category is expanded, and a plus sign icon to its right is highlighted with a red box, indicating it should be clicked to add a new variable. The other elements in the interface, such as the 'Input' and 'Output' sections and the 'Exposed Process Variables' section, remain the same.

- __ b. Enter the following data for the job title:

- **Name:** jobTitle
- **Variable Type:** String
- **List:** Cleared.
- **Visible in Process Portal:** Selected.
- **Alias:** jobTitle

Details

Name:	<input type="text" value="jobTitle"/>
Documentation:	<div style="border: 1px solid #ccc; padding: 5px; height: 100px; width: 100%;"> B I U ≡ ≡ ≡ ≡ ≡ ≡ ≡ ≡ </div>
Variable type:	<input checked="" type="radio"/> String System Data Select... New...
List:	<input type="checkbox"/>
Visible in Process Portal:	<input checked="" type="checkbox"/>
Alias:	<input type="text" value="jobTitle"/>
Track this variable:	<input type="checkbox"/>
Short name:	<input type="text"/>
Process Instance Identifier:	<input type="checkbox"/>

- __ c. Repeat the previous step to create the rest of the private variables as shown in the following table:

Name	Variable Type	List	Visible	Alias
salary	Integer	No	Yes	salary
department	String	No	Yes	department

- __ d. The list of private variables appears as follows:

The screenshot shows the 'Variables' tab selected in a software interface. Under the 'Variables' section, there are three categories: 'Input' (with a plus icon), 'Output' (with a plus icon), and 'Private' (with a plus icon). The 'Private' category is expanded, showing four variables: 'requisitionNumber (String)', 'jobTitle (String)', 'salary (Integer)', and 'department (String)'. The 'salary (Integer)' variable is highlighted with a gray background. Below the private variables is a section for 'Exposed Process Variables' with a plus icon.



Hint

For Variable type Integer, click **Select** next to Variable type to select **Integer**.

The screenshot shows the 'Details' tab selected. In the 'Type' dropdown, 'salary' is selected. A 'Select Library Item' search bar is present. Below it, a list of 'Business Object' types is shown, with 'Integer System Data' highlighted by a red box. At the bottom right of the list, a 'Select...' button is also highlighted by a red box.

Part 6: Create process folders

Folders provide a way of grouping documents that are related to a process. You can create a folder structure that knowledge workers or users in Process Portal can use to add documents that are required to complete the process. You create process folders in the process editor.

- 1. Create process folders.

- a. From the Hiring Request Process, click the **Folders** tab.

The screenshot shows the 'Folders' tab selected in the top navigation bar. Below it, a 'Folders' section contains a single item: 'Root Process Folder'. To the right of this item is a 'Folder Management' panel with three checkboxes: 'Allow locally managed documents', 'Allow locally managed folders', and 'Allow external document references'. The '+' button next to the 'Root Process Folder' is highlighted with a red box.

- 2. Click (+) next to **Root Process Folder** to add a folder.

The screenshot shows the 'Folders' tab selected. The 'Root Process Folder' now has a '+' sign next to it, indicating it can be expanded. The 'Folder Management' panel on the right is identical to the previous screenshot.

- 3. Enter **Hiring Requisition** as the name.

The screenshot shows the 'Folders' tab selected. Under the 'Root Process Folder', there is a new folder named 'Hiring Requisition'. The 'Details' panel on the right shows the 'Name:' field highlighted with a red box, containing the value 'Hiring Requisition'. The 'Folder Management' panel is also visible.

- 4. Under the Folder Management section, select **IBM Business Process Manager** to manage the subfolder hierarchy locally and select all the folder permissions under it.

Details

Name: Hiring Requisition

Folder Management

IBM Business Process Manager

Allow locally managed documents

Allow locally managed folders

Allow external document references

Allow external folder references

Enterprise Content Management system

- 5. Save your changes.

Information

You created a new folder.

Overview Definition Variables Folders

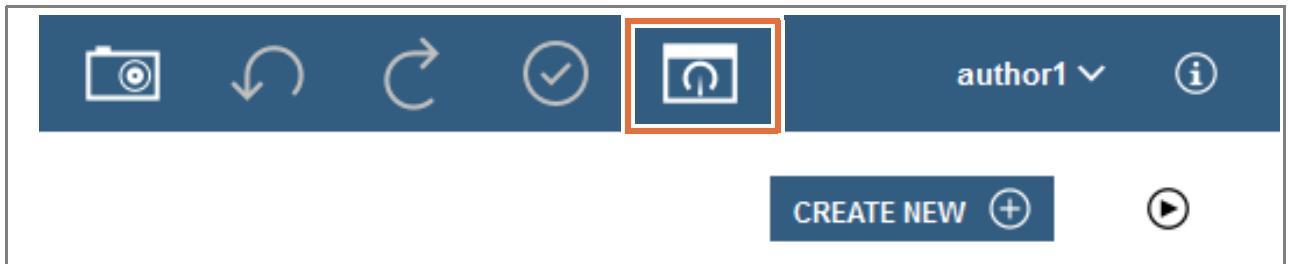
Folders

- Post Process Folder
 - Hiring Requisition

Adding folders is a way to group logically related documents similar to the paper folders on a desk. In a large, complex process, folders bring order to the many documents that accumulate over the resolution of the business problem. A folder can contain one or more subfolders.

The folders that you define for the process are the initial set of folders that are automatically created when a process is started. At run time, you can add or remove subfolders under a specific process folder. In the Folder Management section, you can either select the subfolder hierarchy that IBM Business Process Manager manages locally, or provide reference to the folders that the external ECM server manages.

6. Click the **Process Center** icon on the upper-right corner of the Process Designer window to return to the Process Center view.



Part 7: Test the process activities in Process Portal

You can test the process with ad hoc activities that you built in the Process Portal.

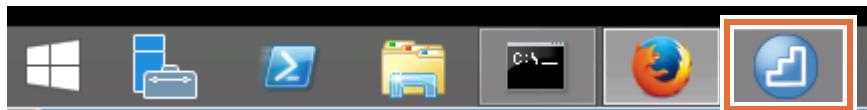
- __ 1. Log in to the process portal.



Cloud

For IBM BPM on Cloud users: Click **Launch** in the Process Portal tile of your IBM BPM on Cloud home screen to access the Process Portal.

- __ a. Return to the IBM Business Process Manager Quick Start window, which you minimized before.



- __ b. Click **Process Portal**.

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.

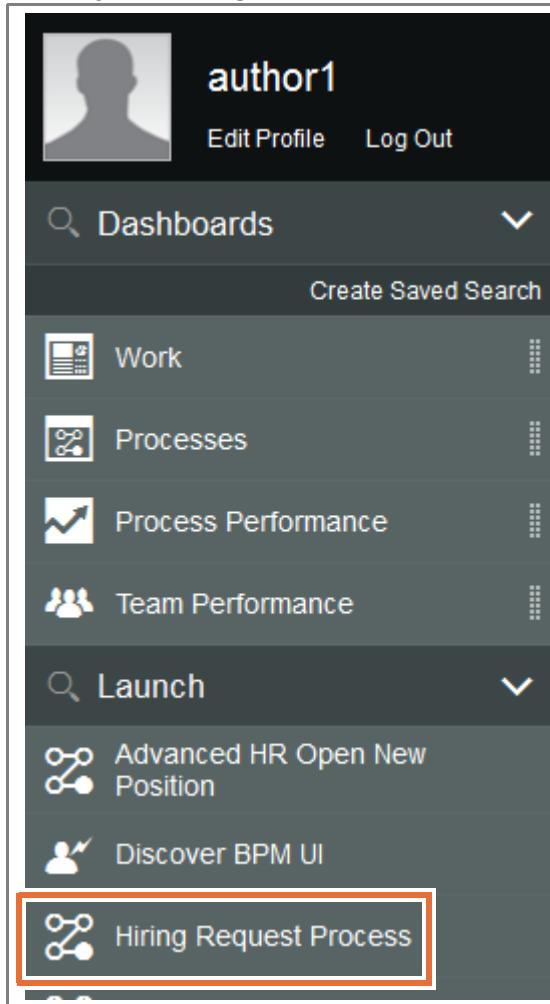
[Knowledge Center](#)



Note

If prompted for credentials, enter **author1** for the **User name** and **author01** for the **Password**. Click **Continue**.

- 2. On the left pane of the Process Portal browser window, start an instance of the Hiring Request Process by clicking the **Hiring Request Process** link under the Launch section.



A process instance is started. You see the message at the bottom of the window.



Note

Depending on the environment, it might take up to a minute to create the first instance of the process, and to wait for the notification to show in the upper-right corner of the portal page. After the data is cached, subsequent instances that are created take less time.

- 3. Under **Work**, the **Submit Hiring Request** activity shows as an open task.

The screenshot shows the Work dashboard. At the top right, it displays '1 Total Open' with '1 On Track', '0 At Risk', and '0 Overdue'. Below the header is a search bar with placeholder text 'Enter search text...' and two search icons. The main area contains a card for a task titled 'Step: Submit Hiring Request' from the 'Hiring Request Process:3'. The card includes details: 'All Users' and 'Due: Nov 3, 2017, 8:44:56 AM'. There are also two small action buttons at the bottom of the card.

- 4. In the Library menu, click **Processes**.

The screenshot shows the Library menu on the left and the Work dashboard on the right. The Library menu has a user profile icon for 'author1', 'Edit Profile', and 'Log Out'. It lists several options: 'Dashboards' (with a dropdown arrow), 'Create Saved Search', 'Work', 'Processes' (which is highlighted with an orange rectangle), 'Process Performance', 'Team Performance', and 'Launch'. The Work dashboard on the right shows the same task card as the previous screenshot.

Information

Sometimes the first time you load a page, it takes longer to load than normal so that the pages can enter the cache. If the page doesn't load within 1 minute, you might be required to click the Processes link twice to view the Processes page.

- 5. The **Hiring Request Process** instance is listed. Click the **Hiring Request Process** instance.

The screenshot shows a search bar at the top with the placeholder "Type a search filter". Below it, a process instance card for "Hiring Request Process:3" is displayed. The card includes a circular icon with arrows, the process name, a star icon, and a creation date of "Created: November 3, 2017 7:44 AM".

Showing 1 out of 1 results

- 6. The process instance window lists the open and ad hoc tasks.

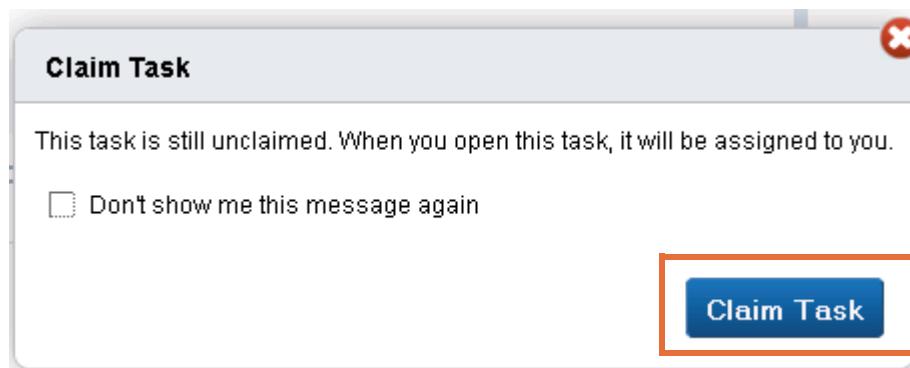
The screenshot shows the "Hiring Request Process:3" process instance window. It includes sections for "Data", "Documents", "Tasks", and "Activities".

- Data:** Displays fields for Department, Job Title, Requisition Number, and Salary, all set to "[not defined]".
- Documents:** Shows a single document titled "Hiring Requisition" created by "author1" at 7:44 AM.
- Tasks:** Shows an open task titled "Step: Submit Hiring Request" due at 8:44 AM, assigned to "All Users".
- Activities:** Shows a Gantt chart for the task, indicating it was created on Nov 3, 2017, at 7:44 AM, and is due on Nov 3, 2017, at 8:44 PM. It also lists three activities: "Approve Hire Request", "Complete Hire Request", and "Review Salary".

7. Click **Step: Submit Hiring Request** under the Tasks section.

The screenshot shows the 'Hiring Request Process:3' application interface. At the top, there's a blue header bar with the title 'Hiring Request Process:3'. Below it is a light gray header with the same title. The main area has a light gray background. On the left, there's a 'Data' section containing placeholder text for Apartment, Job Title, Position Number, and Salary. In the center, there are two tabs: 'Documents' and 'Tasks'. The 'Documents' tab is active, showing a single item: 'Hiring Requisition' by 'author1' at 7:44 AM. The 'Tasks' tab is also present. A red box highlights the task 'Step: Submit Hiring Request' in the 'Tasks' list. This task is marked as 'Open | Completed' and has a due date of November 3, 2017, at 8:44 AM, assigned to 'All Users'.

8. Click **Claim Task**.



- 9. The task is assigned to you and the default human service coach is shown. Click **Done** to complete the task.

The screenshot shows a task interface titled "Step: Submit Hiring Request". At the top, there are three icons: a menu icon, the task title, and a more options icon. Below the title, there are two tabs: "Overview" (selected) and "Details". The main content area contains two boxes: one on the left with the text "This is a sample coach for the following" and one on the right with the text "Step: Submit Hiring Request". At the bottom, there are two buttons: "Done" (highlighted with an orange border) and "Complete Later".

- 10. The task is completed with no more open tasks.

The screenshot shows a "Tasks" section. At the top, there is a header with a dropdown arrow and the word "Tasks". To the right of the header are two buttons: "Open" and "Completed". Below the header, the text "No tasks were found." is displayed. The entire "Tasks" section is enclosed in a light gray box.

- 11. After you click the **Completed** tasks filter, you can view the task in the completed tasks list.

The screenshot shows the IBM Process Activity Monitor interface. On the left, the **Tasks** pane is open, displaying a single task: "Step: Submit Hiring Request" (Completed: November 3, 2017 9:42 AM, Completed by author). The "Completed" filter button is highlighted with a red box. On the right, the **Activities** pane shows three ad hoc activities: "Approve Hire Request" (Ready), "Complete Hire Request" (Ready), and "Review Salary" (Ready). A "Filter" button is also present in the Activities pane.

- 12. Click the ad hoc activity **Approve Hire Request**, which is in a Ready state on the right pane.

The screenshot shows the IBM Process Activity Monitor interface. The **Activities** pane is displayed, showing three ad hoc activities: "Approve Hire Request" (Ready), "Complete Hire Request" (Ready), and "Review Salary" (Ready). The "Approve Hire Request" activity is highlighted with a red box. Above the activities, a timeline bar indicates the task was created on Nov 3, 2017 at 7:44 AM and due on Nov 3, 2017 at 3:44 PM, with a progress bar showing it has been processed.

— 13. Click **Start** to start this task.

The screenshot shows a software interface for managing tasks and activities. At the top, there is a task card with the following details:

- Icon:** A green gear icon.
- Created:** Nov 3, 2017, 7:44 AM.
- Due:** Nov 3, 2017, 3:44 PM.
- Stream:** Stream icon.

Below the task card is a section titled **Activities** with a **Filter** button. There are four activity items listed:

- Approve Hire Request:** This activity is required and will start a single task. The **Start** button is highlighted with a red box.
- Complete Hire Request:**
- Review Salary:**

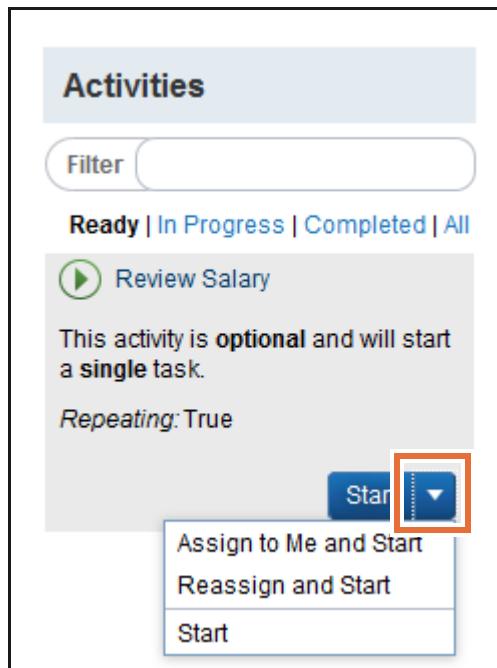
- 14. The **Approve Hire Request** task is created. After you click the **Open** task filter, the open tasks list opens and you can see your open tasks for this instance.

The screenshot shows the IBM Process Activity Monitor interface. On the left, the **Tasks** pane is open, displaying a single task: **Step: Approve Hire Request**. The task is due on November 3, 2017, at 10:49 AM and is assigned to **All Users**. The **Open** button is highlighted with a red box. On the right, the **Activities** pane is open, showing two items: **Complete Hire Request** and **Review Salary**. The **Created** section shows the task was created on Nov 3, 2017, at 7:44 AM, and it is due on Nov 3, 2017, at 3:44 PM. A progress bar indicates the task is 100% complete. Below the progress bar is a **Stream** button.

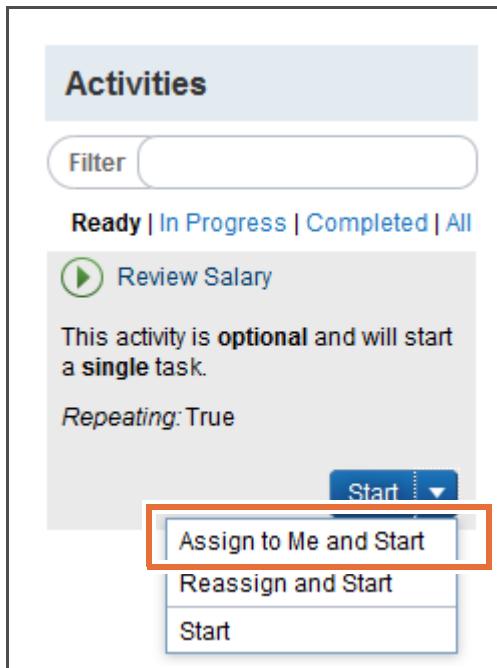
- 15. Select the ad hoc activity **Complete Hire Request** in the right pane and click **Start** to start this task. It is also added in the open tasks list.

The screenshot shows the **Activities** pane. The **Complete Hire Request** activity is selected and highlighted with a red box around its button. A tooltip message states: "This activity is required and will start a single task." Below the activity, the **Start** button is also highlighted with a red box.

- 16. Select the optional activity **Review Salary** in the right pane and open the menu next to the Start button.



- 17. Select **Assign to Me and Start**.



- 18. The task is assigned to you and the default human service coach is shown. Click **Done** to complete the task.

This is a sample coach for the following activity

Step: Review Salary

Done

Complete Later

- 19. Because the **Review Salary** activity is repeatable, it remains in the Ready (optional) state.

▼ Tasks

Open | Completed

Step: Approve Hire Request

Step: Complete Hire Request

Activities

Filter

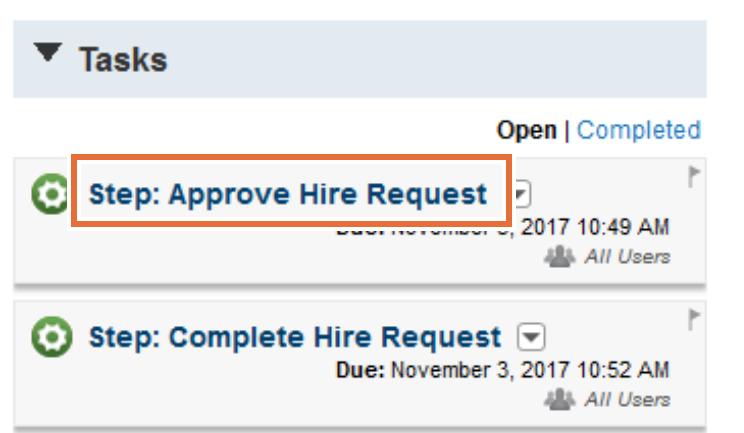
Ready | In Progress | Completed | All

Review Salary

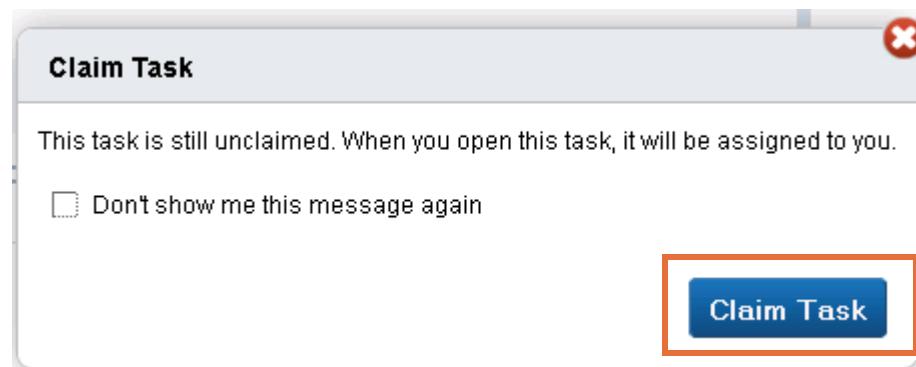


You can click the Review Salary activity and complete the activity again.

- __ 20. Complete the **Approve Hire Request** task.
__ a. Click the **Step: Approve Hire Request** task.



- __ b. Click **Claim Task**.



- __ c. Click **Done** to complete this task.

The screenshot shows a user interface for a task titled "Step: Approve Hire Request". At the top, there are three icons: a menu icon, the title, and a more options icon. Below the title, there are two tabs: "Overview" (selected) and "Details". In the main content area, there are two boxes: one containing "This is a sample coach for the following" and another containing "Step: Approve Hire Request". At the bottom, there are two buttons: "Done" (highlighted with a red box) and "Complete Later".

- __ 21. Complete the **Complete Hire Request** task.

- __ a. Click the **Step: Complete Hire Request** link.
- __ b. Click **Claim Task**.
- __ c. Click **Done** to complete the task.

All the tasks are completed and the Tasks section is empty.

The screenshot shows a user interface section titled "Tasks". It includes a header with a downward arrow and the word "Tasks", a "Open | Completed" button, and a message box stating "No tasks were found." The "No tasks were found." message box is highlighted with a red box.

- 22. After you complete all the activities, click **Completed** under the Tasks section to show the list of completed tasks.

The screenshot shows a list of tasks under a 'Tasks' section. There are four items listed, each with a gear icon and a title. The titles are: 'Step: Complete Hire Request', 'Step: Approve Hire Request', 'Step: Review Salary', and 'Step: Submit Hiring Request'. Each item also includes a timestamp and the text 'Completed by author1'. At the top of the list, there are two buttons: 'Open' and 'Completed'. The 'Completed' button is highlighted with a red border.

Step	Completed By	Date
Step: Complete Hire Request	author1	November 3, 2017 10:11 AM
Step: Approve Hire Request	author1	November 3, 2017 10:09 AM
Step: Review Salary	author1	November 3, 2017 9:55 AM
Step: Submit Hiring Request	author1	November 3, 2017 9:42 AM

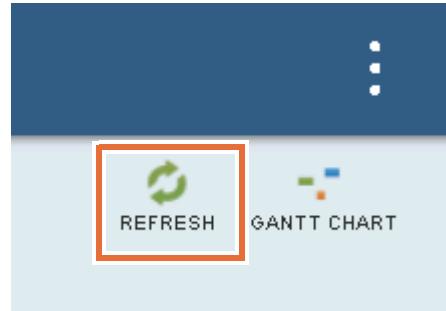
After the tasks status updates, the instance is completed. It might take up to 1 minute for the system to process the update.

The screenshot displays a summary of a completed instance. At the top, it says 'This instance has completed.' Below that is a green gear icon. To the left of the gear is the word 'Created' and the date 'Nov 3, 2017' followed by the time '7:44 AM'. To the right of the gear is the word 'Completed'. A horizontal progress bar is shown, starting at 'Created' and ending at 'Completed', with a blue segment indicating progress. Below the progress bar is a speech bubble icon and the word 'Stream'.

Created	Completed
Nov 3, 2017 7:44 AM	Completed

**Note**

If you do not see the instance completed message, click the **Refresh** icon next to the GANTT CHART icon. All the tasks are complete, but the system needs time to update the instance status.



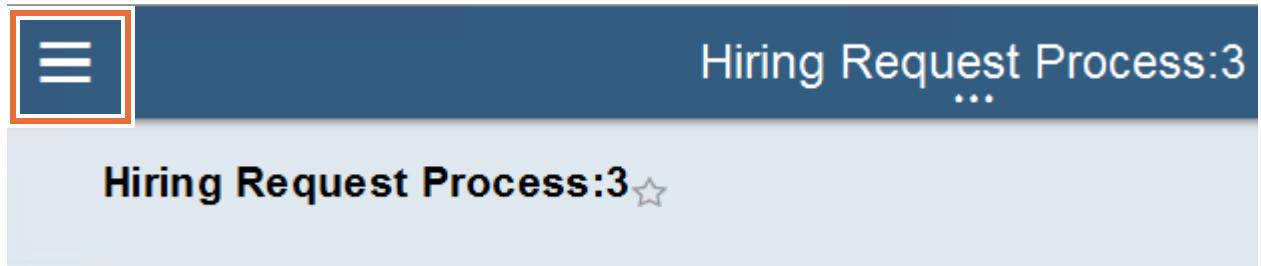
- 23. Click the **Stream** to view the details of the activities. Click the **Context menu** icon at the upper-right corner of Process Portal that is shown as three vertical dots. The sections to view the users that follow this instance and the mentions of this task are shown on the right in the menu.

The screenshot shows the Process Portal Stream interface. At the top, there is a header with the text "Process:3" and a "REFRESH" button. Below the header, a message box says "This instance has completed." with a gear icon. It shows two states: "Created" and "Completed". A timestamp "7:44 AM" is followed by a "Stream" button, which is highlighted with a red rectangle. To the right of the Stream button is a context menu icon (three vertical dots) enclosed in a red box. On the far right, there are sections for "Following" and "Mentions", each with a count of 0 and a dropdown arrow. The main content area displays a list of activity logs:

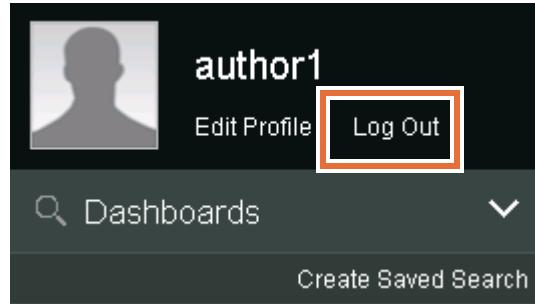
- All work for **Hiring Request Process:3** is complete.
- author1** completed the **Step: Complete Hire Request** task.
Show audit trail
- Step: Complete Hire Request** task was claimed (currently assigned to **author1**)
- author1** completed the **Step: Approve Hire Request** task

A "Post" button is located above the activity logs.

- ___ 24. Click the **Main Menu** icon on the left side of the window.



- ___ 25. Click **Log Out**.



- ___ 26. Close the Process Portal tab in the browser.

- ___ 27. Minimize the browser.

End of exercise

Exercise review and wrap-up

In the first part of this exercise, you created a process application and a process. Then, you built ad hoc activities and tested the process instance in Process Portal.

Exercise 2. Playback 0: Creating a structured process

Estimated time

01:30

Overview

This exercise covers how to create a process application and business process definition (process) in the Process Center, and model teams for the process. It also covers how to model task-type activities, create a linked process, and decompose a process into nested processes.

Objectives

After completing this exercise, you should be able to:

- Create a nested process
- Create the foundation for a process by adding the appropriate lanes to the default pool
- Translate business process workflow steps that are documented in the process discovery and analysis into process model tasks
- Model the expected process flow for the initial process model
- Decompose business process workflow steps that are documented in the process discovery and analysis into process model tasks
- Create a subprocess or a linked process

Introduction

To accomplish the task of creating a process model in a process in the IBM Process Designer, you created a process application in the previous exercise to contain the model. You created an as-is process model by adding some ad hoc activities based on the detailed information that the Hiring Requisition process owner provided.

In this exercise, you create a to-be process model by enhancing the as-is process model that you created in the previous exercise. To accomplish the task of creating the initial process model, you define a pool, lanes, and flow objects such as activities, events, and nested processes. Based on the information that is provided, a process based on process discovery and analysis is created. Based on the information that is provided, you model the teams.

In this exercise, you add activities to the appropriate lanes and use sequence flow to connect the activities. As mentioned in the unit, you model the happy path first. Gateways and various flows are presented in the next exercise.

Finally, you complete decomposition on your process and create subprocesses or linked processes where you see opportunities for them.

Remember that some processes might not need decomposition.

Requirements

None

Exercise instructions

Hiring Requisition process

The company is expanded from the simple (as-is) hiring process to a more detailed (to-be) hiring process. The detailed process now includes more information. The company also wants to move to reduce rework by introducing a more structured process for the Hiring Requisition process. The process that you are going to examine and model is called the Hiring Requisition Process, the foundation for which you created in the first exercise. This process covers a new job position through submission, approval, and completion so applicants can apply for the job position.

Core requirements

- 1.1: A Hiring Manager submits a hiring requisition to the HR Department. The request contains the following information:
 - Customer details:
 - Requisition number
 - Date of request
 - Requester
 - Date position available
 - Job title
 - Job description
 - Job level
 - Number of people managed
 - Division
 - Department
 - Salary to offer
 - Bonus amount
 - Hiring Manager comments
 - New position
- 2.1: If the answer to “New position” is Yes, the request is forwarded to a General Manager. After the General Manager receives the request, the General Manager indicates approval or disapproval.
- 2.2: If the request is not approved, the General Manager specifies a reason and the request is closed. If the request is approved, a salary compliance check is conducted.
- 2.3: The Hiring Manager is notified of the General Manager’s decision after the General Manager approval step.
- 2.4: After the requisition is submitted, an automated system checks for salary compliance. If the request meets salary compliance, the hiring request is automatically posted to the HR Positions database and made available for dissemination.

- 2.5: When a request violates the established salary guidelines of the company, the HR Administrator can approve or reject the requested salary override.
- 2.6: If the salary override is approved, the request is posted to the HR Positions database and made available for dissemination.
- 2.7: If the HR Administrator rejects the requested salary, the HR Administrator must provide comments for the violation, add a proposed salary, and send the request back to the Hiring Manager who originated the request.
- 2.8: When the Hiring Manager gets the request back because of a rejection, the Hiring Manager can negotiate an adjusted salary or can cancel the request. If the negotiation is successful, the request is resubmitted back to the same HR Administrator.
- 2.9: All hiring requests must be added to the HR Positions database regardless of the disposition at the end of the process during a finalization activity.
- 2.10: The HR Administrator has 4 hours to complete the review. If the review is not completed within 4 hours, an email is sent to the HR Administrator. The email notifies the HR Administrator of the missed deadline.

The process owner provides detailed information about the process and its current state to the IBM BPM analyst, who in turn documents the information and analyzes the process for improvement. The process discovery and initial analysis are already completed for you to proceed with the creation of the to-be process model.

Part 1: Open the business process in Process Center

To enhance the previously created Hiring Request Process business process in IBM Business Process Manager Process Designer, return to the Process Designer window, which you minimized in the earlier exercise.

1. Open the **HR Recruitment Processes (HRR)** process application unless it is already open.

The screenshot shows the 'Process Apps' interface. At the top, there are tabs for 'Process Apps', 'Toolkits', 'Servers', 'Admin', and links for 'Preferences' and 'Logout'. Below the tabs, there is a search bar with 'Sort By: Recently Updated' and filters for 'All', 'Favorites', and 'Archived'. A list of applications is shown, with 'HR Recruitment Processes(HRR)' selected. The application card includes a star icon, a last updated timestamp, and an 'Open in Designer' button, which is highlighted with a red box.

2. Click **Processes**.

The screenshot shows the 'DESIGNER' interface. The top navigation bar includes 'DESIGNER', 'INSPECTOR', and several icons. A dropdown menu 'Process App Settings' is open, showing 'Process App Settings' and 'Common'. On the left, a sidebar lists 'HR Recruitment Processes', 'Processes' (which is highlighted with a red box), and 'User Interface'. The main panel displays 'Process App Settings' for 'Common'.

3. Click the **Hiring Request Process**.

The screenshot shows the 'DESIGNER' interface with the 'Processes' tab selected. The top navigation bar includes 'DESIGNER', 'INSPECTOR', and several icons. A dropdown menu 'Process App Settings' is open, showing 'Process App Settings' and 'Common'. On the left, a sidebar lists 'HR Recruitment Processes', 'Processes', and 'User Interface'. The main panel displays 'Process App Settings' for 'Common'. In the center, a list of processes is shown, with 'Hiring Request Process' selected and highlighted with a red box.

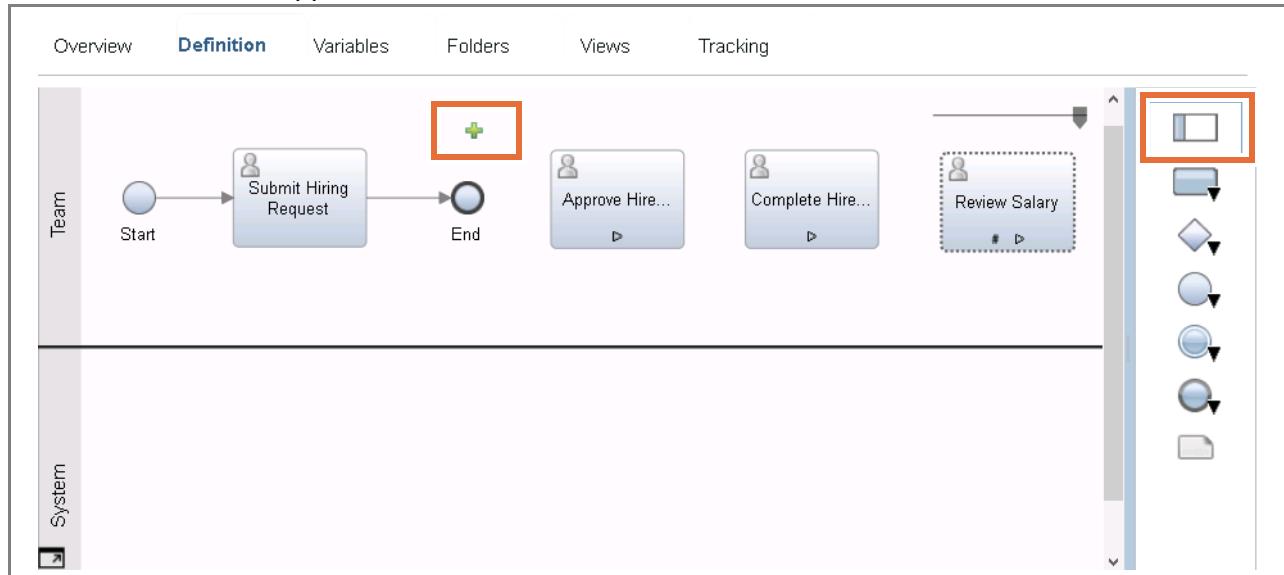
4. The process opens in the Process Designer editor.



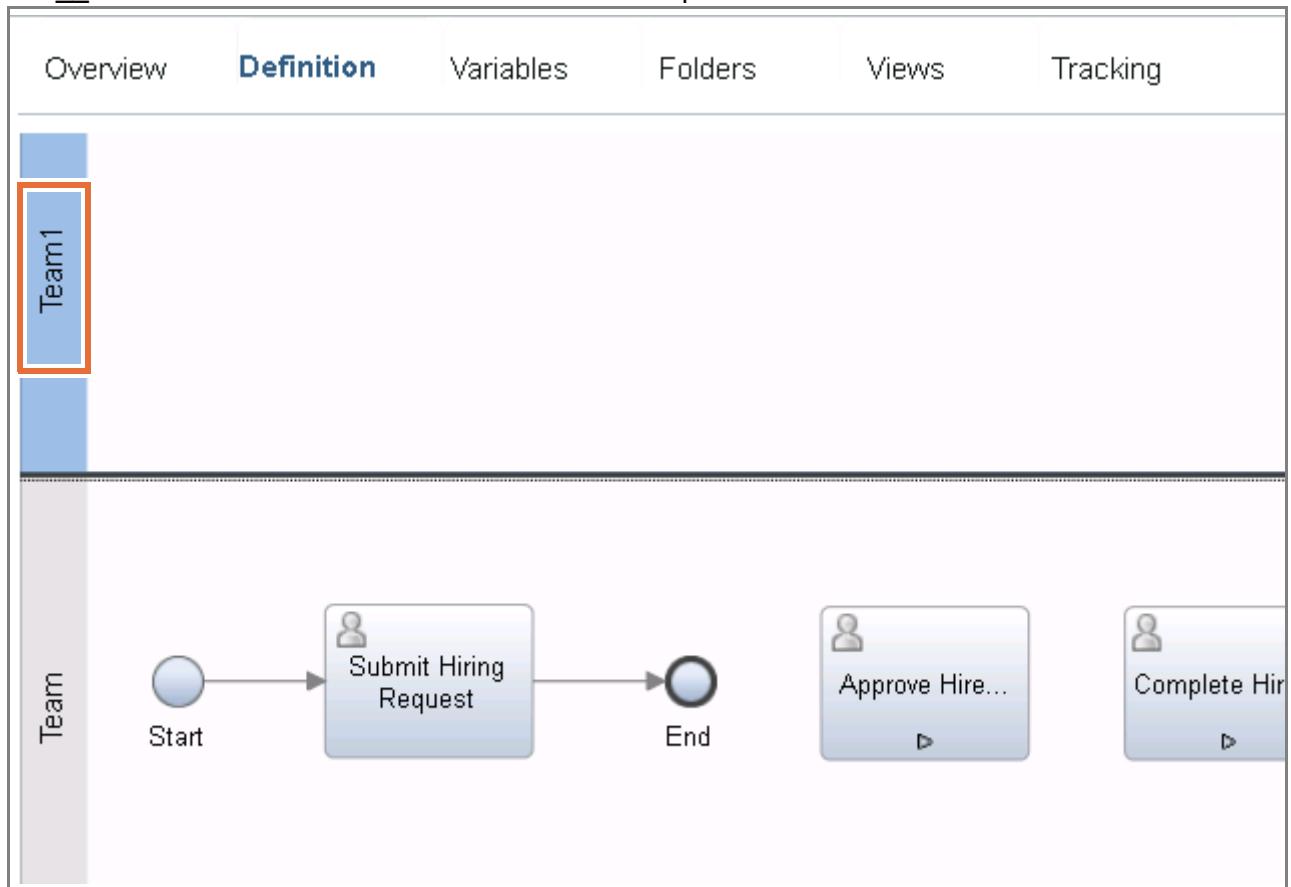
Part 2: Add lanes to the business process

The business process application has a process with some ad hoc activities. By default you have a process with a team lane, a system lane, a start event, and an end event. In this section, you add the necessary team lanes to the two default lanes (team and system) and rename the lanes.

- 1. Model the process teams.
 - a. Read the core requirements in this exercise, pages 2-3 and 2-4, and identify the teams. In the process requirements, two teams are in the main process: Hiring Manager and General Manager. Also, the process has one System lane for a total of three lanes.
 - b. Click the **Lane** icon and drag one lane from the palette to the process above the Team lane unless you see the green (+) symbol. When you release your mouse button, the new lane appears.



___ c. A lane with name Team1 is added to the process.



__ 2. Set the properties of the team lanes.

__ a. Click the top lane to select the entire lane. The **Properties** tab populates at the bottom of the screen.

The screenshot shows a BPMN process diagram with four lanes:

- Team1**: Top lane, highlighted with a red border.
- Team2**: Second lane from top.
- System Data**: Third lane from top.
- All Users**: Bottom lane.

The Properties panel at the bottom displays the following settings for the selected lane:

- General** tab is active.
- Common** section:
 - Name: Team1
 - Color: A color swatch with several colored circles.
 - ID: 7747ce11-5bb9-4fe1-81c8-f49543a10f6d
 - Documentation: A rich text editor interface.
- Behavior** section:
 - Default lane team: All Users (selected)
 - Is system lane:

__ b. In the **Properties > General > Common** section, change the name of the top lane to: Hiring Manager

The screenshot shows the same BPMN process diagram as before, but the top lane has been renamed:

- Hiring Manager**: Top lane, highlighted with a red border.
- Team2**: Second lane from top.
- System Data**: Third lane from top.
- All Users**: Bottom lane.

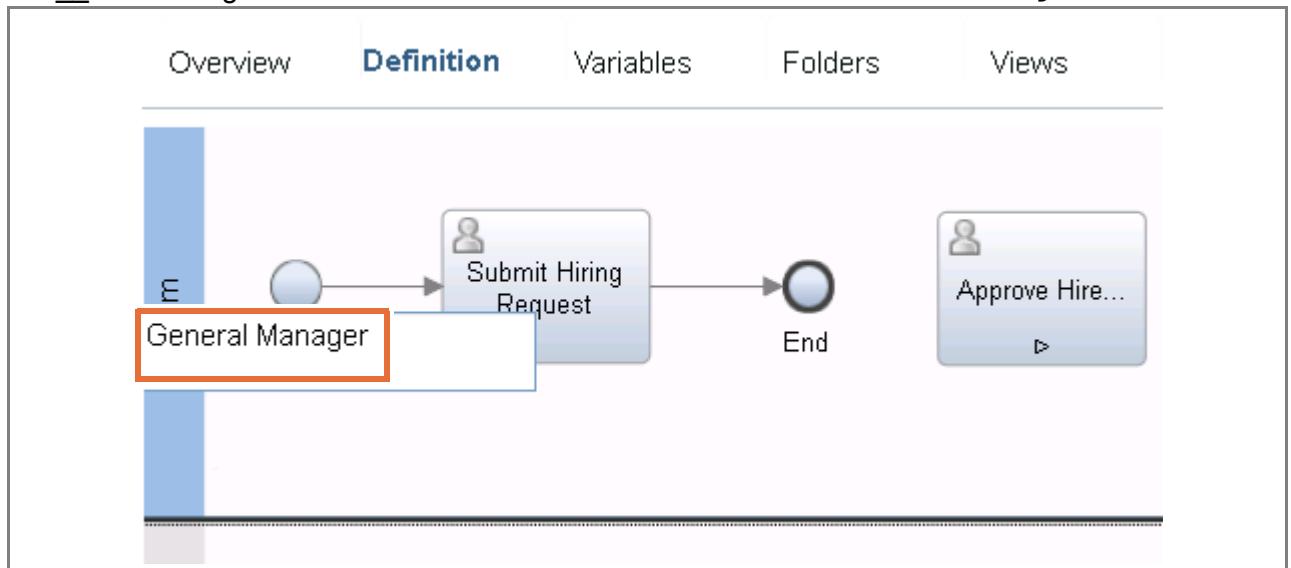
The Properties panel at the bottom displays the following settings for the selected lane:

- General** tab is active.
- Common** section:
 - Name: **Hiring Manager**
 - Color: A color swatch with several colored circles.
 - Documentation: A rich text editor interface.
- Behaviour** section:
 - Default lane team: All Users (selected)
 - Is system lane:

__ 3. Rename the second lane labeled Team.

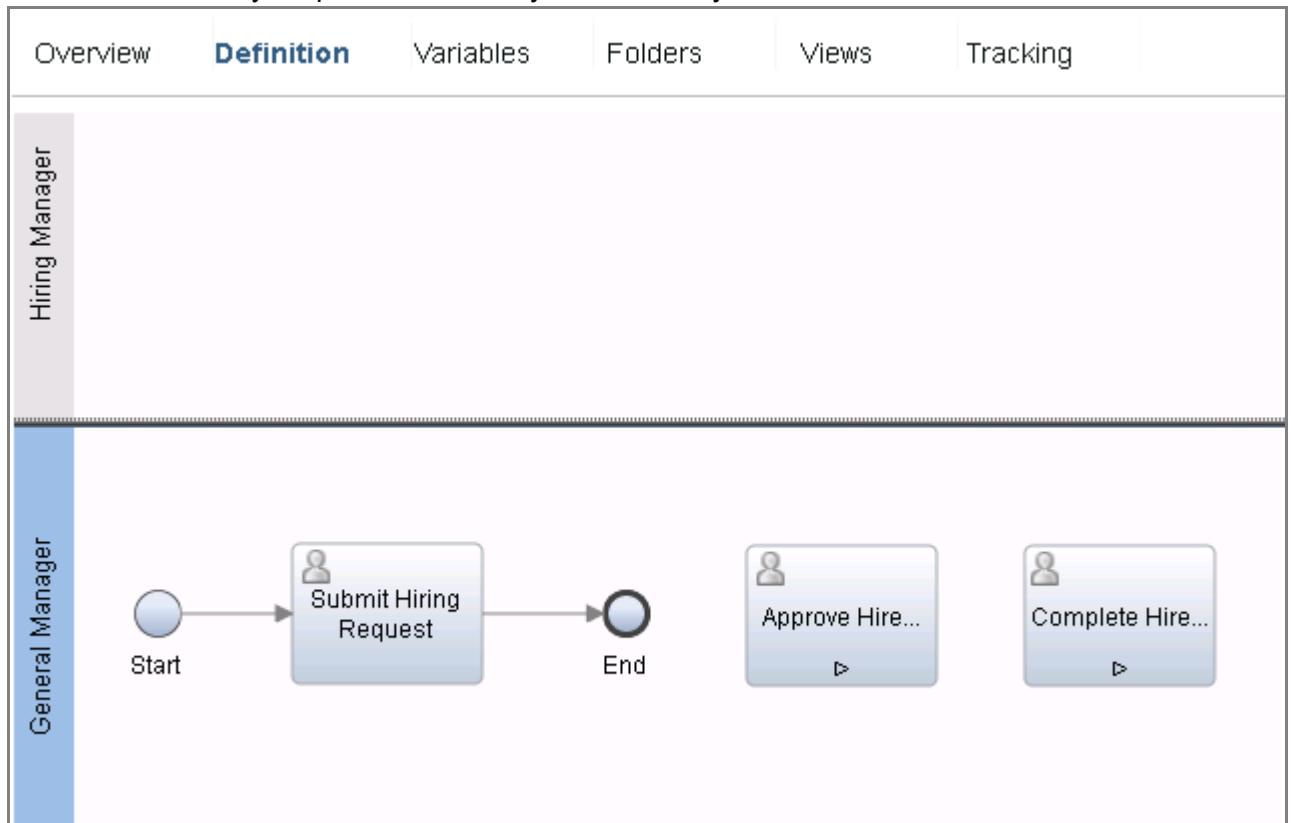
__ a. Select the lane that is labeled **Team** and double-click the name.

- __ b. Change the name of the middle lane from **Team** to: General Manager



Note

This step leaves the bottom lane as the System lane. It is a good practice to keep the System lane at the bottom of your process, even if you have no system lane activities.

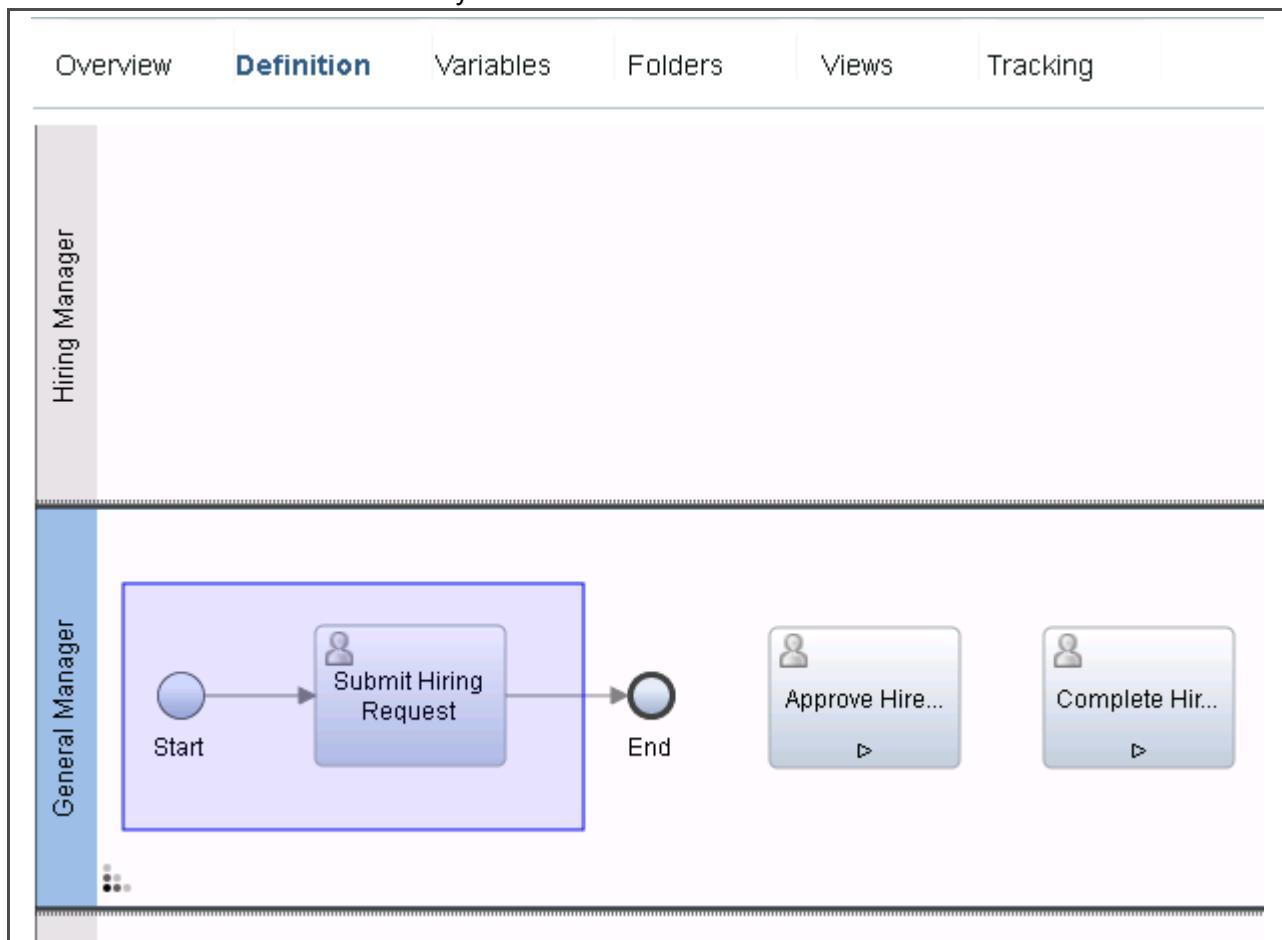


Part 3: Model the activities in the process

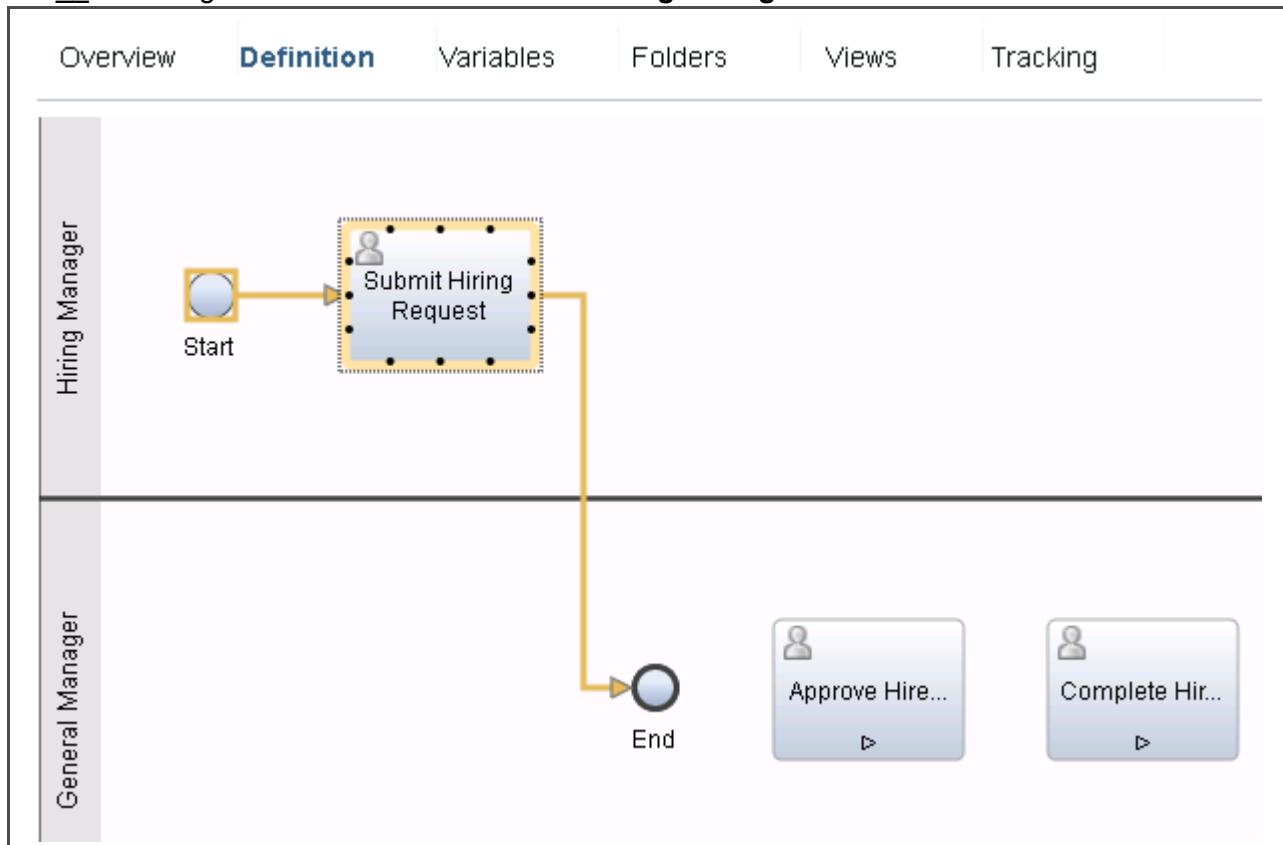
To accomplish the task of adding activities and more events to the process model, use the process requirements that are provided at the beginning of Exercise 2. Read the main process requirements and determine the activities from the requirements. Because the requirements were written down, it might be easier to write down the activities on paper before proceeding. Remember, activities use a verb-noun naming convention. If you read the requirements carefully, notice that the main process is described from item 2.1 through item 2.4. In the process requirements, four activities are in the main process: Submit Hiring Request, Approve New Hire Request, Approve Hire Request, and Complete Hire Request.

You identify the teams that are associated with the lanes instead of allowing all users to claim the tasks that are associated with each activity. Determine which teams conduct each of the four activities. From the process requirements, determine the following assignments:

- Hiring Manager (team): Submit Hiring Request
 - General Manager (team): Approve New Hire Request
 - System (team): Approve Hire Request and Complete Hire Request
- 1. Arrange the activities to the team lanes to establish the correct process flow. Model the activities and set their properties.
- a. Select the first two items in the process. To drag the Start event and the Submit Hiring Request activity to the **Hiring Manager** lane, select both items by dragging a **box** around both items with your cursor.



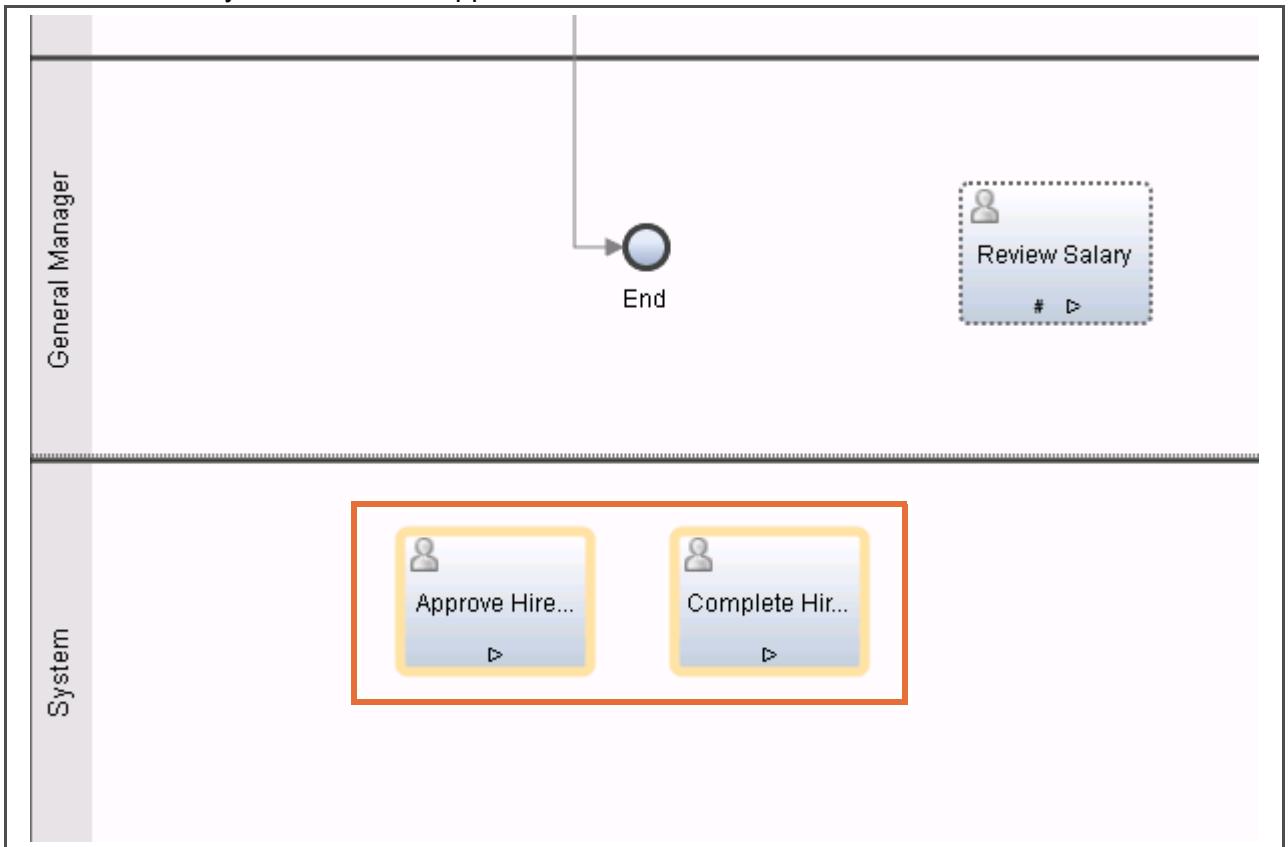
- __ b. Drag the selected activities in the **Hiring Manager** lane.



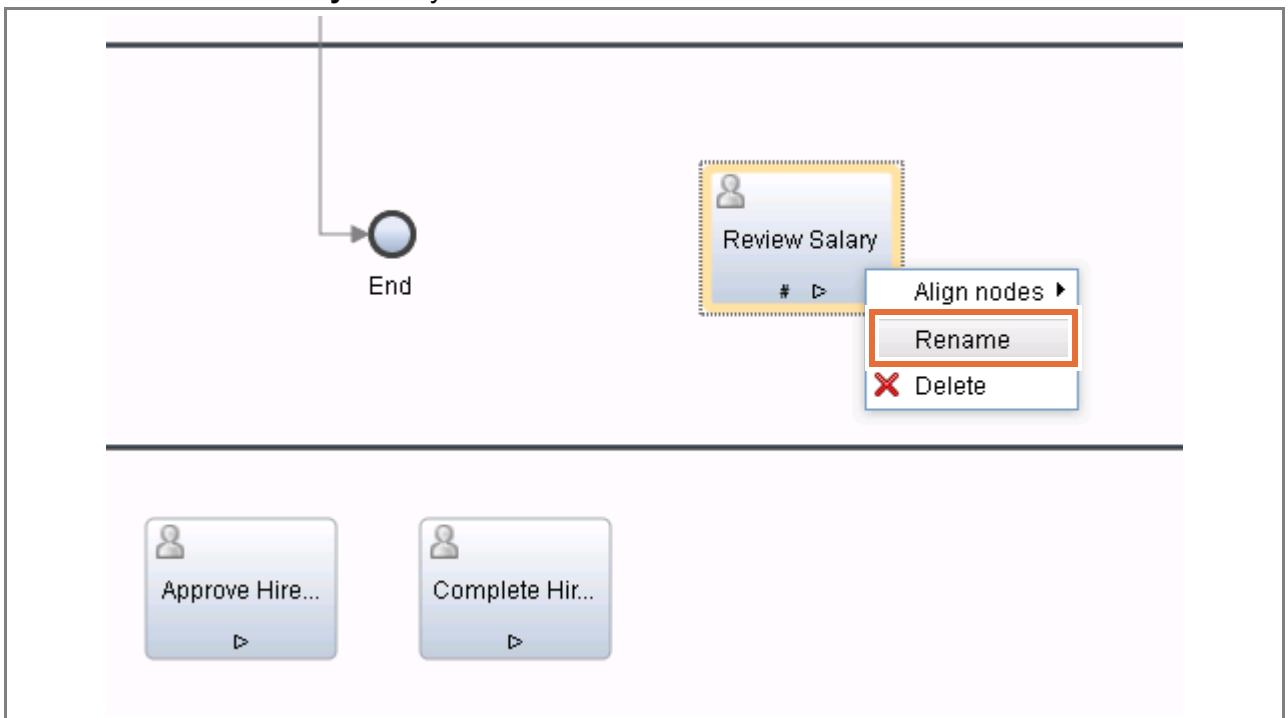
- __ c. Drag a box to select the **Approve Hire Request** activity and the **Complete Hire Request** activity.



- ___ d. Drag both of the activities to the **System** lane. Later in the exercise you rearrange this activity in a new lane: **Approvers**.



- ___ e. In this scenario, a single activity to review the salary is insufficient, and the feedback you receive is that the label is misleading to people who view the process. Right-click the **Review Salary** activity and select **Rename**.



- ___ f. Enter the name: Approve New Hire Request



Reminder

According to the scenario, if the answer to “New position” is Yes, the request is forwarded to a General Manager. After the General Manager receives the request, the General Manager indicates approval or disapproval.

- ___ g. While the **Approve New Hire Request** activity is selected, view the **Properties > General > Behavior** section. Change this activity from optional to required. Select the option: **Yes. The activity is required**.

▼ Behavior

The activity runs even though it does not have an inbound flow

How is the activity started?

Automatically by the system
 Manually by the user

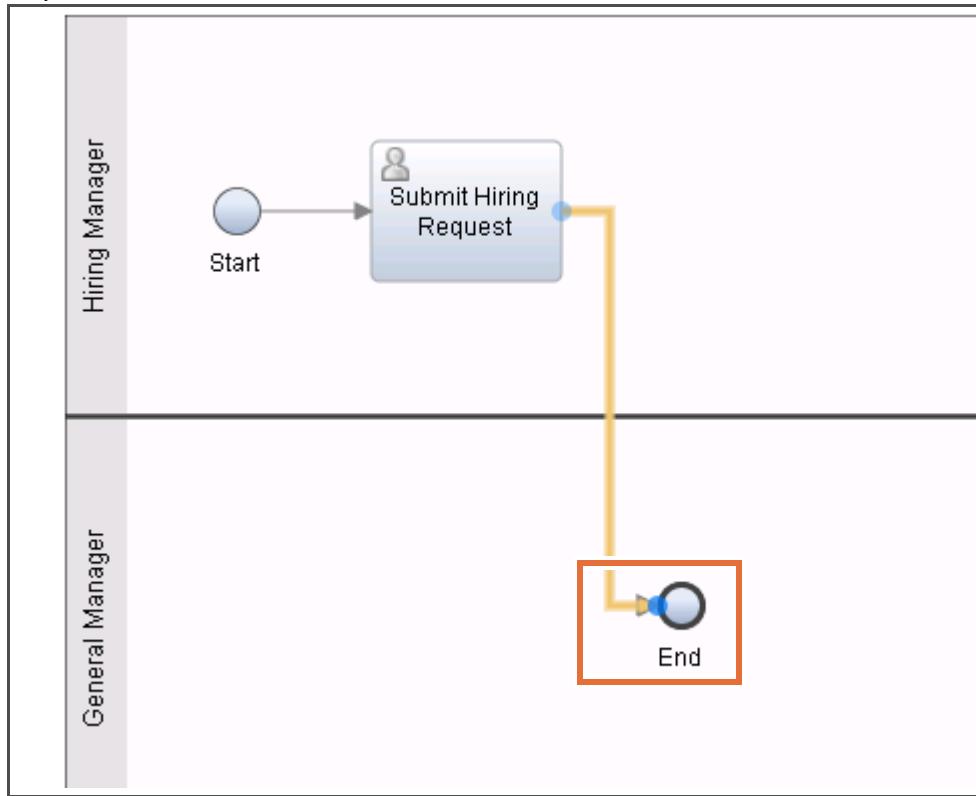
Does the activity have to be completed?

Yes. The activity is required
 No. The activity is **optional**

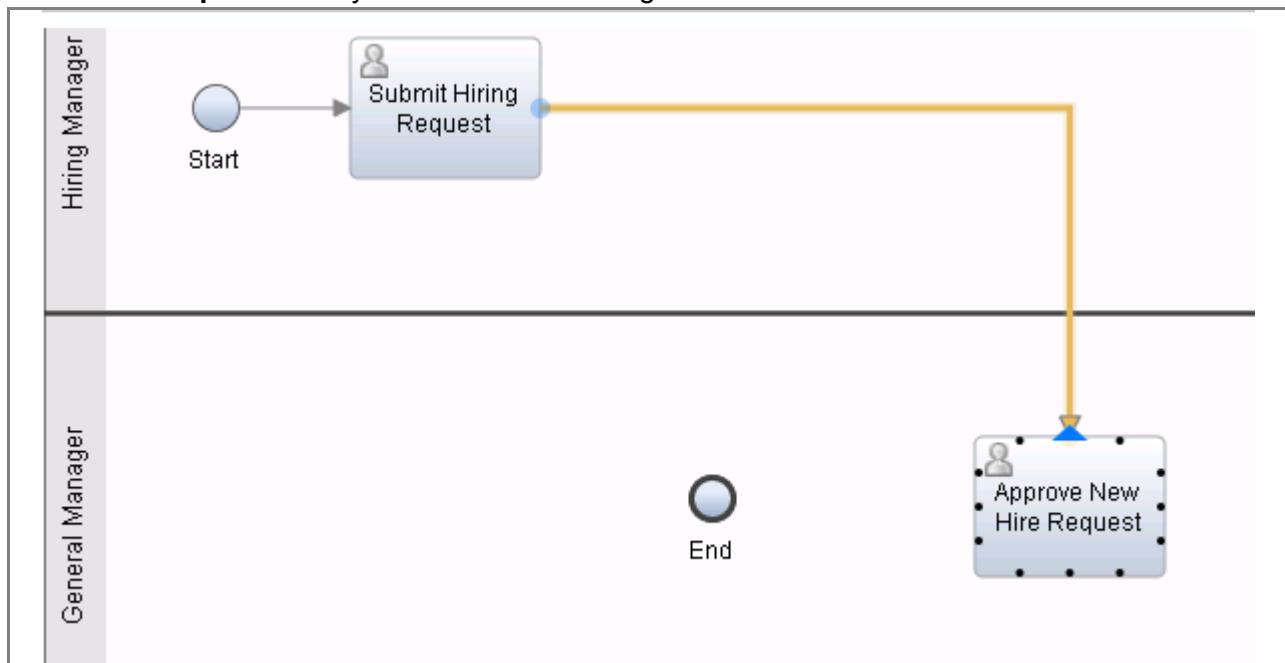
Repeatable. The activity can be invoked multiple times (i)

Hidden. This is a background activity that users will not see

- 2. Add sequence flow to the components in the diagram.
- a. Select the sequence flow between **Submit Hiring Request** activity and the **End** event.
 - b. Hover over the anchor point on the **End** event to highlight the blue endpoint of the sequence flow.

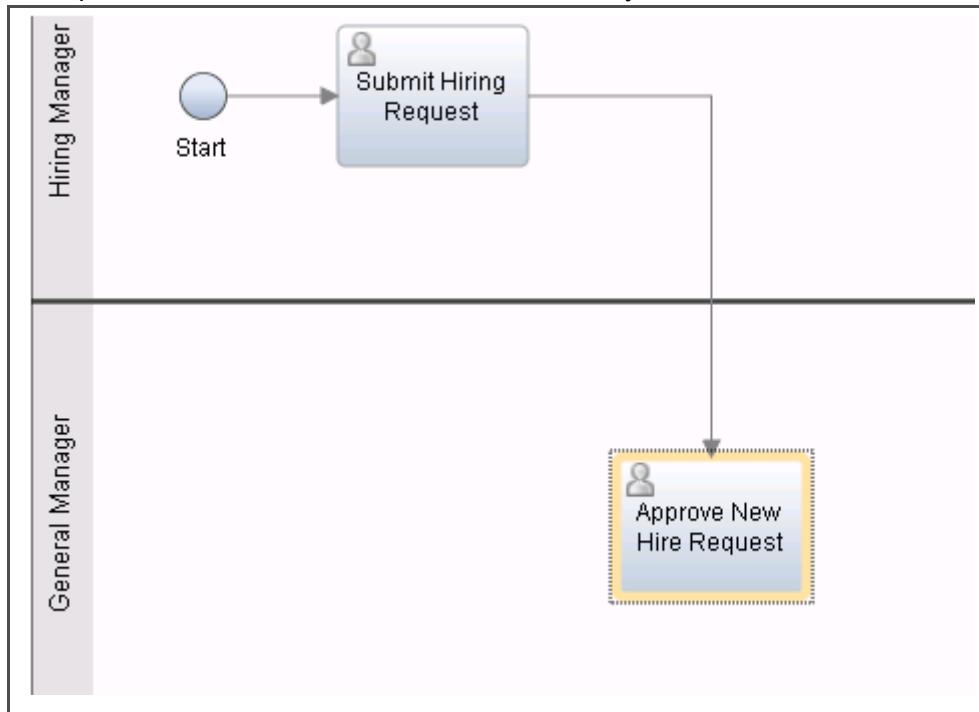


- c. Drag this end of the sequence flow to an anchor point on the **Approve New Hire Request** activity in the General Manager lane.

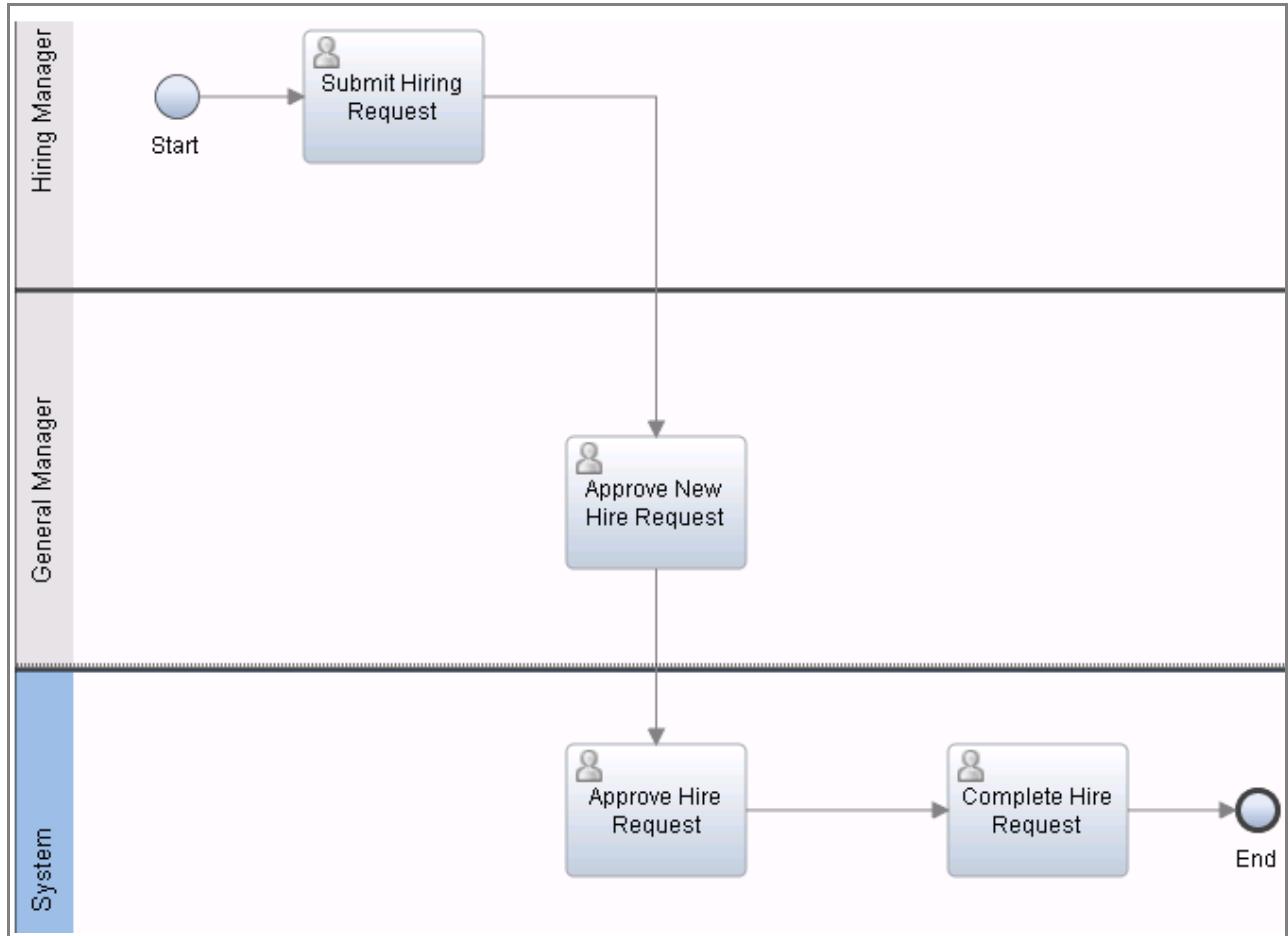


- d. Drag the **End** event and place it next to the **Complete Hire Request** activity.

- e. Select the **Approve New Hire Request** activity and drag it near the Submit Hiring Request activity to align the activities with straight flow lines in the process. You might be required to move the end event out of the way.



- ___ f. Connect the sequence flow between the **Approve New Hire Request** activity and **Approve Hire Request** activity in the System lane.
- ___ g. Connect the sequence flow between the **Approve Hire Request** activity and the **Complete Hire Request** activity.
- ___ h. Connect the sequence flow between the **Complete Hire Request** activity and the **End** event.



- ___ i. Save your process. The first process is complete for this exercise. Next, you create another process, a nested process.

Part 4: Create a nested process

Determine whether any additional work is required for each activity and if so, what the work involves.

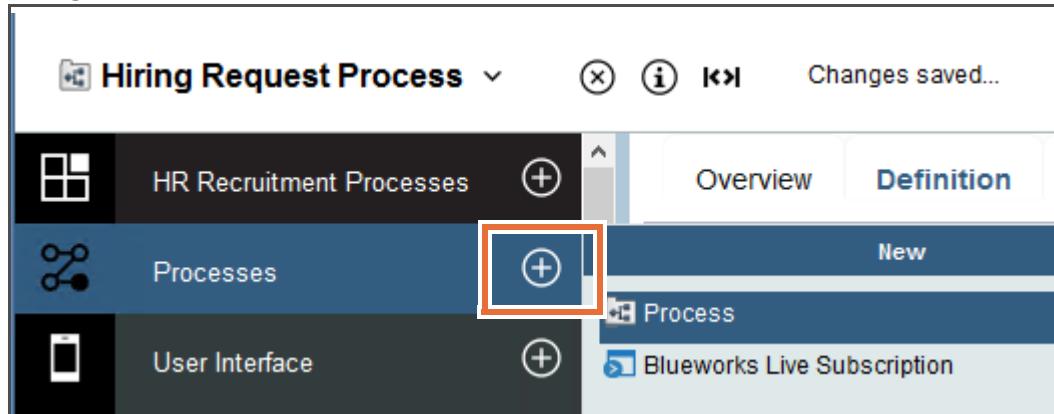
- **Submit Hiring Request:** Generally no rework is needed.
- **Approve New Hire Request:** Extra work that is required for the General Manager to record a rejection reason.
- **Approve Hire Request:** Extra team work that the HR Administrator and the Hiring Manager require.
- **Complete Hire Request:** Generally no rework is needed.

Because more than one team is involved in the **Approve Hire Request** activity, it is no longer a “task” but a nested process. The nested process has lanes, such as System, HR Administrator, and Hiring Manager. Activities include Check Hire Request, Override Hire Request, and Negotiate Hire Request.

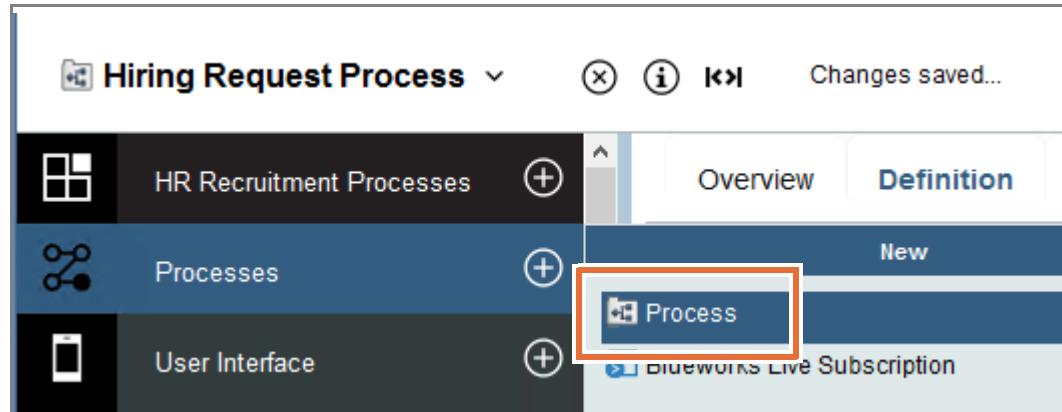
You must determine whether the nested process is a linked process or a subprocess as they are created differently. If you think your nested process might ever be reused, create a linked process. Otherwise, create a subprocess. Using a subprocess has benefits because processing and development time can be faster, but it cannot be reused.

Approve Hire Request is a nested process and might be reused in other HR processes, so you create a linked process for this scenario.

- 1. Create a process that is called **Approve Hire Request**.
- a. From the process library, hover over **Processes** category and then click the (+) plus sign.



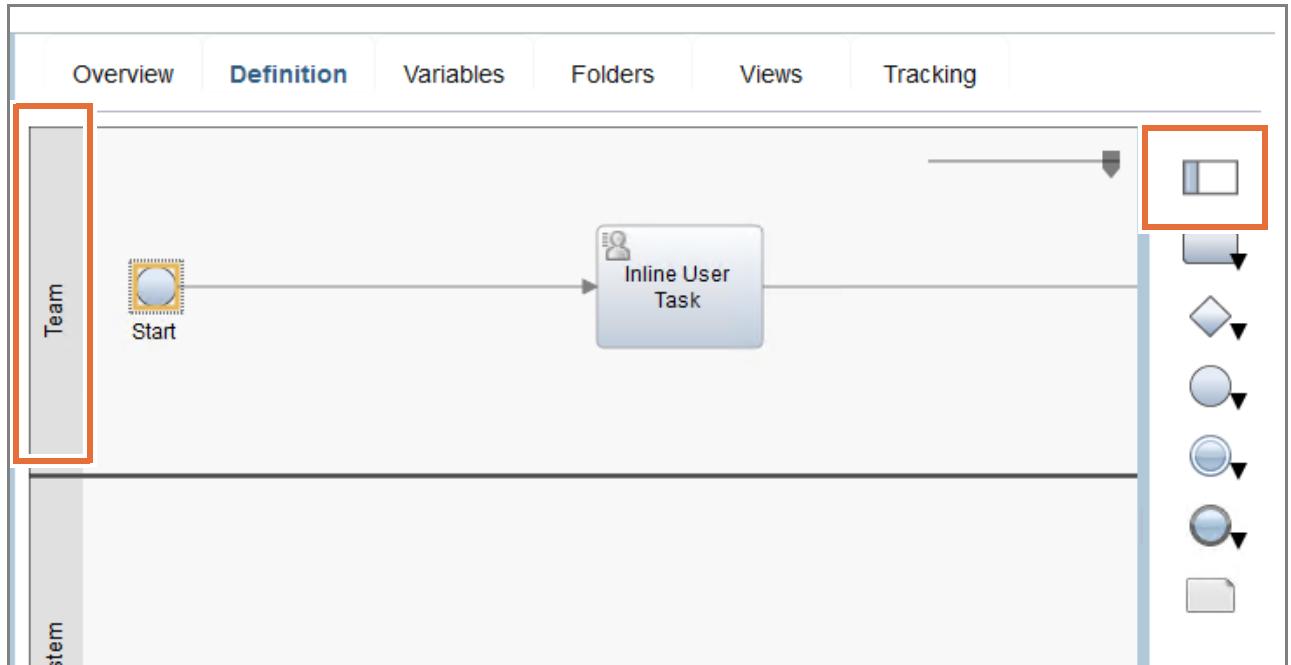
- __ b. From the **New** menu, click **Process**.



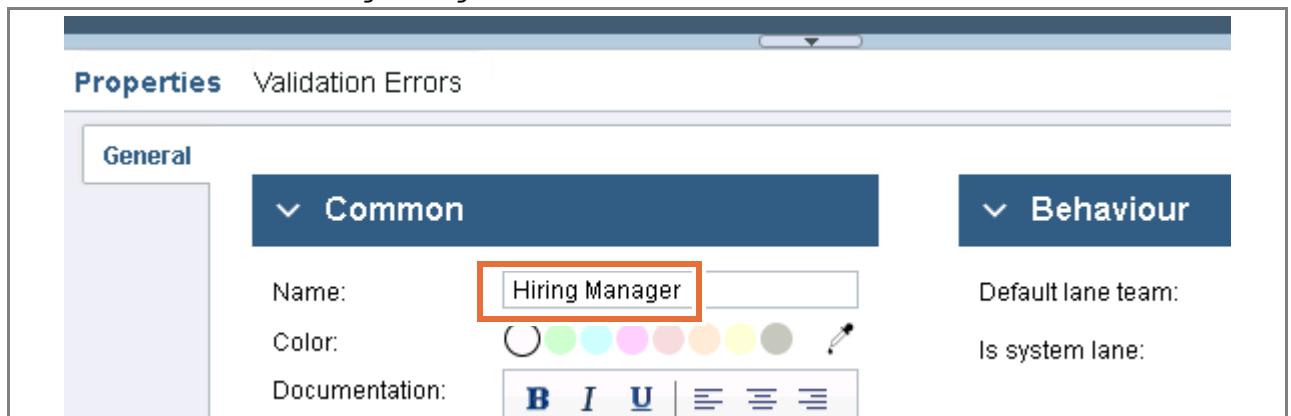
- __ c. Enter **Approve Hire Request** for the **Name**.
__ d. Click **Finish**.

A screenshot of a "New Process" dialog box. The title is "New Process". The text inside says: "A process captures a set of activities and the data and content to support the activities. These activities can be part of a structured flow, ad-hoc activities that are not part of a structured flow, or a combination of the two." There is a blue icon of a process flow on the right. Below the text, there is a form field with "Name : Approve Hire Request" and a red box highlighting the input field. At the bottom, there are two buttons: "FINISH" and "CANCEL", with "FINISH" also having a red box around it.

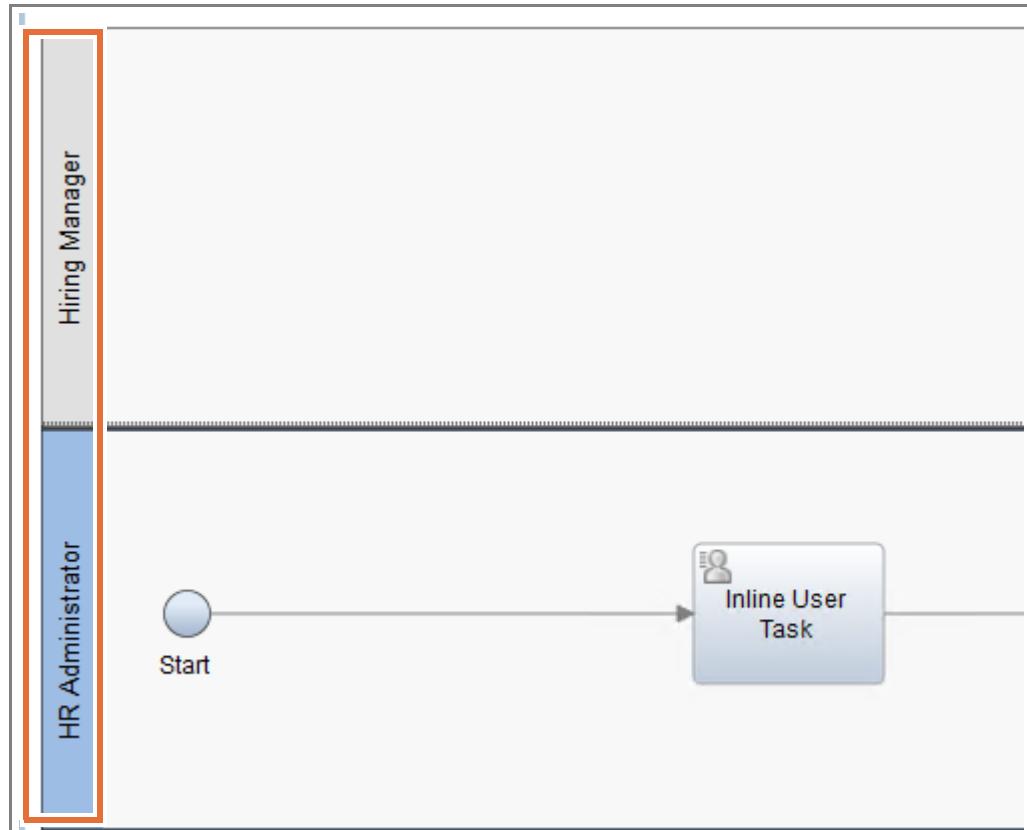
- __ 2. Create the System, HR Administrator, and Hiring Manager lanes.
- __ a. Click the **Lane** icon from the palette and drag one lane from the palette to the canvas above the default Team lane.



- __ b. In the **Properties > General > Common** section for the new Team1 lane, change the name to: **Hiring Manager**



- c. Rename the second lane to: HR Administrator

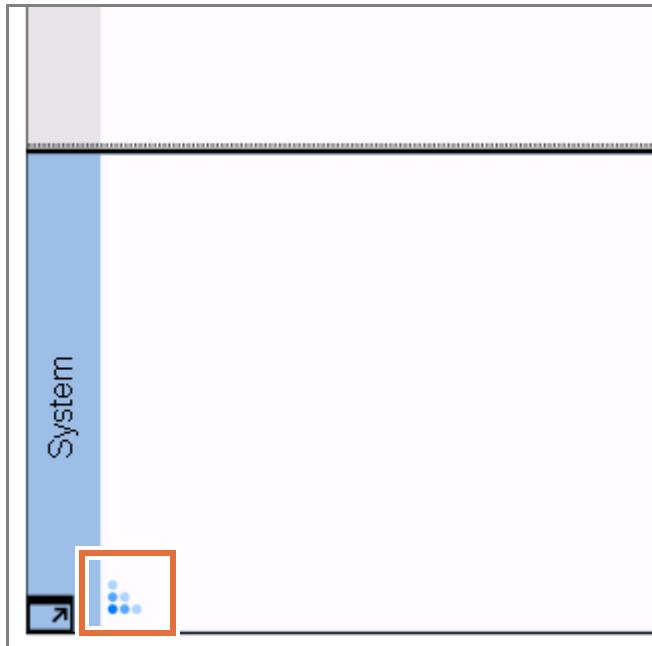


Hint

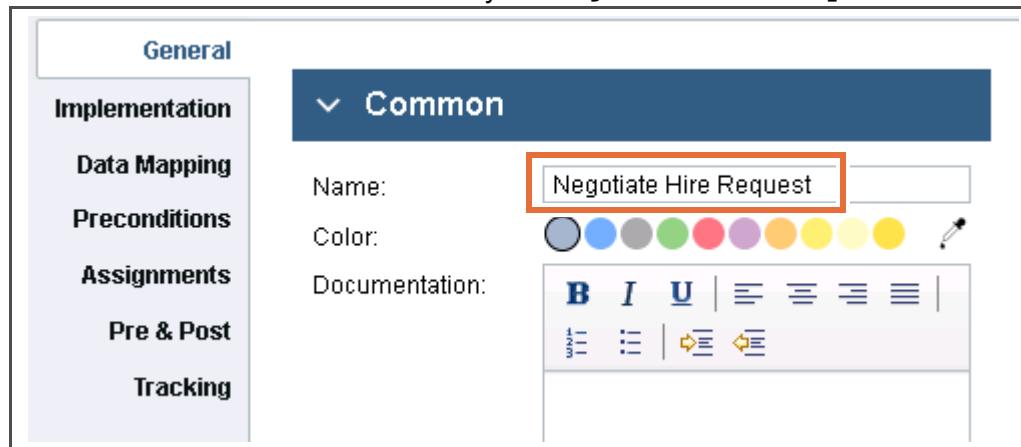
You can change the name of a lane under the **Properties > General > Common > Name** section.
You can also double-click the lane name to rename it.

**Note**

You can resize the lanes by clicking and dragging the control point at the left border of the lane. Hover your mouse at the left border of the lane to see this control point.

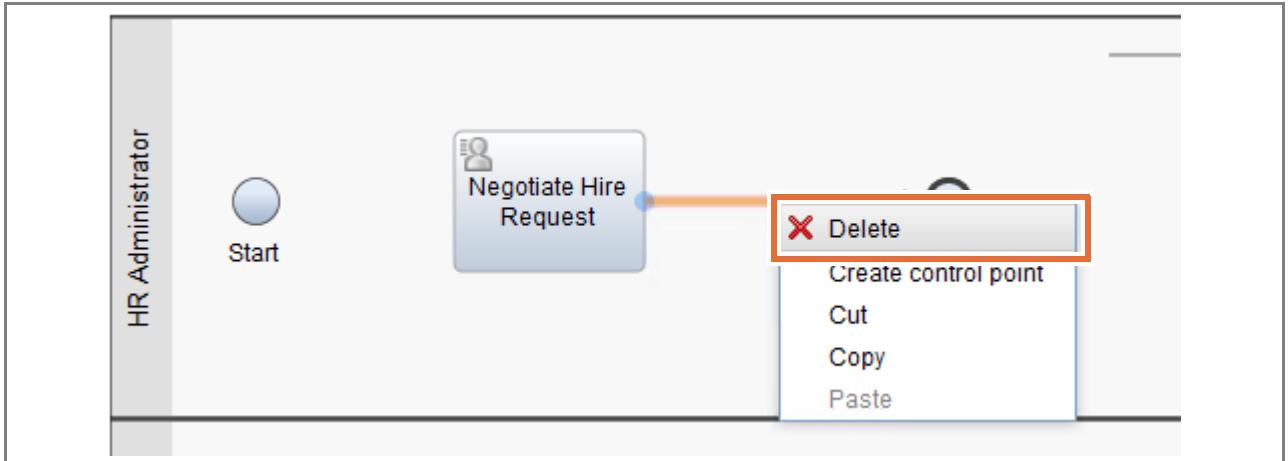


- ___ 3. Create the Check Hire Request, Override Hire Request, and Negotiate Hire Request activities.
- ___ a. Rename the Inline User Task activity to: Negotiate Hire Request

**Reminder**

You can double-click the activity name to rename it.

- ___ b. Delete all the flows on the canvas.

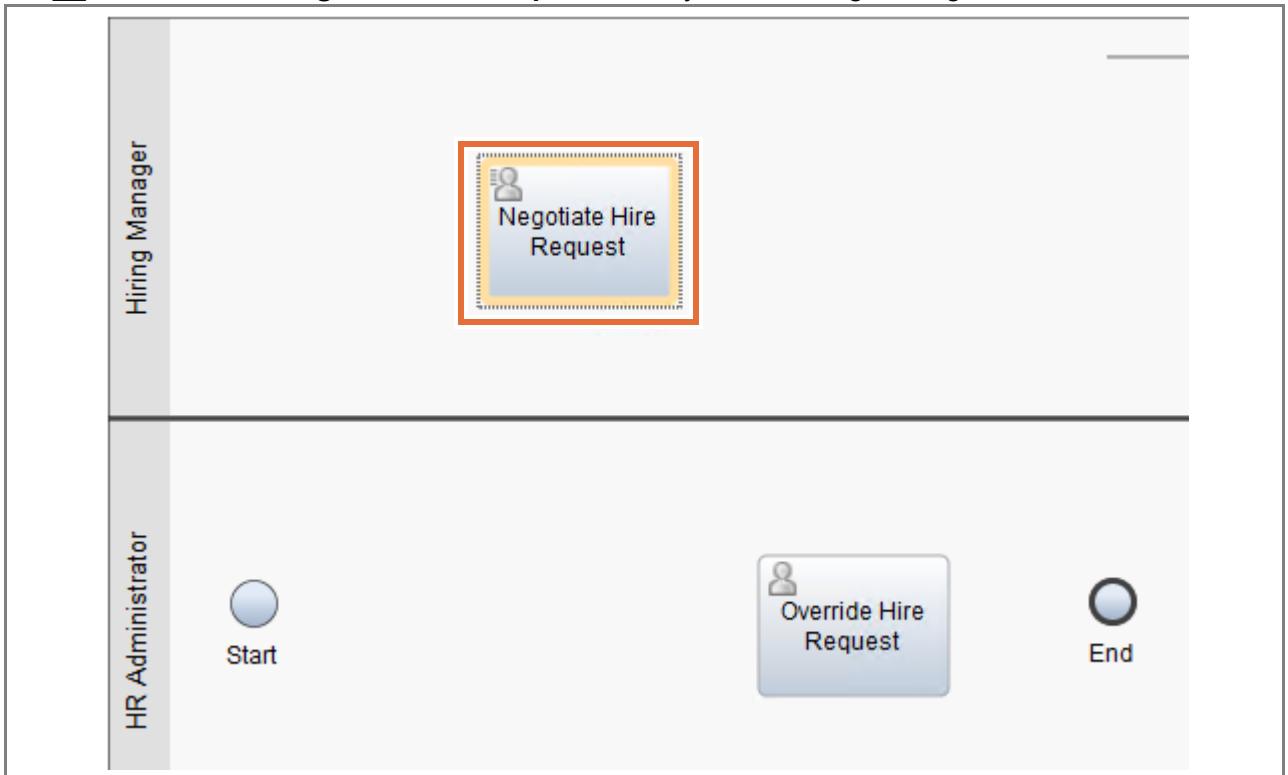


- ___ c. Expand the activity icon in the palette and drag a **User Task** to the right of **Negotiate Hire Request** on the canvas.



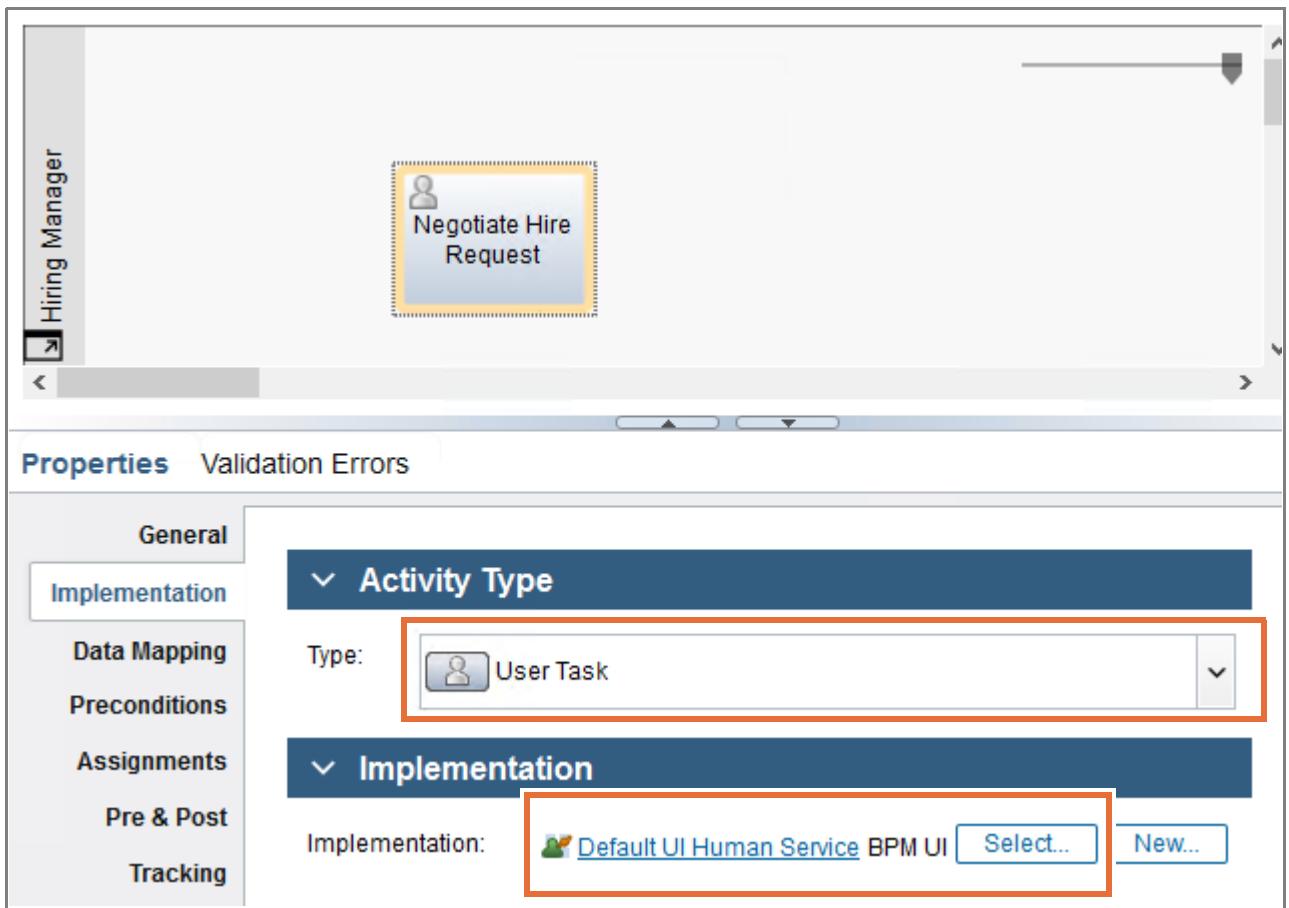
- ___ d. Name the activity **Override Hire Request**.

- ___ e. Move the **Negotiate Hire Request** activity to the Hiring Manager lane.



- ___ f. In the **Properties > Implementation > Activity Type**, change the activity to a **User Task**.

- __ g. In the Implementation section, change the implementation to a **Default UI Human Service**.

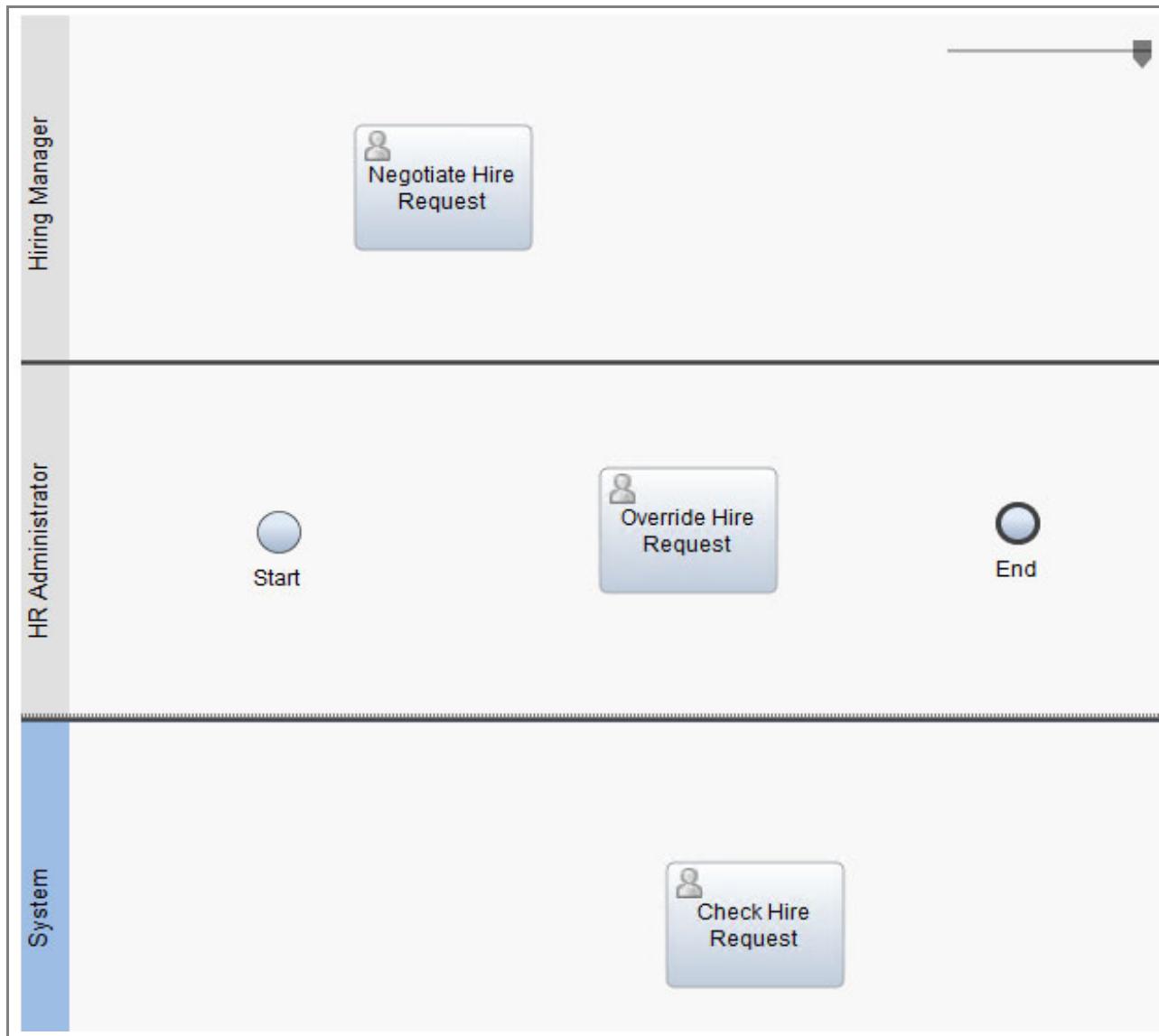


- __ h. Drag another User Task to the System lane and name the system task: Check Hire Request.

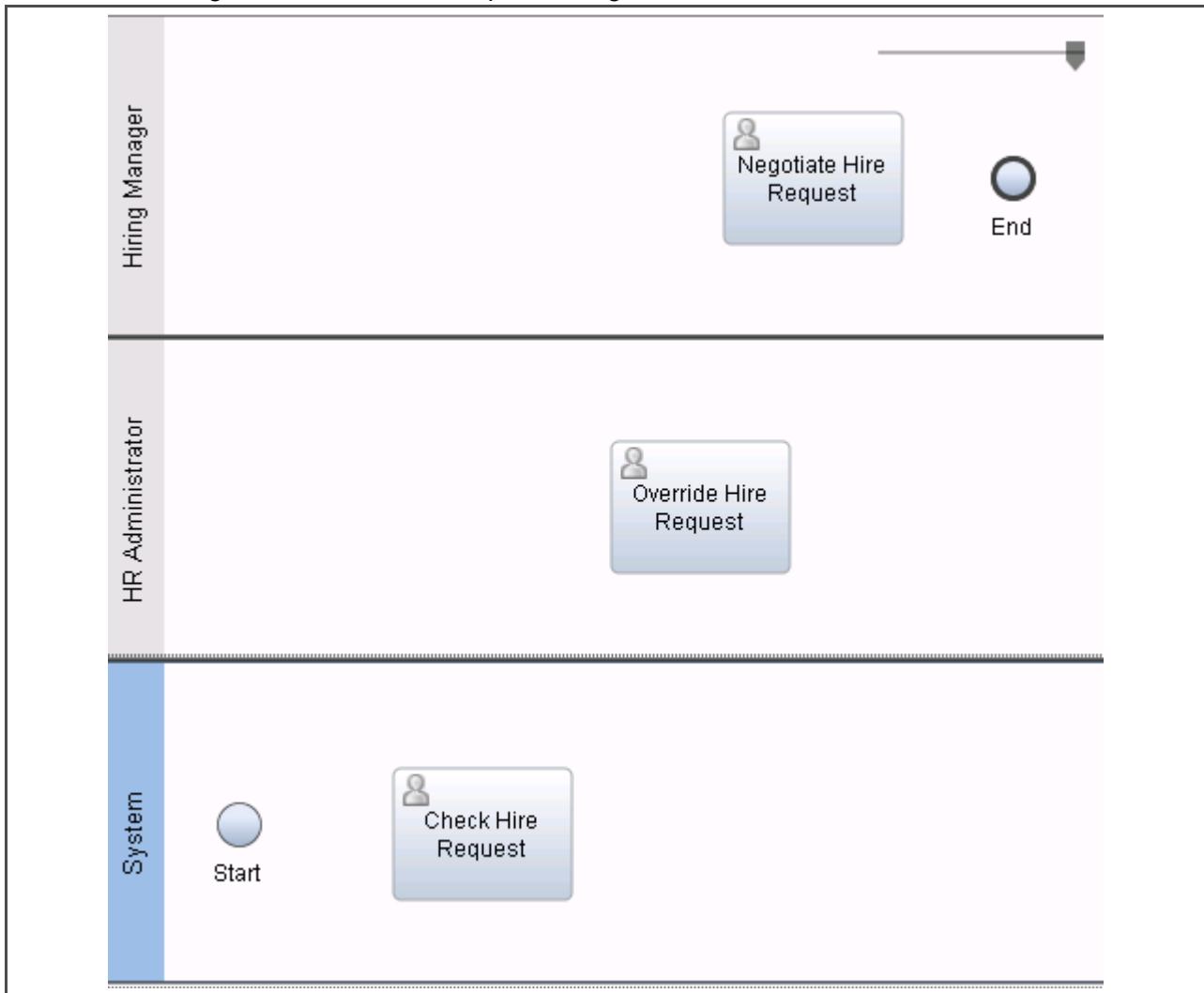
- __ i. The locations of your activities should match the nested diagram table.

Table 1. Nested diagram

Lane	Activities
Hiring Manager	Negotiate Hire Request
HR Administrator	Start event, Override Hire Request, End event
System	Check Hire Request



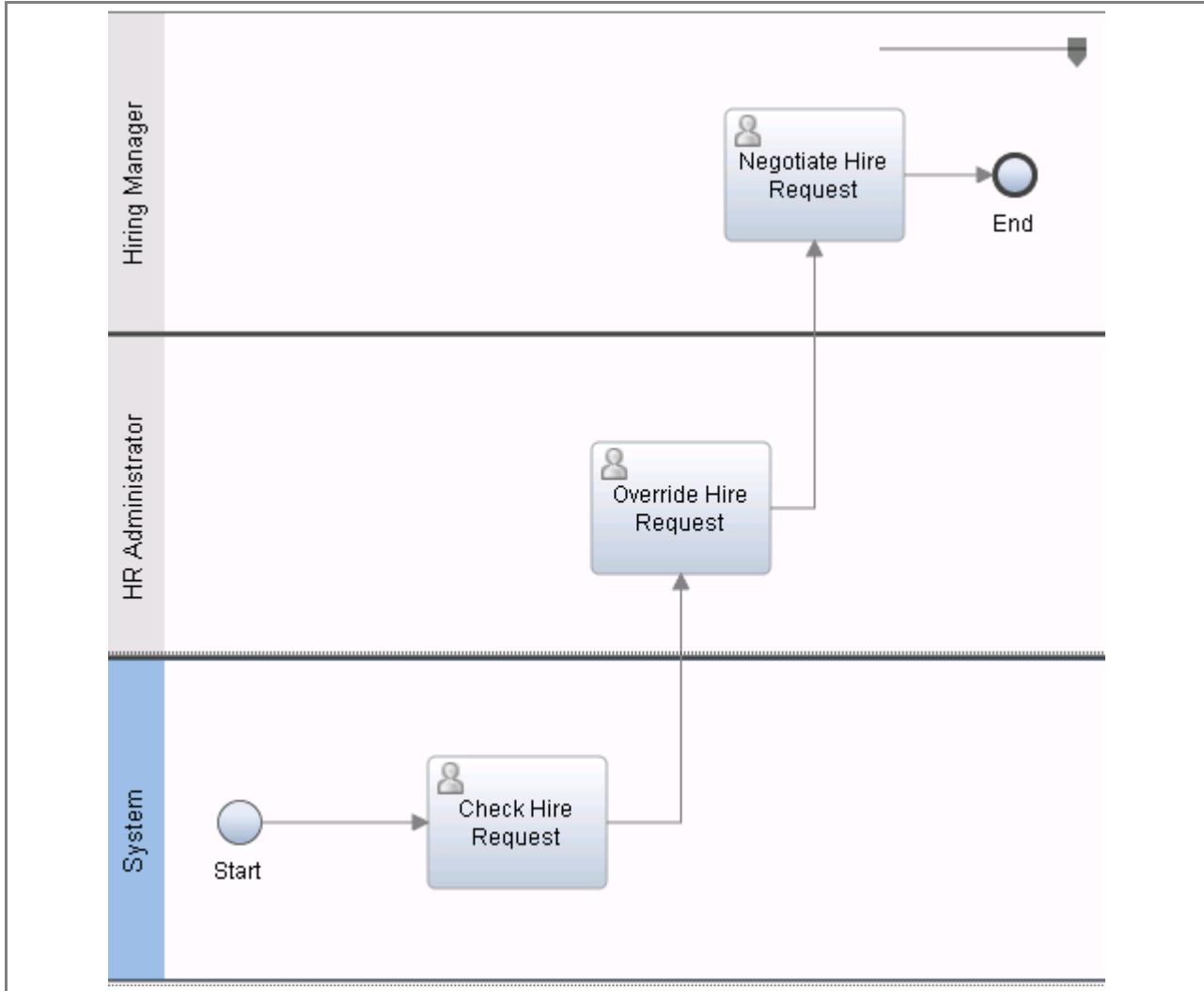
- j. Move your **Start** event before the first activity, **Check Hire Request** in the System lane, and the **End** event after the **Negotiate Hire Request** activity in the Hiring Manager lane. Arrange the activities in an upward diagonal line.



Note

Although the layout breaks the preferred left-to-right, top-to-bottom approach because the process starts in the system lane, most developers keep the system lane as the bottom lane. Communicating a clear, concise model is the most important goal, and even though this model breaks the preferred practice, it meets the readability goal.

- k. Create a sequence flow from the **Start** event to **Check Hire Request**.
- l. Create a sequence flow from **Check Hire Request** to **Override Hire Request**.
- m. Create a sequence flow from **Override Hire Request** to **Negotiate Hire Request**.
- n. Finally, create a sequence flow from **Negotiate Hire Request** to the **End** event.



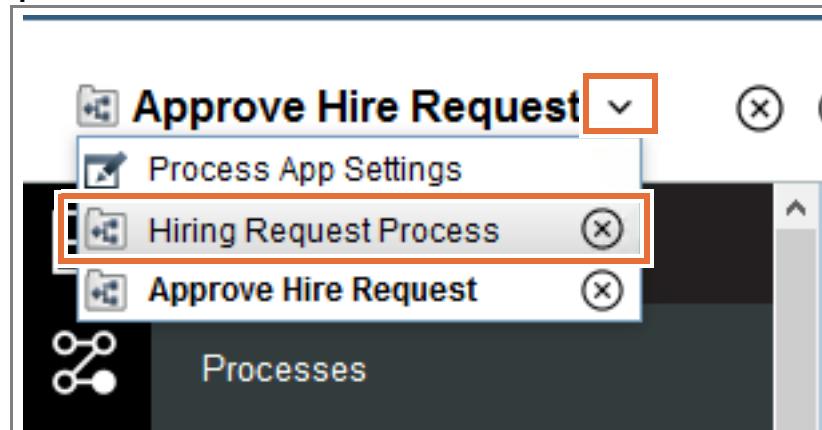
Reminder

You can align the activities by using the arrows keys on the keyboard or by selecting and dragging the activities in the optimal positions so that all the sequence flows as straight lines. You can fine-tune your movements by holding Shift when you press the arrow key.

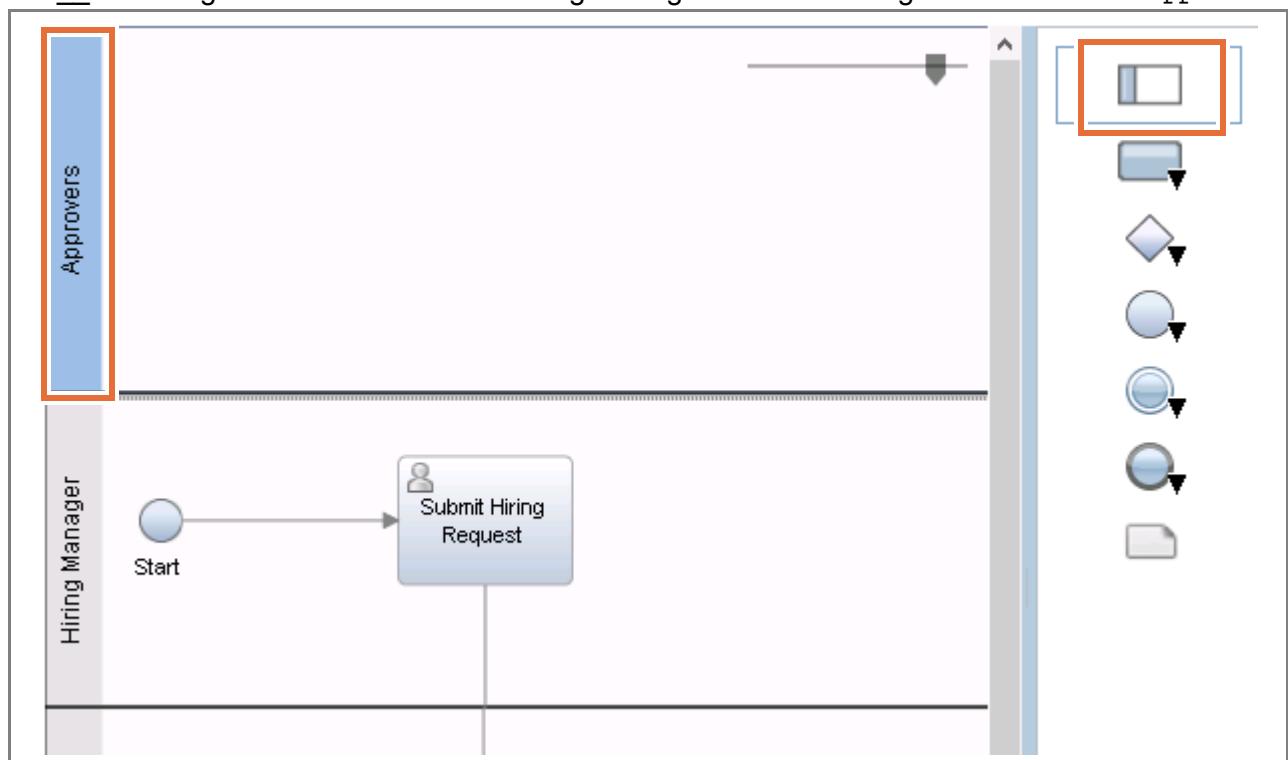
- o. Save your process application.

Part 5: Attach the linked process

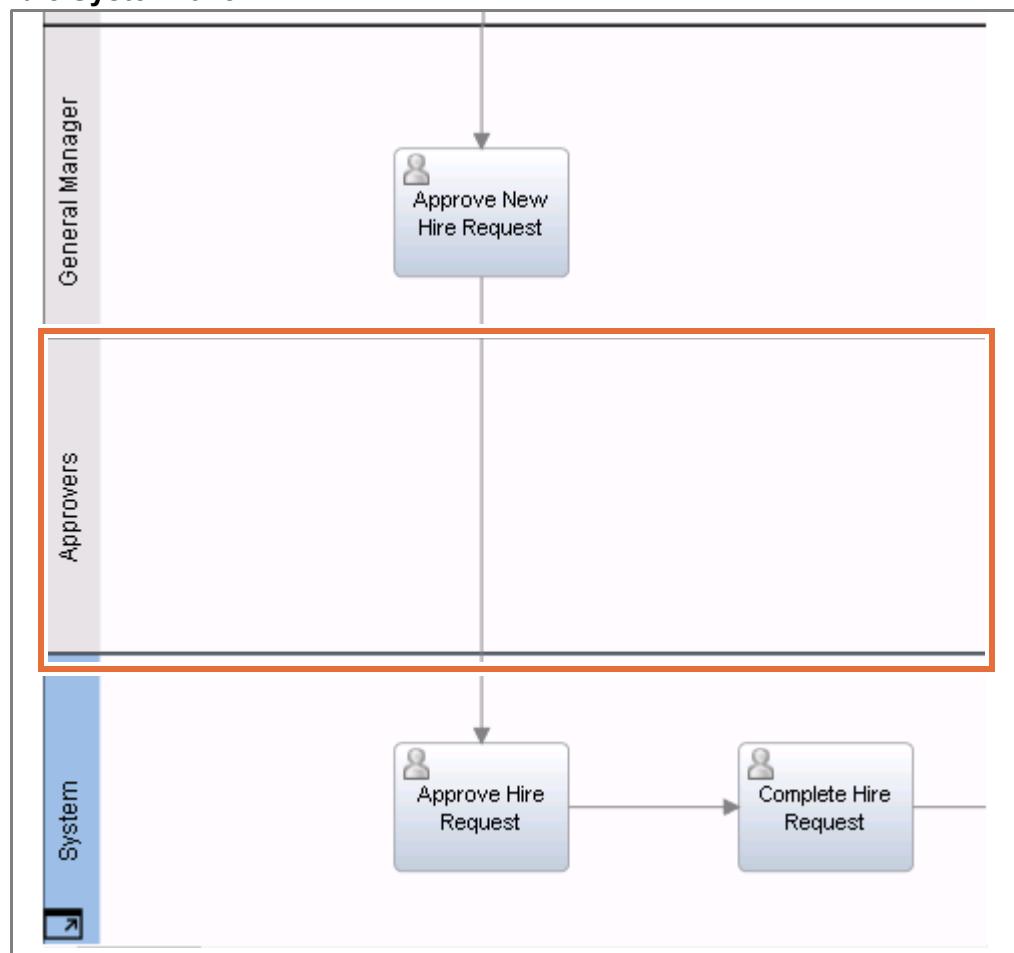
- 1. Return to the Hiring Request Process by opening the **History** menu at the top and selecting **Hiring Request Process**.



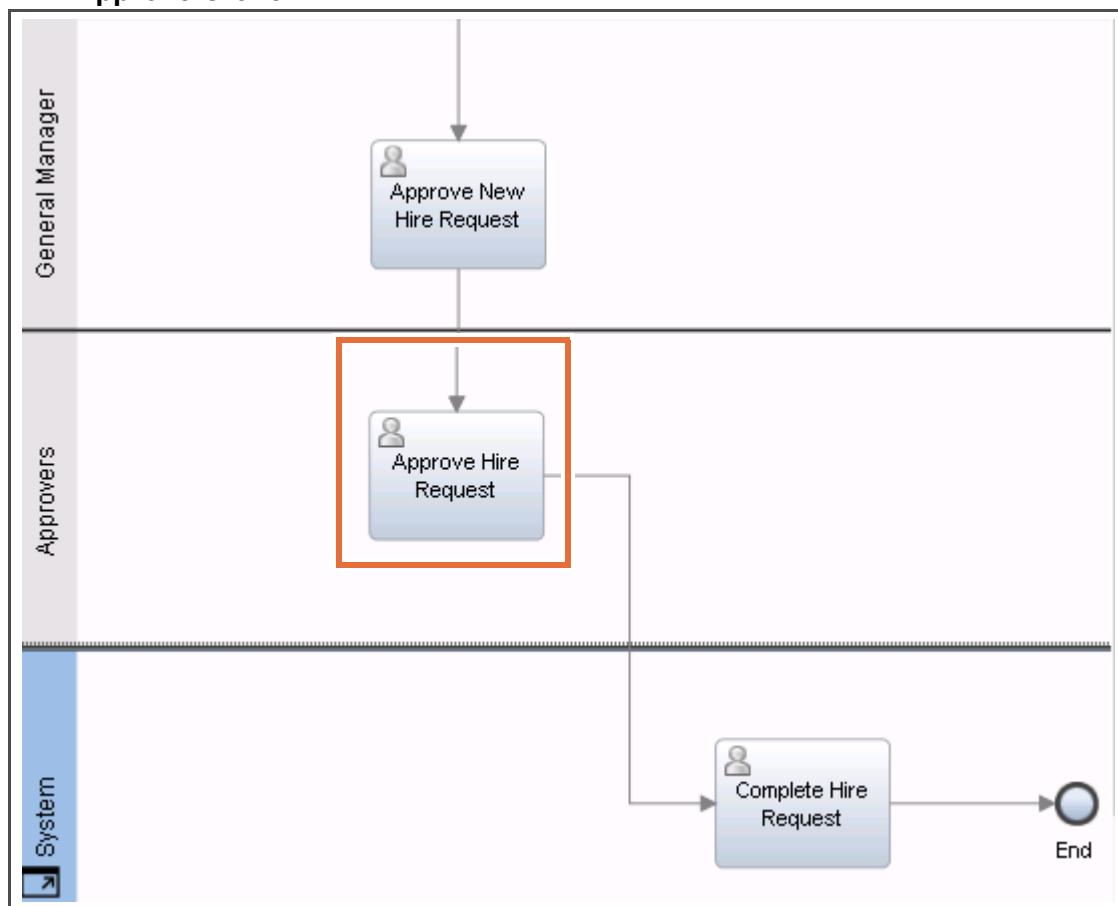
- 2. Change the implementation of the existing **Approve Hire Request** activity to a nested process.
 - a. Drag a new lane above the Hiring Manager lane in the diagram and name it: **Approvers**



- b. Continue to right-click the **Approvers** lane and click **Move Lane Down** until it is above the **System** lane.



- ___ c. Drag the existing **Approve Hire Request** activity that is in the System lane into the **Approvers** lane.



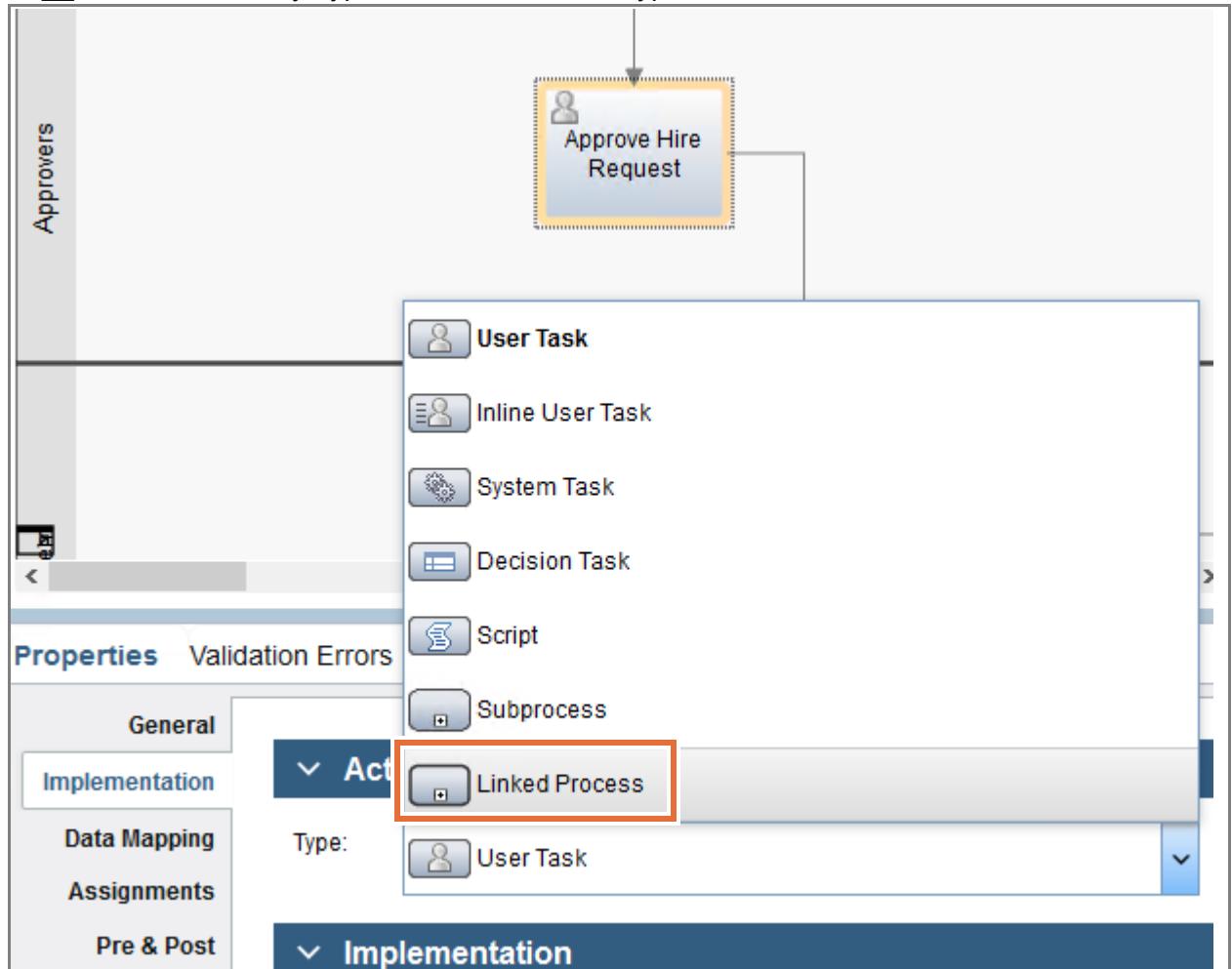
Reminder

Realign the activities in their respective lanes so that the sequence flow lines are always straight.

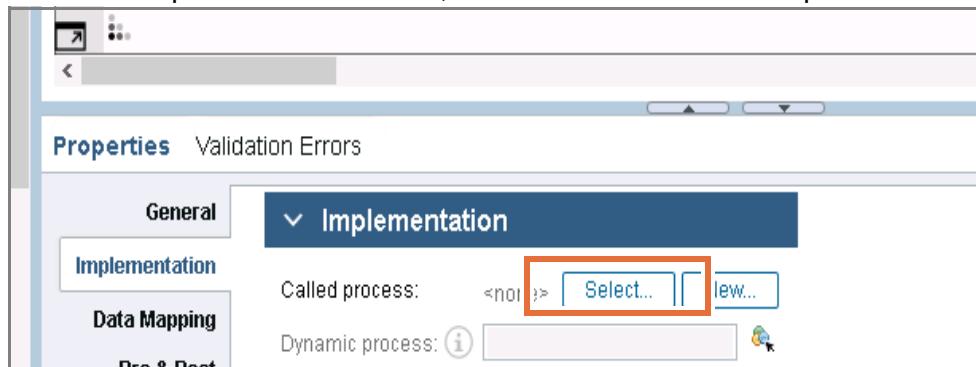
- ___ d. When the **Approve Hire Request** activity is selected, open the **Properties > Implementation** menu.

Properties Validation Errors	
General	Activity Type
Implementation	Type: User Task
Data Mapping	
Assignments	
Pre & Post	
Tracking	Implementation

- ___ e. In the Activity Type section, select the Type as **Linked Process**.



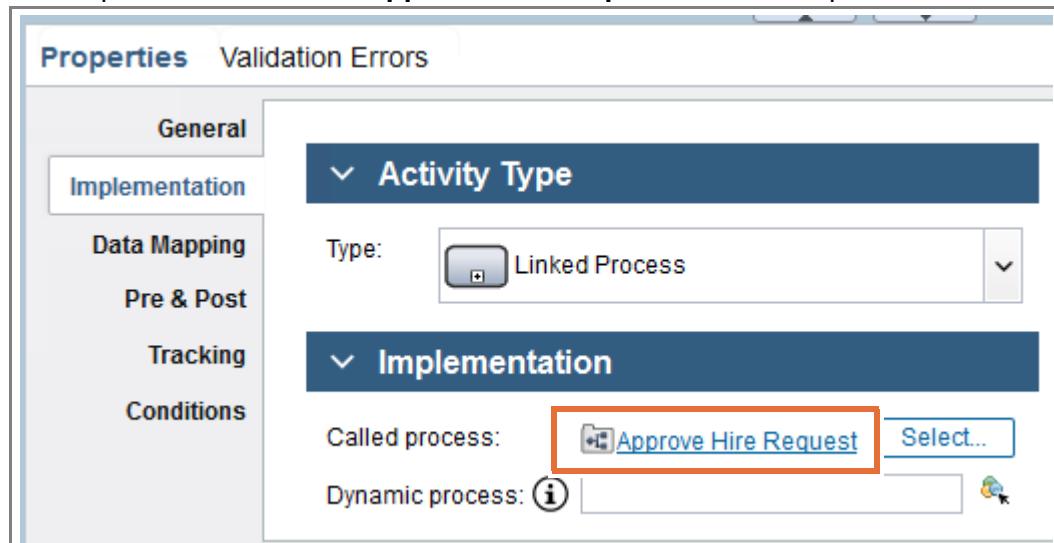
- ___ f. Under the Implementation section, click **Select** next to Called process.



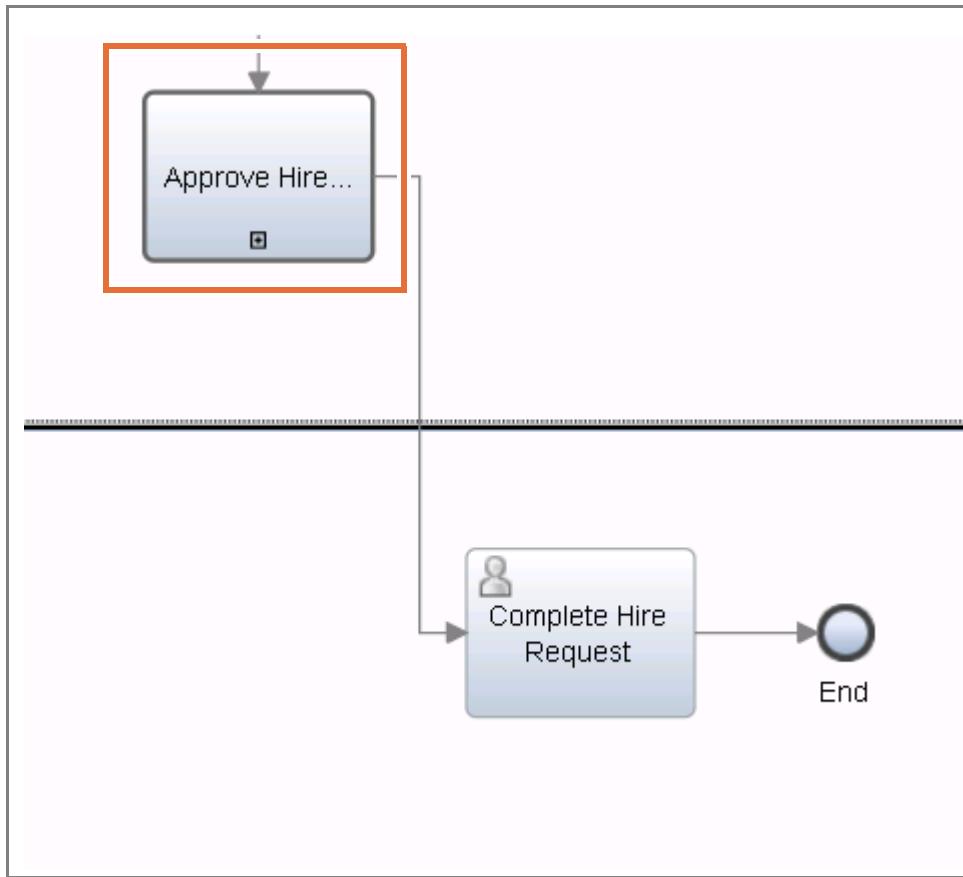
- __ g. Select **Approve Hire Request** from the Process menu.



- __ h. The process now shows **Approve Hire Request** as a linked process.



- __ i. The Approve Hire Request activity is now changed to a linked process. Save your process.



Important

You completed the following tasks:

- Created the foundation for a process by adding the appropriate lanes to the default pool
- Modeled the expected process flow for the initial process model
- Decomposed business process workflow steps that are documented in the process discovery and analysis into process model tasks
- Created a subprocess or a linked process

In the next exercise, you learn how to create gateways and timer intermediate events in the process.

End of exercise

Exercise review and wrap-up

In this exercise, you enhanced a process in a process application in the Process Center. You added swimlanes to the default pool. Then, you defined and modeled teams in the business process definition. Next, you converted business process workflow steps that are documented in the process discovery and analysis into process model tasks. Finally, you created the expected process flow for the initial process model, and decomposed a business process to create a linked process.

Exercise 3. Playback 0: Controlling process flow

Estimated time

01:30

Overview

This exercise covers how to create gateways in a business process definition, and how to create timer intermediate events.

Objectives

After completing this exercise, you should be able to:

- Add gateways to a process
- Model the appropriate sequence flows for each gateway
- Add a timer intermediate event to a process based on business requirements
- Model an escalation path in a process with IBM Process Designer
- Document details for the implementation team

Introduction

The purpose of this exercise is to add all the gateways necessary to model the flow control for the process.

In this exercise, add a timer intermediate event that helps satisfy newly identified requirements for the Hiring Requisition process. This action encompasses process flow control and more activities that are based on conditions by non-human interactions.

Requirements

Successful completion of the previous exercise is required.

Exercise instructions

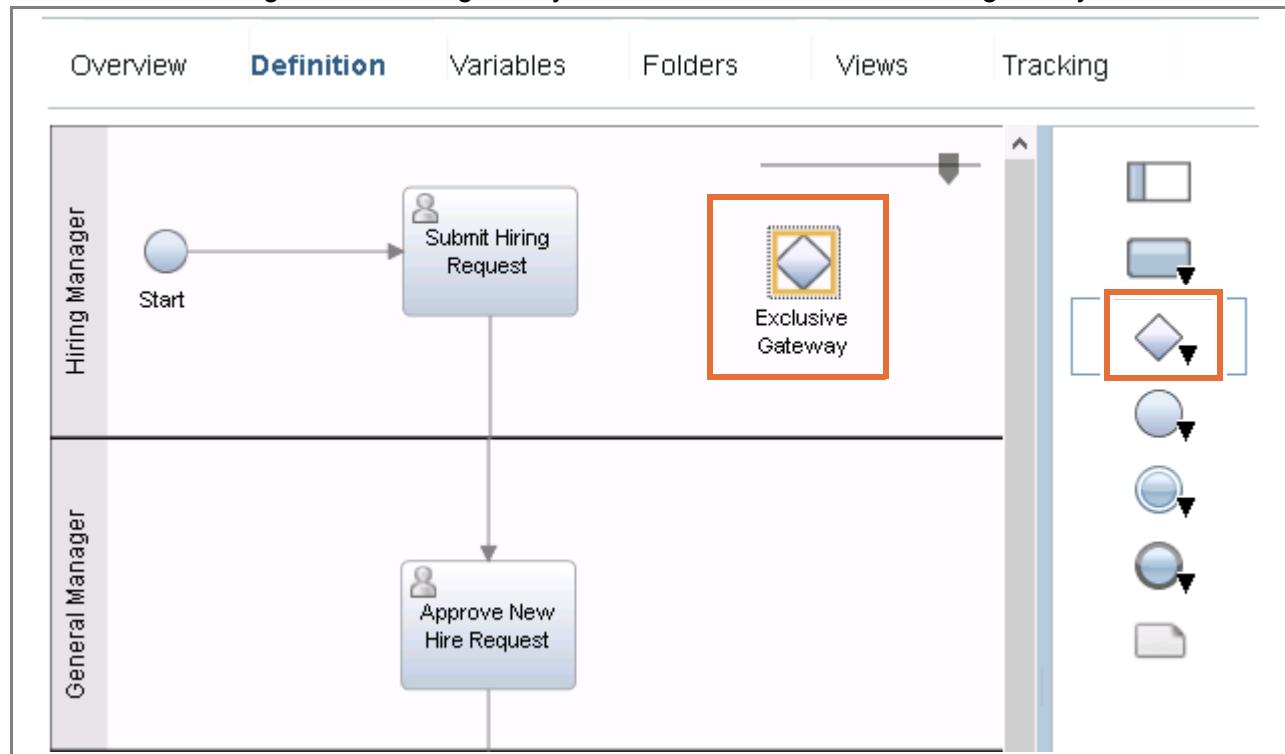
Part 1: Create gateways for parent process

Process flow controls are called gateways. A gateway is represented as a diamond, and can be thought of as a question at a point in the process flow.

Gateways are added to the parent process, the Hiring Request Process. In this part of the exercise, you create a gateway that is called Is Position New.

In the Hiring Request Process, you need a gateway to direct the process for the General Manager to review the salary after the Submit Hiring Request activity.

- 1. In the Hiring Request Process, drag an **Exclusive Gateway** from the palette onto the canvas to the right of the **Submit Hiring Request** activity. You do not need to open the menu to drag an exclusive gateway to the canvas, as the exclusive gateway is the default.

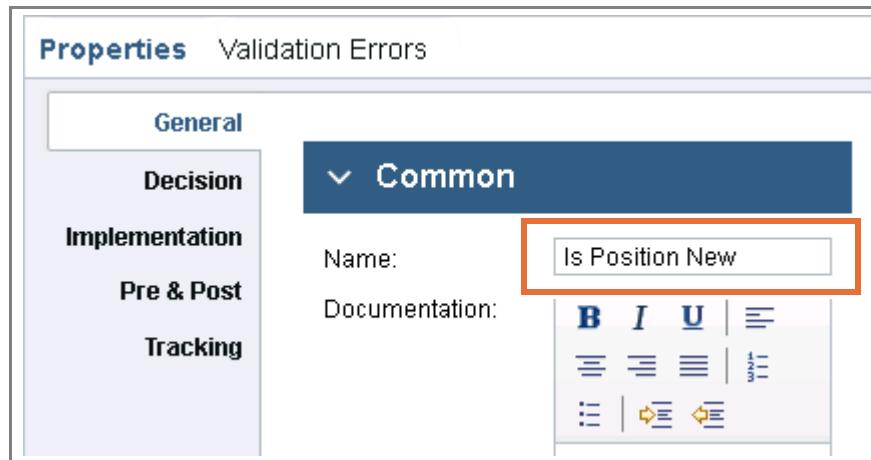


Note

You can reconnect the flows and align activities to make room for other components in the process. Make sure that the connecting nodes remain the same.

__ 2. Select the **Gateway** and set the properties.

__ a. Select the gateway and in the **Properties > General > Common** section, enter **Is Position New** as the **Name**.



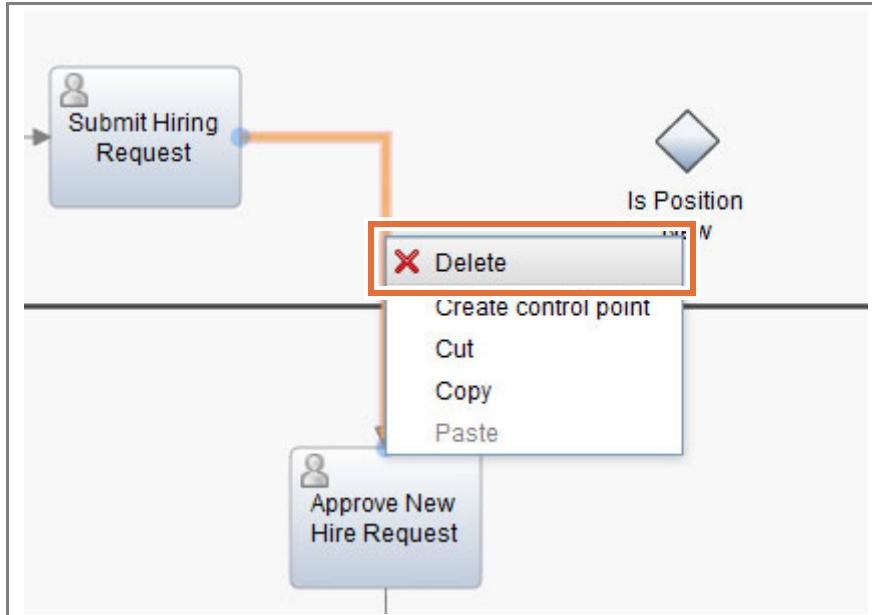
Reminder

Because all gateways imply a question, the question mark is unnecessary, and anyone who views the process automatically assumes the question mark.

__ b. In the **Properties > General > Behavior** section, verify that the Gateway type is: **Exclusive Gateway**.



- 3. Connect the gateway.
 - a. Select the sequence flow between **Submit Hiring Request** and **Approve New Hire Request**.
 - b. While selected, right-click the sequence flow and click **Delete**.



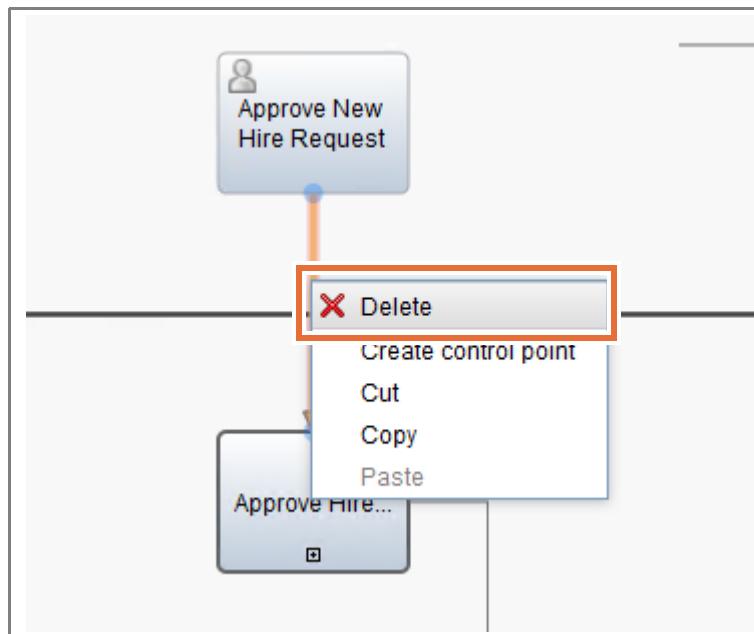
Hint

While selected, you can also delete the sequence flow by pressing the Delete key on the keyboard.

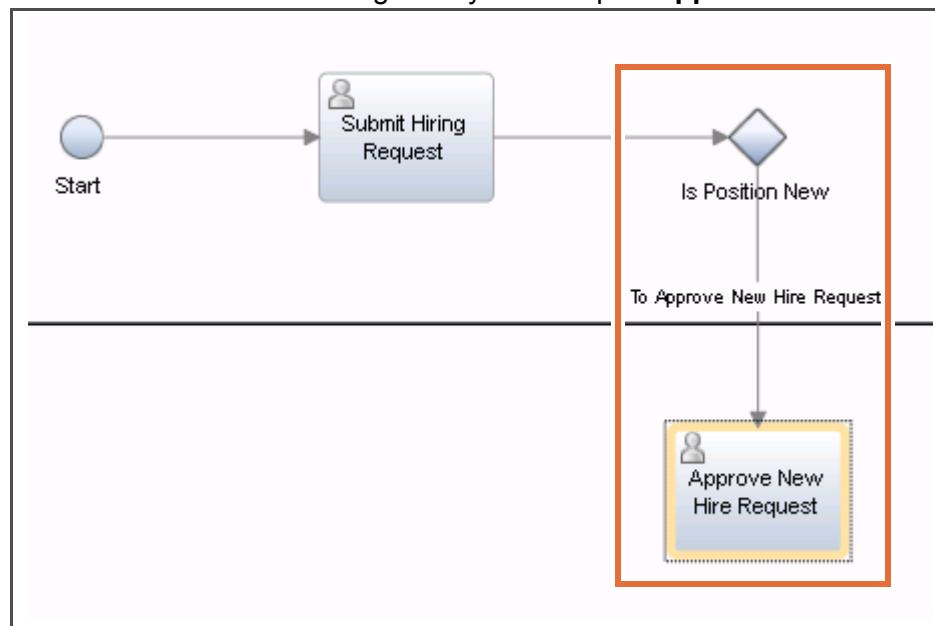
- c. Create a sequence flow between the **Submit Hiring Request** activity and the **Is Position New** gateway. Hover over the right anchor point on the activity until an arrow appears, then drag the arrow to the left anchor point on the gateway.



- __ d. Delete the sequence flow between **Approve New Hire Request** and **Approve Hire Request**.

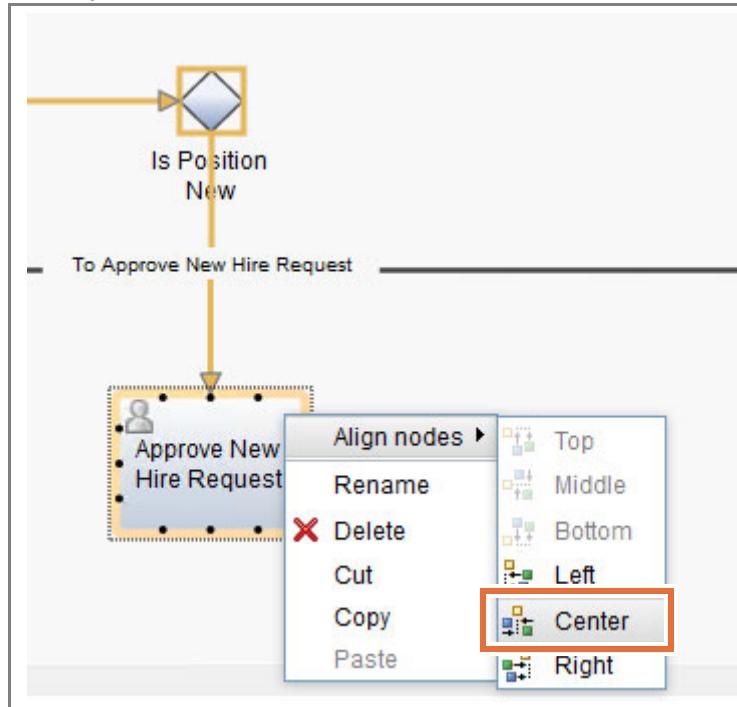


- __ e. Connect the **Is Position New** gateway to the top of **Approve New Hire Request**.

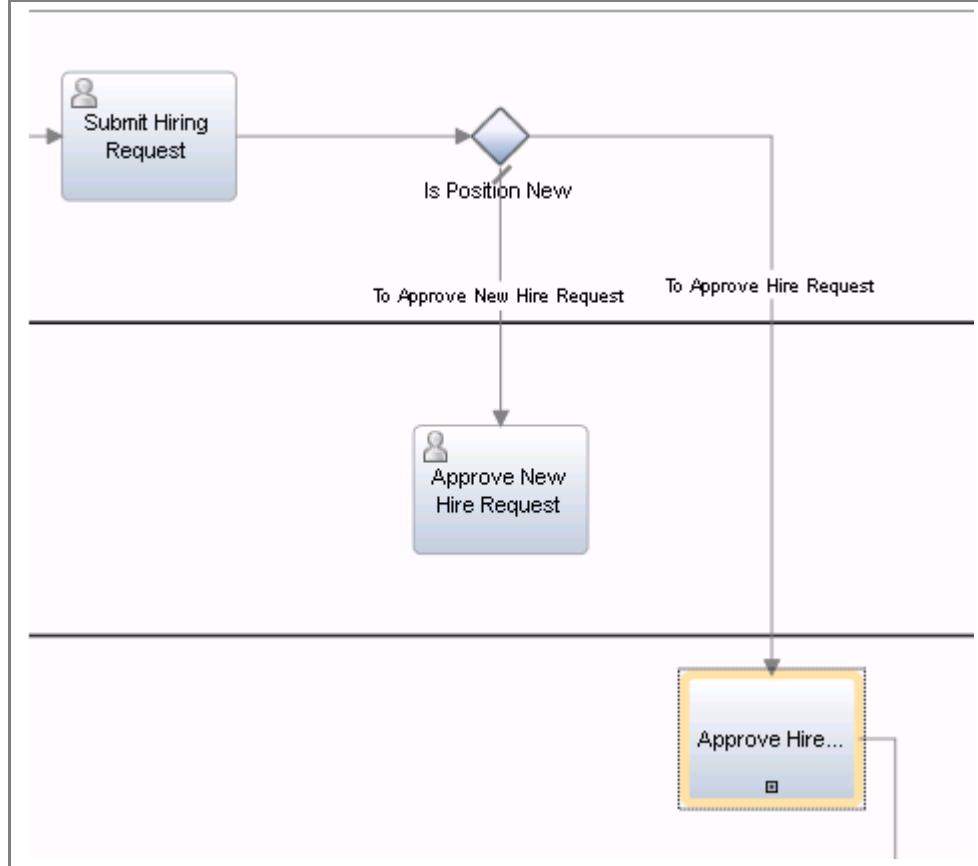


**Important**

To present the best process, always align the objects on your canvas. To align objects vertically, select the items and right-click one of the selected objects. Then, click **Align nodes > Center** to align the objects vertically.



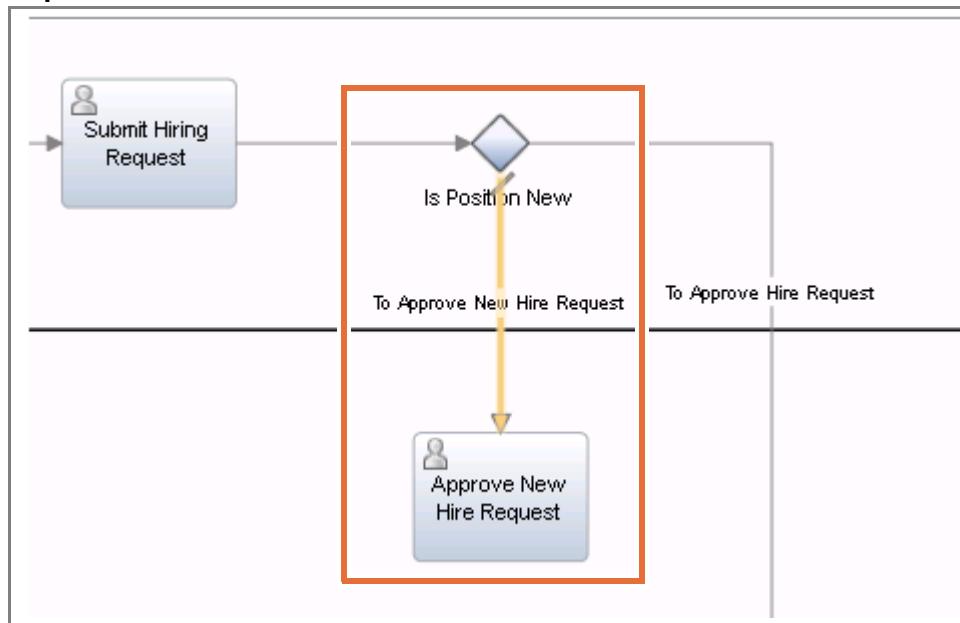
- f. Connect the **Is Position New** gateway to the top of **Approve Hire Request**.



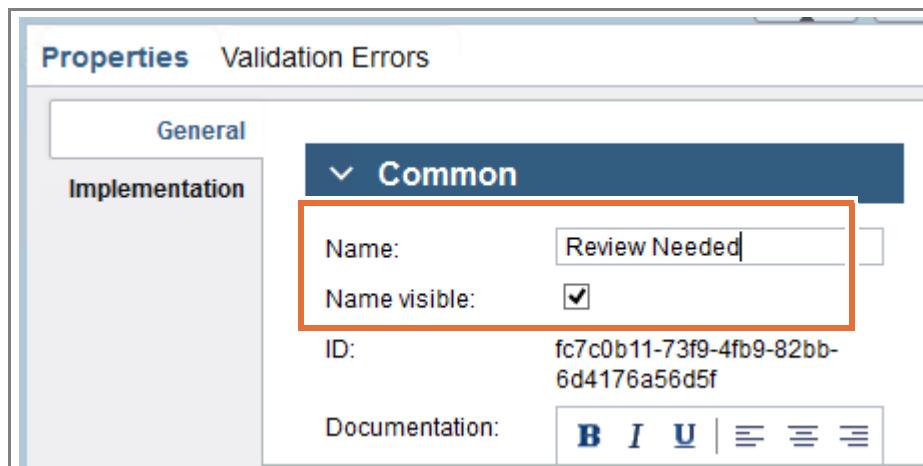
Note

Your sequence flows do not have to exactly match the exercise diagrams in this lab. The sequence flows change after implementation of the gateways. The flows become default or conditional (represented by a diagonal hash marker) according to the order you draw them, so your flows might vary from the diagrams in the labs. You learn how to set the default flow in a later exercise.

- 4. Label the flows.
- a. Select the flow between the **Is Position New** gateway and **Approve New Hire Request**.



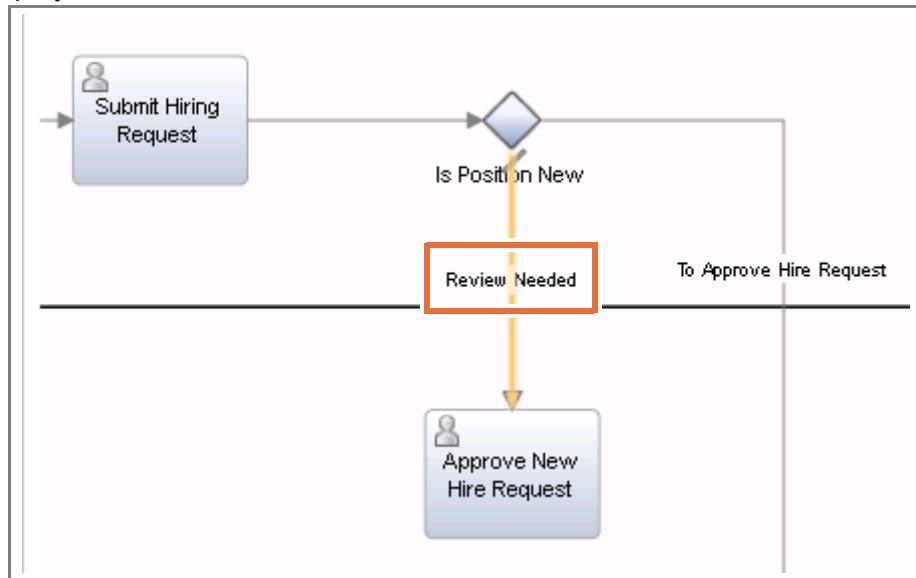
- b. In the **Properties > General > Common** section, enter **Review Needed** as the **Name** and select **Name visible**.



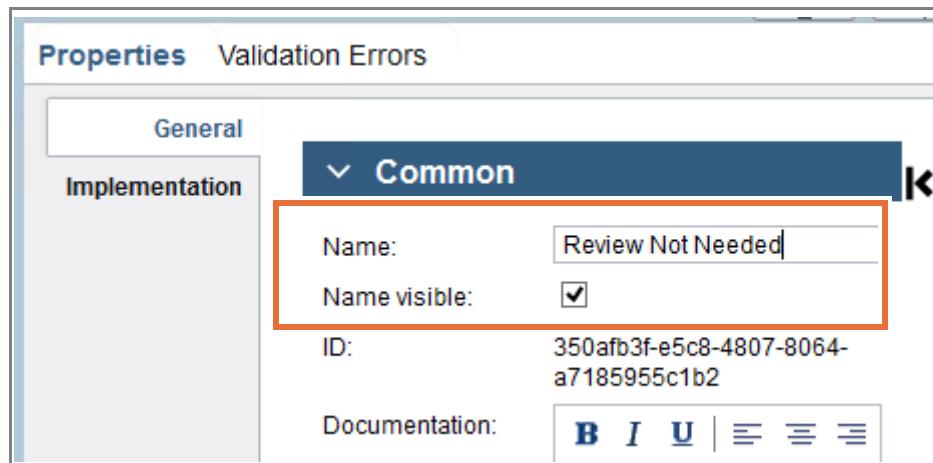


Information

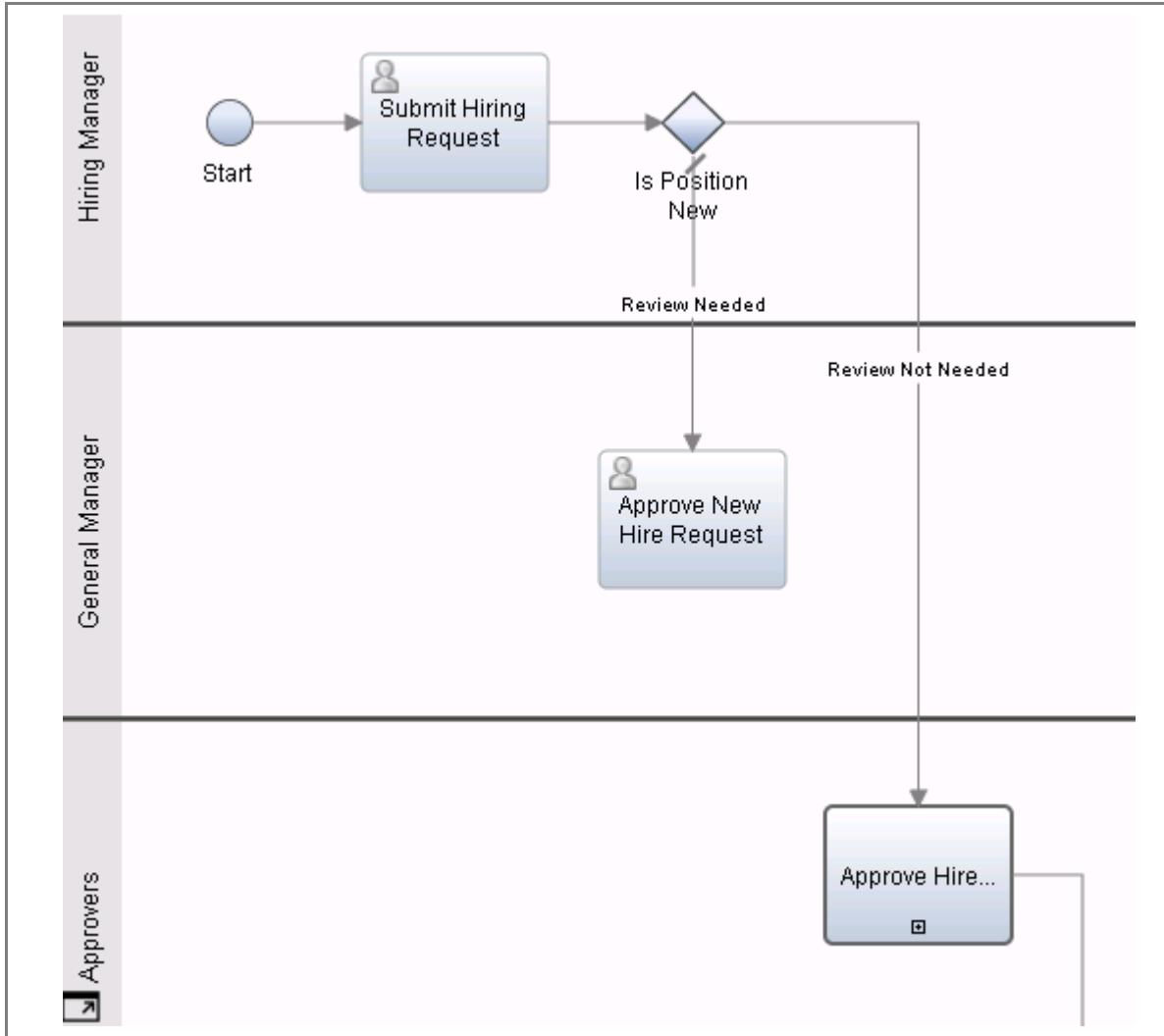
The flow displays the label.



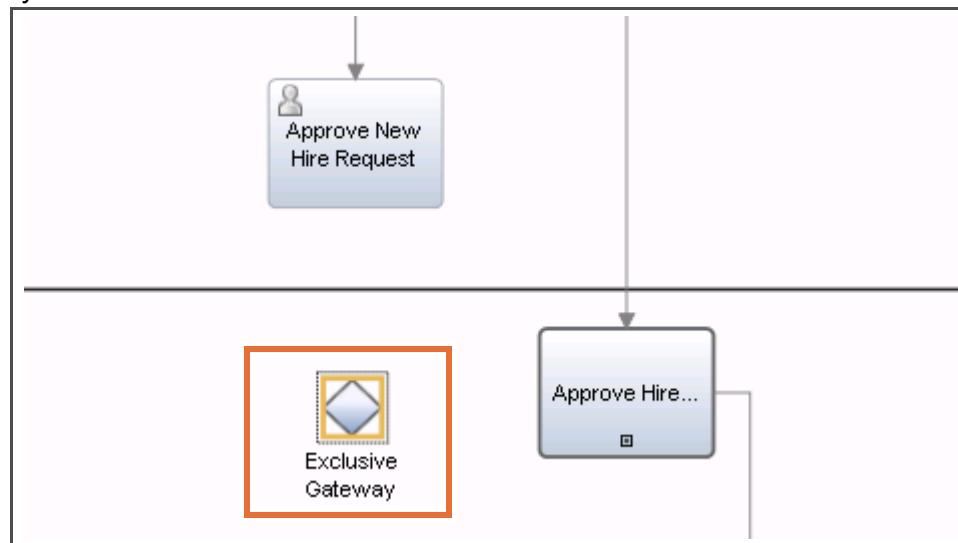
- ___ c. Select the flow between the **Is Position New** gateway and **Approve Hire Request**.
- ___ d. In the **Properties > General > Common** section, enter `Review Not Needed` as the Name and select **Name visible**.



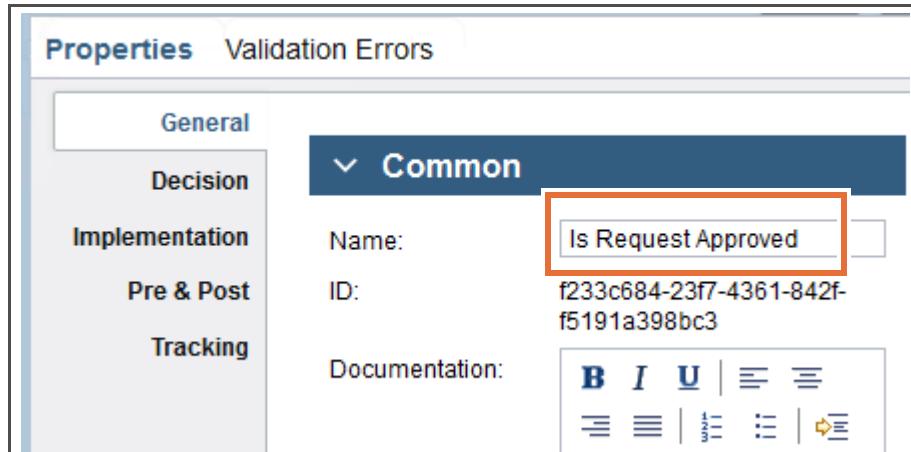
- e. Save your changes. The flow displays the label.



5. Drag a **Gateway** from the palette onto the canvas to the left of the **Approve Hire Request** activity.



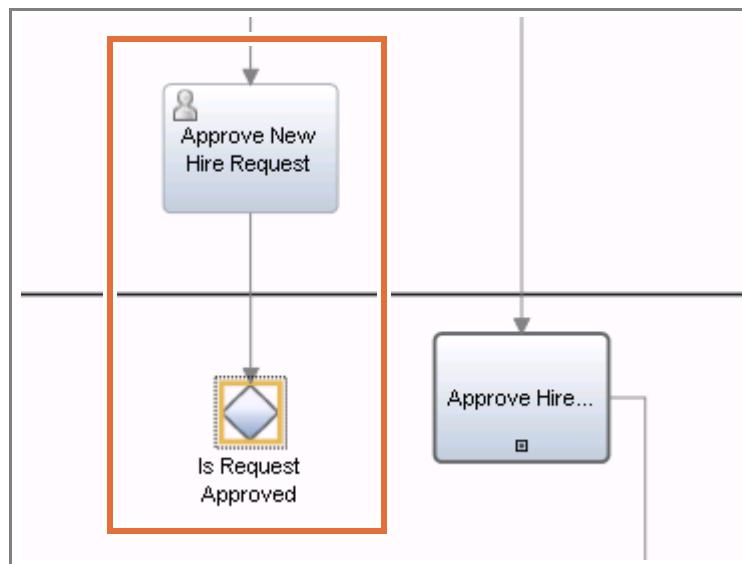
- ___ 6. Select the **Gateway** and set the properties.
 - ___ a. Select the **Exclusive Gateway** and in the **Properties > General > Common** section, enter **Is Request Approved** as the Name.



- ___ b. In the **Properties > General > Behavior** section, verify **Exclusive Gateway** as the Gateway type.



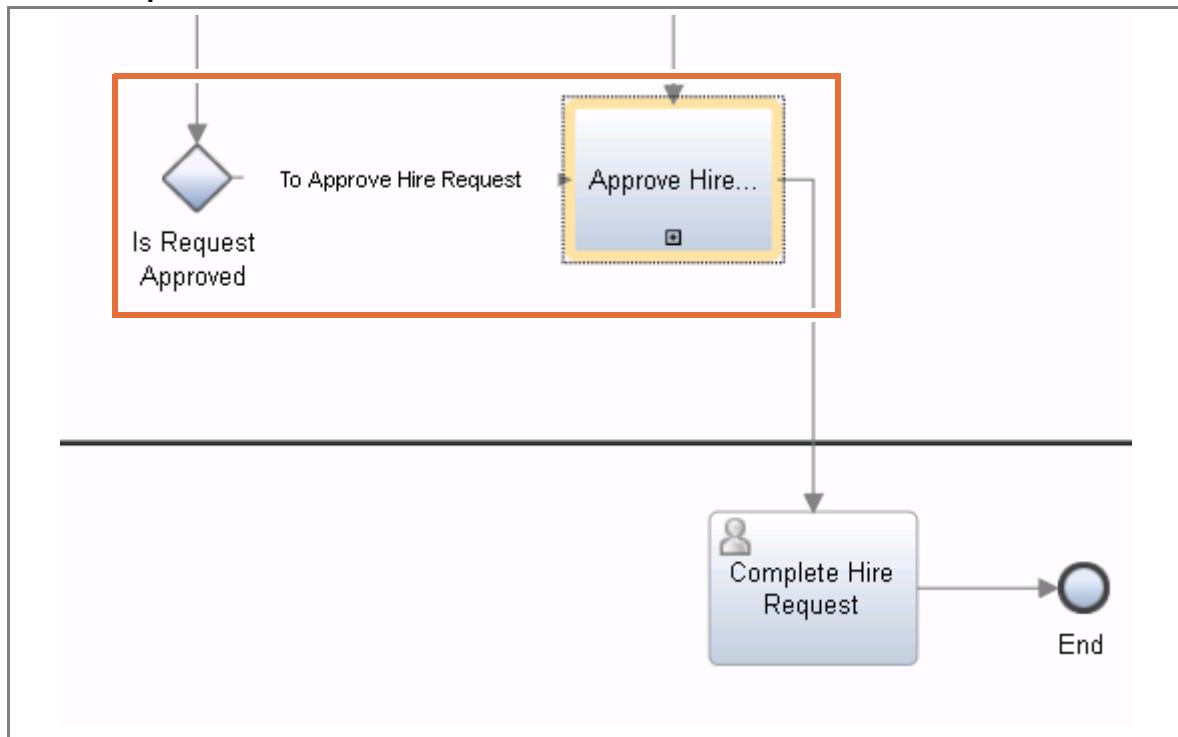
- ___ 7. Connect the gateway.
 - ___ a. Connect the **Approve New Hire Request** to the top of the **Is Request Approved** gateway.



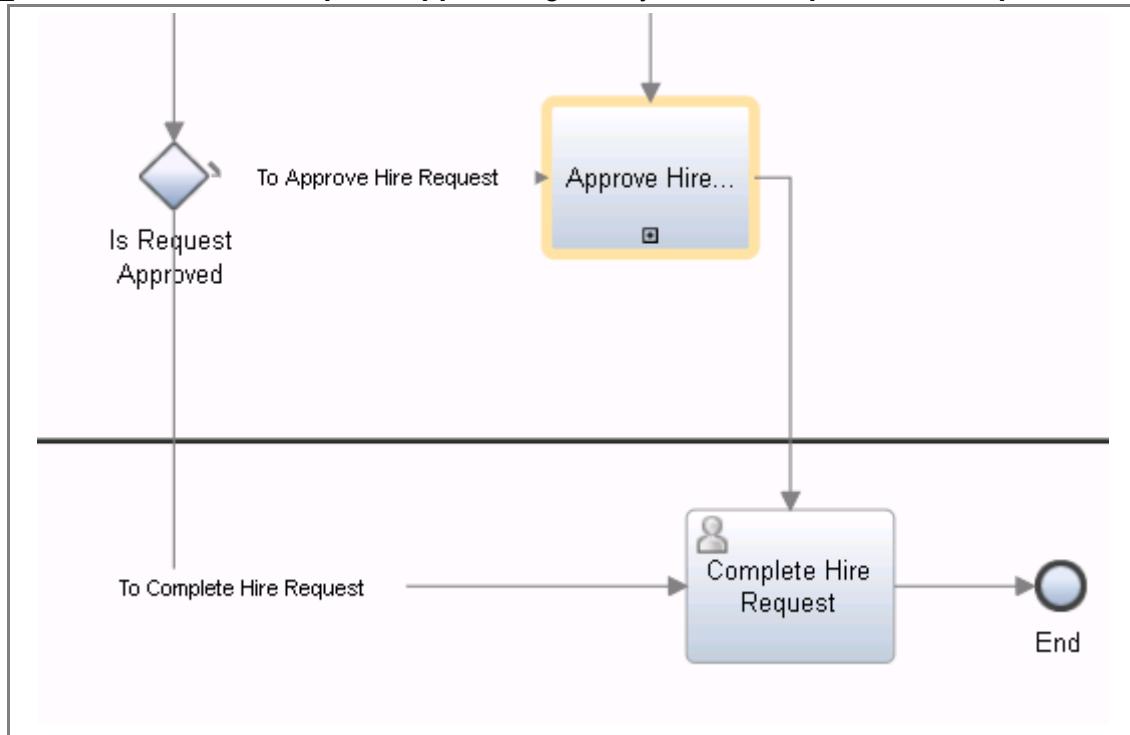
**Reminder**

Straighten your flows and align your objects.

- __ b. Connect the **Is Request Approved** gateway to the left anchor point of **Approve Hire Request**.

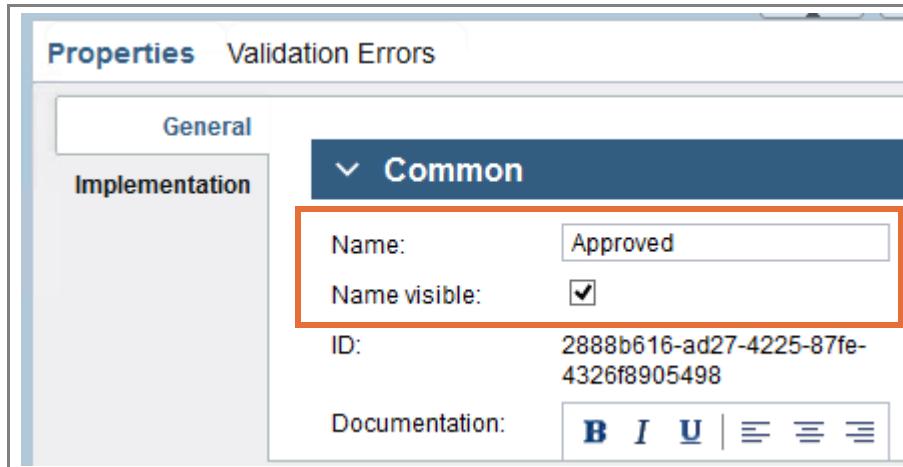


- c. Connect the **Is Request Approved** gateway to the **Complete Hire Request** activity.



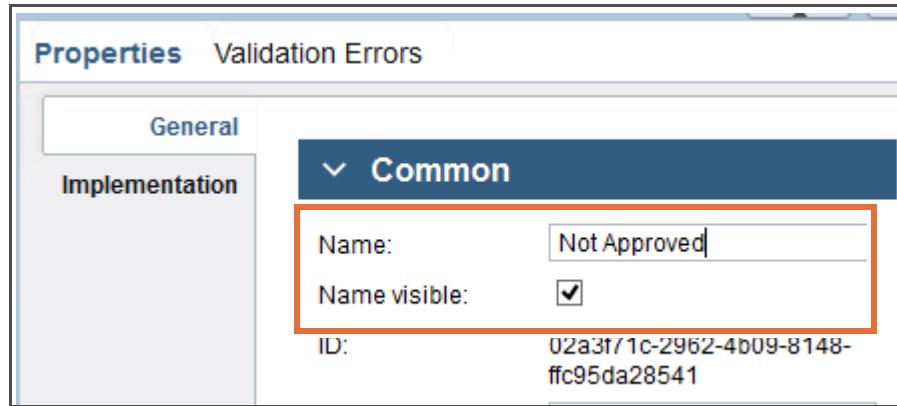
- 8. Label the flows.

- a. Select the flow between the **Is Request Approved** gateway and the **Approve Hire Request** linked process.
- b. Name the flow Approved and select **Name visible**.

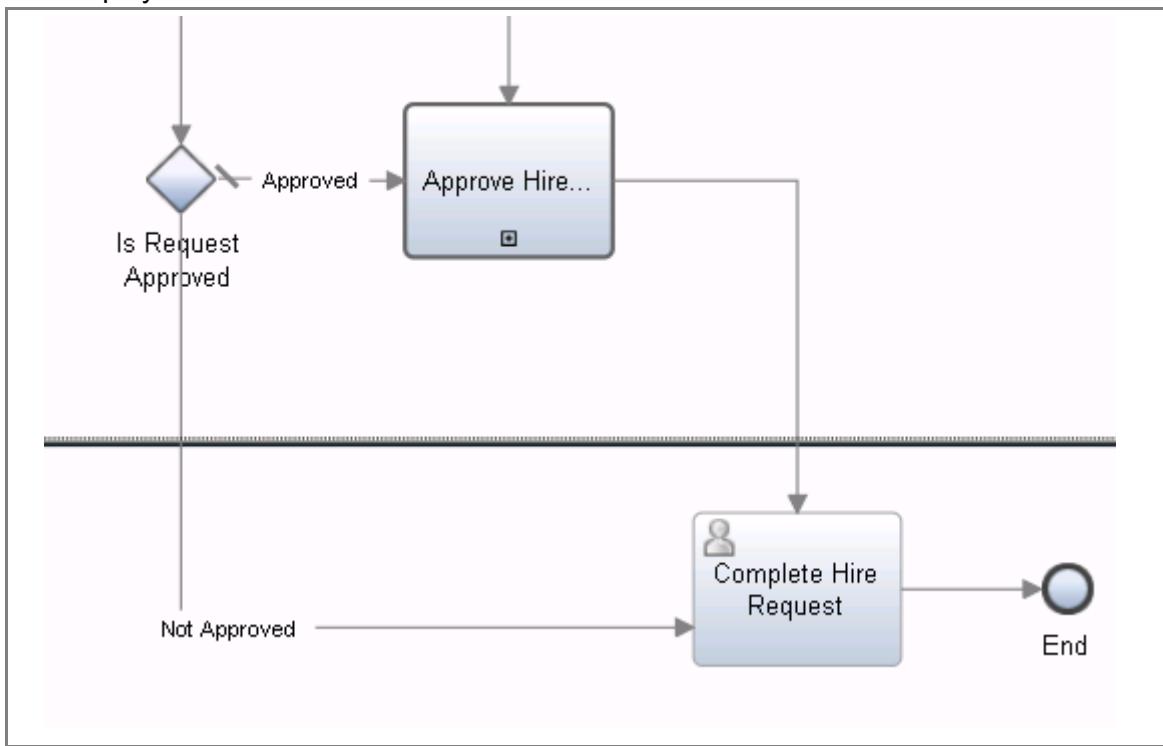


- c. Select the flow between the **Is Request Approved** gateway and **Complete Hire Request**.

- __ d. Name the flow Not Approved and select **Name visible**.



The flow displays the label:

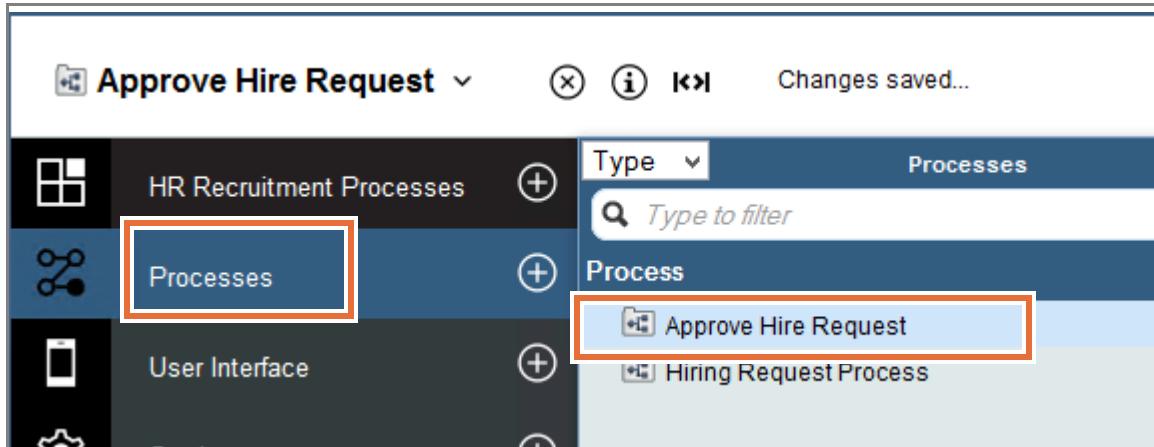


- __ e. Save your process.

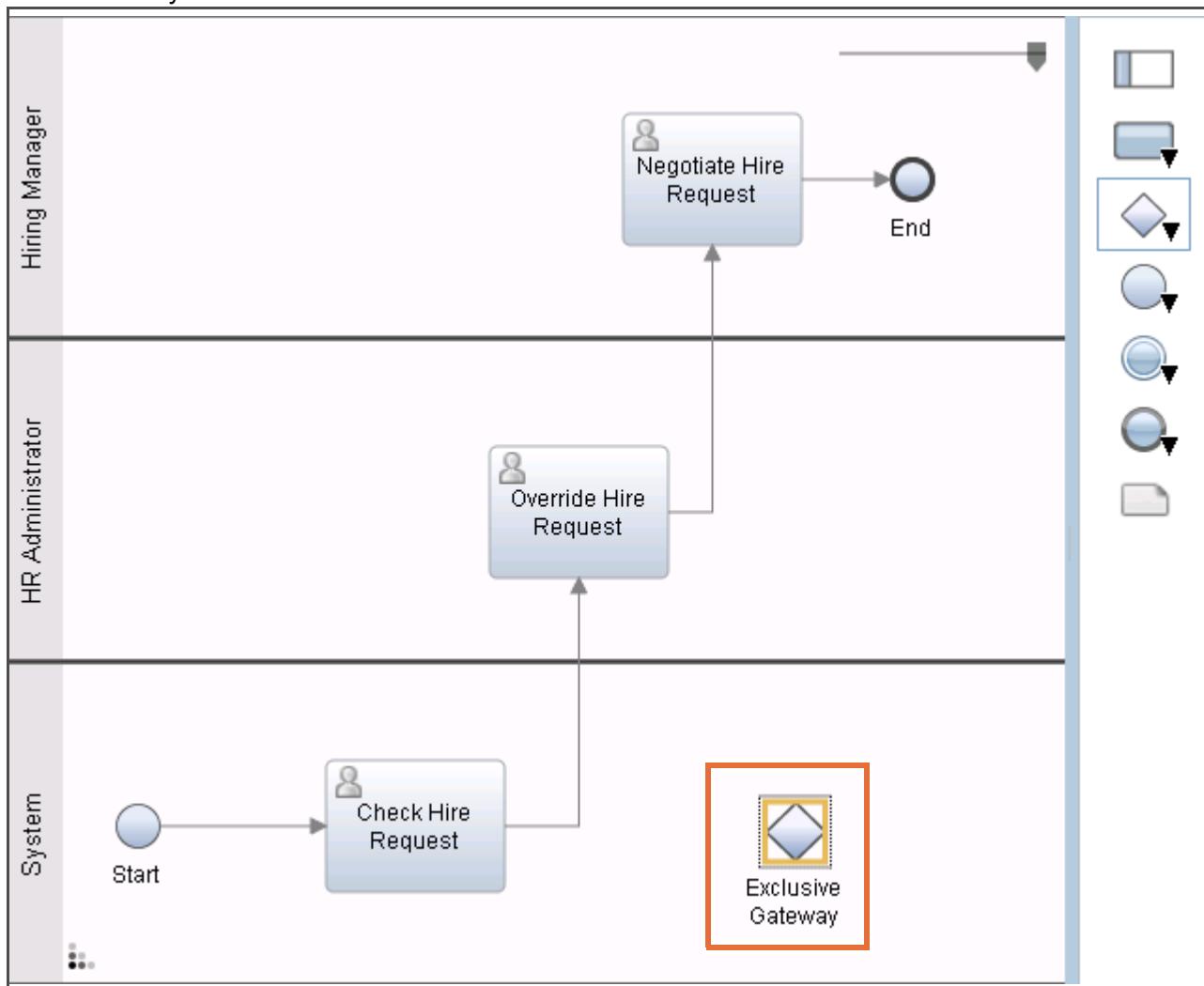
Part 2: Create gateways for the nested process

Add gateways to the nested process Approve Hire Request. In this part, create a gateway that is called Is Salary Compliant.

- To open the Approve Hire Request process, click **Processes** in the library and click **Approve Hire Request** from the menu.

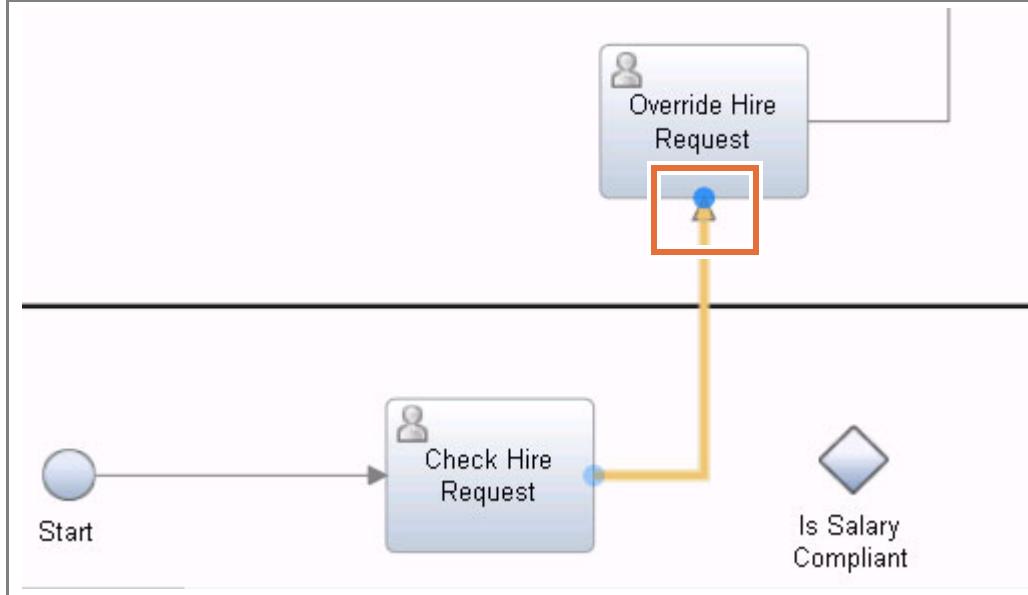


- 2. Drag a **Gateway** from the palette onto the canvas to the right of the **Check Hire Request** activity.

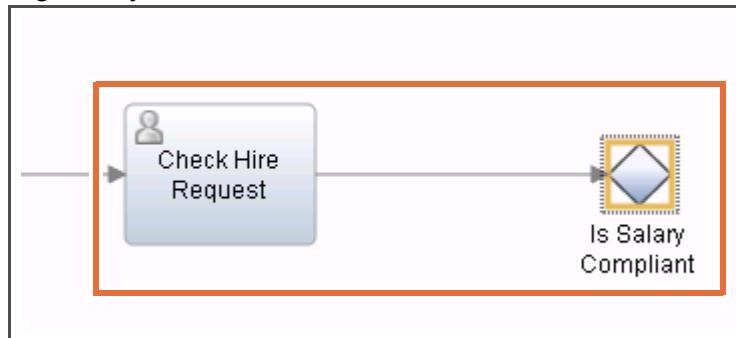


- 3. Rename the gateway to: Is Salary Compliant
 — 4. Verify that the Gateway type is **Exclusive Gateway**.
 — 5. Connect the gateway.
 — a. Select the sequence flow between **Check Hire Request** and **Override Hire Request**.

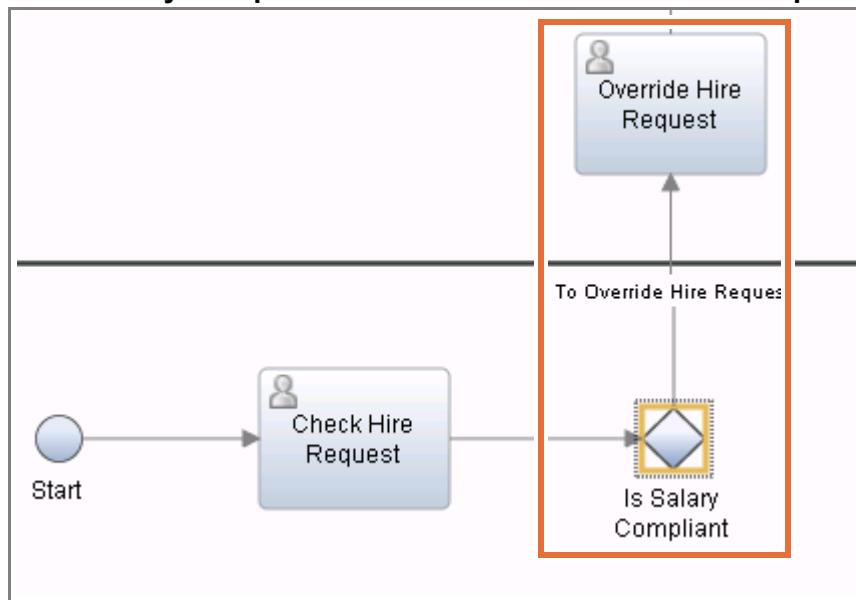
- ___ b. Hover over the tip of the sequence flow to see a blue point.



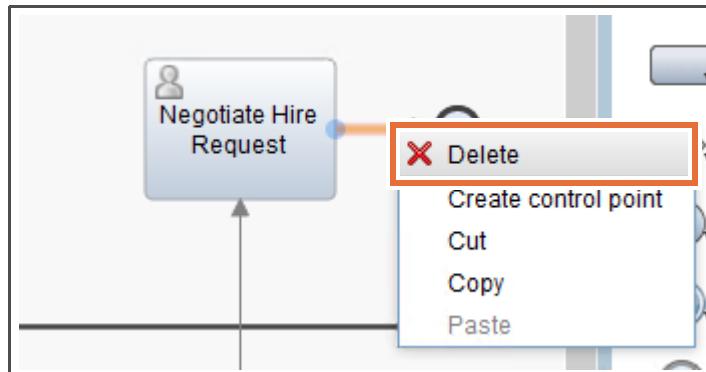
- ___ c. Select the blue point and drag the flow to connect **Check Hire Request** to the **Is Salary Compliant** gateway.



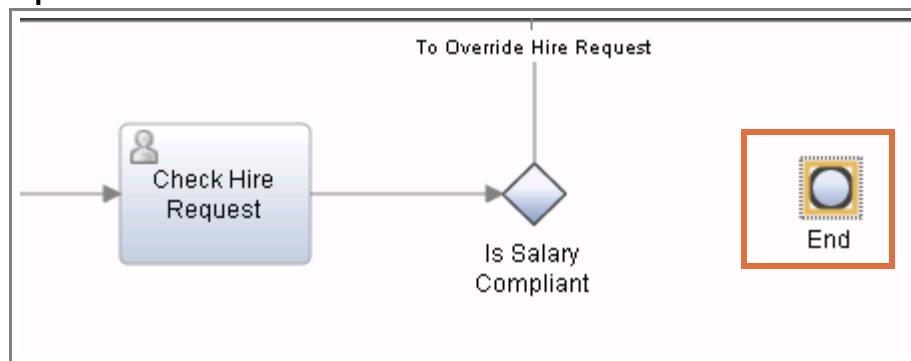
- ___ d. Connect **Is Salary Compliant** to the bottom of **Override Hire Request**.



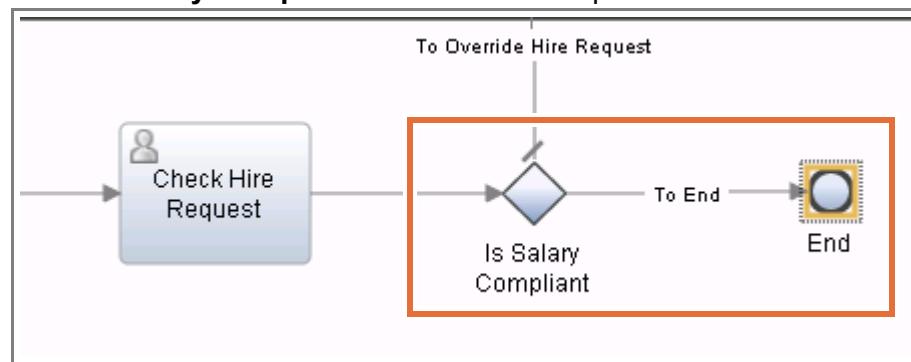
- e. Remove the existing sequence flow between **Negotiate Hire Request** and the **End** event.



- f. Move the **End** event from the top of the diagram and place it to the right of **Is Salary Compliant**.



- g. Connect **Is Salary Compliant** to the left anchor point of the **End** event.

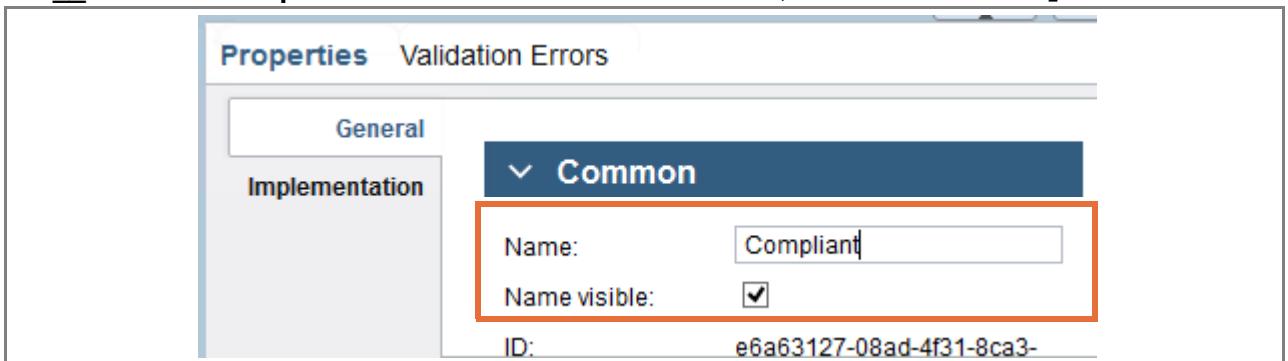


- h. Save your changes.

6. Label and straighten the flows.

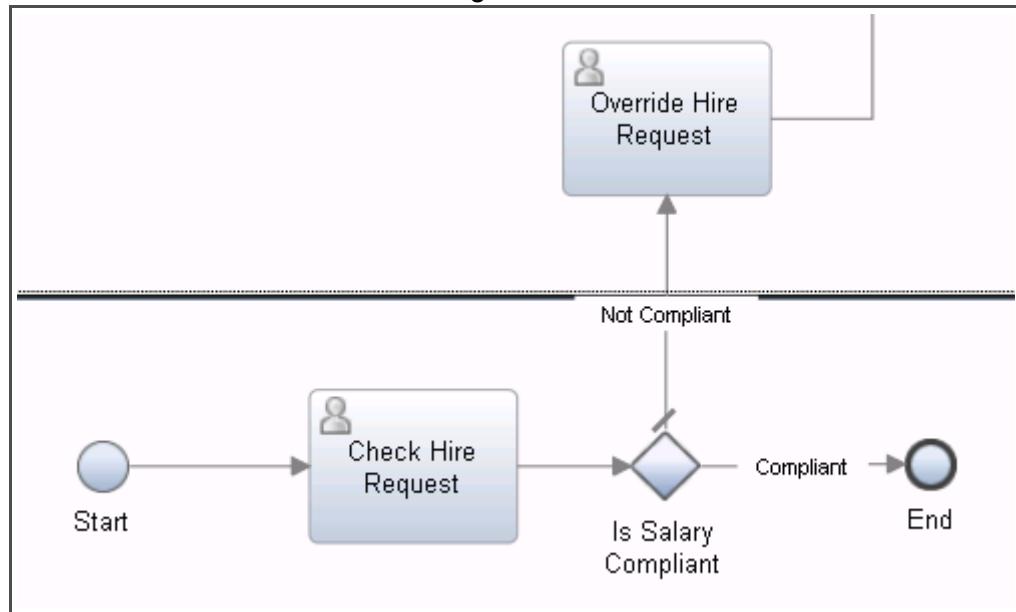
- a. Select the flow between the **Is Salary Compliant** gateway and **Override Hire Request**.
- b. Name the flow: **Not Compliant**
- c. Select the flow between the **Is Salary Compliant** gateway and the **End** event.

- ___ d. In the **Properties > General > Common** section, name the flow: Compliant

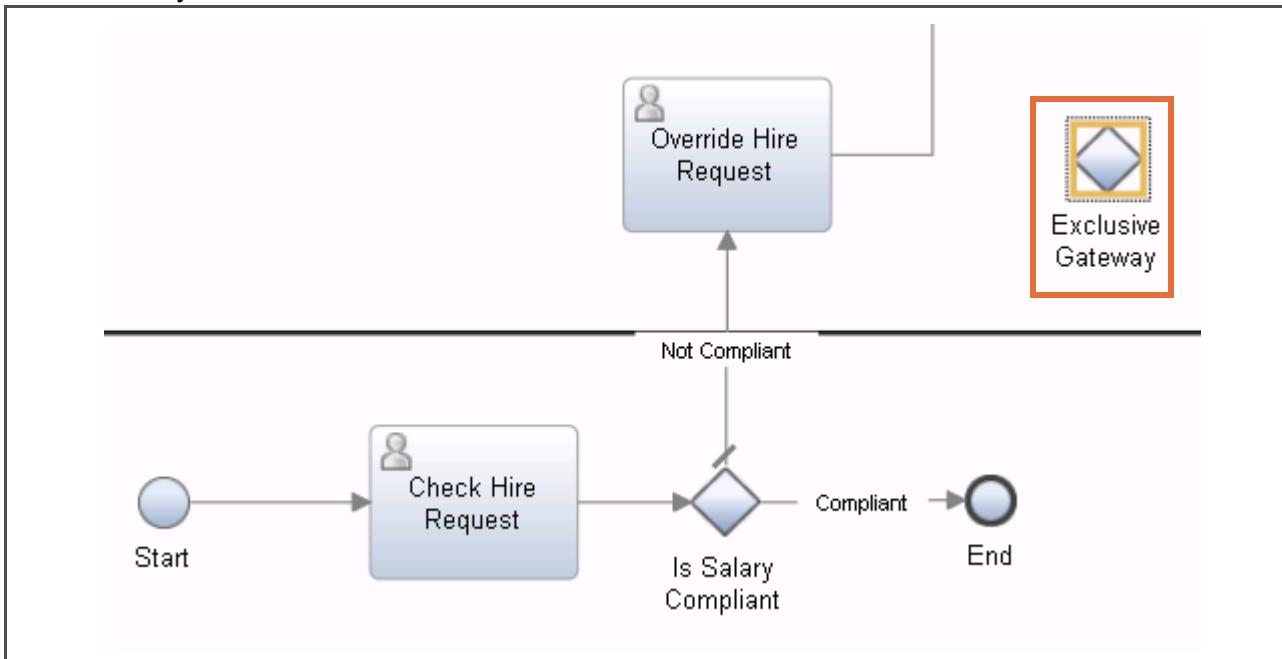


Information

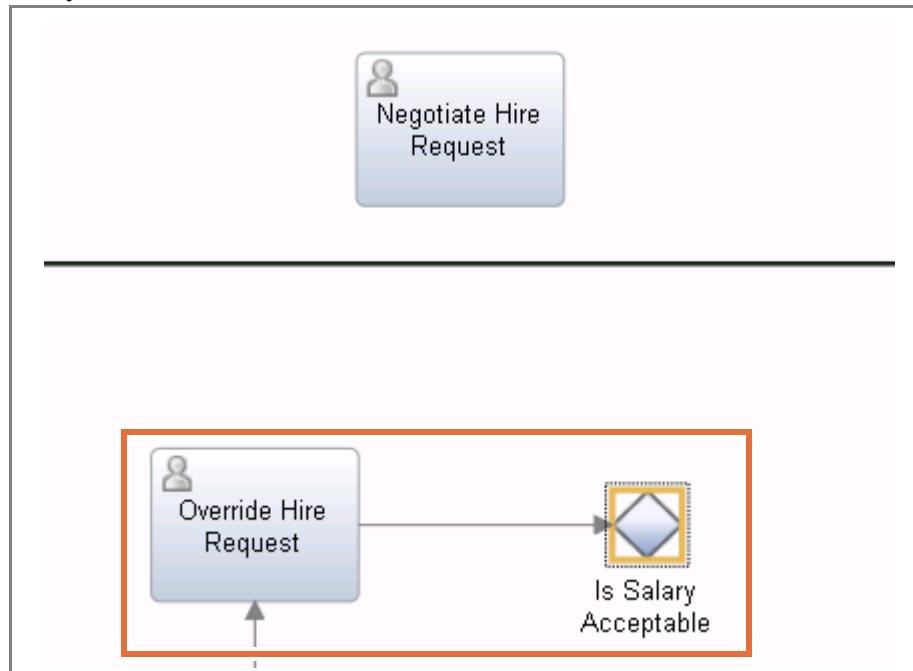
Your process should look similar to this image.



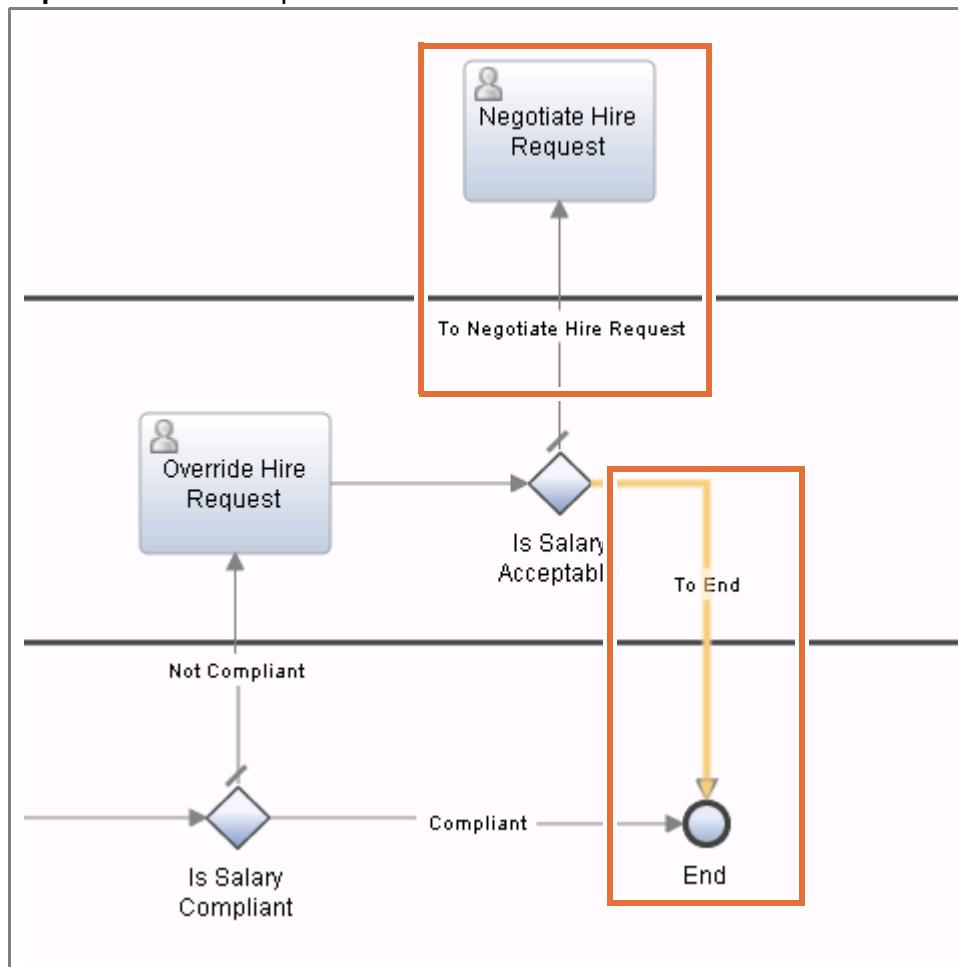
7. Drag a **Gateway** from the palette onto the canvas to the right of the **Override Hire Request** activity.



8. In the **Properties** section, set the gateway name to: **Is Salary Acceptable**
 9. Connect the gateway.
 a. Delete the flow between the **Override Hire Request** and the **Negotiate Hire Request** activities.
 b. Connect the **Override Hire Request** to the left anchor of the **Is Salary Acceptable** gateway.



- ___ c. Connect the **Is Salary Acceptable** gateway to the bottom anchor of **Negotiate Hire Request** and to the top anchor of the **End** event.



- ___ 10. Label the flows.

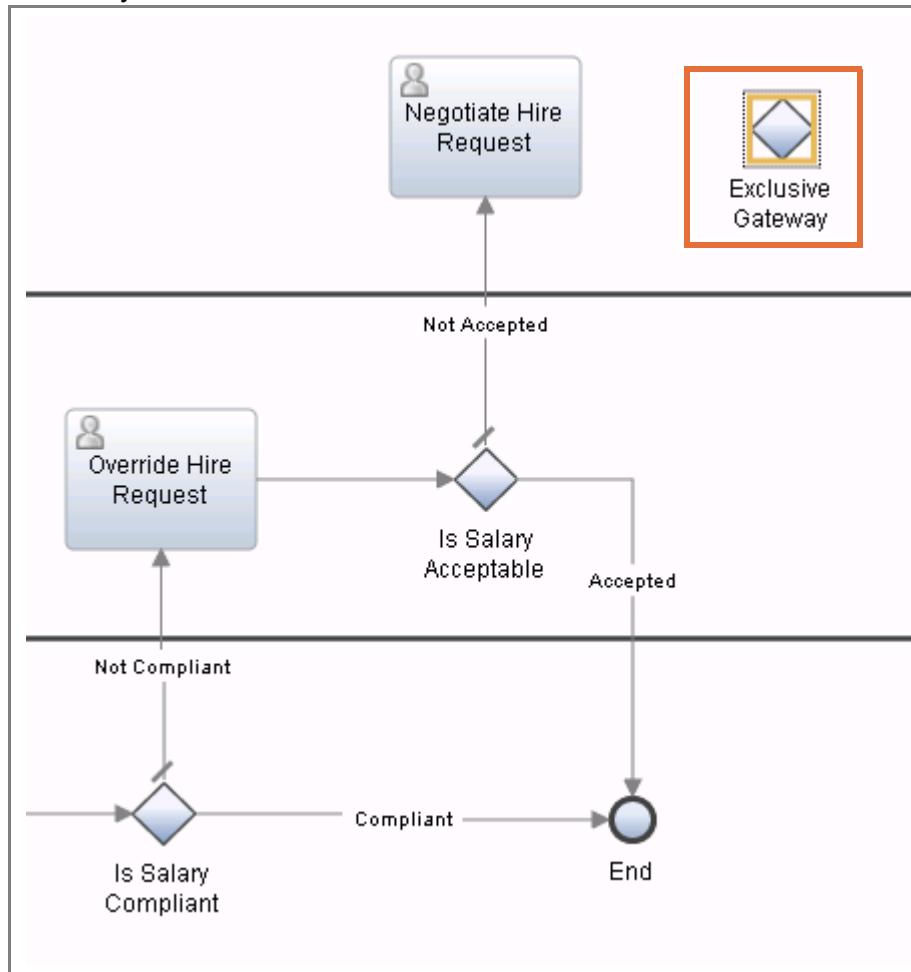


Reminder

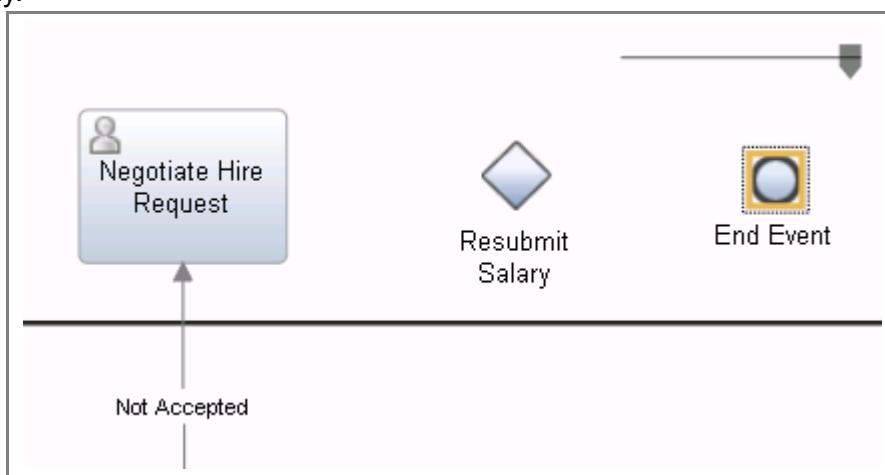
If the HR Administrator rejects the requested salary, the HR Administrator must provide comments for the violation, add a proposed salary, and send the request back to the Hiring Manager who originated the request. Label the flows as Accepted or Not Accepted.

- ___ a. Name the flow between the **Is Salary Acceptable** gateway and **Negotiate Hire Request** as: Not Accepted
- ___ b. Name the flow between the **Is Salary Acceptable** gateway and the end event as: Accepted

- 11. Drag a **Gateway** from the palette onto the canvas to the right of the **Negotiate Hire Request** activity.

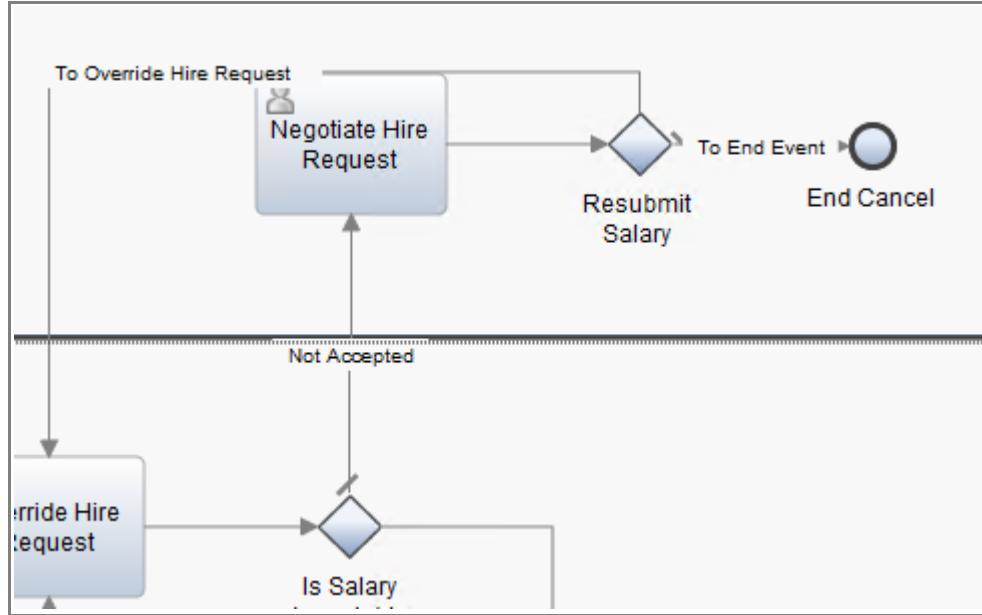


- 12. Rename the gateway to **Resubmit Salary** and verify that the Gateway type is **Exclusive Gateway**.
- 13. Drag an **End event** from the palette onto the canvas to the right of the **Resubmit Salary** gateway.

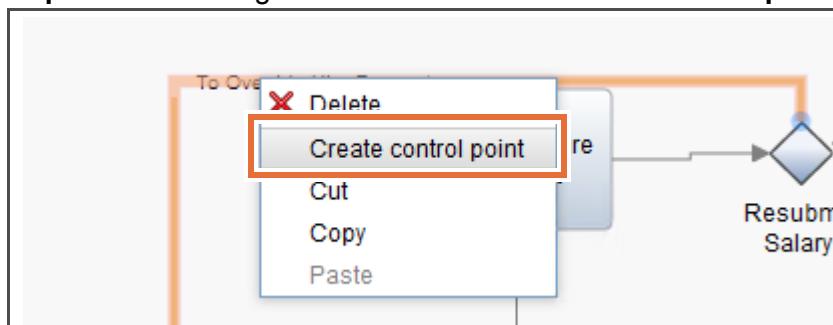


- 14. Rename the **End event** as: **End Cancel**

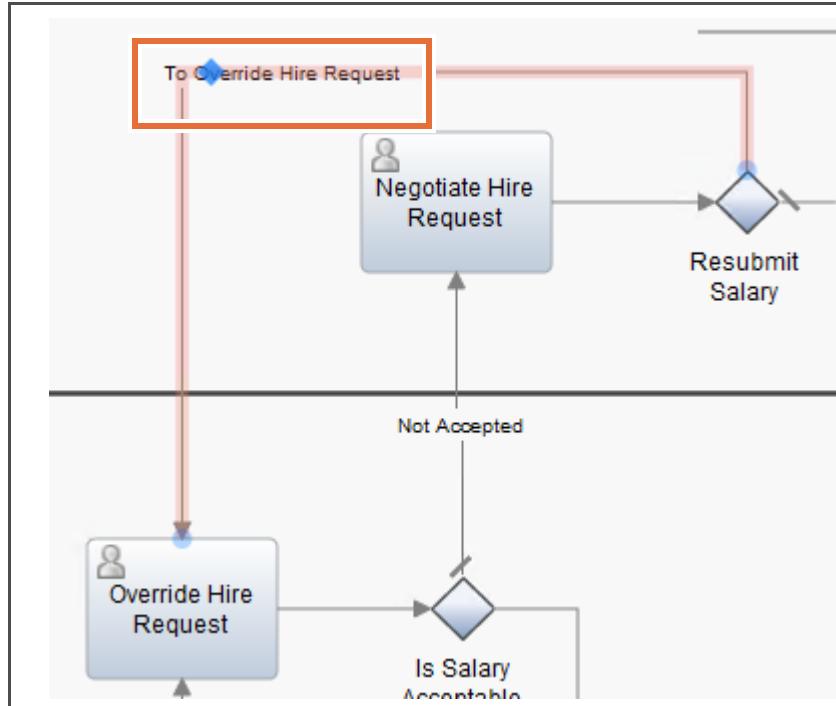
- 15. Connect the gateway.
- a. Connect **Negotiate Hire Request** to the **Resubmit Salary** gateway.
 - b. Connect the **Resubmit Salary** gateway to the **End Cancel** event.
 - c. Connect the **Resubmit Salary** gateway to the top of **Override Hire Request**.



- d. To fix the flow that overlaps the Negotiate Hire Request activity, select the **To Override Hire Request** flow and right-click the flow. Click **Create control point**.



- __ e. Drag the blue diamond to move the flow above the activity.



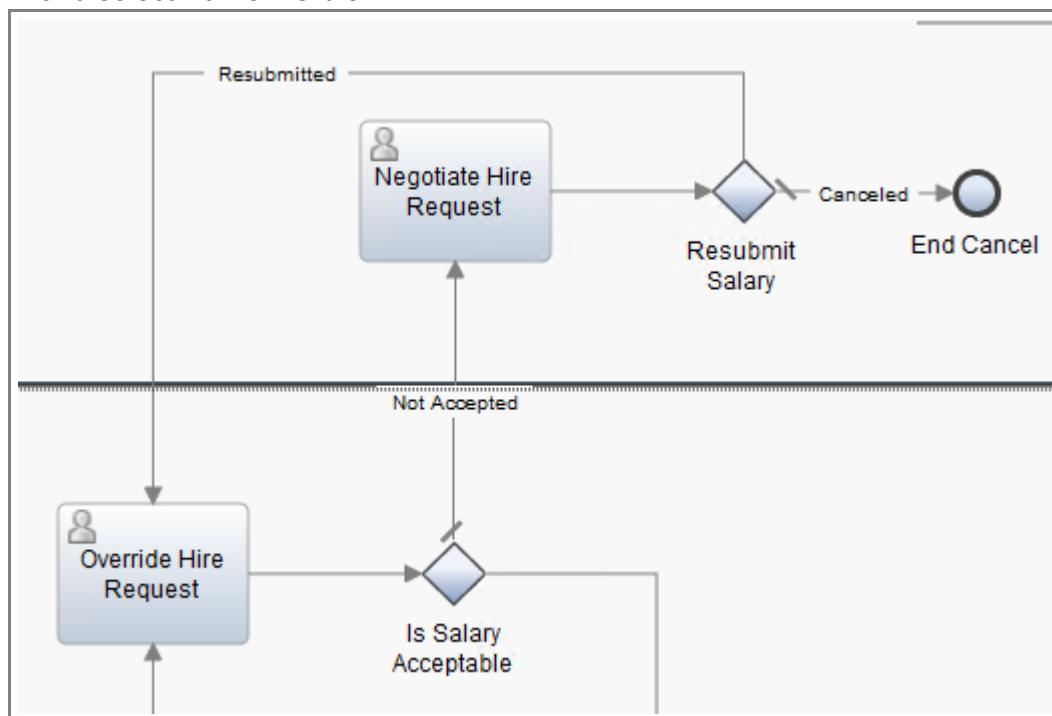
Important

Make sure that you create the flows to these elements in the correct directions. You can align the activities by selecting and dragging them to a location where you can see them better.

- __ 16. Label the flows.

- __ a. Name the flow between the **Resubmit Salary** gateway and **Override Hire Request** as **Resubmitted** and select **Name visible**.

- __ b. Label the flow between the **Resubmit Salary** gateway and the **End** event as **Canceled** and select **Name visible**.



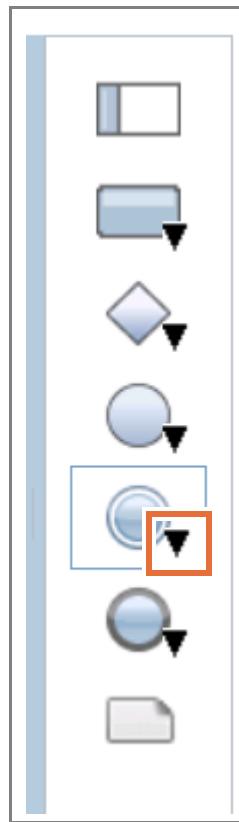
Part 3: Modeling timer intermediate events

Examine the process requirements and select the one that models an escalation.

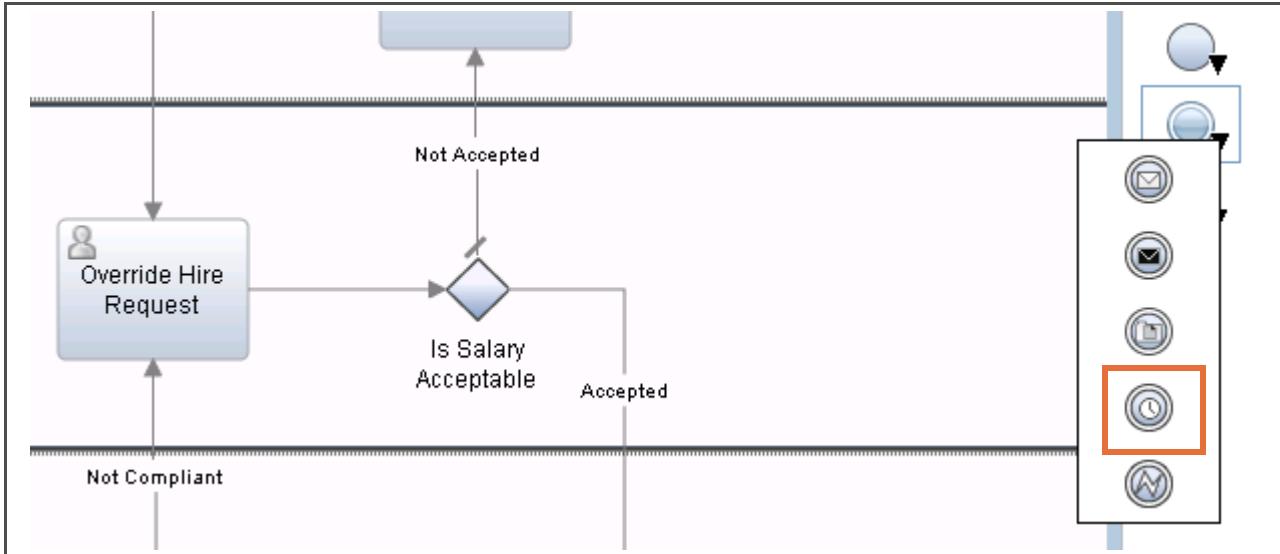
Refer to the core requirements for an escalation in 2.10 listed in Exercise 2. The HR Administrator has 4 hours to complete the review. If the review is not completed within 4 hours, an email is sent to the HR Administrator notifying the HR Administrator of the missed deadline.

Use the timer implementation option to model escalation paths or delays in your processes. Using a timer intermediate event, you can specify a time interval after or before which some activity is conducted. The timer implementation option is available for events that are included in the process flow and events that are attached to an activity.

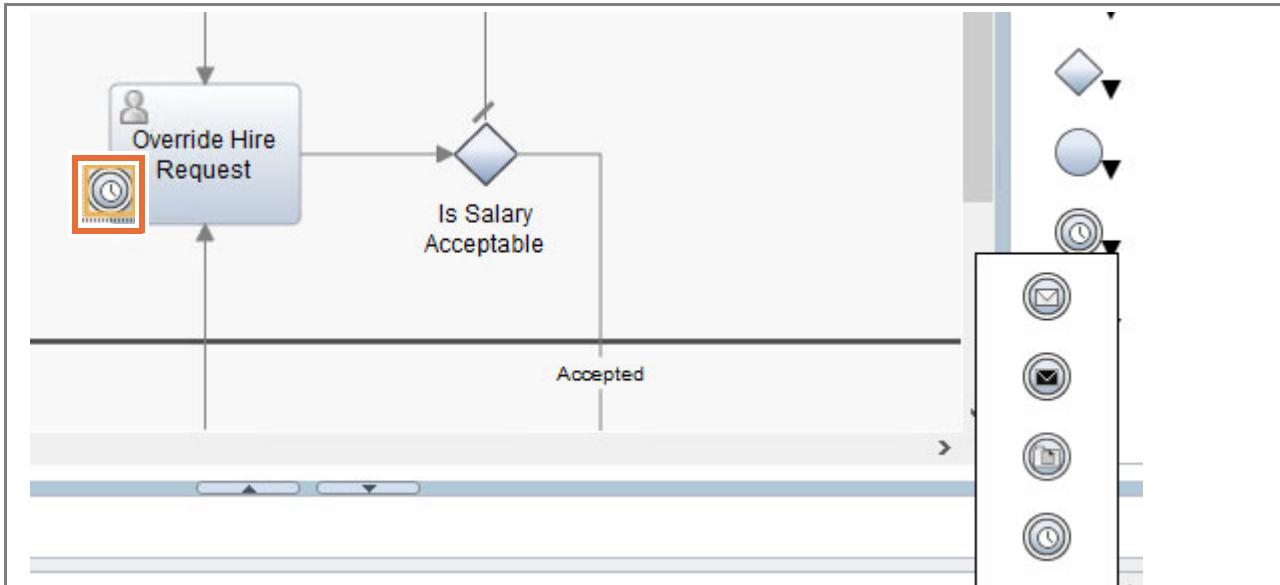
- 1. In the **Approve Hire Request** process, select the **drop-down icon** beside the **Intermediate** event.



2. Select the **Timer event** from the set of intermediate events.



3. Drag the **Timer event** from the palette onto the lower-left anchor of the **Override Hire Request** activity.

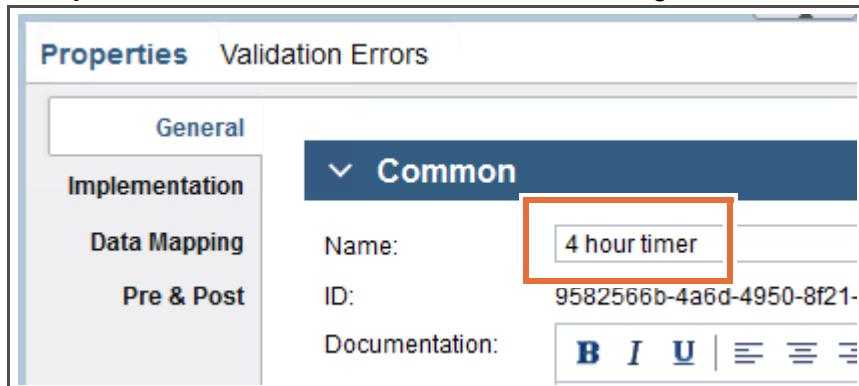


Important

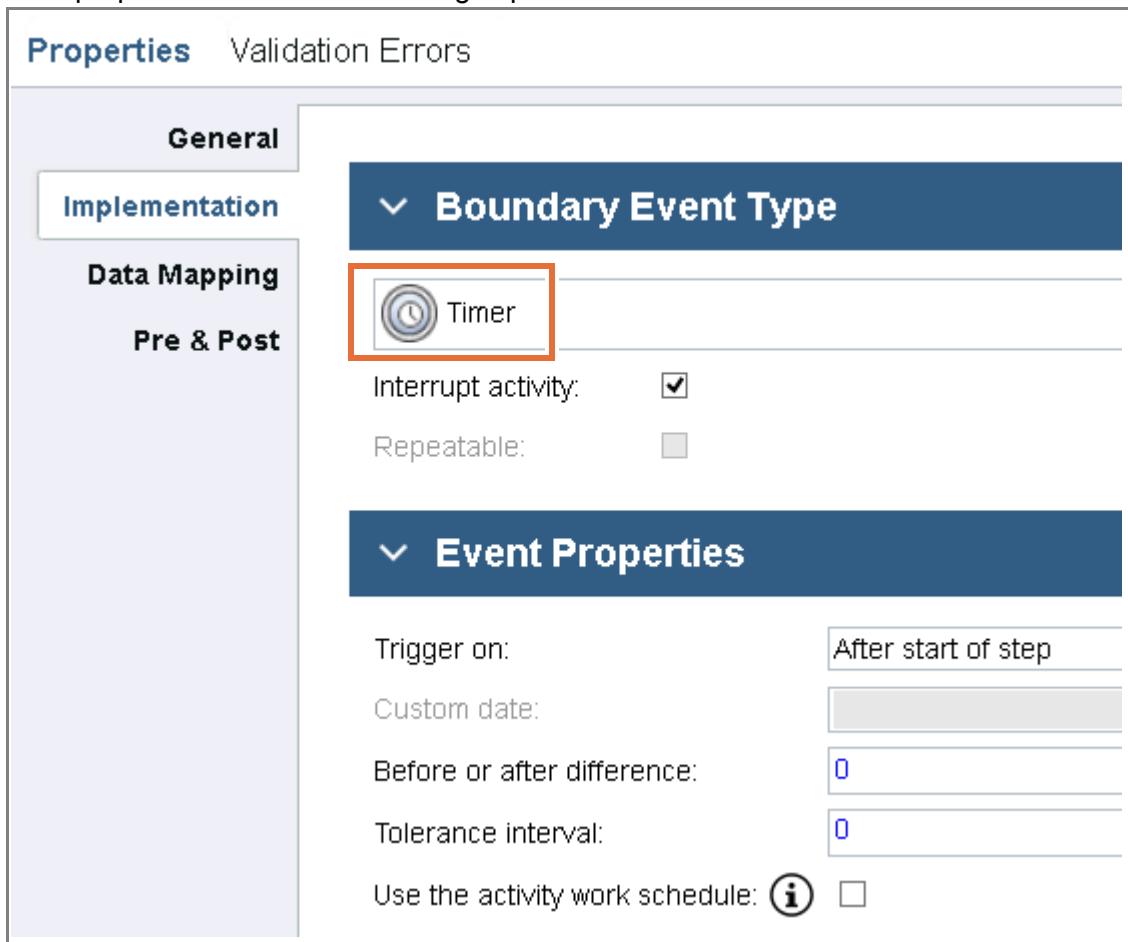
Sometimes attaching the timer event to the activity can be tricky. If it drops on the canvas instead, you can click the timer event and drag it onto the activity. Look for the anchor points to appear on the activity while the timer is hovering over the activity, then drag the timer to one of the anchor points.

For this exercise, make sure that you drag the timer event onto the activity instead of outside of the activity. In this case, you want a boundary or attached intermediate event instead of a sequence flow intermediate event. If you correctly create an attached timer event, the name for an attached intermediate event is also changed to Boundary Event.

- 4. Click the **Intermediate** boundary event and set its properties.
- a. Select the **Timer** event.
- b. In the **Properties > General > Common** section, change the name to: 4 hour timer

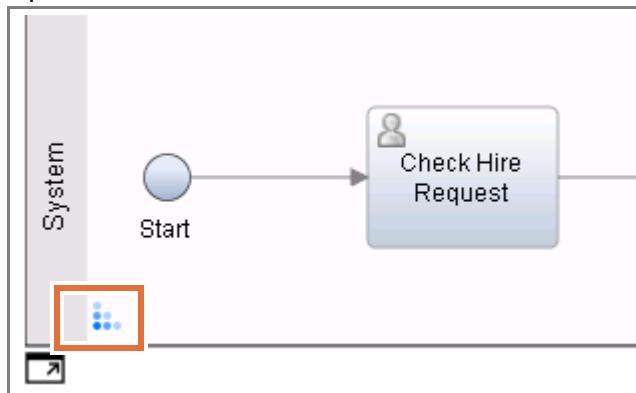


- c. Select the **Properties > Implementation** menu. In the **Boundary Event Type** section, make sure that the Timer boundary event is selected. Keep the remaining default properties that were set during implementation.

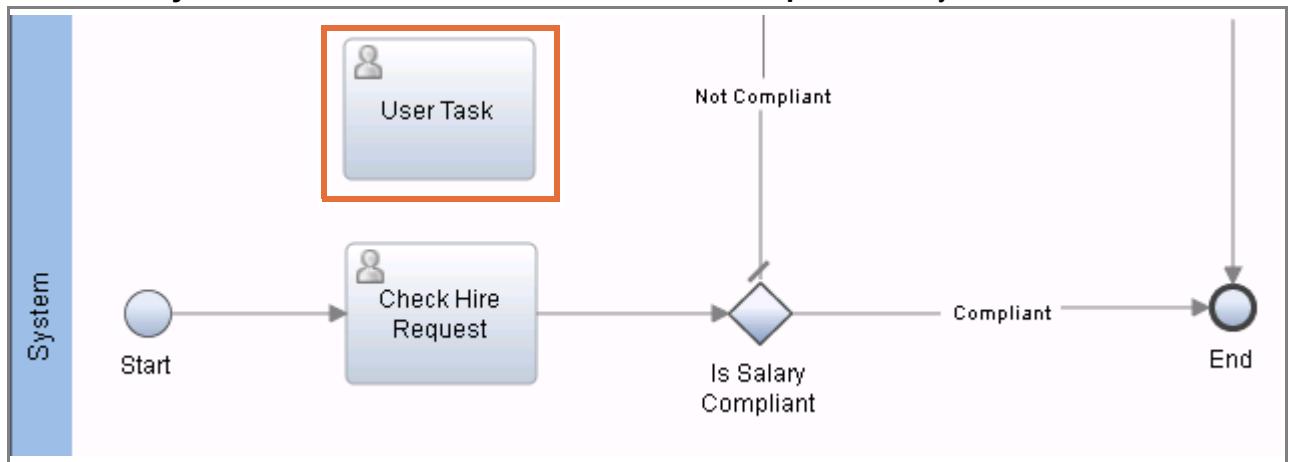


__ 5. Add the Notify HR Administrator activity.

__ a. Resize the **System** lane by clicking and dragging the control point next to the System lane label to expand the lane.



__ b. Expand the Activity icon in the palette and drag a **User Task** from the palette to the **System** lane. Place it above the **Check Hire Request** activity.

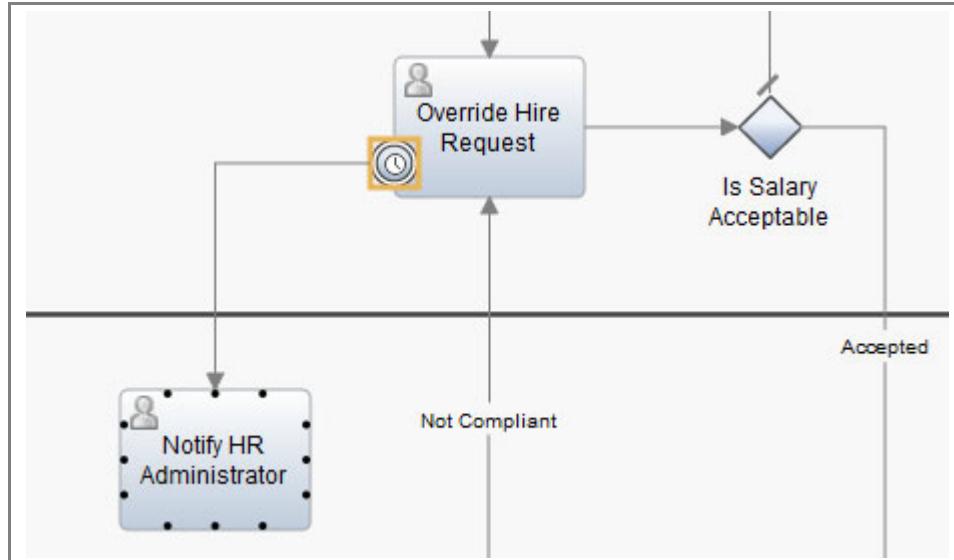


Hint

Drag a box around the activities, events, and the gateway in the System lane, and move them down together to make space for this new activity.

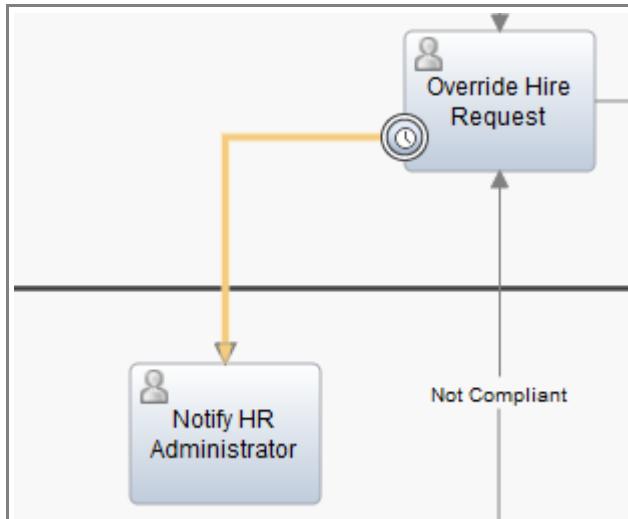
__ c. Rename the activity to: **Notify HR Administrator**

- ___ 6. Connect the **Timer Intermediate Event** to the **Notify HR Administrator** activity.

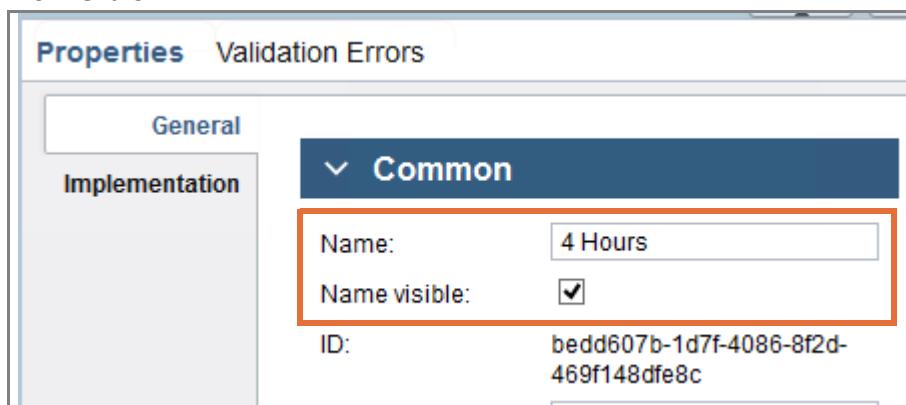


- ___ 7. Label the flow.

- ___ a. Select the flow between the **Timer Intermediate Event** and **Notify HR Administrator**.

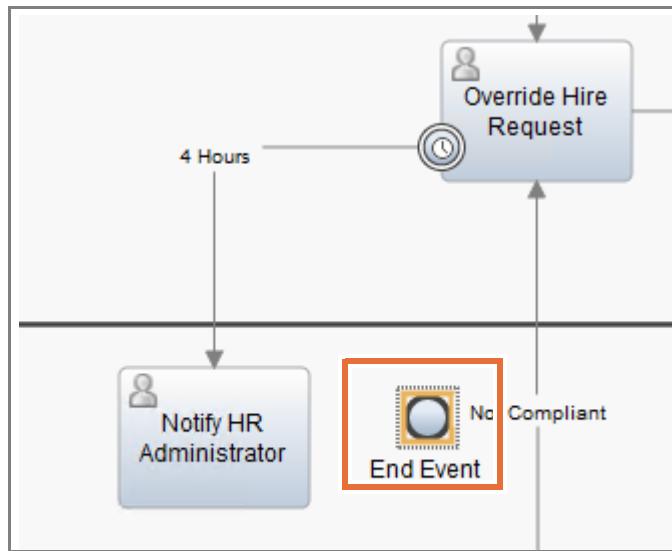


- ___ b. In **Properties > General > Common** section, name the flow as **4 Hours** and select **Name visible**.

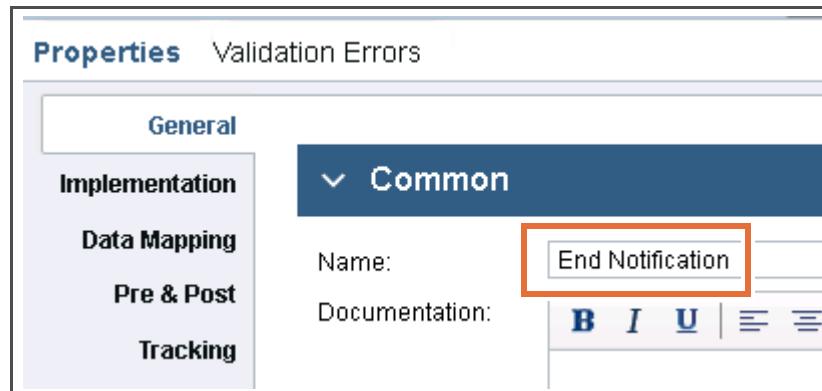


— 8. Add an End Notification event.

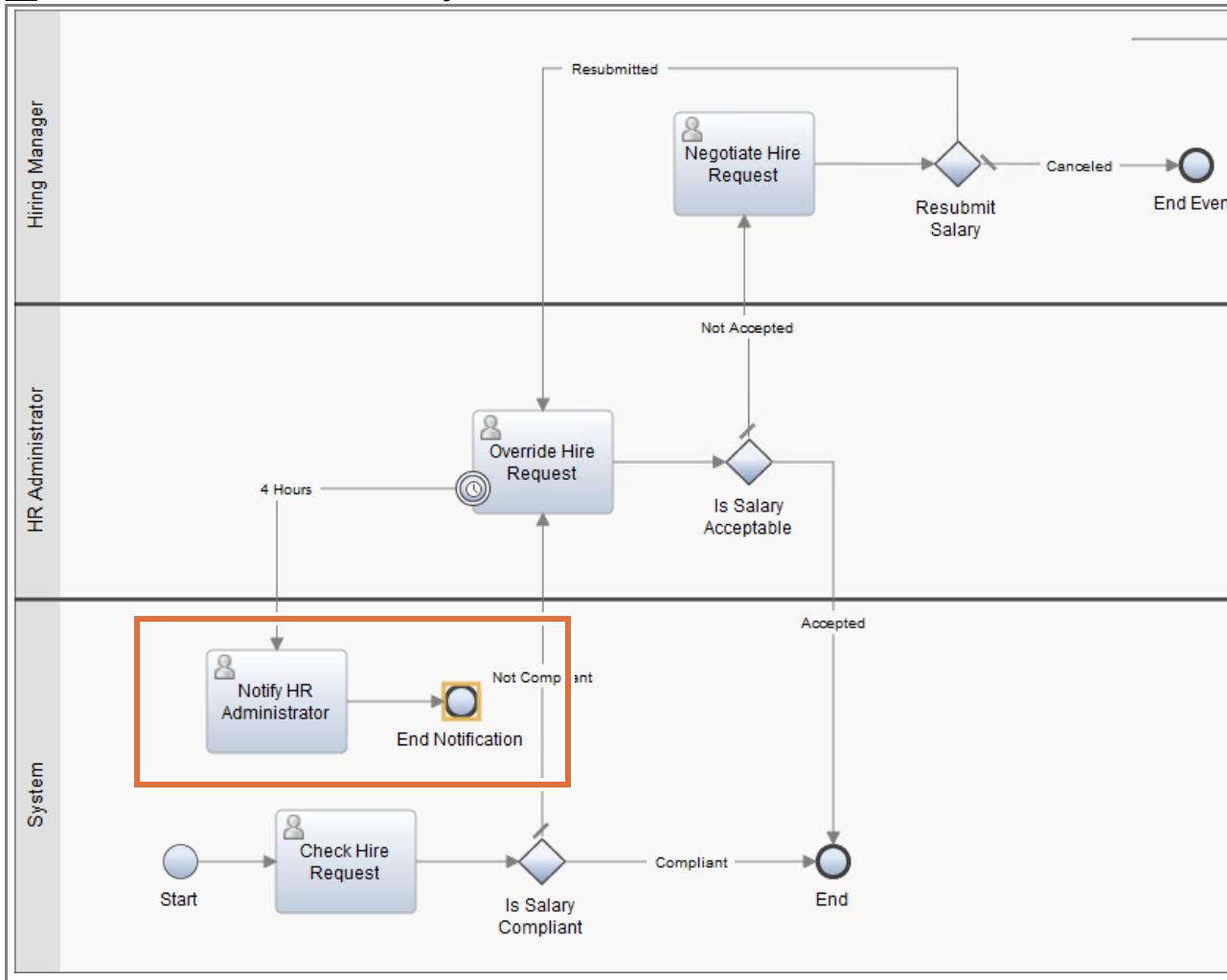
— a. Drag an **End Event** icon from the palette onto the canvas to the right of **Notify HR Administrator**.



— b. Rename the End event to: End Notification



9. Connect the flow from **Notify HR Administrator** to the **End** event.



10. Save your work.



You completed the following tasks:

- Added gateways to a process
- Modeled the appropriate sequence flows for each gateway
- Added a timer intermediate event to a process based on business requirements
- Modeled an escalation path in a process with IBM Process Designer

In the next exercise, you learn how to validate your process model as this step is the final stage of Playback 0.

End of exercise

Exercise review and wrap-up

The first part of this exercise showed adding gateways to the process in IBM Process Designer. Next, you modeled the appropriate sequence flows for each gateway. The last part of this exercise showed adding a timer intermediate event for a process requirement.

Exercise 4. Validating the process model

Estimated time

00:15

Overview

This exercise covers how to validate the business process.

Objectives

After completing this exercise, you should be able to:

- Validate that the business process reflects the intended requirements
- Implement the requirements with playback feedback and new process requirements as input

Introduction

Before moving on to Playback 1, it is a good idea to verify your process in a final playback meeting.

This action can build consensus among the different stakeholders as the model does not change much visually after Playback 0. Sometimes you have a few oscillations between Playback 1 and Playback 0 until a team is firmly in the implementation phase of the playbacks.

Requirements

Successful completion of the previous exercise is required.

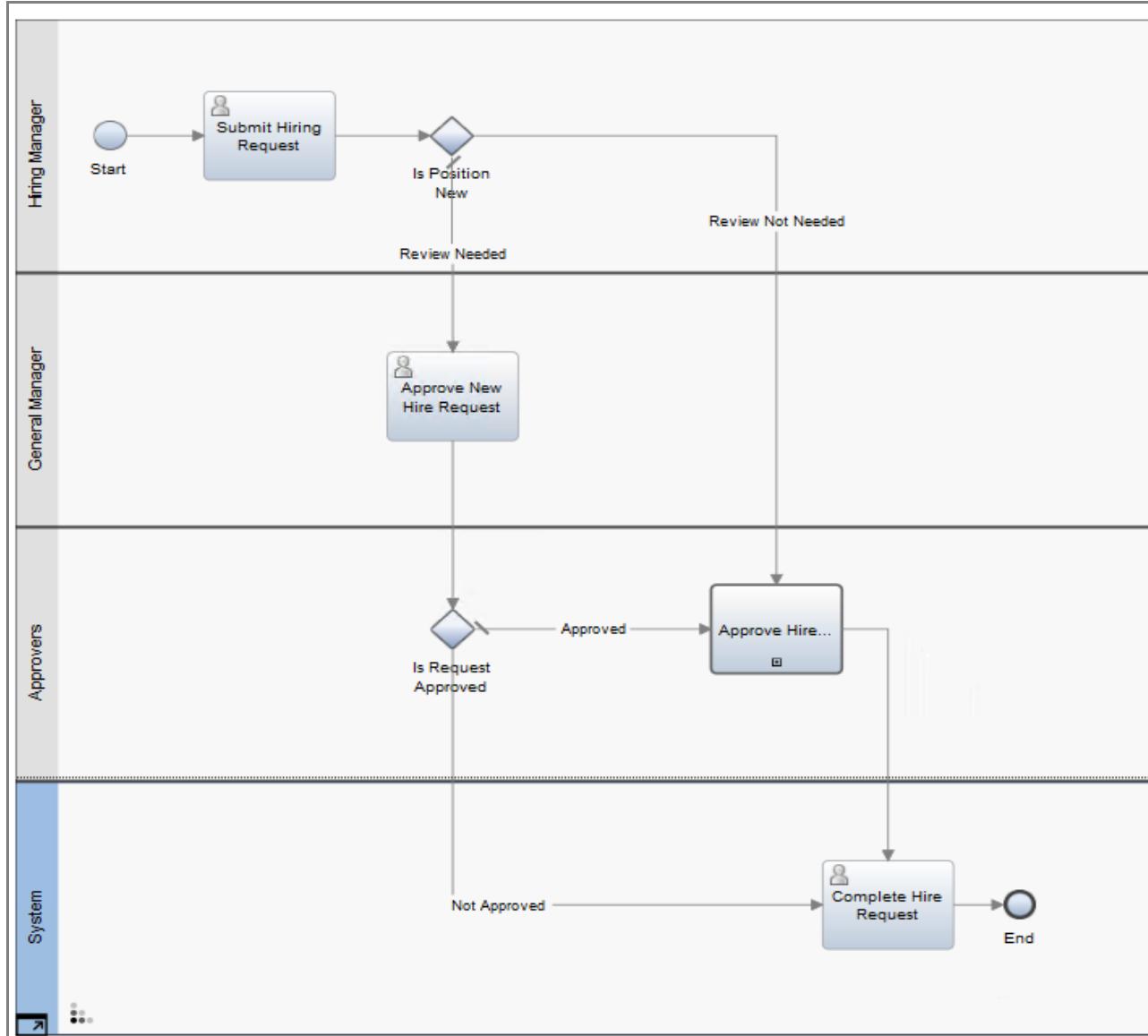
Exercise instructions

Part 1: Verify the process model

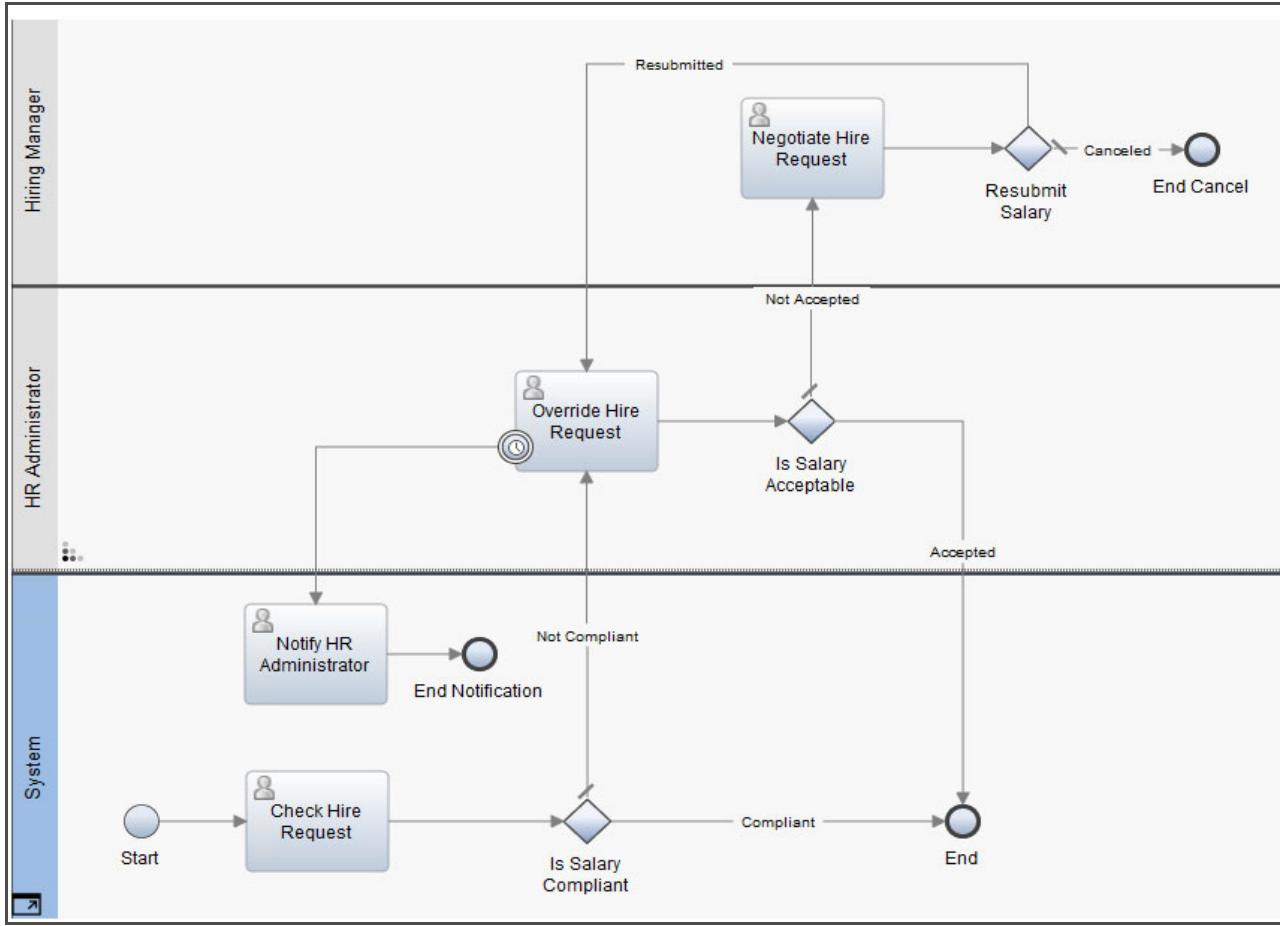
If you have a colleague or others nearby, see whether you can explain your new model to them to simulate a playback. If not, examine your process in IBM Process Designer and compare it to the diagrams.

Ensure that the **Hiring Request Process** and **Approve Hire Request** processes look similar to the diagrams before moving on:

1. View the **Hiring Request Process**.



2. View the **Approve Hire Request** process.



Note

The flows out of the decision gateways might not match your screen because they are not yet implemented. You implement the gateway default flows in the next playback.

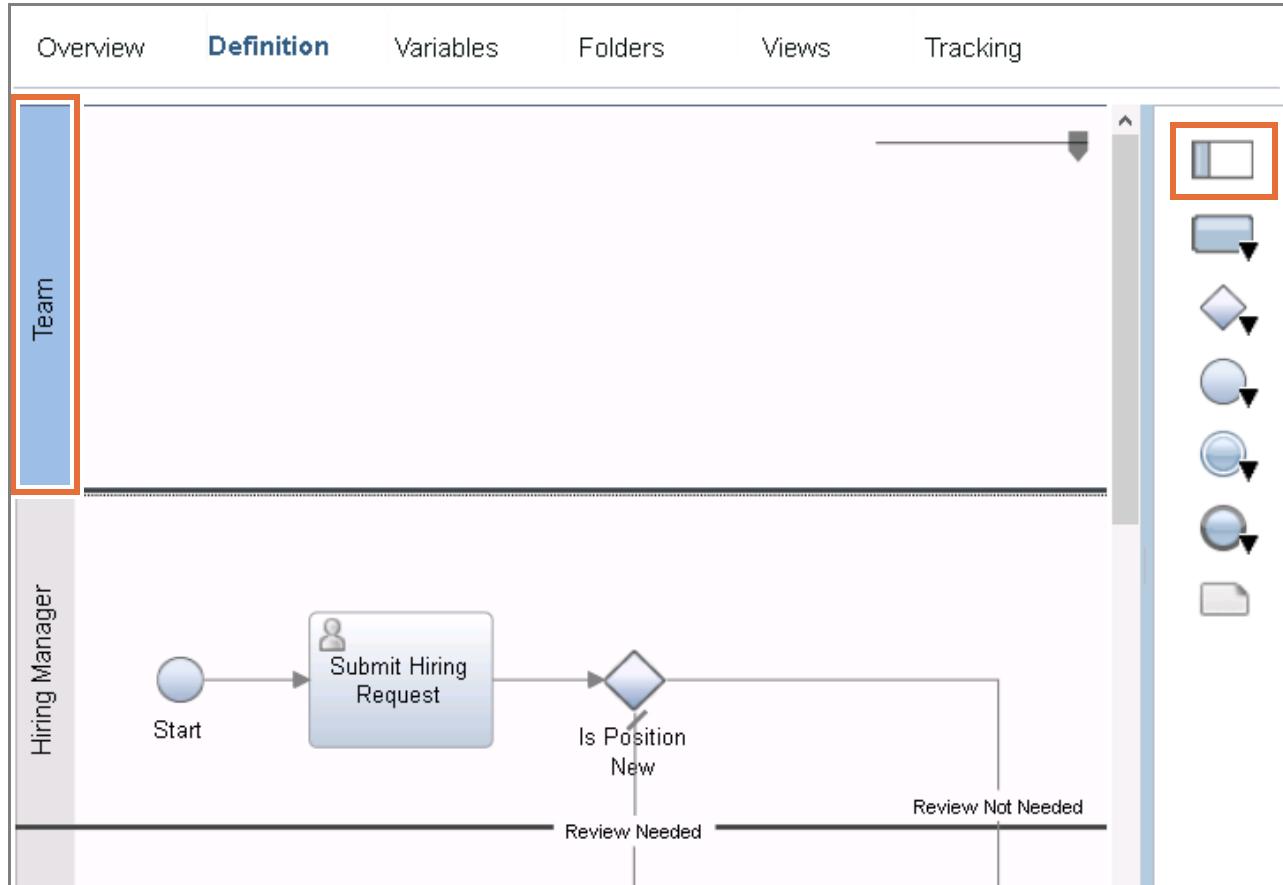
Part 2: Add new process requirements to the process

The process has new requirements. During Playback 0, the vice president mentions:

Several job postings contain wording that violates a new set of hiring laws. To address this issue, the vice president suggests that a lawyer must review every job post at some point before the job opening is released and posted to the public. The lawyer checks for legal compliance and edits if necessary.

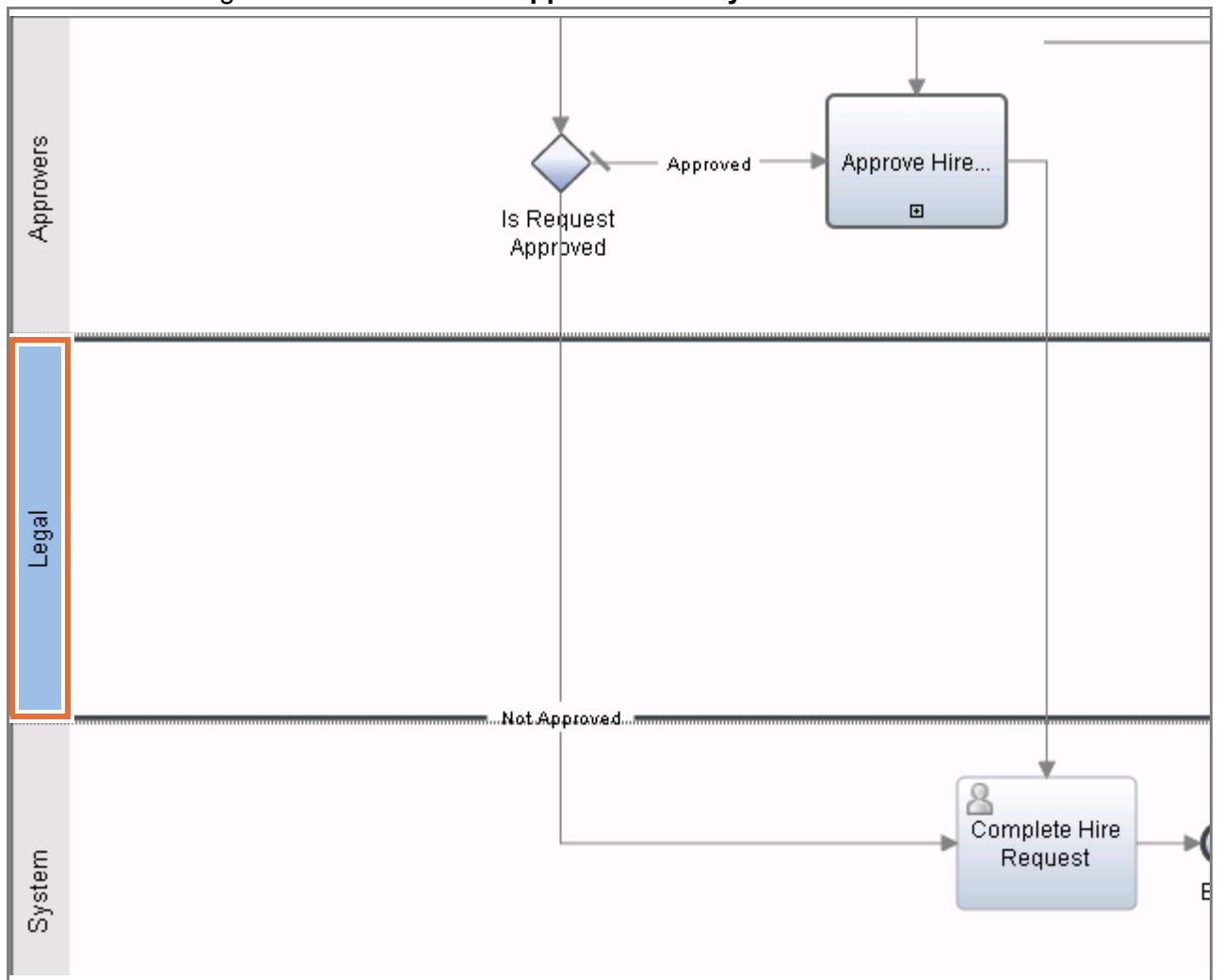
Examine the requirement and come up with the solution for this requirement.

- ___ 1. Add a lane in the Hiring Request Process for the legal team.
 - ___ a. Open the **Hiring Request Process**.
 - ___ b. Drag a **Lane** from the palette to the canvas above the Hiring Manager lane.



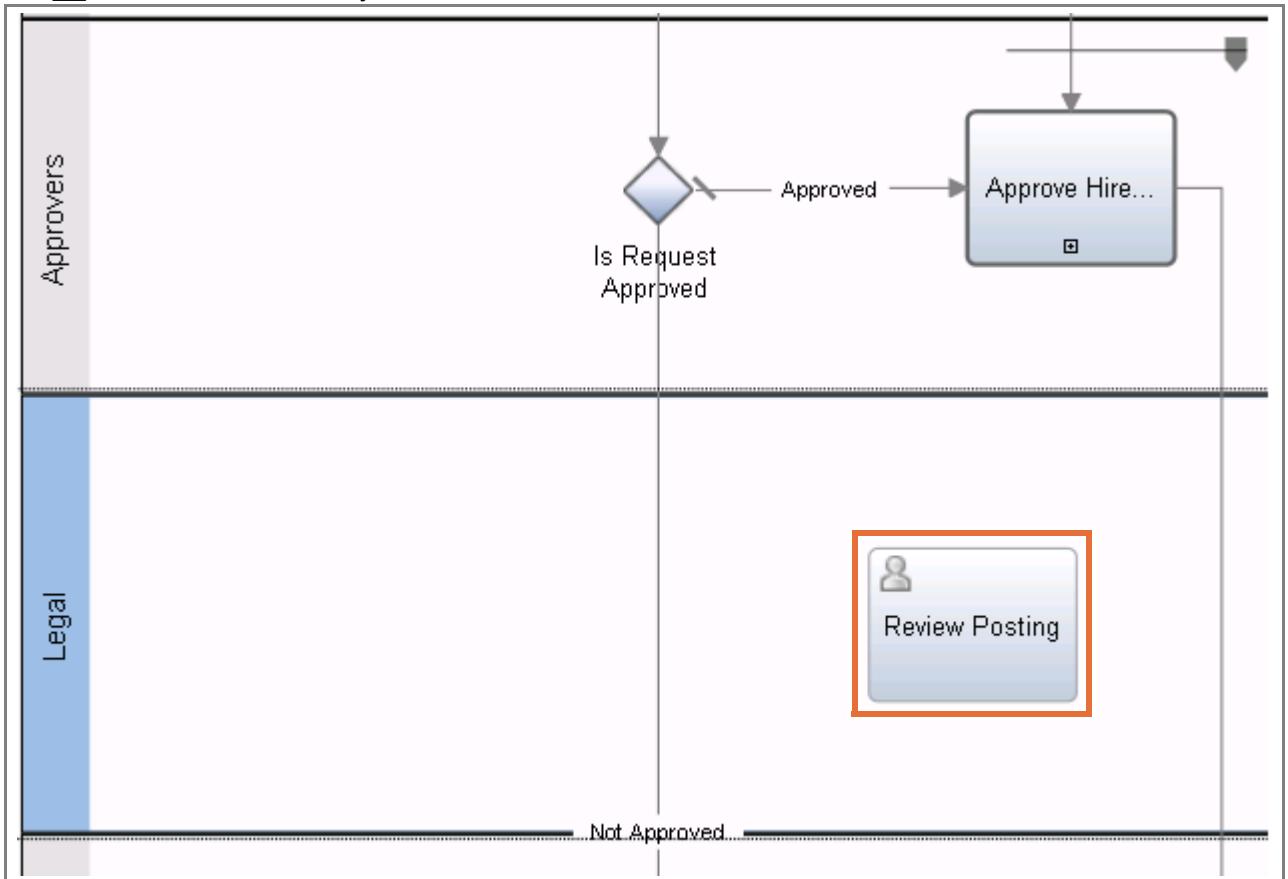
- ___ c. Name the lane: Legal

- ___ d. Right-click the **Legal** lane. Click **Move lane down**. Repeat moving the lane down until the Legal lane is between the **Approvers** and **System** lanes.



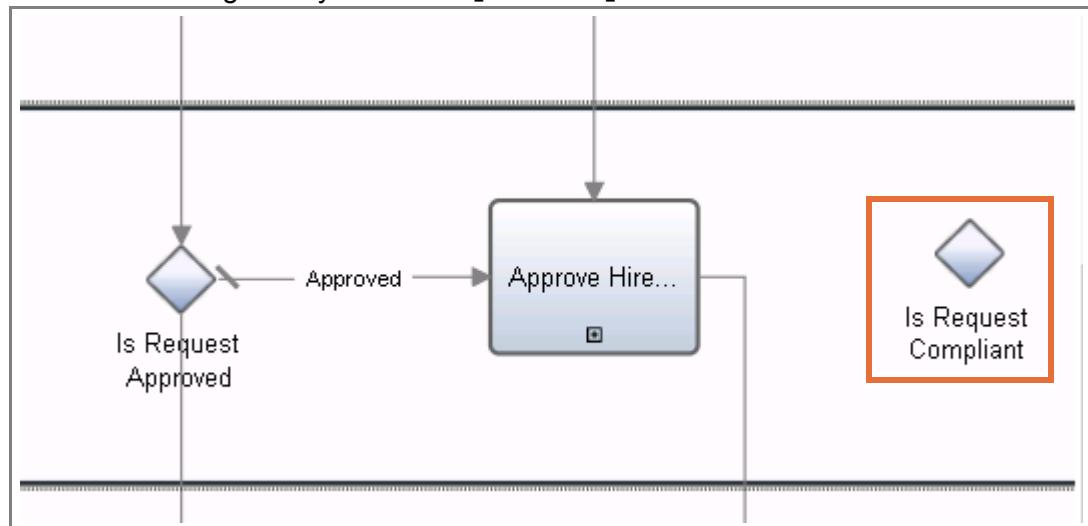
- ___ 2. Create an activity for a lawyer to review the hire request.
___ a. Drag a **User Task** from the palette to the **Legal** lane in the process.

- __ b. Name the activity: Review Posting

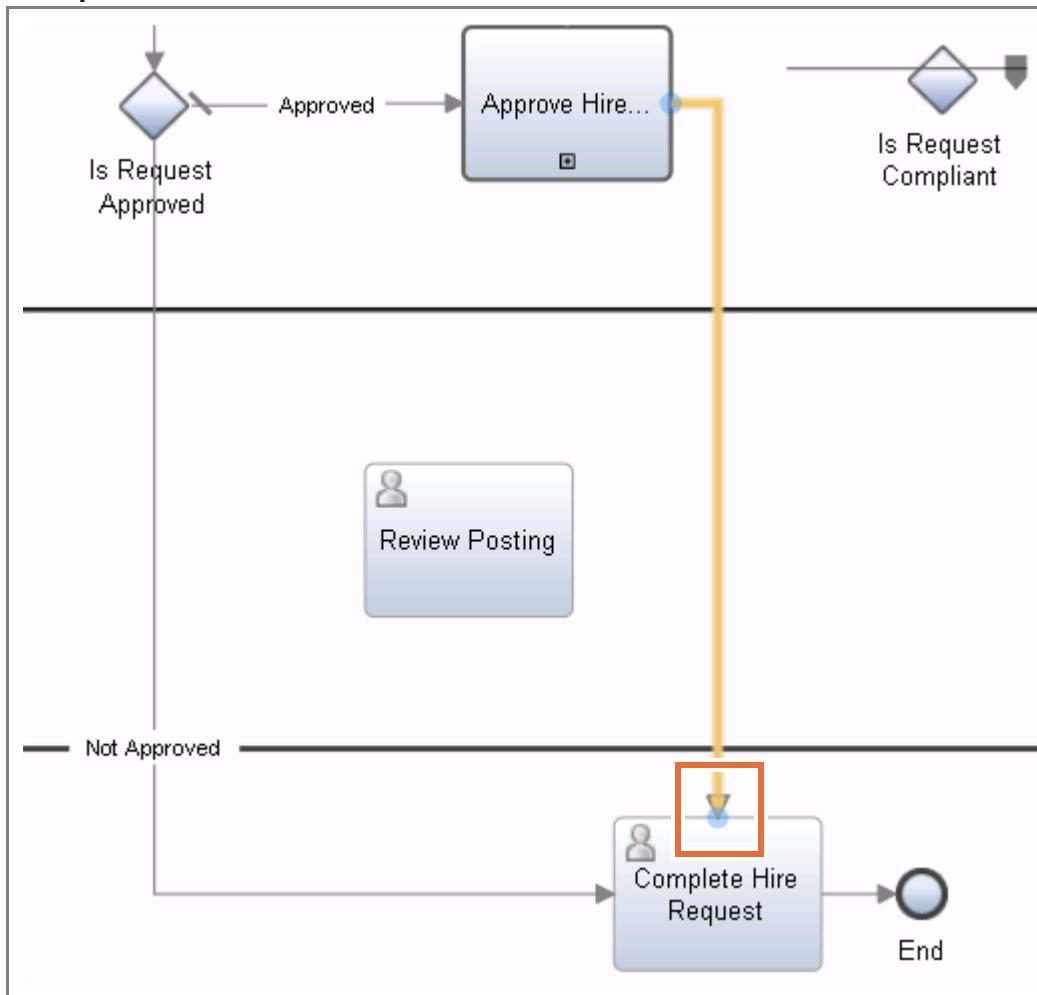


- __ 3. Add a gateway to check the request compliance.

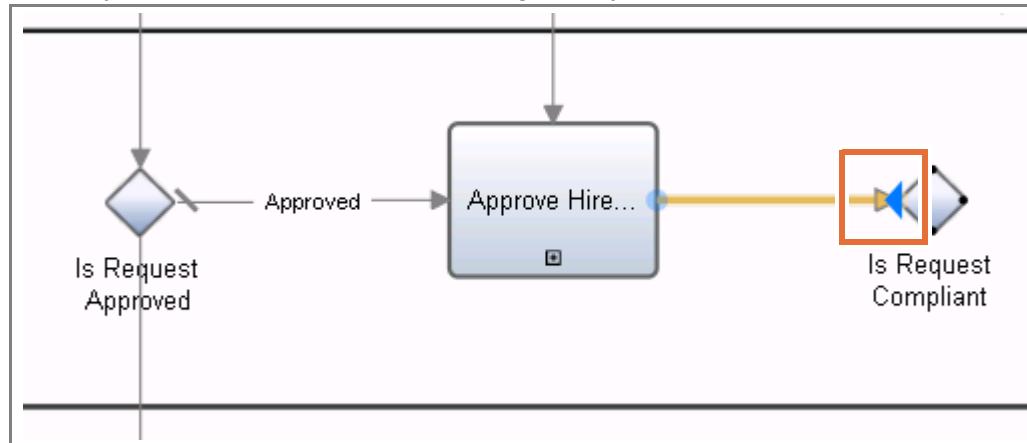
- __ a. Drag a **Gateway** to the right of **Approve Hire Request**.
 __ b. Rename the gateway to: **Is Request Compliant**



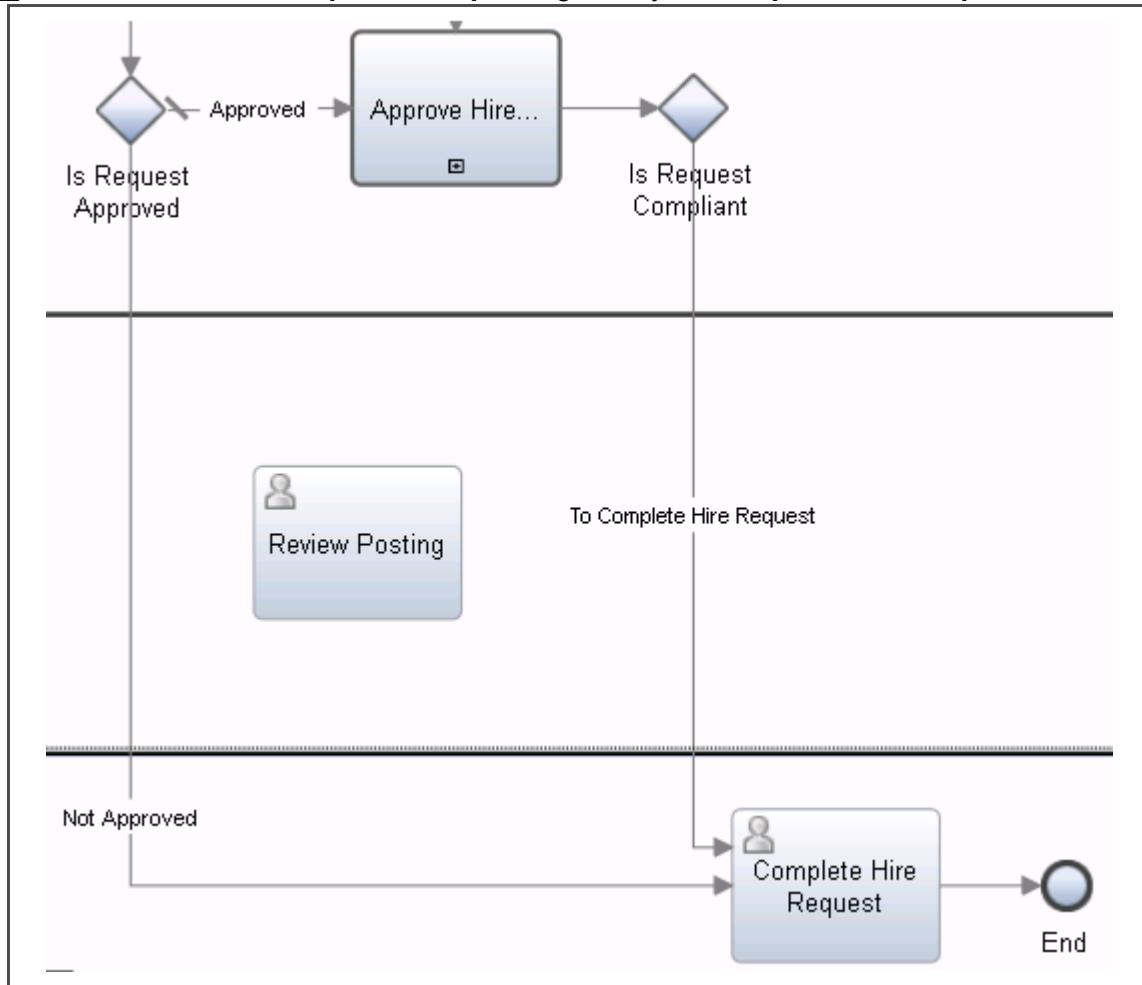
- 4. Connect the **Review Posting** activity.
- a. Select the sequence flow between **Approve Hire Request** and **Complete Hire Request** activities.



- b. Drag the blue point that appears at the tip of the flow line at the **Complete Hire Request** activity to the **Is Request Compliant** gateway.

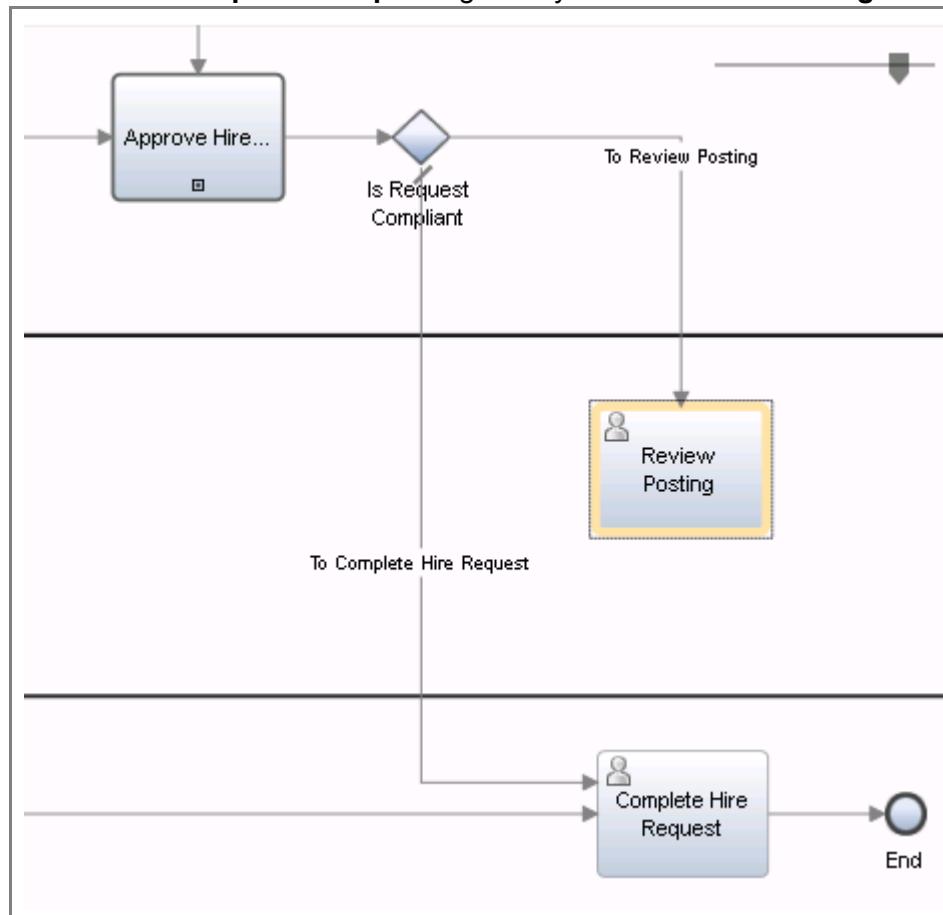


- c. Connect the **Is Request Compliant** gateway to **Complete Hire Request**.

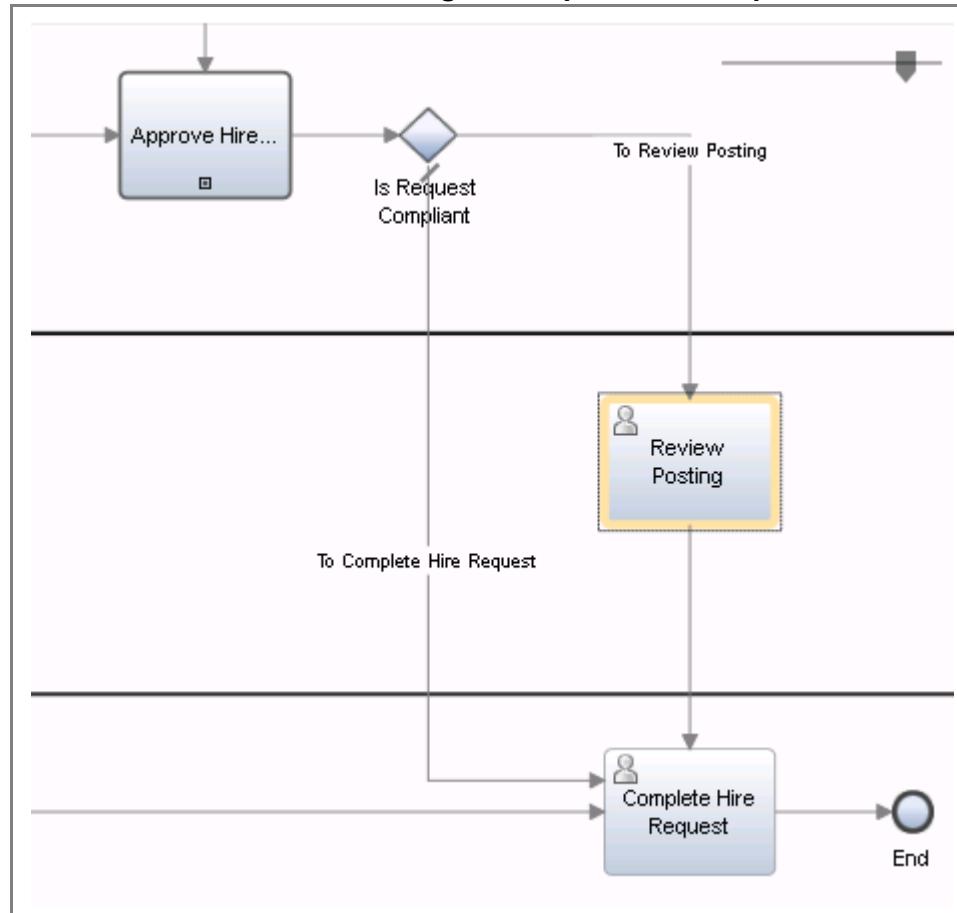


- d. Drag the **Review Posting** activity above the **Complete Hire Request** activity.

- __ e. Connect the **Is Request Compliant** gateway to the **Review Posting** activity.



- ___ f. Create a flow from **Review Posting** to **Complete Hire Request**.

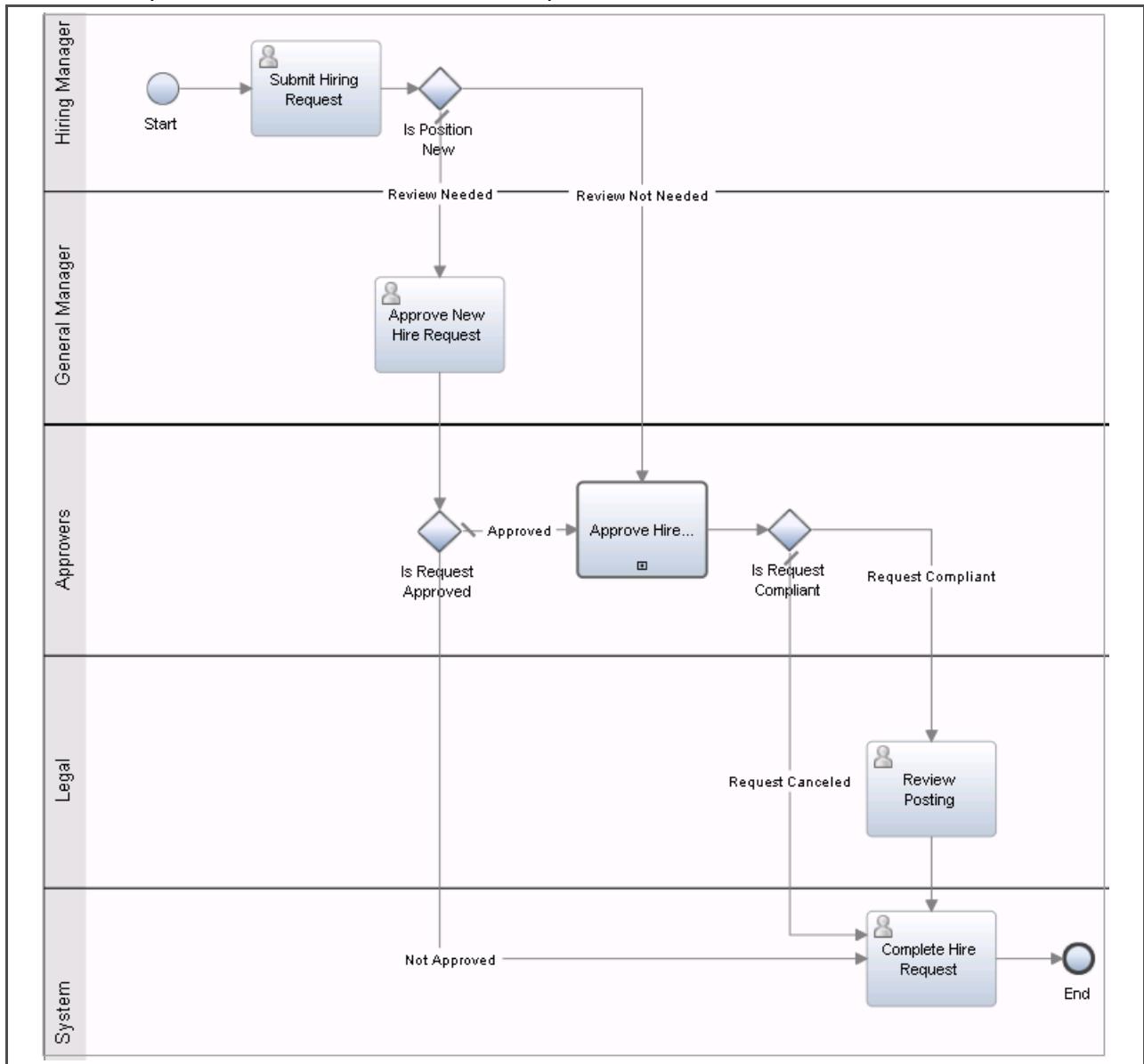


- ___ 5. Label the flow lines.

- ___ a. Rename the sequence flow between **Is Request Compliant** and **Complete Hire Request** as: Request Canceled
- ___ b. Rename the sequence flow between **Is Request Compliant** and **Review Posting** as: Request Compliant

**Important**

Your final process should look similar to this process.



6. Save your work.

**Important**

After adding an element to your process, it is a good practice to verify the process a second time. When you encounter process problems while developing for the Playback 1 process phase, it is common to return to Playback 0 to correct the problems.

Playback 0 is complete.

If you need more information about conducting the playback, consult the Appendix A: Conducting playbacks for Playback 0.

End of exercise

Exercise review and wrap-up

In this exercise, you validated the business process, and you took the playback feedback and added a requirement to the process. All the stakeholders agreed to the revised version of the process, and the process owner along with the lead developer suggested that the team transition to Playback 1.

Exercise 5. Playback 1: Controlling process flow with business data

Estimated time

01:00

Overview

In this exercise, you create assets that are required during Playback 1 controlling the process flow of the process lifecycle. You create variables, implement timer intermediate events, establish routing, and implement exclusive gateways.

Objectives

After completing this exercise, you should be able to:

- Create simple variables in a process
- Implement timer intermediate events in a process
- Implement gateways for a process
- Implement routing for an activity

Introduction

In this part of Playback 1, controlling process flow, the process data model is implemented along with the appropriate process flow for the data. Flow data is different from business data in that flow data moves the process along from flow object to flow object. The most obvious examples of flow data are the data elements that decision points use in the process or service diagrams. When a token is at a decision point, the values of data elements are used to determine the next paths to take. Flow data also includes the following circumstances:

- Data that is used to determine which activities to run
- Data that is used to determine who starts each activity
- Data that is used to determine when an activity is due or when an activity must be escalated

The process flow data ensures that the business process gets the right activities to the right people at the right time.

This exercise can be tedious as it requires you to flip between many different things in the Process Designer. This exercise simulates the real-world development effort when you create your first Process Application after this course. Even though the exercise implements lots of things and maps

numerous variables, the result is a process that shows your business data flows through the process and the process data controls the process flow.

Requirements

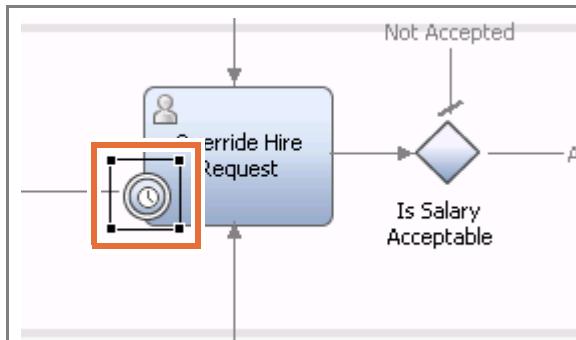
Successful completion of the previous exercise is required.

Exercise instructions

Part 1: Implement the intermediate timer event

In Playback 0 of the Hiring Request Process, an attached timer intermediate event was modeled. The HR Administrator has 4 hours to complete the review. If the review is not completed within 4 hours, an email is sent to the HR Administrator. The email notifies the HR Administrator of the missed deadline. Now, your responsibility is to implement the timer event that was modeled in the process.

- ___ 1. Implement the attached timer intermediate event.
 - ___ a. If it is not already open, open the **HR Recruitment Processes (HRR)** in the Process Designer and open the **Approve Hire Request** process.
 - ___ b. Return to the **Definition** tab.
 - ___ c. Click the **timer intermediate event** that is attached to the **Override Hire Request** activity.



- ___ d. Click **Properties > Implementation**.

___ e. Change the **Event Properties** to the following values:

- **Trigger on:** After start of step
- **Before or after difference:** 4 Hours
- **Tolerance interval:** 0 Hours

Properties Validation Errors

General

Implementation

Data Mapping

Pre & Post

Boundary Event Type

Timer

Interrupt activity:

Repeatable:

Event Properties

Trigger on: After start of step

Custom date:

Before or after difference: 4 Hours

Tolerance interval: 0 Hours

Use the activity work schedule:

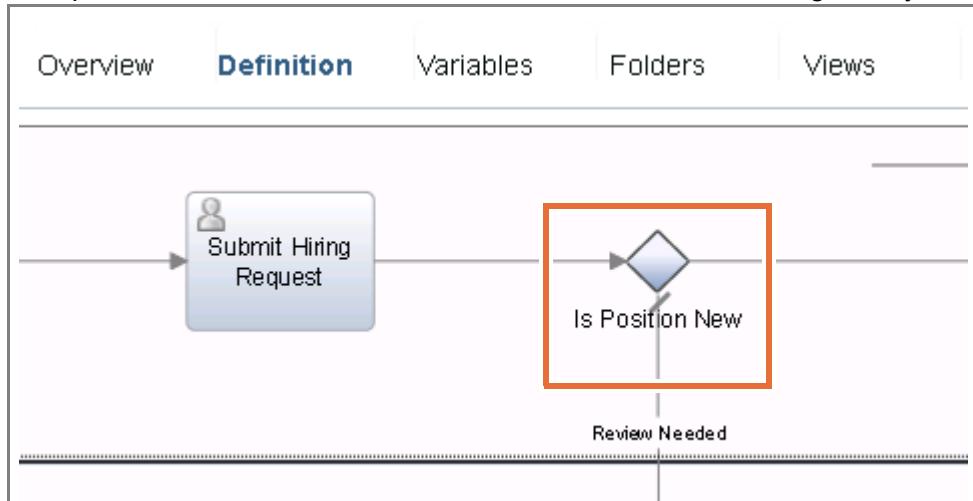
___ f. Save your work.

Part 2: Create the process flow variables for the Hiring Request Process

The process model for the Hiring Request Process is in place. Process data flow implementation is one of the goals of Playback 1. Now you must start to implement the process, first by creating simple variables to implement the logic behind the decision gateways on the current process.

___ 1. Open the **Hiring Request Process**.

- 2. Create a private variable to control the **Is Position New** exclusive gateway.



- a. Click the **Variables** tab.

The screenshot shows the "Variables" tab selected in the top navigation bar. The main area is titled "Variables" with a dropdown arrow icon. Below this, there are sections for "Input" (with a plus sign icon), "Output" (with a plus sign icon), and "Private" (with a plus sign icon). Under "Private", four variables are listed: "requisitionNumber (String)", "jobTitle (String)", "salary (Integer)", and "department (String)". To the right of these variables are three small square icons with arrows pointing up, down, and left respectively. At the bottom of the list is a section for "Exposed Process Variables" with a plus sign icon. On the far right, there are three small square icons with arrows pointing up, down, and left respectively.

- b. Click the (+) plus sign next to Private.

The screenshot shows the 'Variables' tab selected in a process editor. The interface includes tabs for Overview, Definition, Variables, Folders, and Views. Below the tabs is a section titled 'Variables' with a dropdown arrow. Underneath are sections for Input, Output, and Private. The Private section is expanded, showing four variables: requisitionNumber (String), jobTitle (String), salary (Integer), and department (String). To the right of the Private section is a vertical toolbar with several icons, including three blue '+' buttons. The middle blue '+' button is highlighted with a red box. Below the Private section is a section for Exposed Process Variables with its own blue '+' button.

- c. Name the private variable: isNewPosition
Confirm that the Variable Type is **String**.

The screenshot shows the 'Details' tab for a variable. The 'Name' field contains 'isNewPosition' and is highlighted with a red box. Below it is a rich text editor toolbar with buttons for bold (B), italic (I), underline (U), and alignment. The 'Documentation' field is empty. At the bottom of the tab, there are sections for 'Variable type', 'List', and 'Visible in Process Portal'. The 'Variable type' section shows 'String' with a small icon, and 'System Data' is listed below it. This entire section is also highlighted with a red box. There are 'Select...' and 'New...' buttons next to the type field.

**Important**

String variables are used here instead of Boolean for flexibility of the implementation. If requirements change later, and a third outflow is added to the gateway, a String is easier to implement the change than a Boolean would be.

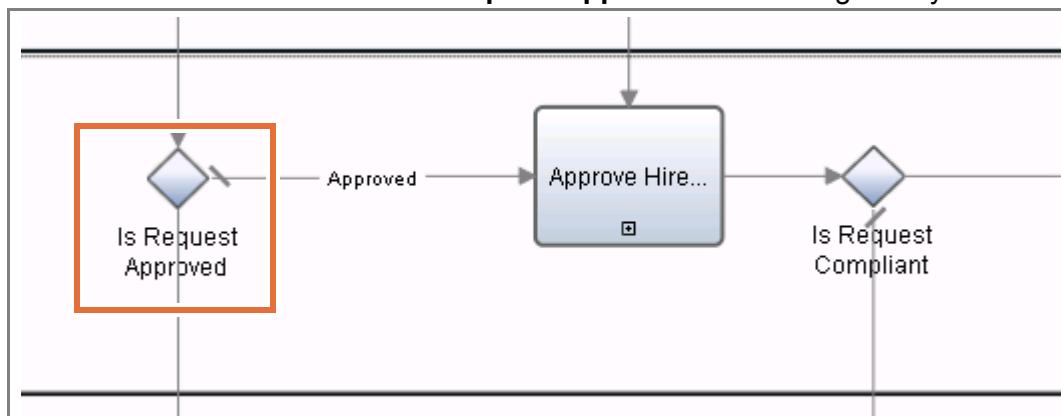
- ___ d. In the **Documentation** field, insert the following text:

0 = Not a new position
1 = New position, requires approval

Details

Name:	isNewPosition
Documentation:	<p>B I U ≡ ≡ ≡ ≡ ≡ ≡</p> <p>0 = Not a new position 1 = New position, requires approval</p>

- ___ e. Do not change the other options. Save your changes.
- ___ 3. Create a variable to control the **Is Request Approved** exclusive gateway.

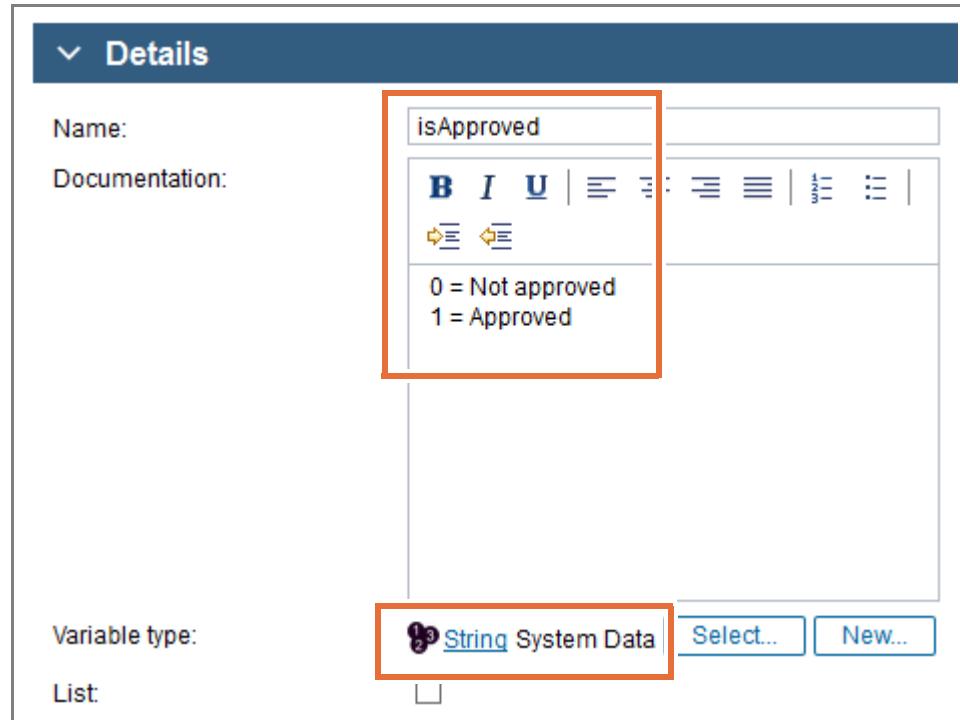


- ___ a. On the **Variables** tab, create another Private variable: `isApproved` (String)

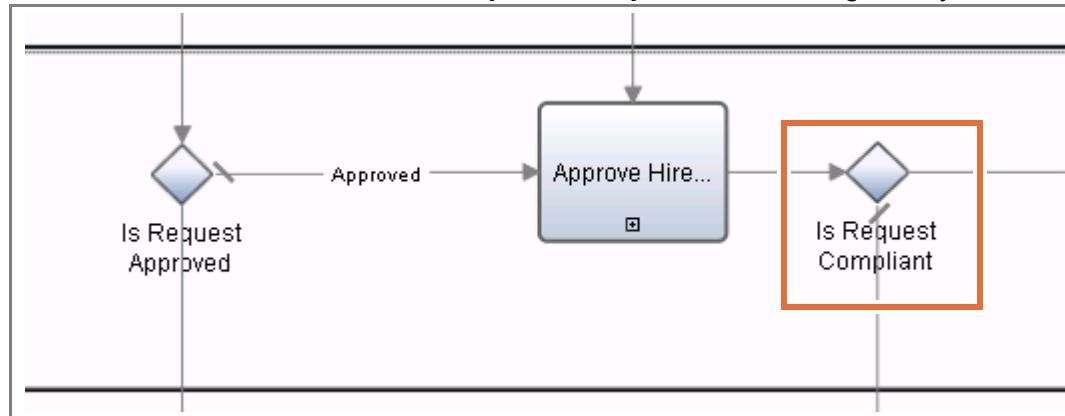
- __ b. In the **Documentation** field, insert the following text:

0 = Not approved

1 = Approved



- __ 4. Create a variable to control the **Is Request Compliant** exclusive gateway.



- __ a. On the **Variables** tab, create another Private variable: `isCompliant` (String)



Note

The `isCompliant` process flow variable maps to a variable in the Approve Hire Request linked process. The mapping occurs later in this lab.

- __ b. In the **Documentation** field, insert the following text:

0 = Hire request is canceled
1 = Hire request is compliant

The screenshot shows the 'Details' dialog box for a variable. The 'Name:' field contains 'isCompliant'. The 'Documentation:' field contains the text '0 = Hire request is canceled' and '1 = Hire request is compliant', which is highlighted with a red box. Below these fields is a rich text editor toolbar with bold (B), italic (I), underline (U), and alignment buttons. The 'Variable type:' field is set to 'String System Data', also highlighted with a red box. There are 'Select...' and 'New...' buttons next to it. The 'List' field has an empty checkbox.

- __ c. Save your changes.

You created three private variables in this exercise.

The screenshot shows the 'Variables' dialog box. On the left, there are categories: 'Input' (yellow icon), 'Output' (green icon), and 'Private' (blue icon, expanded). Under 'Private', there is a list of variables: 'requisitionNumber (String)', 'jobTitle (String)', 'salary (Integer)', 'department (String)', 'isNewPosition (String)', 'isApproved (String)', and 'isCompliant (String)'. The last three variables ('isNewPosition', 'isApproved', and 'isCompliant') are highlighted with a red box. On the right side of the dialog box are buttons for moving items up (+), down (-), and delete (X). At the bottom, there is a button for 'Exposed Process variables' and a '+' button.

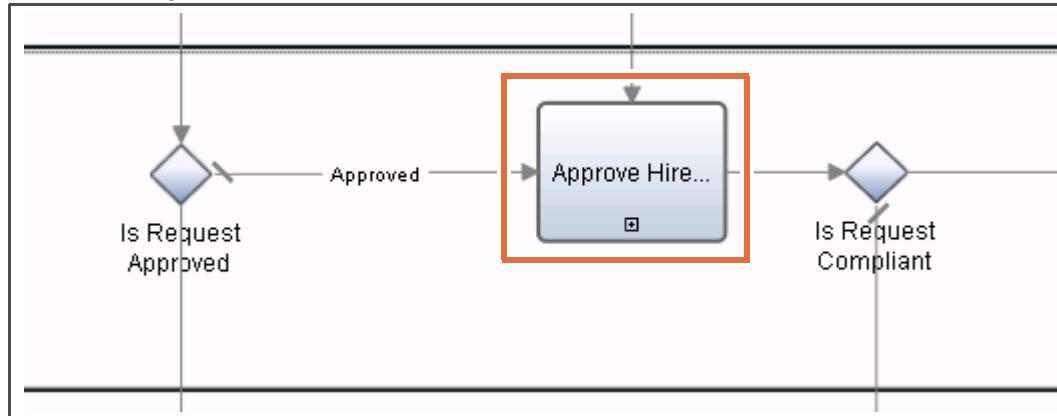
Part 3: Create a process flow variable for the Approve Hire Request linked process

- 1. Create a variable for the Is Salary Compliant exclusive gateway in the Approve Hire Request process.

- a. Return to the **Hiring Request Process** process by clicking the **Definition** tab.



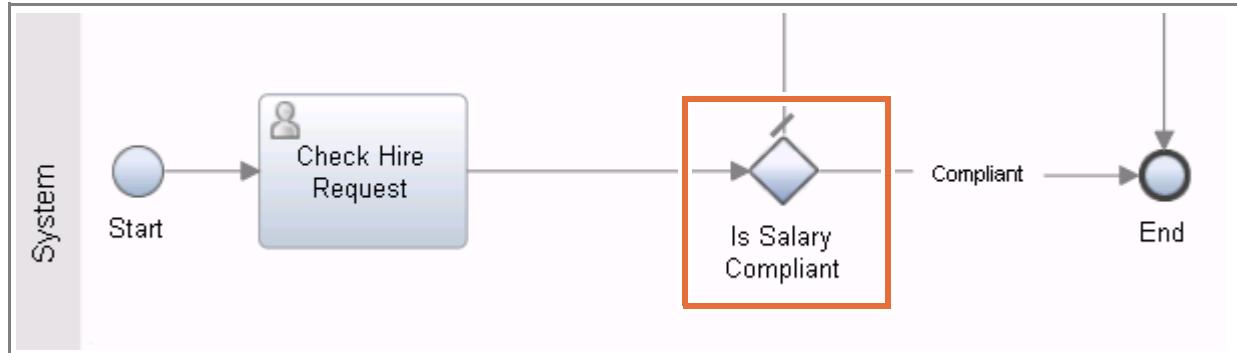
- b. Open the **Approve Hire Request** linked process by double-clicking the linked process in the diagram.



Important

Be sure that you select the **Approve Hire Request** linked process, not the **Approve New Hire Request** activity. Look for the linked process marker on the bottom center.

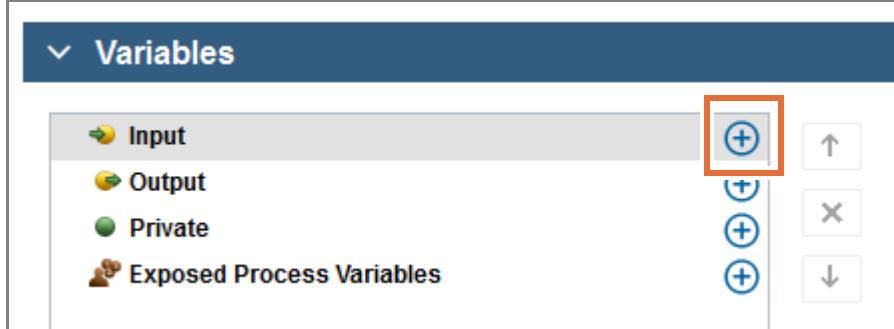
- c. Create a variable for the **Is Salary Compliant** exclusive gateway in the **Approve Hire Request** process. Because the parent process uses this data, create a variable that passes data through the linked process as an input and output variable.



- ___ d. Click the **Variables** tab for the **Approve Hire Request** linked process.



- ___ e. Click the (+) plus sign next to Input.



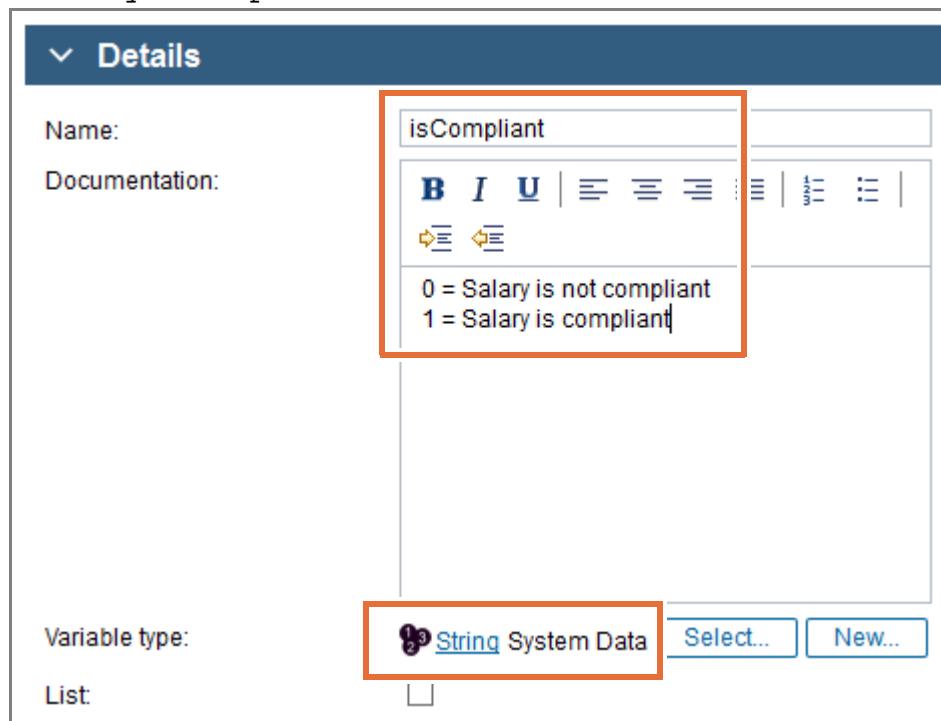
- ___ f. Name the input variable: `isCompliant`

Verify that the Variable Type is **String**.

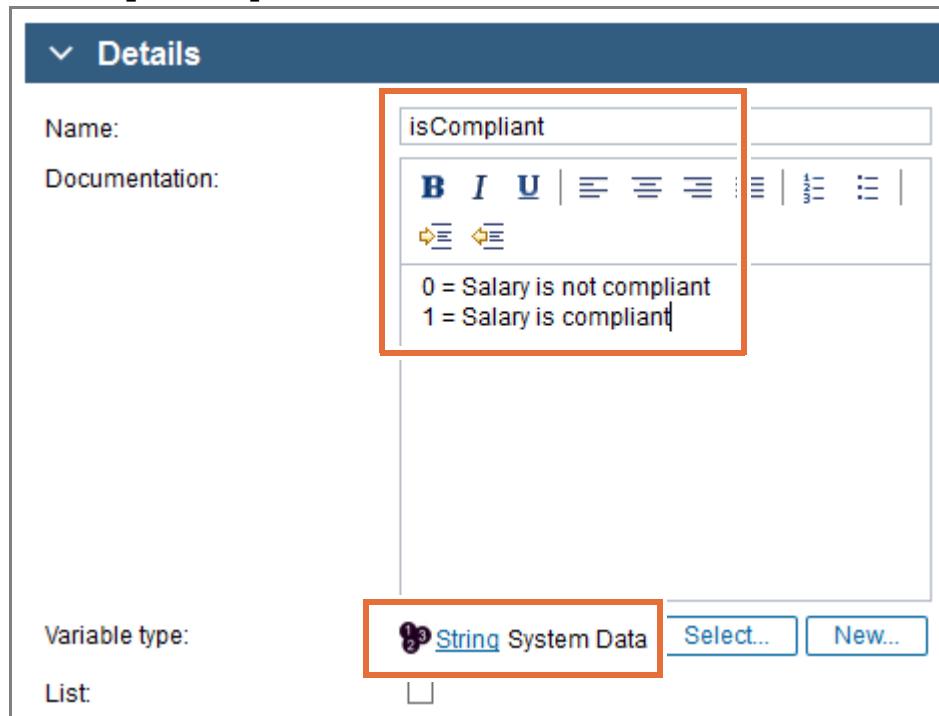
- ___ g. Insert the following text in the **Documentation** field:

0 = Salary is not compliant

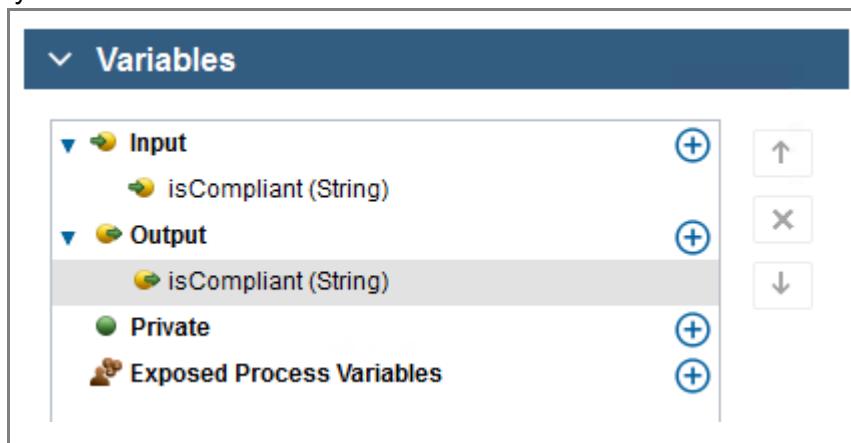
1 = Salary is compliant



- __ h. Create an output variable: `isCompliant` (String)
__ i. Insert the following text in the **Documentation** field:
0 = Salary is not compliant
1 = Salary is compliant



- __ j. Save your work.



Part 4: Implement gateways

When you modeled the process during Playback 0, you added several gateways to the process, and you must implement these gateways. In this section, you enhance the process by implementing the exclusive gateways in both the Hiring Request Process and the linked process. Use the simple variables that were created in a previous section to implement the gateways.

Implement the exclusive gateways in the Hiring Request Process process and the Approve Hire Request linked process to meet the following requirements:

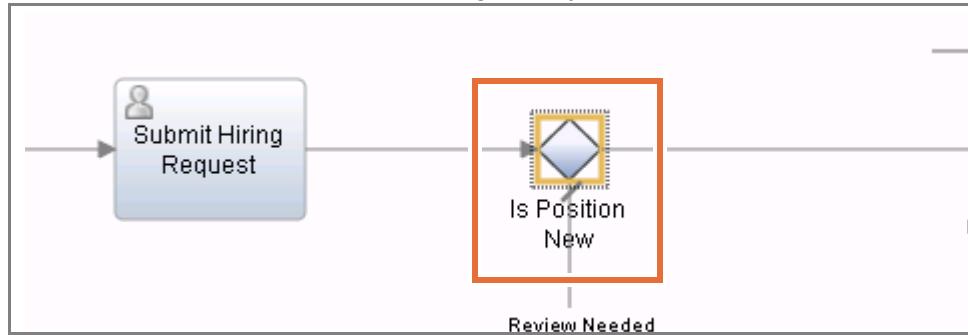
- If the answer to “Is Position New” is 1 (true), the request is forwarded to a General Manager. After the General Manager receives the request, the General Manager approves or rejects the request.
 - If the request is rejected, the system processes the request and the Hiring Request process is complete. If the request is approved, the process flows to the Approve Hire Request linked process.
 - In the linked process, after the requisition is submitted, an automated system-level check for salary compliance starts. If the request meets salary compliance, the hiring request is forwarded to Legal for review. If the request is canceled, it is automatically sent to the HR Positions database and processed.
1. Implement the **Is Position New** exclusive gateway.



Reminder

When the position is new, the process must flow to the Approve New Hire Request activity down the Review Needed flow.

- a. Open the **Hiring Request Process**.
- b. Switch to the **Definition** tab.
- c. Click the **Is Position New** exclusive gateway.



__ d. Open the **Properties > Implementation** menu.

The screenshot shows the 'Properties' dialog with the 'Validation Errors' tab selected. On the left, there is a vertical navigation bar with tabs: General, Decision, Implementation (which is highlighted with a red box), Pre & Post, and Tracking. To the right of this bar is a 'Decisions' section. Under 'Decisions', there are two dropdown menus: 'Review Not Needed:' and 'Default flow:'. The 'Default flow:' dropdown has an arrow icon indicating it can be expanded. Below the dropdowns are several small icons for file operations like copy, paste, and delete.

__ e. In the **Decisions** section, click the arrow to expand the **Default flow** selection box.

This screenshot is similar to the previous one, but the 'Default flow:' dropdown menu is now expanded, showing three options: 'Review Needed', 'Review Needed', and 'Review Not Needed'. The third option, 'Review Not Needed', is highlighted with a blue selection bar and a red box, indicating it is the current selection.

__ f. Select **Review Not Needed**.

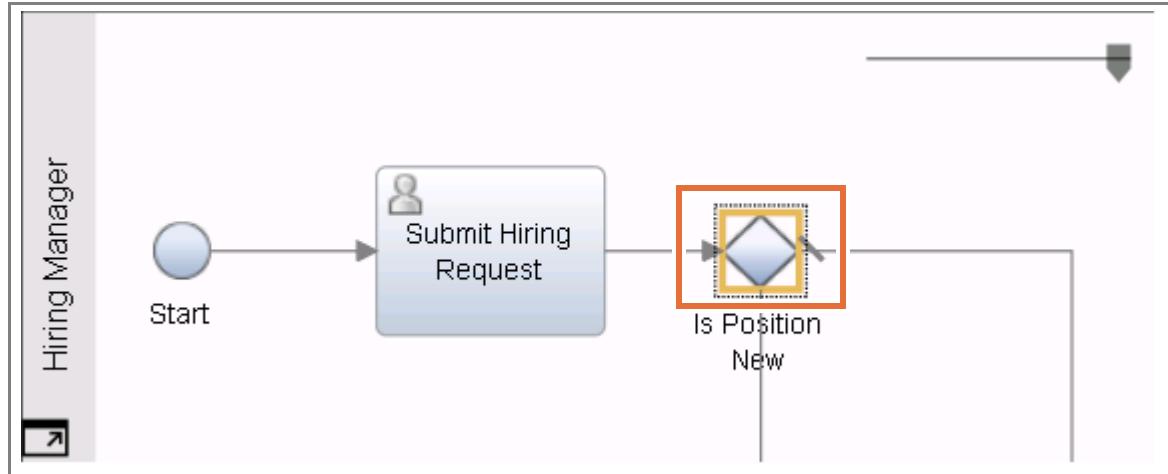
In this final screenshot, the 'Default flow:' dropdown menu is still expanded, showing the three options. The 'Review Not Needed' option at the bottom is now highlighted with a red box, indicating it has been selected.



Information

This selection makes Review Not Needed the default sequence flow. The order of sequence flow statements starts with Review Needed at the top and Review Not Needed on the bottom.

- g. Verify that the default flow is now changed for the **Is Position New** gateway.



- h. In the first field to the right of the **Review Needed** label, click the **Variable Picker** icon.

Properties Validation Errors

- General
- Decision
- Implementation**
- Pre & Post
- Tracking

▼ Decisions

Review Needed: NONE ▾

Default flow: Review Not Needed

- i. Select **isNewPosition (String)**. The field is populated with the variable:
`tw.local.isNewPosition`

Properties Validation Errors

- General
- Decision
- Implementation**
- Pre & Post
- Tracking

▼ Decisions

Review Needed: tw.local.isNewPosition ▾

Default flow: Review Not Needed

**Hint**

You can also type this variable and as you type and press Ctrl+Spacebar, the auto-complete feature suggests different options to you that match what you already typed. The options are filtered as you complete your entry.

The screenshot shows the 'Decisions' configuration window. In the 'Review Needed:' field, 'tw.local.' is typed. An auto-complete dropdown menu is open, listing several variables: 'requisitionNumber - String' (highlighted in yellow), 'jobTitle - String', 'salary - Integer', 'department - String', 'isNewPosition - String', 'isApproved - String', and 'isCompliant - String'. The 'Default flow:' field contains 'Review'.

The auto-complete window provides other contextual information on the object or object type. Use the mouse or the up and down arrows on the keyboard to select the correct option, and press Enter to select the option.

- j. Next, change the next select menu value to: ==
- k. In the last field to the right of **Review Needed**, type: "1"

The screenshot shows the 'Decisions' configuration window. The 'Review Needed:' field contains 'tw.local.isNewPosition == "1"'. The 'Default flow:' field contains 'Review Not Needed'. The condition part of the 'Review Needed' field is highlighted with a red rectangle.

**Important**

You must include the value in double quotation marks because `isNewPosition` is a String type of variable. Without the quotation marks, the system compares a String type to an Integer type and generates an error.

**Reminder**

The condition is set for the Is Position New gateway. If the first condition is not met, the default condition is Review Not Needed.

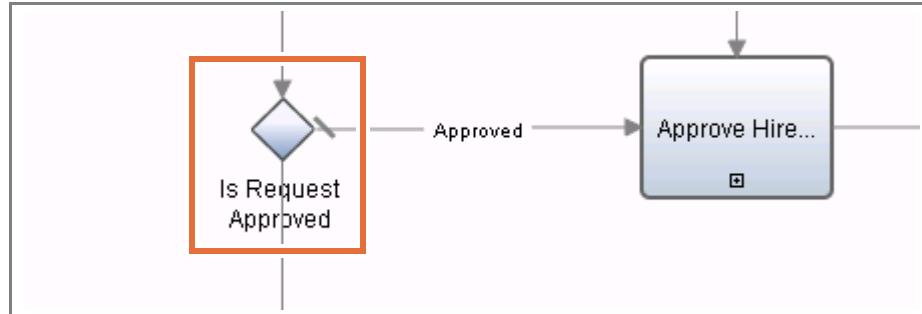
- __ 2. Implement the **Is Request Approved** exclusive gateway.



Reminder

When the request is approved, the process must flow to the Approve Hire Request linked process down the Approved flow.

- __ a. Click the **Is Request Approved** exclusive gateway on the process.



- __ b. Open the **Properties > Implementation** menu.

Properties Validation Errors

General	
Decision	
Implementation	Decisions Not Approved: [] [] Default flow: Approved
Pre & Post	
Tracking	

- __ c. In the **Decisions** section, expand the Default flow selector.
 __ d. Select **Not Approved**. This selection makes **Not Approved** the default sequence flow. The order of sequence flow starts with **Approved** at the top and **Not Approved** on the bottom.

Properties Validation Errors

General	
Decision	
Implementation	Decisions Approved: [] [] Default flow: Not Approved []
Pre & Post	
Tracking	

- __ e. In the first field to the right of **Approved**, map to:
`tw.local.isApproved`

- ___ f. Next, change the drop-down menu value to: ==
- ___ g. In the last field, type: "1"

The screenshot shows the 'Properties' dialog box for a validation rule. The left sidebar has tabs for General, Decision, Implementation, Pre & Post, and Tracking. The 'Decision' tab is active. Under the 'Decisions' section, there is a condition: 'Approved: tw.local.isApproved == "1"'. The 'Implementation' tab is also visible.

- ___ 3. Implement the **Is Request Compliant** exclusive gateway.



Reminder

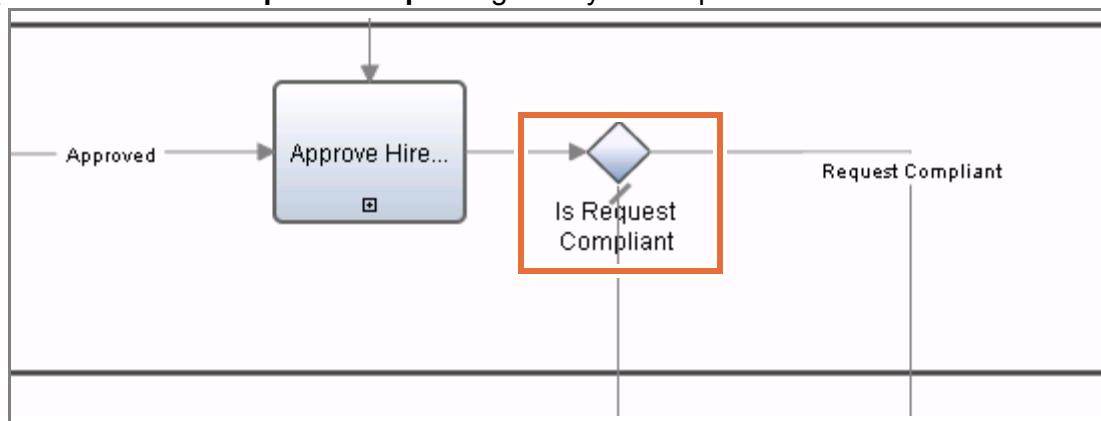
When the request is compliant, the process must flow to the Review Posting activity down the Request Compliant flow.



Note

The conditions for this gateway are set in a decision service in Playback 2. All the conditions that must be met are established as a business rule. The intent for this playback is merely to control the flow. For now, focus on implementing the gateway, not how the data is generated.

- ___ a. Click the **Is Request Compliant** gateway on the process.



- ___ b. Click the **Properties > Implementation** menu.

The screenshot shows the 'Properties' dialog with the 'Validation Errors' tab selected. On the left, there is a vertical navigation bar with tabs: General, Decision, Implementation (which is highlighted with a red box), Pre & Post, and Tracking. The main area is titled 'Decisions'. It contains two sections: 'Request Compliant:' and 'Default flow:'. The 'Request Compliant:' section has a dropdown menu set to 'NONE'. The 'Default flow:' section has a dropdown menu set to 'Request Canceled'.

- ___ c. In the **Implementation** section, expand the **Default flow** selector.
 ___ d. Select **Request Compliant**. This arrangement makes **Request Compliant** the default sequence flow. The order of sequence flow expresses that the **Request Canceled** option is first and that the **Request Compliant** option is second.

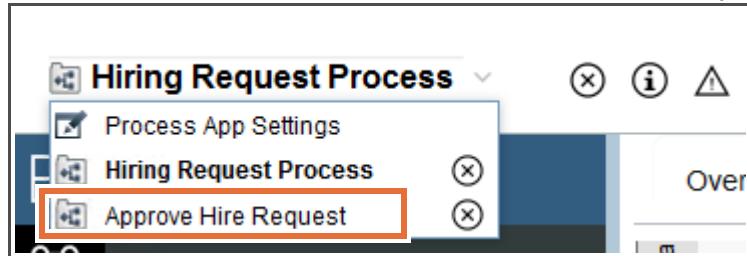
The screenshot shows the same 'Properties' dialog as before, but now the 'Default flow:' dropdown menu is expanded, revealing a list of options. The option 'Request Compliant' is highlighted with a red box.

- ___ e. In the first field to the right of **Compliant**, enter: `tw.local.isCompliant`
 ___ f. Change the drop-down value to: `==`
 ___ g. In the last field to the right of **Compliant**, type: `"0"`

The screenshot shows the 'Properties' dialog with the 'Validation Errors' tab selected. The 'Implementation' tab is selected. In the 'Default flow:' section, the dropdown menu is expanded, showing the expression `tw.local.isCompliant == "0"` entered into the text input field. The field to the right of the expression is set to `==`, and the field to the right of that is set to `"0"`.

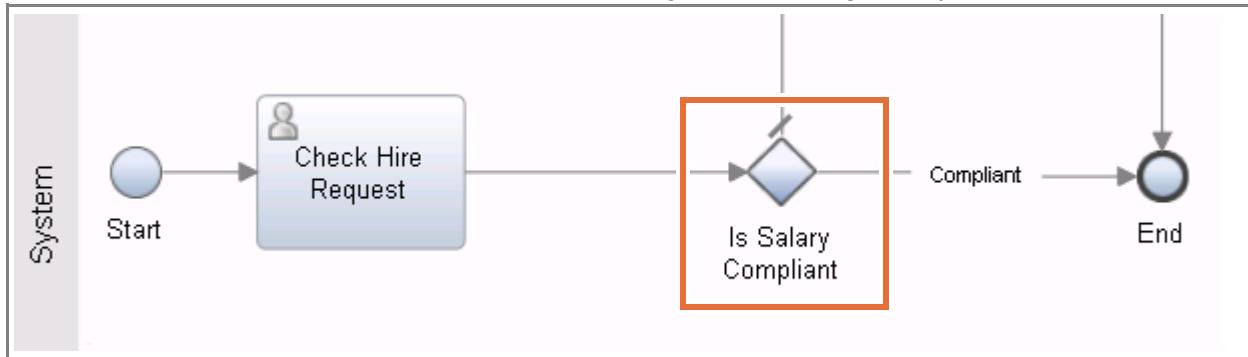
- ___ h. Save your work.

- 4. Implement the **Is Salary Compliant** exclusive gateway in the **Approve Hire Request** linked process. When the salary is compliant, the flow must move to the end event down the compliant flow.
- a. Open the **Approve Hire Request** linked process from the History menu.



If you don't see the process in your history menu, open the Approve Hire Request from the library on the left.

- b. On the **Definition** tab, select the **Is Salary Compliant** gateway.



Select the **Is Salary Compliant** gateway, not the **Is Salary Acceptable** gateway.

- c. Click the **Properties > Implementation** menu.
- d. In the **Decisions** section, open the Default flow menu.
- e. Select **Not Compliant**, if not selected.



Information

This arrangement makes Not Compliant the default sequence flow. The order of sequence flow in the Implementation section ensures that the Compliant option is assessed first, and if the first assessment is false, the Not Compliant option is taken.

- f. In the first field to the right of **Compliant**, enter: `tw.local.isCompliant`

- __ g. Change the drop-down value to: ==
- __ h. In the last field to the right of **Compliant**, type: "1"

The screenshot shows the SAP Business Data Services Properties dialog. The left sidebar has tabs: General, Decision, Implementation (which is selected), Pre & Post, and Tracking. Under the Implementation tab, there is a 'Decisions' section. Inside this section, there is a row for 'Compliant' with the expression 'tw.local.isCompliant' and an operator '==' followed by the value '1'. A red box highlights this row. Below it, there is another row for 'Default flow' with the value 'Not Compliant'.

- __ i. Save your work.



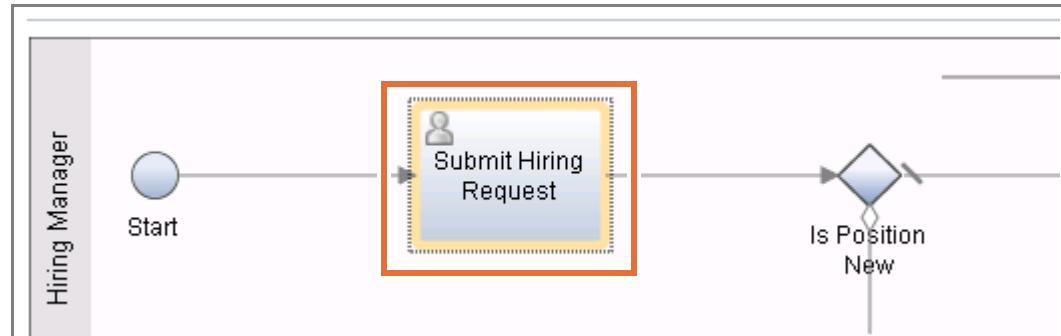
Optional

The challenge exercise in the Appendix is to create the variables and implement the rest of the gateways on the subprocess.

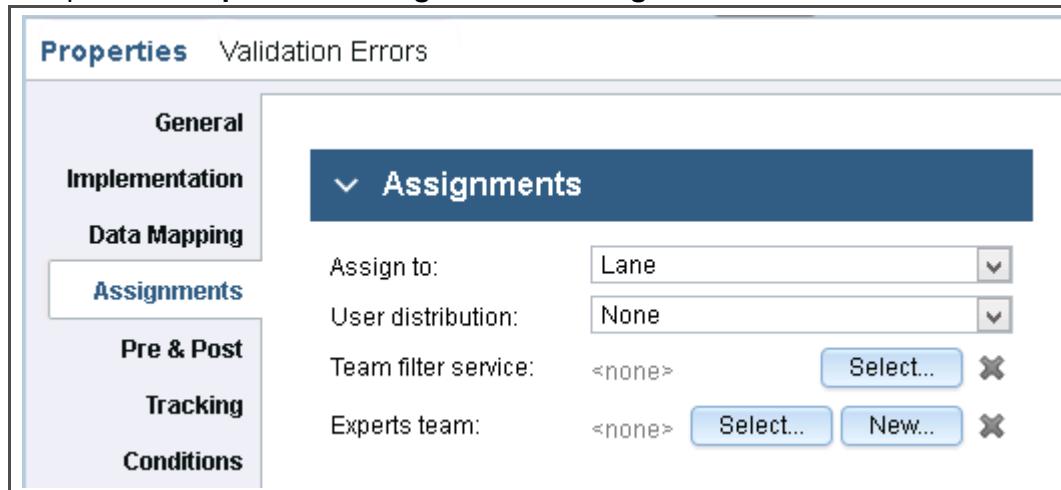
Part 5: Implement routing for an activity

Effective business process management is about getting the right task to the right teams at the right time. Using IBM Business Process Manager, authors use lanes (sometimes known as swimlanes for a pool) to identify the process teams. Process developers implement routing in the process for each activity to the right team in the lane. The goal is to implement the teams for the lanes and implement assignment routing for all activities in the process.

- ___ 1. Route the Submit Hiring Request activity in the Hiring Request Process to **Lane** and distribution to **Last User**.
 - ___ a. Open the **Hiring Request Process**.
 - ___ b. On the **Definition** tab, click the **Submit Hiring Request** activity on the **Hiring Request Process**.

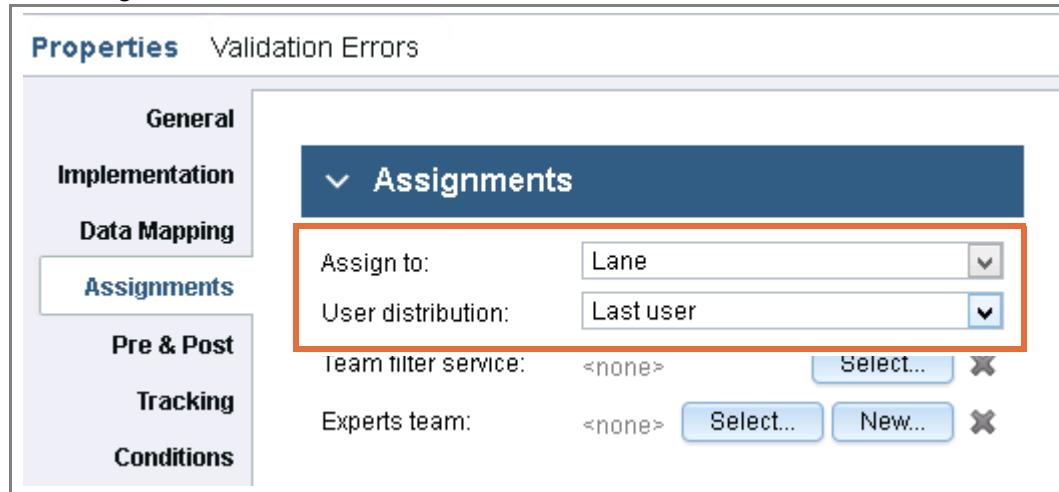


- ___ c. Open the **Properties > Assignments > Assignments** section.



- ___ d. Verify that the default assignment for **Assign to** is: Lane

- e. Change the **User distribution** to: Last user



Information

The Last user option assigns the runtime task to a user who completed an earlier task.

- If the assignment option is set to Lane, the task is assigned to the user who completed the activity that immediately precedes the selected activity in the lane.
- If the assignment option is set to Team, the task is assigned to the user who completed the last task that was assigned to the same team.

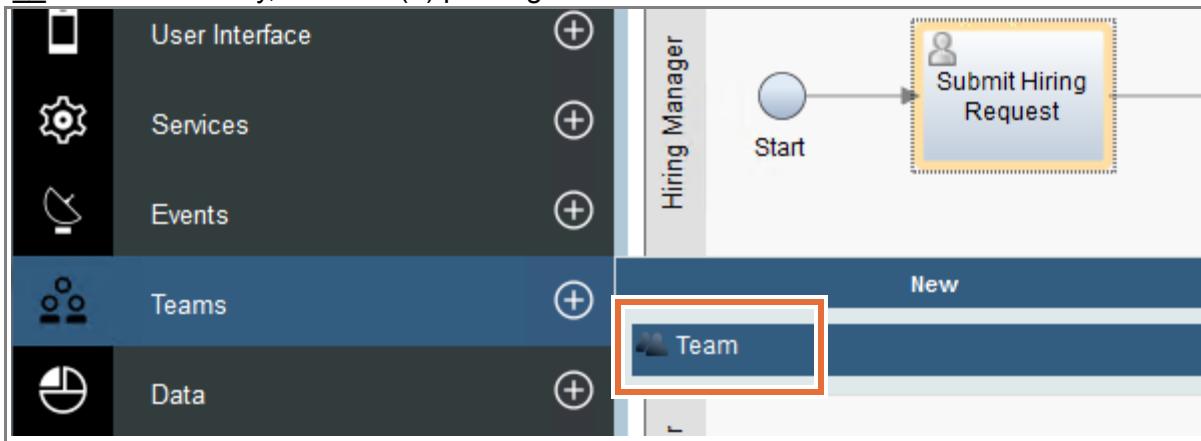
In both cases, the task is assigned only to an established last user, if the user is a member of the group that is associated with the task. An established last user is determined by using the policy valid for the Lane or Team assignment option. If the user is not a member of the group that is associated with the task, then the task is assigned to the task group.

Because this activity is the first in this lane for this process, the system automatically assigns the first task to the user that creates the instance. When a user creates an instance of this process, the server displays any screens that are configured for the task. You create those screens in the next playback phase.

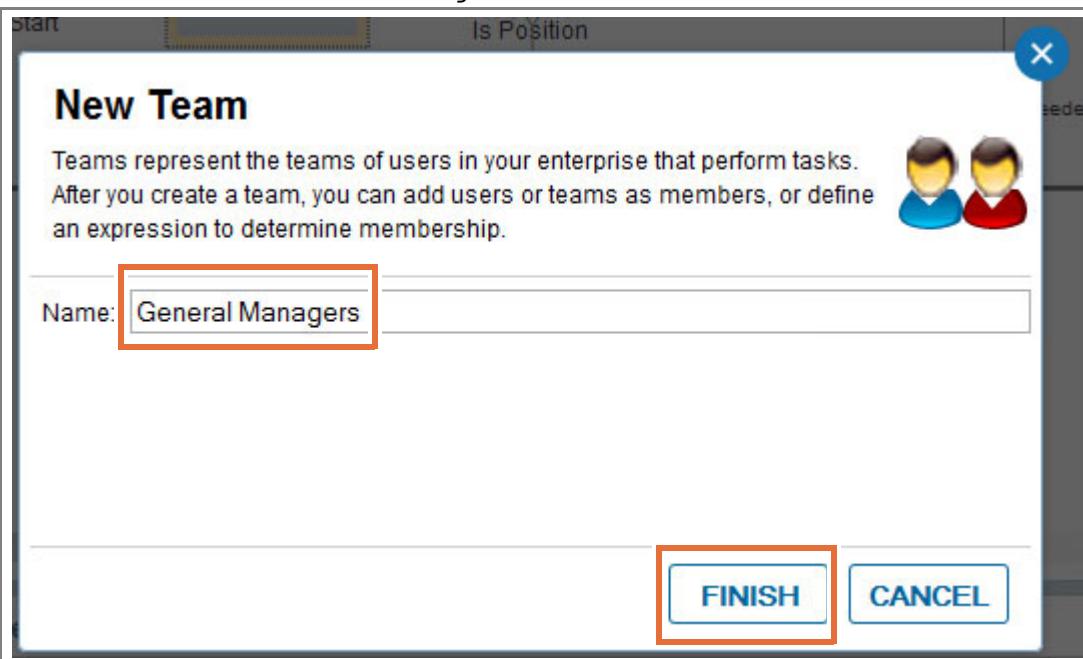
- f. Save your changes.

2. Create a team and add a member.

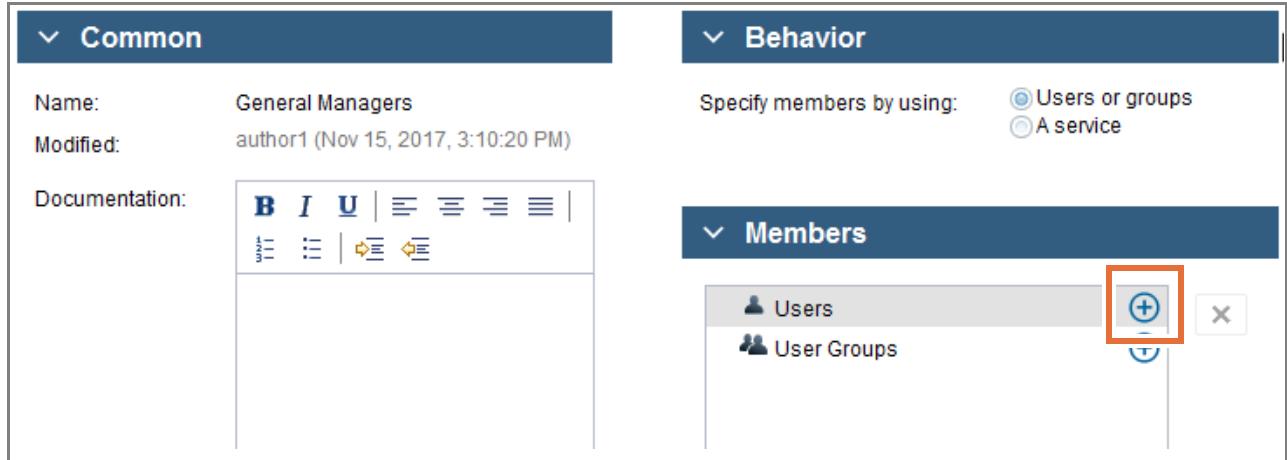
a. In the library, click the (+) plus sign next to **Teams**. Click **Team**.



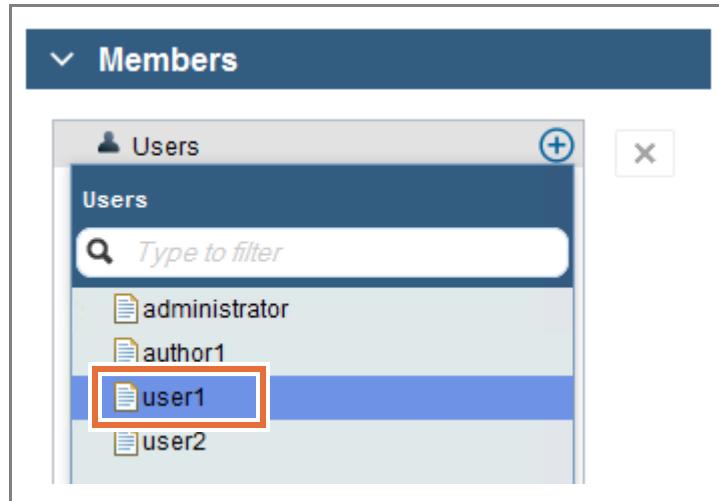
b. Name the team **General Managers** and click **Finish**.



c. In the **Members** section, click the (+) plus sign next to **Users**.



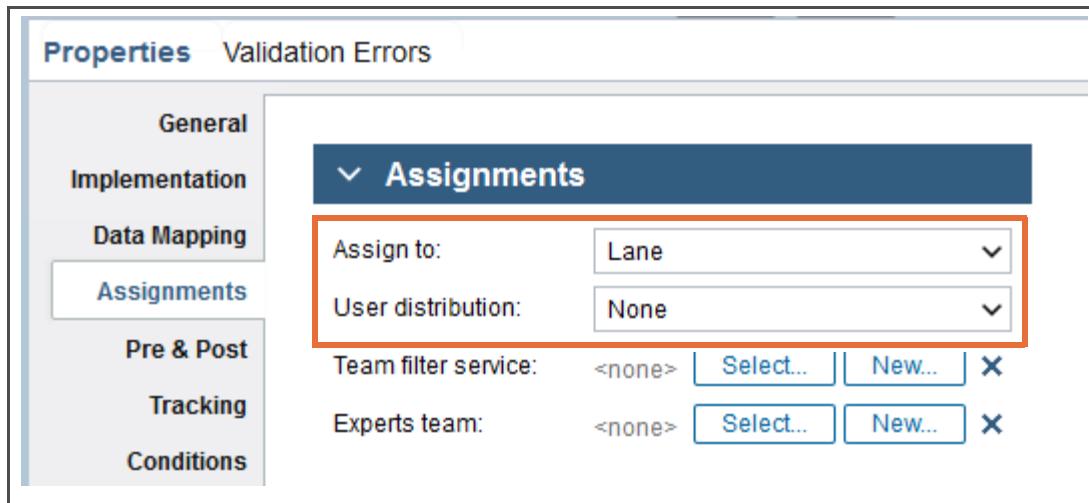
- __ d. Add **user1** to the team from the list.



You now have a populated list in the Members section with user1 for the General Managers team.



- __ e. Save your work.
- __ 3. Implement the routing for **Approve New Hire Request**.
- __ a. Open the **Hiring Request Process**.
 - __ b. On the **Definition** tab, click the **Approve New Hire Request** activity.
 - __ c. In the **Properties > Assignments > Assignments** section, leave the value of **Assign to:** as **Lane** and the value of **User distribution** as **None**.

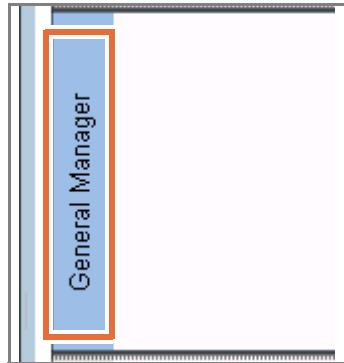




Information

The default **User distribution** option is **None**. IBM Business Process Manager does not assign the task to any user when using this option. All the other options in this selection assign the task to a user of the system.

- ___ d. On the General Manager lane, click the **label** on the left.

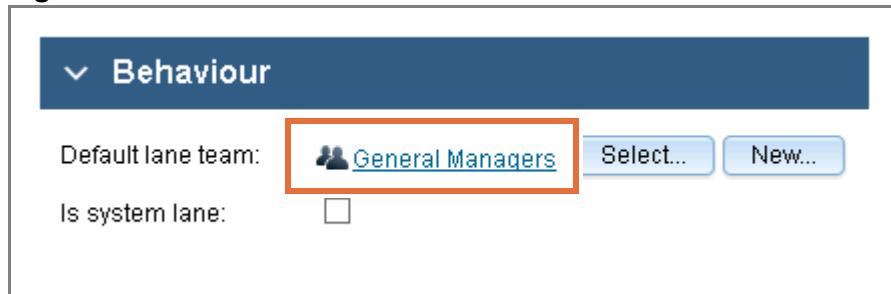


- ___ e. From the **Properties > General > Behavior** section, click **Select** next to Default lane team.

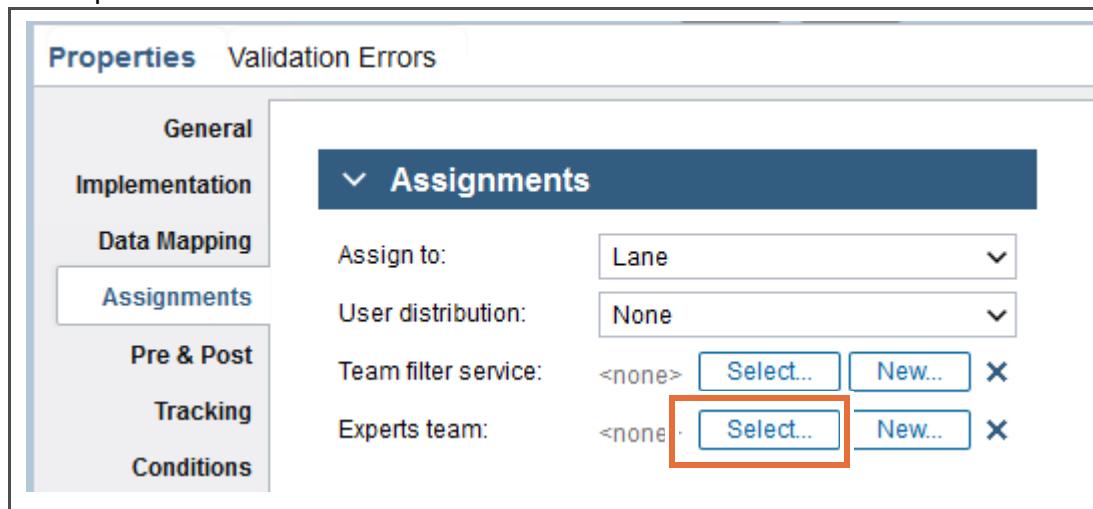
- ___ f. Click the **General Managers** team to select it.

Type	Team	Count
	All Users System Data	7
	Background System Data	
	General Managers	
	Managers System Data	
	Managers of All Users System Data	
	Process Owner System Data	
	System System Data	

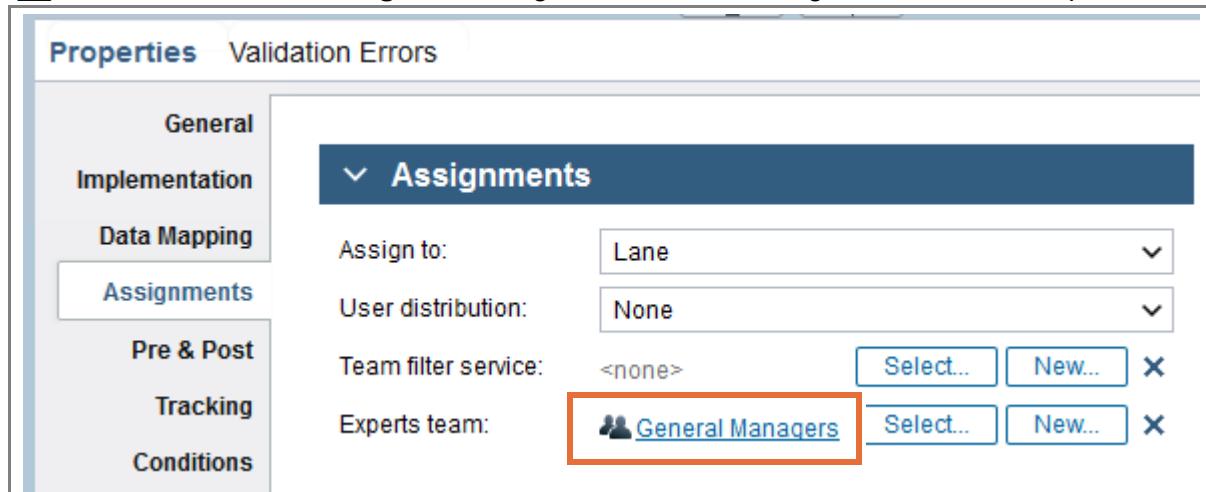
The **General Managers** team is assigned as the default team for the **General Managers** lane.



- 4. Assign the **General Managers** team as the **Experts Group** for the Approve New Hire Request activity.
 - a. On the **Hiring Request Process**, click the **Approve New Hire Request** activity.
 - b. Open the **Properties > Assignments > Assignments** section. Click **Select...** next to the Experts team.



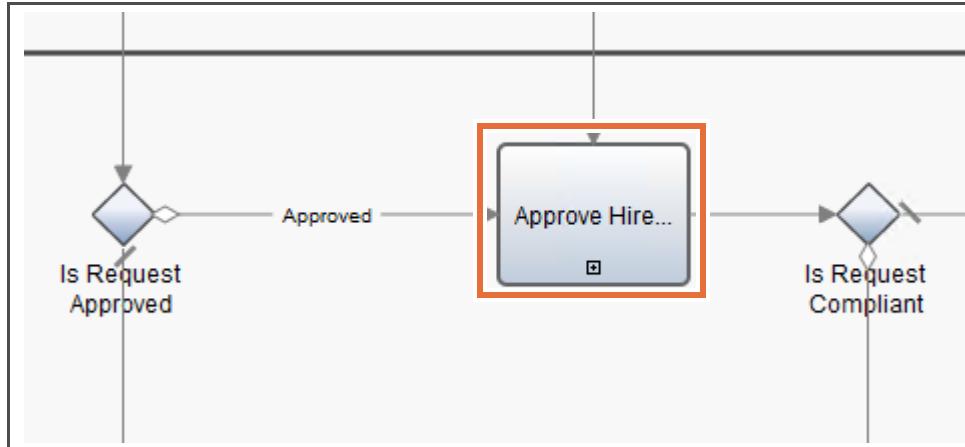
- c. Click **General Managers** to assign the General Managers team as the Experts team.



- 5. Save your work.

Part 6: Mapping variables

- 1. Map the process flow variable to the Approve Hire Request linked process.
 - a. On the **Hiring Request Process**, click the **Approve Hire Request** linked process.



- b. Click the **Properties > Data Mapping** menu.
- c. Enter `tw.local.isCompliant` for both the Input Mapping and Output Mapping variables to map the `tw.local.isCompliant` variable from the parent to the linked process.

Input Mapping	↔	Output Mapping
<code>tw.local.isCompliant</code>		<code>isComplian...</code> <code>tw.local.isCompliant</code>



Hint

You can also use the **variable picker** to select the variable `isCompliant` (`String`). The field is populated with the variable `tw.local.isCompliant`.

The screenshot shows the 'Output Mapping' dialog box. On the left, there is a placeholder icon with the text 'isComplian...'. In the center, there is a list of variables with their types: requisitionNumber (String), jobTitle (String), salary (Integer), department (String), isNewPosition (String), isApproved (String), and isCompliant (String). The 'isCompliant' variable is highlighted with a red box. On the right, there is another placeholder icon with the text 'isComplian...'.

- ___ d. Save your work.



Information

For more information about conducting the playback, see the Appendix A: Conducting Playbacks for Playback 1.

End of exercise

Exercise review and wrap-up

This exercise looked at setting variables in business processes, implementing an intermediate timer event on a process, implementing exclusive decision gateways, and establishing routing for an activity.

The assets for Playback 1: Controlling process flow are complete. You now can control gateways by assigning values to variables inside the process. The next exercise looks at Playback 1: Business data and services to define variables and create a reusable client-side human service.

Exercise 6. Playback 1: Business data, services, and coaches

Estimated time

01:30

Overview

In this exercise, by using the core requirements, you determine and create all of the necessary assets to support a coach in the Hiring Request Process. You use complex business objects (variable types) to organize your data, and pass data into and out of each diagram when you have nested processes. You build a service and define guided user interactions with a coach. You also implement a service for an activity, and map variables between a nested service and an activity. You model the coach by using the concept of grids.

Objectives

After completing this exercise, you should be able to:

- Determine and organize data when provided with a written process
- Add business objects and object types
- Create a client-side human service
- Add variables and business objects to a process application
- Create and configure a coach to obtain process participant input
- Model a coach by using the concept of grids
- Add coach controls to control process flow
- Create a client-side human service and coach for the General Manager review activity
- Implement an activity by attaching a service and mapping data

Introduction

In Playback 1: Business data and services, you create human services. Users can use human services to access task assignments in an activity-centered web form. At this stage of development, it is important that the functions of the human service are implemented. Users interact with web-based interfaces called coaches to complete their assigned tasks. Enhancements can be added later. This exercise is about making sure that users have what they need in terms of business data and task assignment information to complete the process activity.

The human services with coaches are implemented on two activities in the Hiring Request Process process. The coaches are used during playbacks to provide business data to the process and control the process flow.

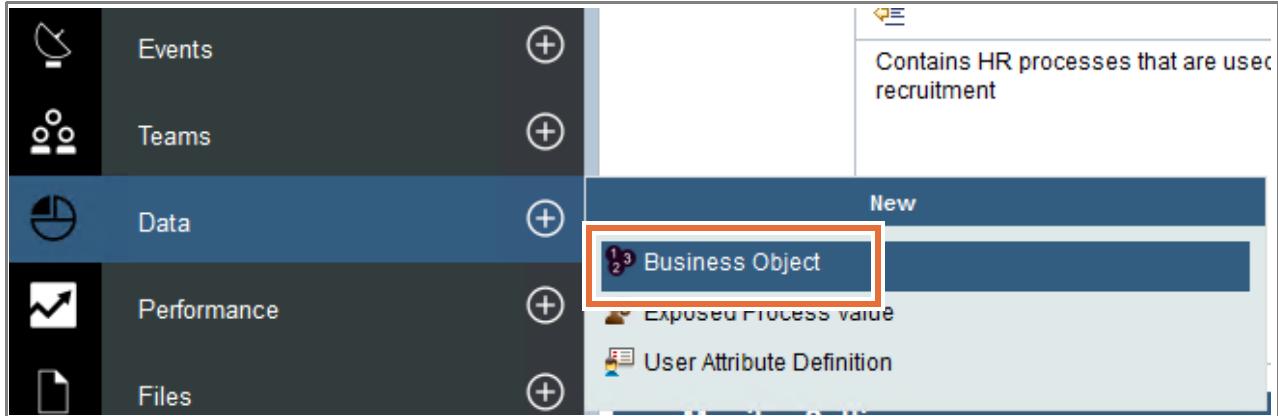
Requirements

Successful completion of the previous exercise is required.

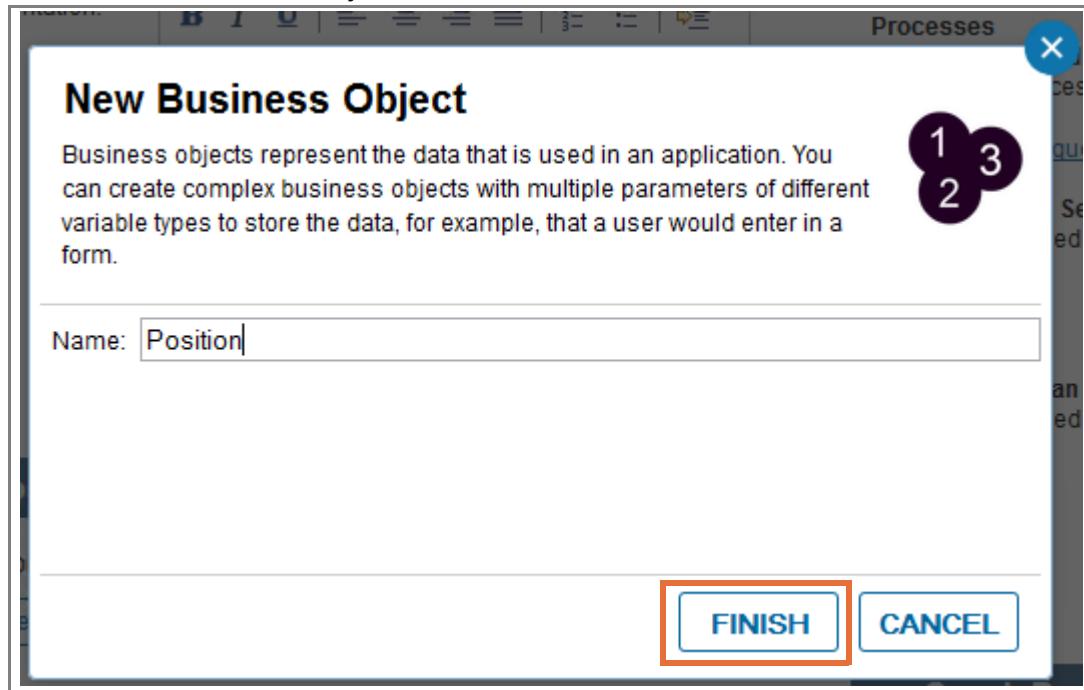
Exercise instructions

Part 1: Build business objects and variables

- 1. Create a complex variable to hold the position data. Use this variable on the coach to allow users to input the position data. Later in the lab, you map these variables to move the data throughout the process.
- a. In the IBM Process Designer, open the **HR Recruitment Processes** process app.
- b. In the library, click the **(+)** plus sign next to the Data category, and select **Business Object**.



- c. Name the business object **Position** and click **Finish**.





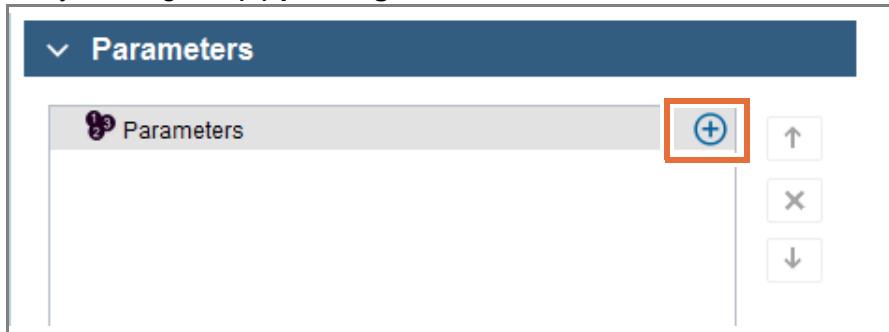
Note

Always follow the naming convention for the variable names. Variable names are case-sensitive.

Capitalize the first letter when creating business objects (for example, Employee), but use camel case for the parameters or instantiation of the variable (for example, employeeId).

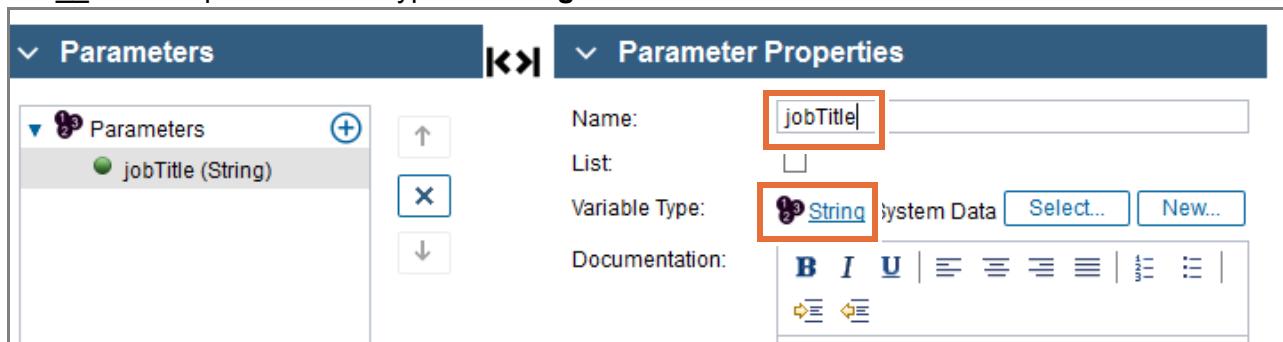
Camel case refers to a word or string of letters that has no space and has an uppercase letter in a position other than the first letter.

- __ d. You now see the settings page for the business object. Add a parameter to the business object by clicking the **(+)** plus sign next to Parameters.



- __ e. Change the name of the parameter to: jobTitle

- __ f. Keep the variable type as **String**.

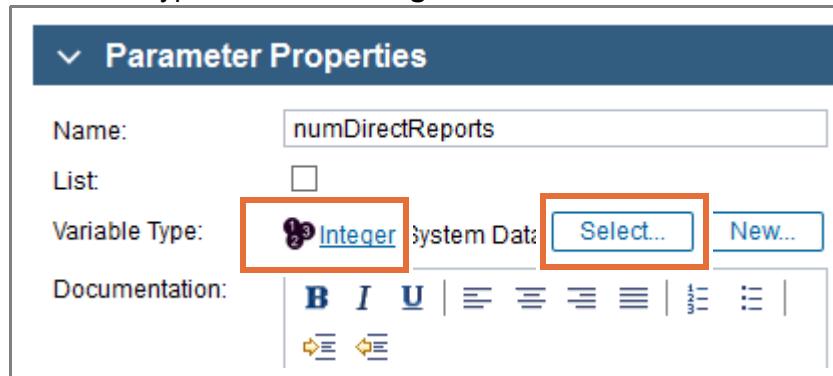


- q. Repeat steps 1d – 1f to add the following parameters:

- jobDescription (String)
 - jobLevel (String)

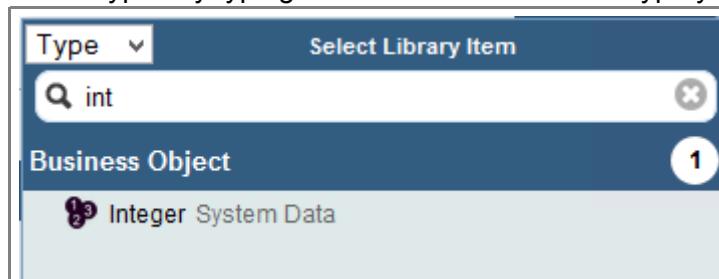
- h. Add another parameter that is named: numDirectReports

- __ i. Change the type of the numDirectReports attribute to an Integer. Click **Select** to the right of Variable Type and select **Integer**.

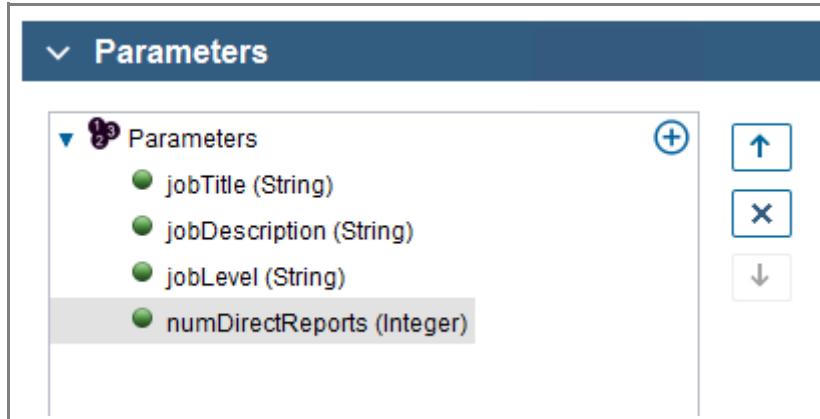


Hint

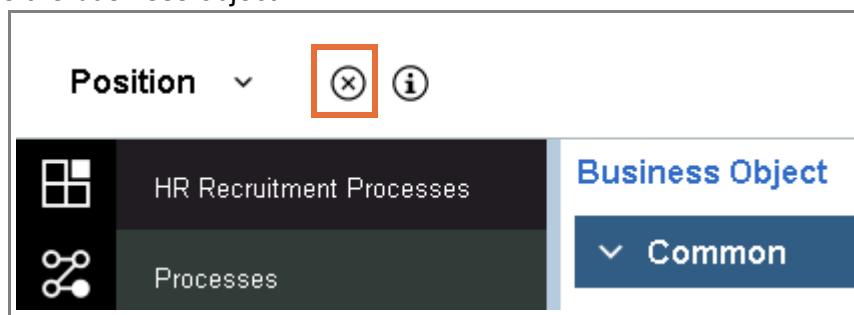
You can filter the variable types by typing the letters of the variable type you are looking for.



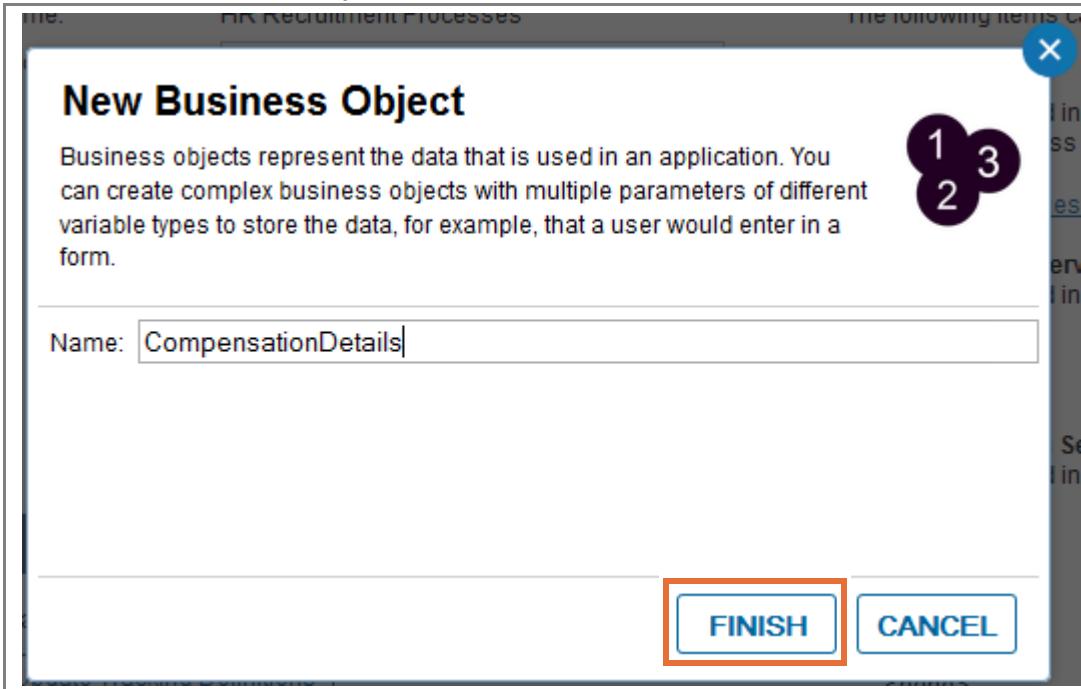
- __ j. Save your process application.



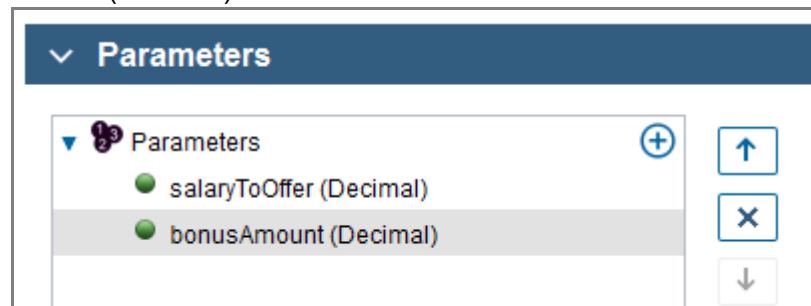
- __ k. Close the business object.



- __ 2. Create a second business object that is named `CompensationDetails` to allow users to input compensation data on a coach.
 - __ a. Click the **(+)** plus sign next to the Data category in the library, and select **Business Object**.
 - __ b. Name the business object `CompensationDetails` and click **Finish**.

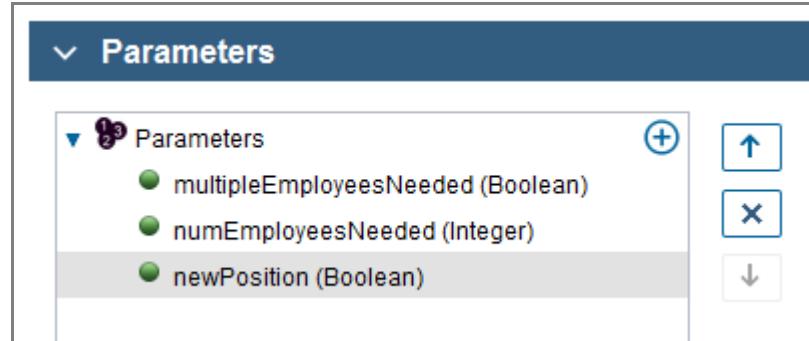


- __ c. Add the following parameters for the complex variable `CompensationDetails`. Remember to change the variable type by clicking **Select** to the right of **Variable Type** to change it to Decimal rather than the default (String):
 - `salaryToOffer` (Decimal)
 - `bonusAmount` (Decimal)

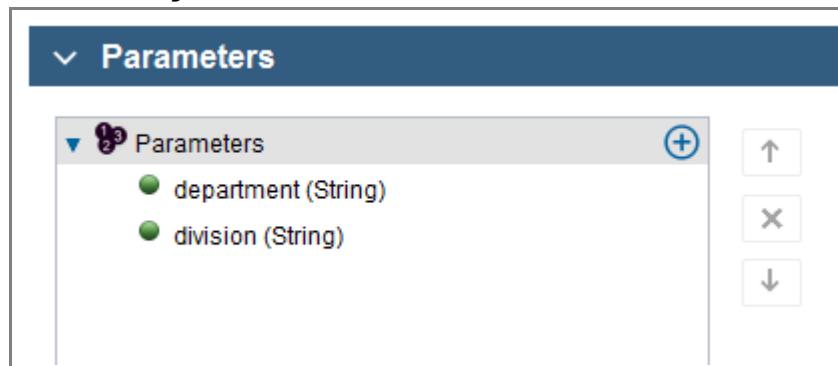


- __ d. Save the process app and close the business object.
- __ 3. Create a business object that is named `RecruitingDetails` to allow users to input recruiting information on a coach.
 - __ a. Create a business object: `RecruitingDetails`
 - __ b. Include the following parameters in the complex variable `RecruitingDetails`:
 - `multipleEmployeesNeeded` (Boolean)
 - `numEmployeesNeeded` (Integer)

- newPosition (Boolean)



- ___ c. Save the process app and close the business object.
- ___ 4. Create a fourth business object that is named `DepartmentDetails` to allow users to input department data on a coach.
 - ___ a. Create a business object: `DepartmentDetails`
 - ___ b. Include the following parameters in the complex variable `DepartmentDetails`:
 - department (String)
 - division (String)



- ___ c. Save your process application and close the business object.
- ___ 5. Create a business object that is named `HiringRequisition` to save data about the requisition. Model this variable to hold the other objects you created as subobjects. Use this variable as the main parent business object to hold all the data for the process.
 - ___ a. Create a business object: `HiringRequisition`
 - ___ b. Add the following parameters:
 - requisitionNumber (String)
 - dateOfRequest (Date)
 - requester (String)
 - datePositionAvailable (Date)

- hiringManagerComments (String)

The screenshot shows a 'Parameters' dialog box with a title bar. Below the title bar is a list of parameters under the heading 'Parameters'. The list includes:

- requisitionNumber (String)
- dateOfRequest (Date)
- requester (String)
- datePositionAvailable (Date)
- hiringManagerComments (String)

On the right side of the dialog box, there are three buttons: an upward arrow, a downward arrow, and a delete (X) button.

- c. Add a parameter that is called `position`. But this time, click **Select** to the right of **Variable Type** and use the **Position** business object that you created.

The screenshot shows a 'Select Library Item' dialog box with a title bar. Below the title bar is a search bar. The main area is titled 'Business Object' and contains a list of items. One item, 'Position', is highlighted with a red rectangle.

- ANY System Data
- Boolean System Data
- CompensationDetails
- Date System Data
- Decimal System Data
- DepartmentDetails
- HiringRequisition
- Integer System Data
- Position** (highlighted)
- RecruitingDetails
- String System Data
- Time System Data



Reminder

The variable type is capitalized but the parameter name that is part of a parent object is lowercase.

Parameter Properties

Name:	<input type="text" value="position"/>
List:	<input type="checkbox"/>
Variable Type:	Position Select... New...
Documentation:	B I U ≡ ≡ ≡ ≡ ≡ ≡ ≡ ≡ ≡ ≡ ↔ ↔

- d. Continue to add the other complex objects:

- compensationDetails (CompensationDetails)
 - departmentDetails (DepartmentDetails)
 - recruitingDetails (RecruitingDetails)

Parameters	
▼	13 Parameters
▲	requisitionNumber (String)
▲	dateOfRequest (Date)
▲	requester (String)
▲	datePositionAvailable (Date)
▲	hiringManagerComments (String)
▶	position (Position)
▶	compensationDetails (CompensationDetails)
▶	departmentDetails (DepartmentDetails)
▶	recruitingDetails (RecruitingDetails)

- ___ e. Save the process app and close the business object.
 - ___ 6. Add the requisitionDetails (HiringRequisition) variable to the Approve Hire Request linked process as an input and output variable.



Important

To pass data from the parent process ([Hiring Request Process](#)) to your nested process ([Approve Hire Request](#)), you must create and map variables. Because the nested process is a linked process, you must pass the data object to the nested process by way of the input and output variable settings in the nested linked process. Remember, variable mapping is not required for subprocesses, as the subprocess uses the variables from the parent process. Variable mapping is required for linked processes because they are a separate process from the parent process.

- ___ a. Open the **Approve Hire Request** linked process.
- ___ b. Click the **Variables** tab.

Input	
isCompliant (String)	(+)

Output	
isCompliant (String)	(+)

Private	
	(+)

Exposed Process Variables	
	(+)

- c. Add an Input variable `requisitionDetails` (`HiringRequisition`) to the process. You can expand the `requisitionDetails` variable to view its contents.

The screenshot shows the Variables panel with the following structure:

- Input** (selected)
 - isCompliant (String)
 - requisitionDetails (HiringRequisition)**
 - requisitionNumber (String)
 - dateOfRequest (Date)
 - requester (String)
 - datePositionAvailable (Date)
 - hiringManagerComments (String)
 - position (Position)
 - compensationDetails (CompensationDetails)
 - departmentDetails (DepartmentDetails)
 - recruitingDetails (RecruitingDetails)
- Output**



Reminder

To change the type of a variable, click **Select** next to **Variable type:** and select the type **HiringRequisition** from the list.

- 7. Add an output variable `requisitionDetails` (`HiringRequisition`) to the process.



Important

When you type in the output name, the variable is immediately flagged because you have an input and an output with the same name, but different variable types. The system defaults the variable to a String. When you set the variable type to match the type of the input variable, the error disappears.

The screenshot shows the Details panel with the following fields:

Name:	requisitionDetails (highlighted with a red border and has a red X icon to its left)
Documentation:	(empty)

- 8. You added the variables to the subprocess. Save your work.

The screenshot shows the 'Variables' tab with the following structure:

- Input** section:
 - isCompliant (String)
 - requisitionDetails (HiringRequisition)
- Output** section:
 - isCompliant (String)
 - requisitionDetails (HiringRequisition)
- Private** section:
 - requisitionDetails (HiringRequisition)
- Exposed Process Variables** section:
 - (empty)

! Important

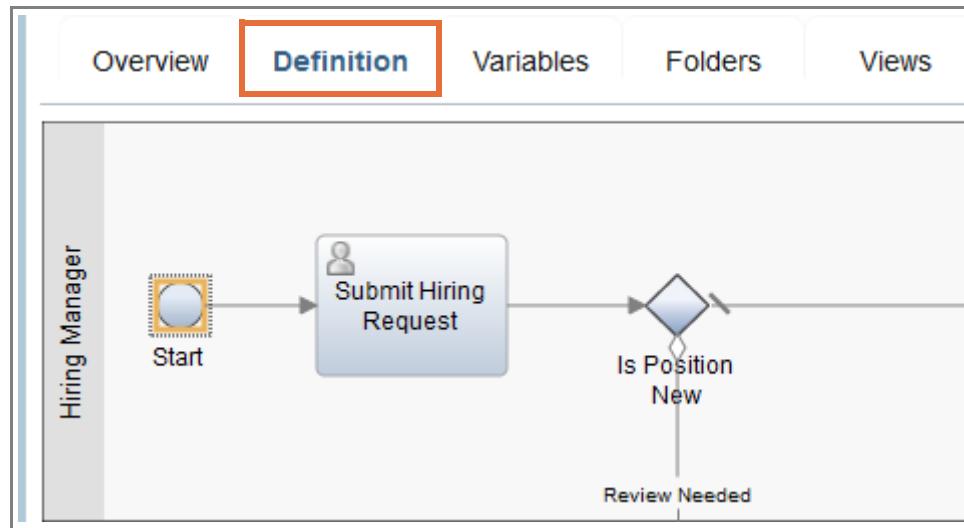
Make sure that the input and output variable names exactly match. If the names do not match, then you have two different variables in the linked process. The input data to the process is not sent as output to the parent process unless you copy the contents of one variable into the other. Copying and pasting the variable name is a good practice.

- 9. Create and map the requisitionDetails variable from the higher-level process to the nested process.
- Open the **Hiring Request Process** and click the **Variables** tab.
 - Create a private variable: `requisitionDetails (HiringRequisition)`

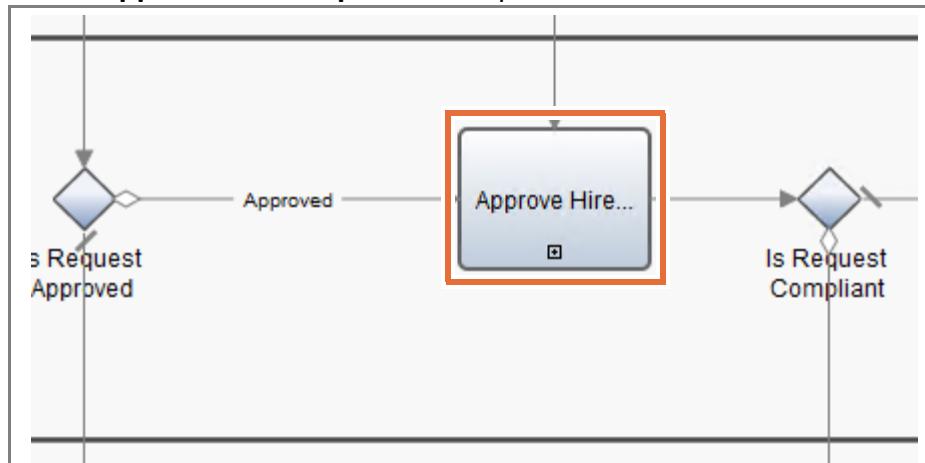
The screenshot shows the 'Variables' tab with the following structure:

- Input** section:
 - (empty)
- Output** section:
 - (empty)
- Private** section:
 - requisitionNumber (String)
 - jobTitle (String)
 - salary (Integer)
 - department (String)
 - isNewPosition (String)
 - isApproved (String)
 - isCompliant (String)
 - requisitionDetails (HiringRequisition)** (highlighted with a red box)
- Exposed Process Variables** section:
 - (empty)

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



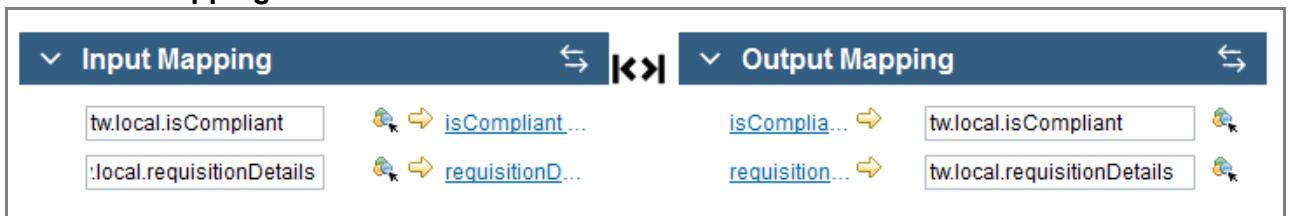
- d. Click the **Approve Hire Request** linked process.



- e. Click the **Properties > Data Mapping** menu.

Properties		Validation Errors
General		
Implementation		
Data Mapping	Input Mapping <div style="display: flex; justify-content: space-between;"> tw.local.isCompliant isCompliant... </div> <div style="display: flex; justify-content: space-between;"> ⚠ requisitionD... </div>	
Pre & Post		
Tracking		

- f. Map `tw.local.requisitionDetails` to both the **Input Mapping** and **Output Mapping** sections.



Reminder

You can use **Ctrl+Space** to view the auto complete options.

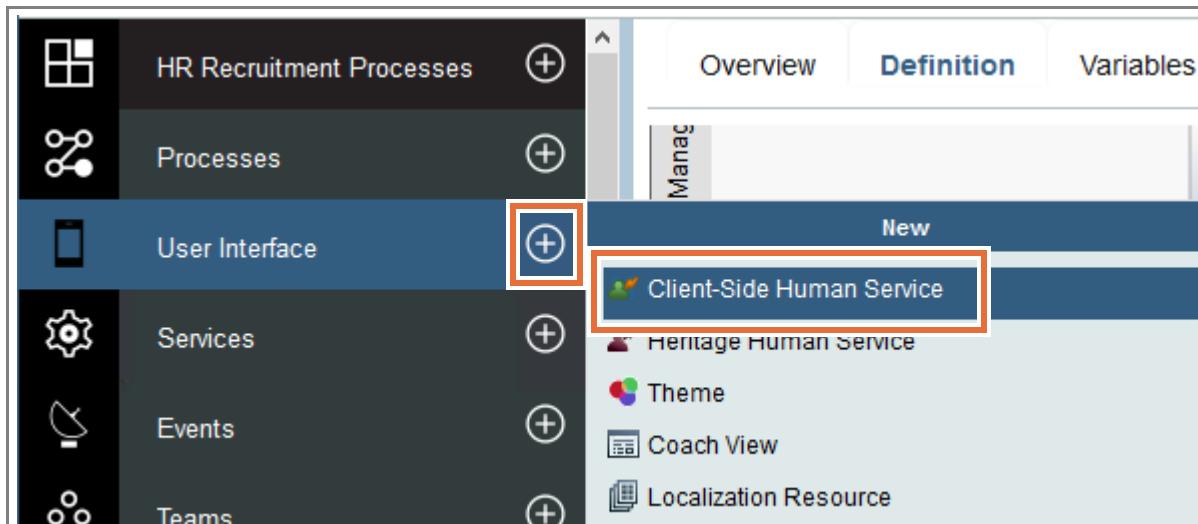


Part 2: Create a reusable client-side human service and define variables

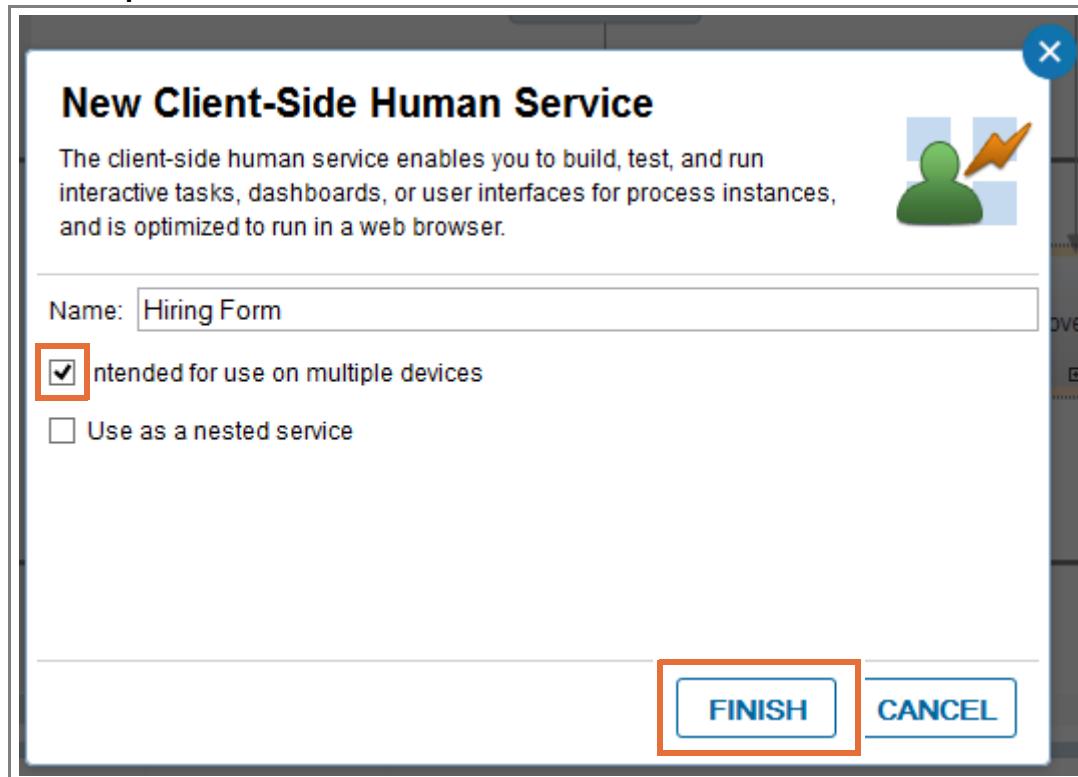
A default human service coach was automatically implemented for each activity when you created each activity inside of a lane that was not a system lane. In this section, you define and implement custom services and coaches for activities in the process. In the next few sections, you create a coach to collect information from process teams. The custom service passes the collected information in the coach to the process.

Two activities on the process need a coach that has the same fields: the Submit Hiring Request and the Approve Hiring Request. One requires the ability to input data, and the other needs the data to be read-only and an approval mechanism. In this part of the exercise, you create a single coach for both activities. Because the same data can be shown for multiple activities on the process, create a coach that can be reused for data input and viewing data (read-only).

- 1. Create a client-side human service.
- a. In the Process Designer library, click the (+) plus sign next to User Interface, and select Client-Side Human Service.



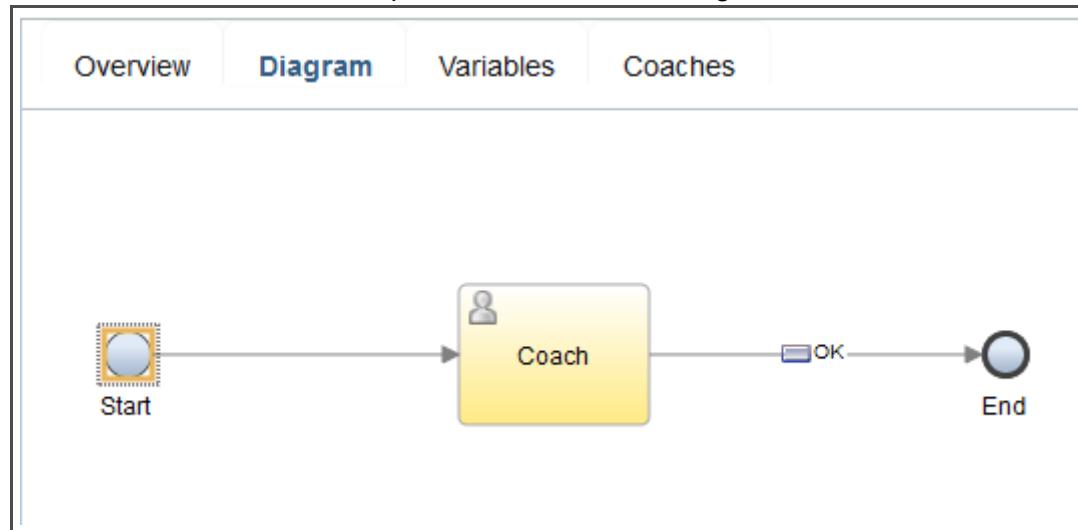
- b. Name the client-side human service: **Hiring Form**. Select the **Intended for use on multiple devices** check box and click **Finish**.



Note

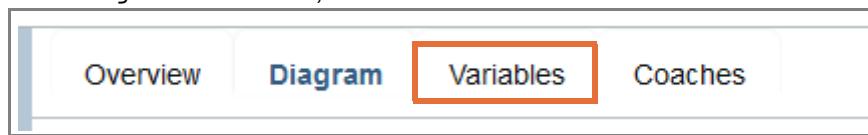
Intended for use on multiple devices enables the new service to be used on multiple device types. With this option, you can use responsive controls for all the new coaches and coach content that you add to the service.

- c. The new human service opens in the Process Designer editor.

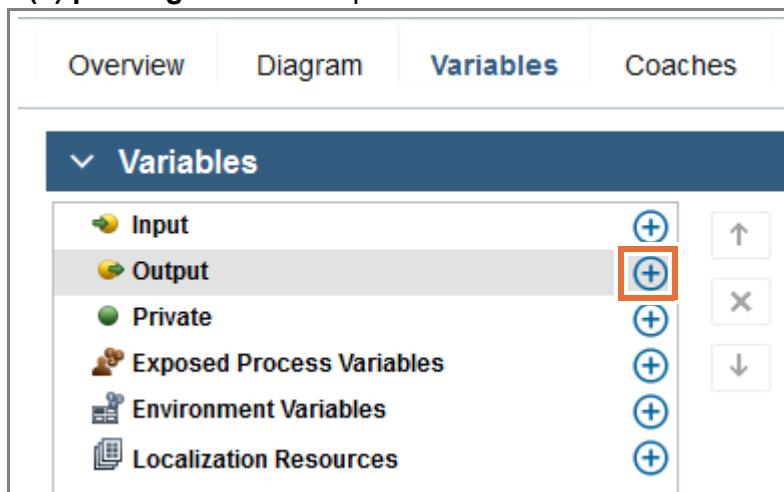


__ 2. Add the `requisitionDetails` output variable to the human service.

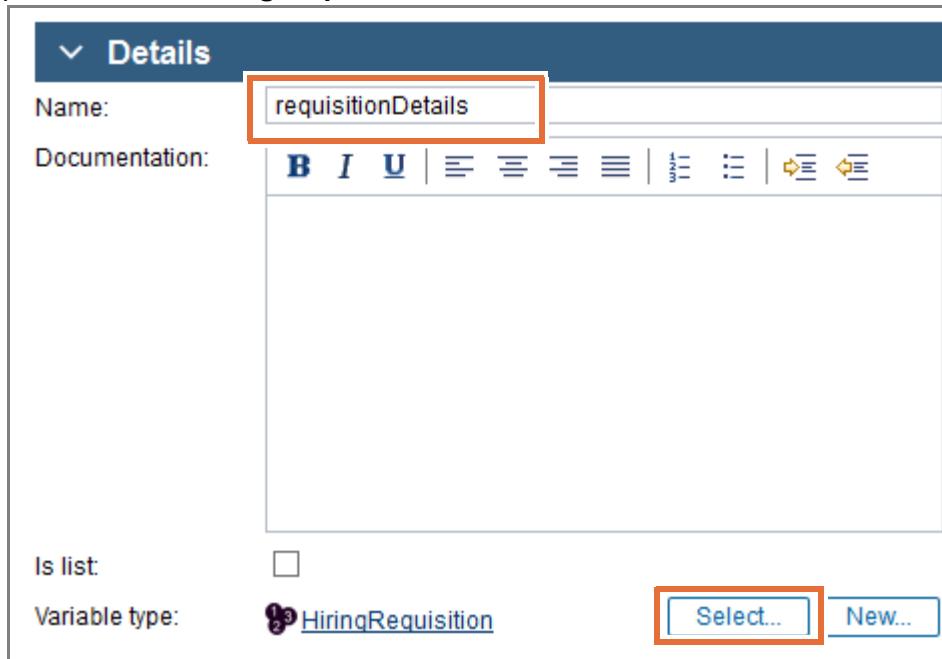
__ a. In the `Hiring Form` service, click the **Variables** tab.



__ b. Click the (+) plus sign next to Output.



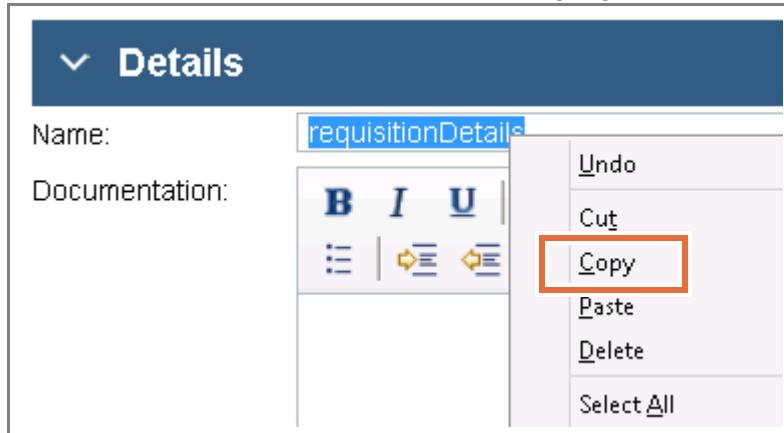
__ c. Name the variable: `requisitionDetails`. Click **Select** to the right of the Variable type option to select **HiringRequisition**.



Information

Because you need all of the subobjects of the parent **HiringRequisition** object, you use the full object in the coach. If you need only part of the parent object, include that subobject and map the subobject from the parent object when mapping inputs and outputs.

- 3. Because you reuse this coach for multiple activities, add matching input variables to the human service. If any data is entered as part of the process instance from the first activity, you can present it on the coach.
- a. Click the output variable `requisitionDetails`, highlight the name, and copy it.



- b. Click the (+) plus sign next to Input and paste `requisitionDetails` into the Name field.



Information

When you paste the name of the input variable, you see an error icon. Similar to input and output variables in processes and services in the Process Designer, this error occurs because the variables are not unique: two variables of different types have the same name. This error disappears after you set the types of the variables to be equal.



- c. Click **Select** to the right of the Variable type option.

- __ d. Change the type to **HiringRequisition**.

The screenshot shows the 'Variables' dialog box with the following structure:

- Input:** requisitionDetails (HiringRequisition) (with a plus sign)
- Output:** requisitionDetails (HiringRequisition) (with a plus sign)
- Private:** (with a plus sign)
- Exposed Process Variables:** (with a plus sign)
- Environment Variables:** (with a plus sign)
- Localization Resources:** (with a plus sign)

- __ e. Save your work.
- 4. Now that the variables are created, add the inputs that are bound to the variables to the coach.

- __ a. In the Hiring Form service, click the **Diagram** tab.

The screenshot shows the service editor with the tabs: Overview, Diagram (highlighted), Variables, and Coaches.

- __ b. Click the **Coach** on the canvas, and in the **Properties > General > Common** section, change the name of the coach to: **Hiring Form**

The screenshot shows a BPMN diagram with a 'Start' node, a 'Hiring Form' activity, and an 'End' node. Below the diagram is the 'Properties' panel with the 'Common' tab selected. The 'Name:' field contains 'Hiring Form' (highlighted with a red box).



Reminder

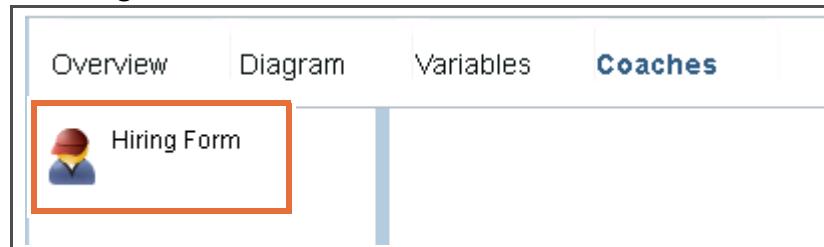
You can also click the name of a selected coach on the canvas to rename it.

- 5. Build the online form coach for the process team to use in the process application. Create your coach fields from the **Output** variables. Build the first section that contains all the simple variables from the Requisition Details object.

- a. For the `Hiring Form` service, click the **Coaches** tab.



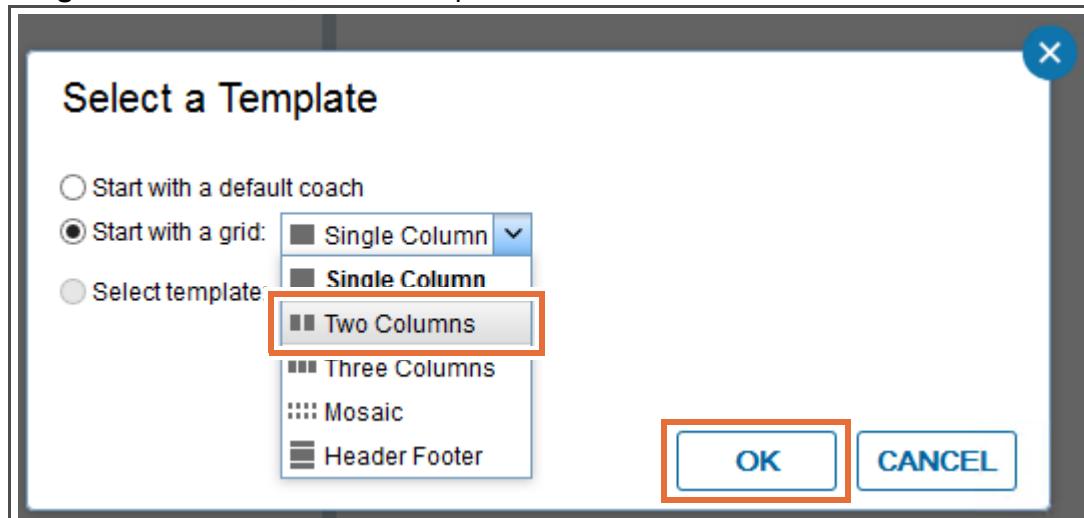
- b. Click the **Hiring Form** coach.



Hint

Consider how items are laid out. This consideration determines whether the coach is laid out vertically or horizontally. This coach lays out items vertically except for the position field.

- c. A message box for selecting a template appears. Select the second option: **Start with a grid**. From the list, select the option **Two Columns** and click **OK**.



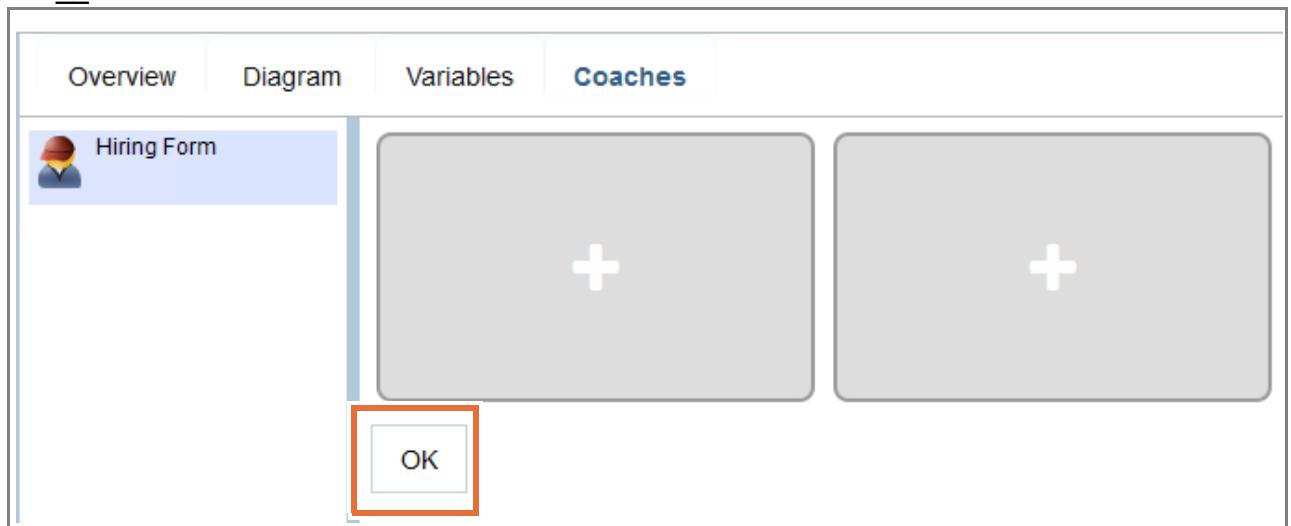


Information

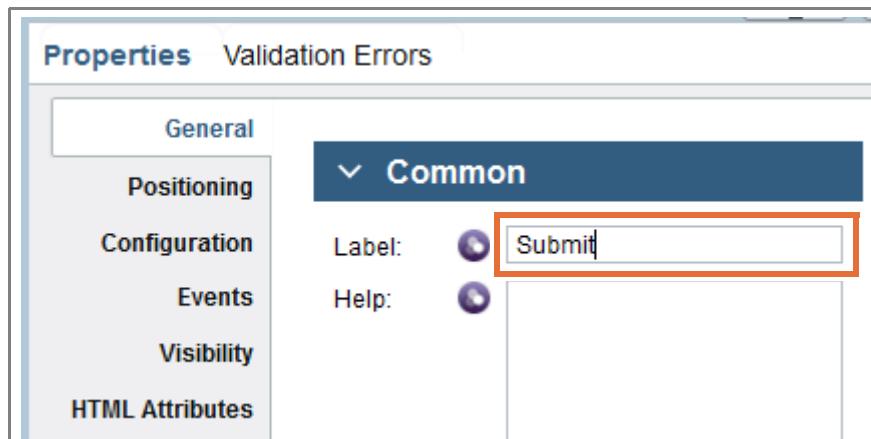
A two-column grid appears for the **Hiring Form** coach layout. This layout is the default layout that you start from to build your coach.



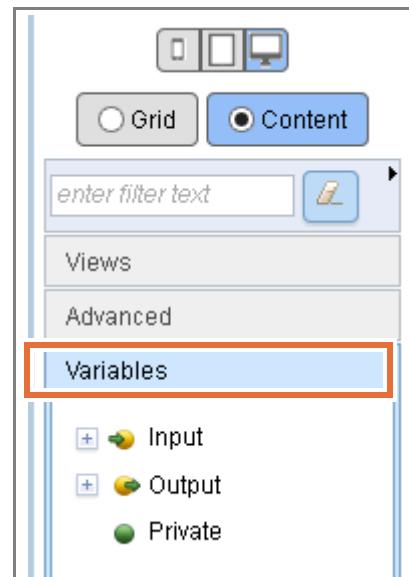
- ___ d. Click **OK** on the coach canvas to select the button.



- __ e. In the **Properties > General > Common** section, change the label of the control button type to: Submit



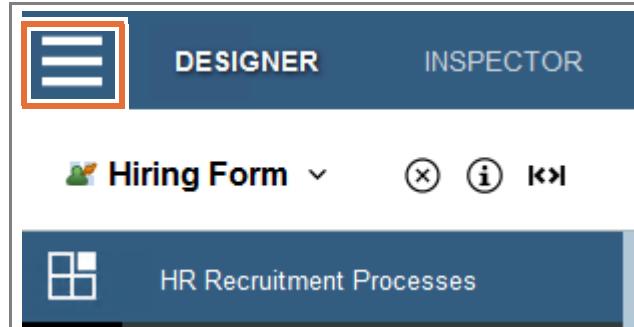
- __ f. To the right of the coach canvas in the palette, click the section to expand the **Variables** palette tray and view the service variables.



- __ g. The first inputs that you add to the coach are the components of the `requisitionDetails` variable. From the palette, click the **(+)** plus sign next to Input to view the service input variables.

**Hint**

Collapse the **Library** on the left side by clicking the library icon (three horizontal lines in the upper-left side) to allocate more space on the canvas to work.



- h. Drag the **requisitionDetails > requisitionNumber** variable to the left column in the grid over **Submit**.

The screenshot shows the Designer interface with a form containing a text input field labeled "Requisition Number". This input field is highlighted with a red box. To the right of the form is a sidebar with various options and a "Variables" section. In the "Variables" section, under the "Input" category, there is a tree view of variables. One of these variables, "requisitionDetails (HiringRequisition)" has its "requisitionNumber (String)" item highlighted with a red box, matching the highlighted item in the form.

- i. Drag the **requisitionDetails > requester** variable from the palette to the space underneath **Requisition Number** in the left column. Wait until you see the orange line below the Requisition Number variable in the same grid.

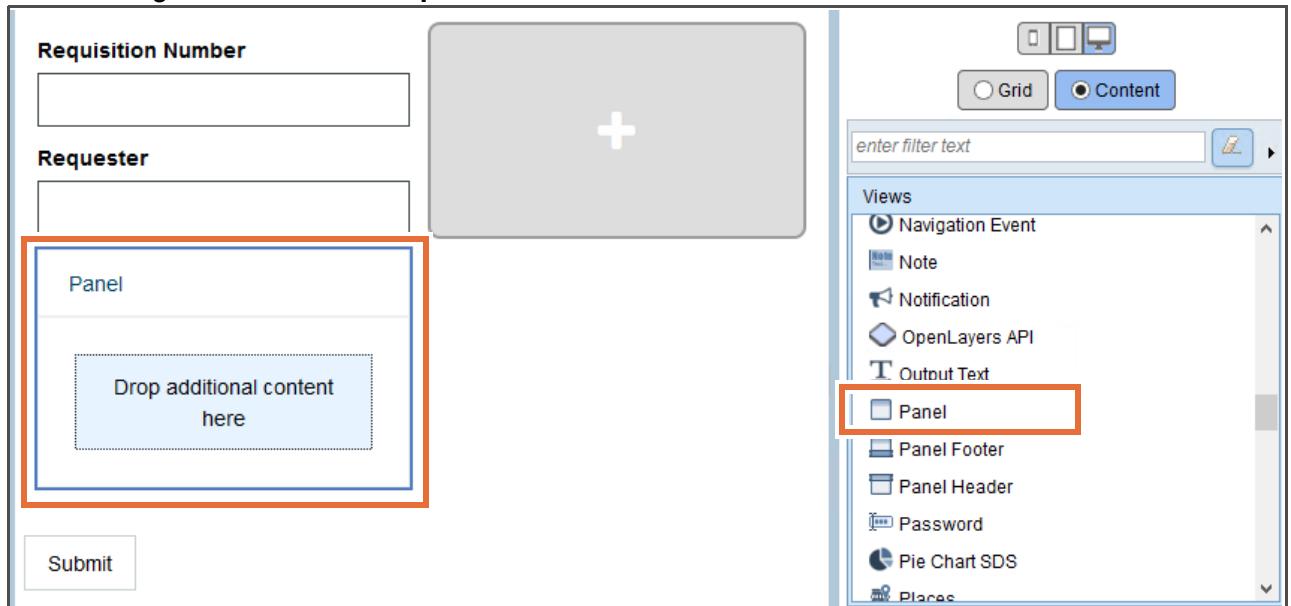
The screenshot shows the IBM Worklight Studio interface. On the left, there is a form with a 'Requisition Number' input field and a 'Submit' button. A red box highlights the 'Requisition Number' input field. To the right of the form is a palette with various icons. Below the palette is a sidebar with tabs for 'Grid' and 'Content' (which is selected). The 'Content' tab shows a list of variables under 'Variables'. One of these variables, 'requester (String)', is highlighted with a red box. The variable details show it is part of 'requisitionDetails (HiringRequisition)'.

Information

The **Requester** field is added to the coach.

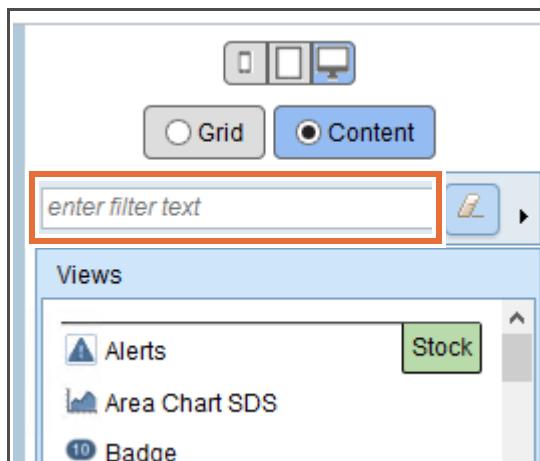
The screenshot shows the updated form. The 'Requester' field has been added to the left column, positioned below the 'Requisition Number' field. It is highlighted with a blue border. The 'Submit' button remains at the bottom of the form.

- __ j. Expand the **Views** palette tray. Drag a **Panel** control onto the canvas inside the same grid beneath the **Requester** field.

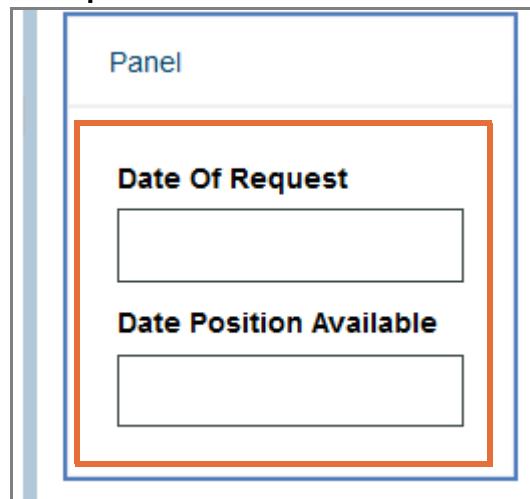


Hint

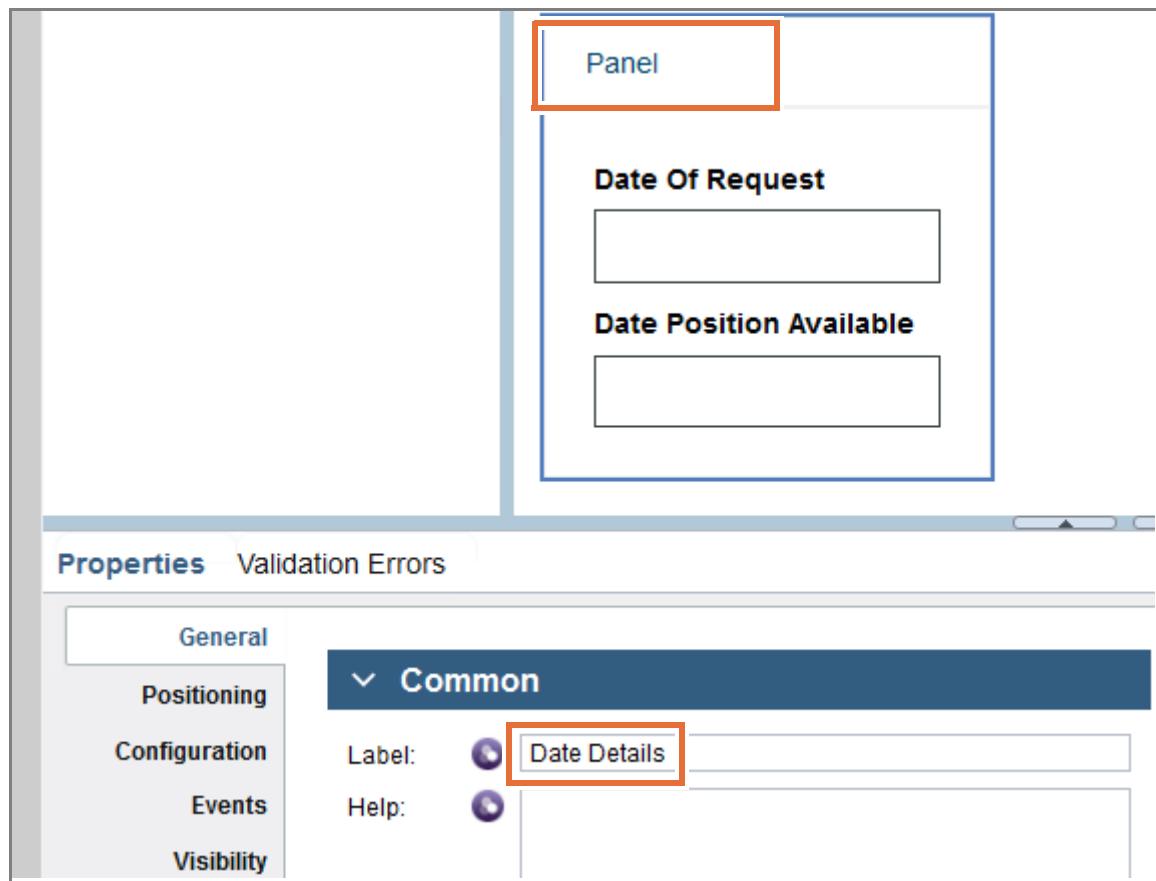
Similar to the filtering capabilities in the Process Designer, use the input box above the Views palette tray to filter the coach view controls.



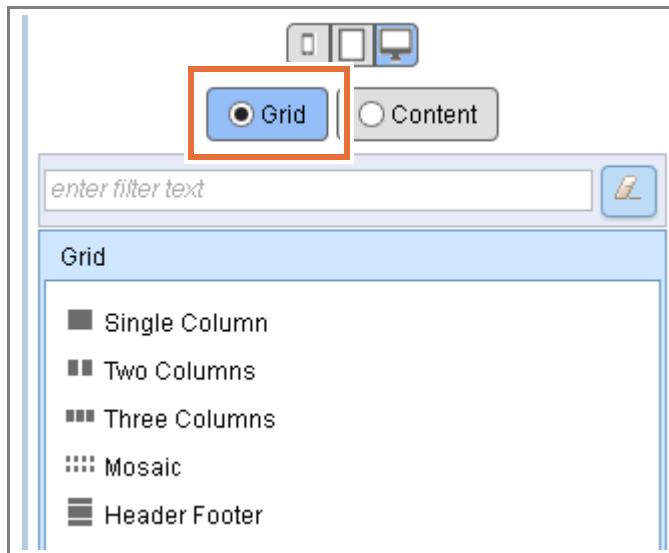
- k. Expand the **Variables** palette tray and drag the date fields (**Input > requisitionDetails > dateOfRequest** and **Input > requisitionDetails > datePositionAvailable**) into the area that is labeled **Drop additional content here**.



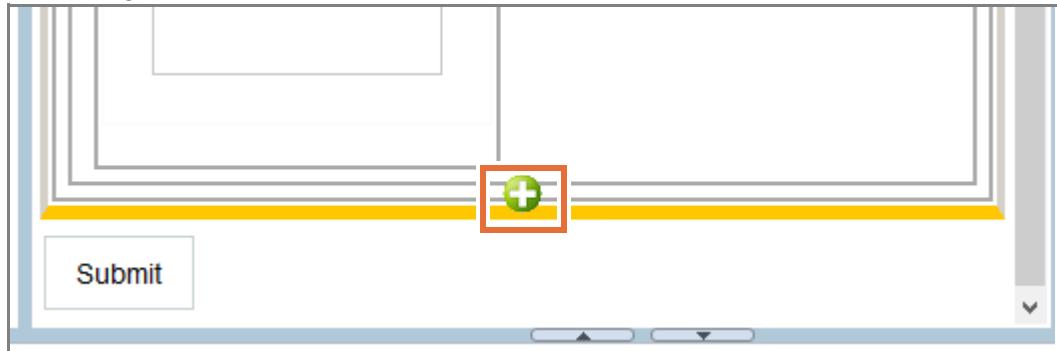
- l. Select the **Panel**. From the **Properties > General > Common** section, change the label to: Date Details



- 6. Add a grid above the **Submit** button.
- a. Select the **Grid** radio button on the right side above the palette trays. This setting shows the grids on the canvas.



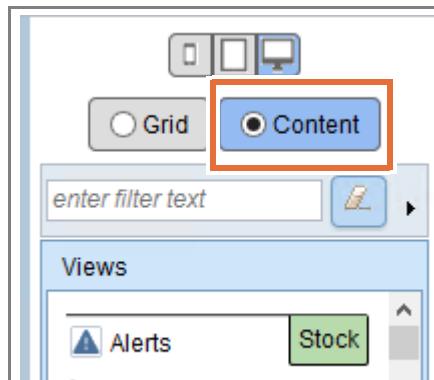
- b. Hover your mouse and click the **(+)** plus sign on the bottom of the outer-most grid to add a grid above the Submit button.



**Hint**

Your grid should have only a single line between the orange space and the submit button.

- ___ c. Click **Content** to return to the add content mode.

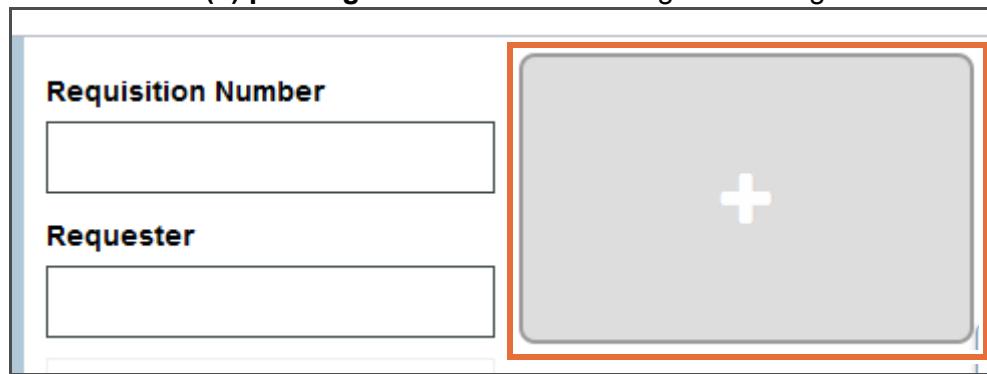


- ___ d. Open the **Variables** palette tray. Drag the **Input > requisitionDetails > hiringManagerComments** variable in the new grid.

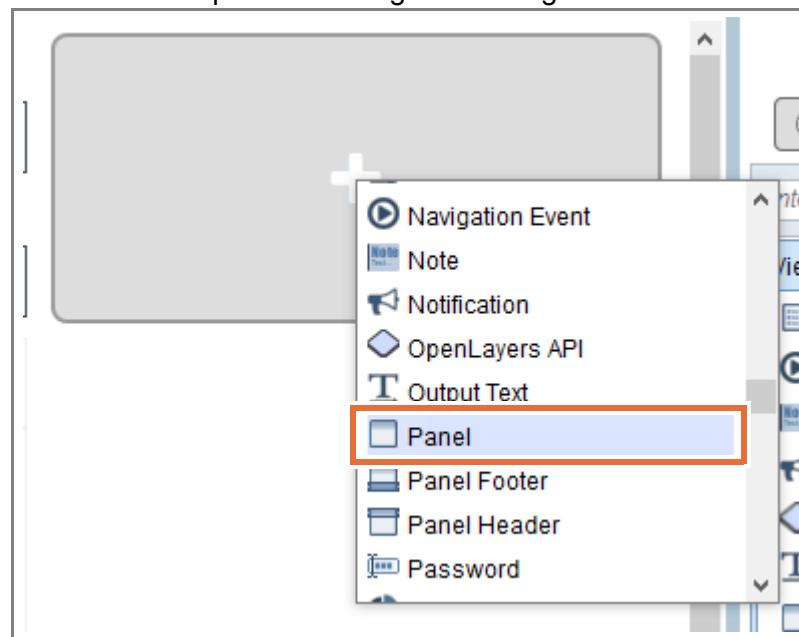
- ___ e. Save your work.

— 7. Create a panel and populate it with the Position Details business objects.

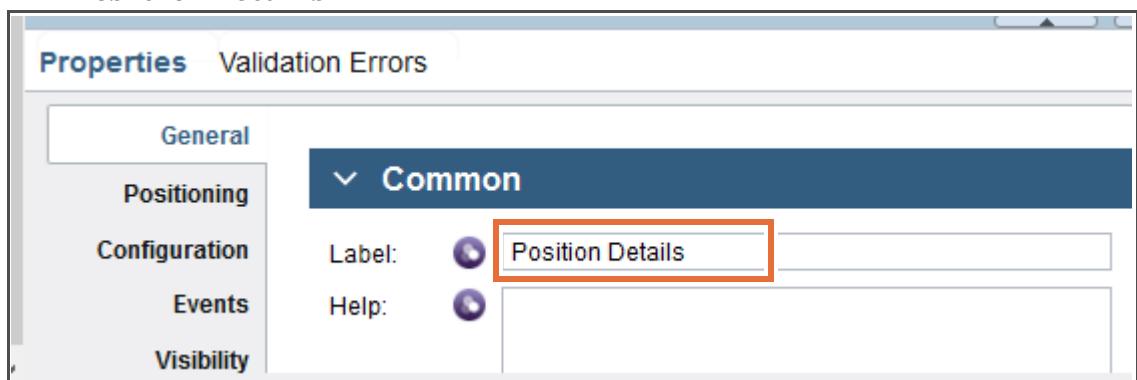
— a. Click the white (+) plus sign on the **Panel** in the right column grid.



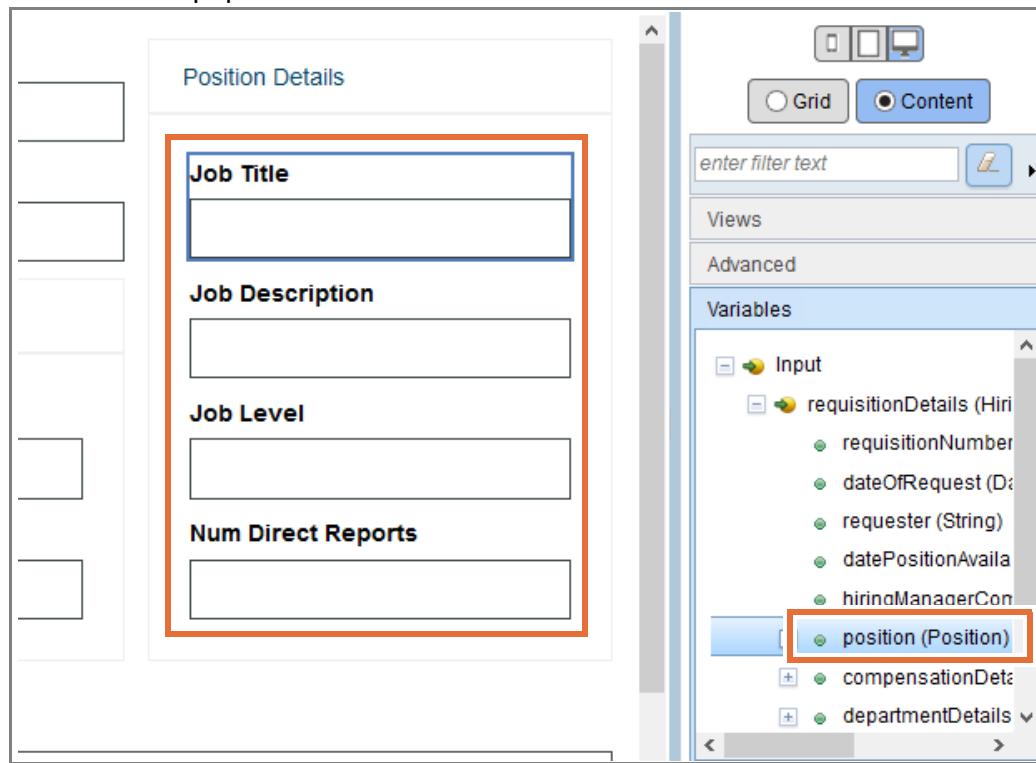
— b. Click **Panel** to add the panel to the right column grid.



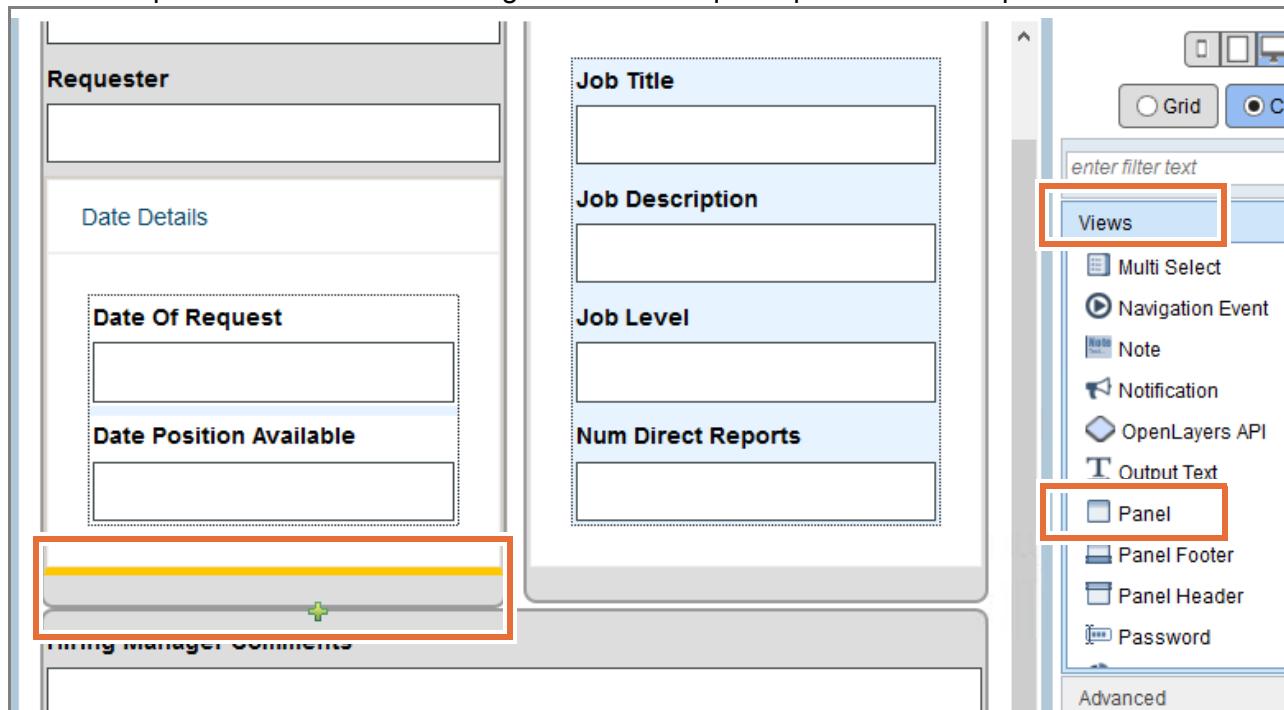
— c. Click the new panel. In the **Properties > General > Common** section, label the panel: Position Details



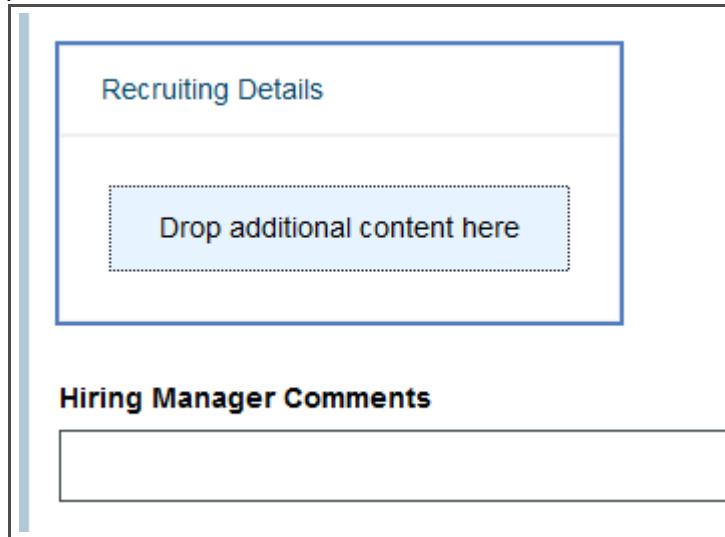
- ___ d. From the **Variables** palette tray, drag the **Input > requisitionDetails > position** complex variable into the new panel area that is labeled **Drop additional content here**. The section populates with the subvariables of the Position variable.



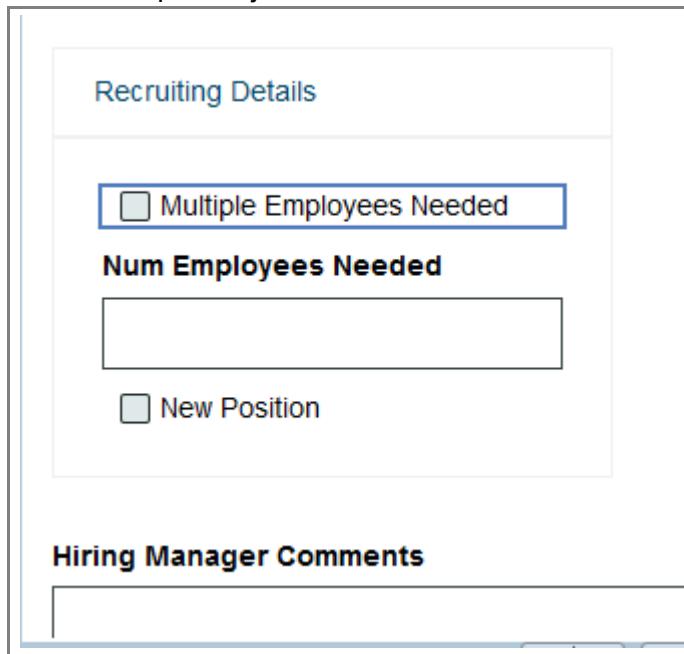
- ___ 8. Group the `recruitingDetails` variables together, setting them apart from the other data on the screen in a section.
- ___ a. Drag a **Panel** from Views onto the canvas in the first column grid below the **Date Details** panel. Wait to see the orange line at the required position and drop the section there.



- __ b. Label the panel: Recruiting Details



- __ c. From the **Variables** palette tray, drag the **Input > requisitionDetails > recruitingDetails** complex object to the new section.



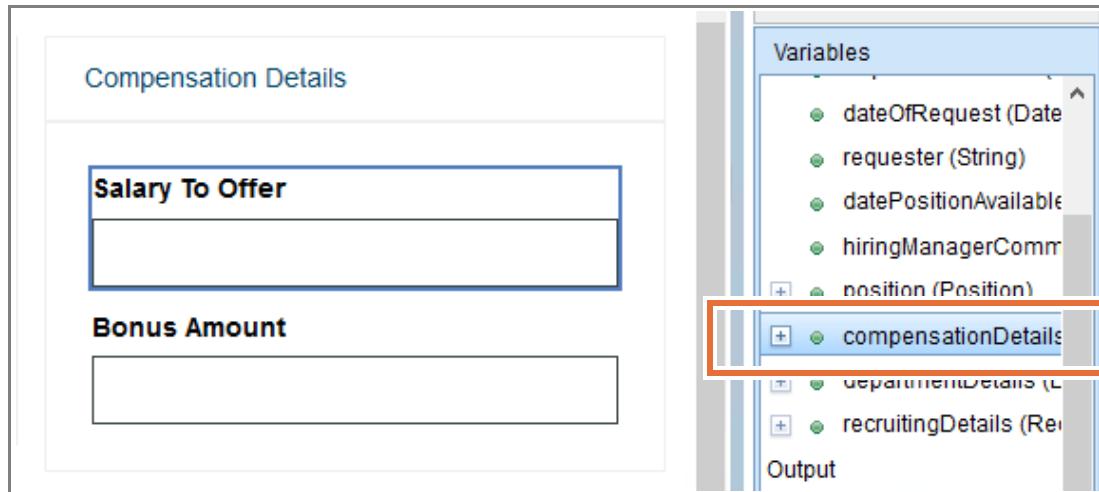
Important

When you drag a complex object to the canvas, inputs are created for all the subvariables.

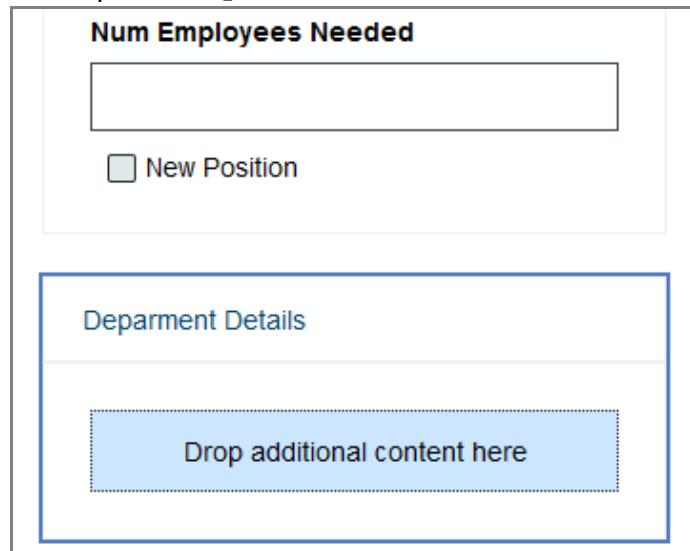
- 9. Add the Compensation Details to the coach.
- a. Expand the **Views** palette tray. Drag a **Panel** control onto the canvas in the second column grid next to the **Recruiting Details** section, and label the new panel: **Compensation Details**

The screenshot shows a user interface with two main sections: 'Recruiting Details' and 'Compensation Details'. The 'Recruiting Details' section contains fields for 'Multiple Employees Needed' (checkbox), 'Num Employees Needed' (text input), and 'New Position' (checkbox). Below this is a section labeled 'Hiring Manager Comments'. The 'Compensation Details' section has a placeholder 'Drop additional content here'.

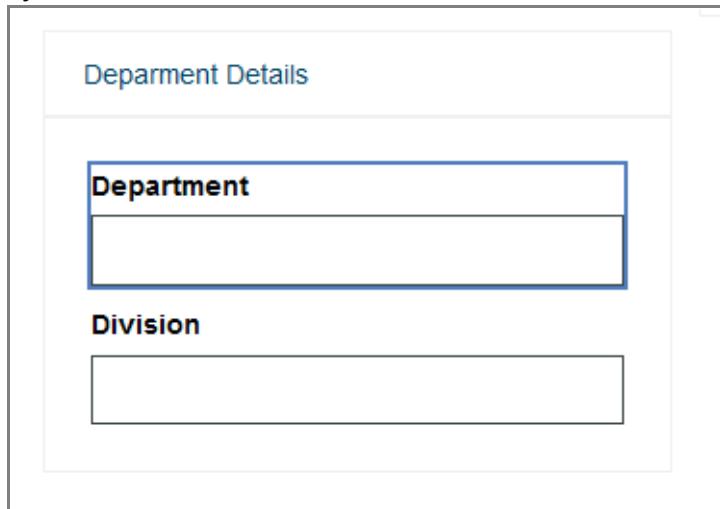
- b. Add the **Variables > Input > requisitionDetails > compensationDetails** complex variable to the new section.



- 10. Finally, add the Department Details section to the coach.
 - a. For the final section, drag a panel in the first column grid below the **Recruiting Details**, and label the new panel: Department Details

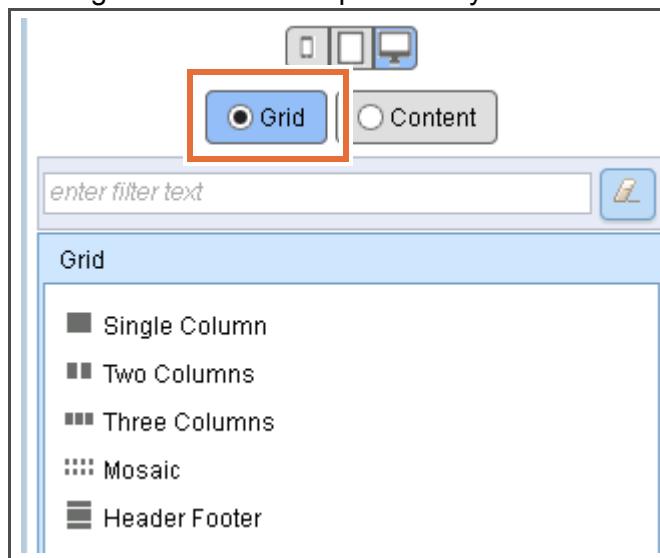


- b. From the **Variables** palette tray, drag the **requisitionDetails > departmentDetails** complex object to the new section.

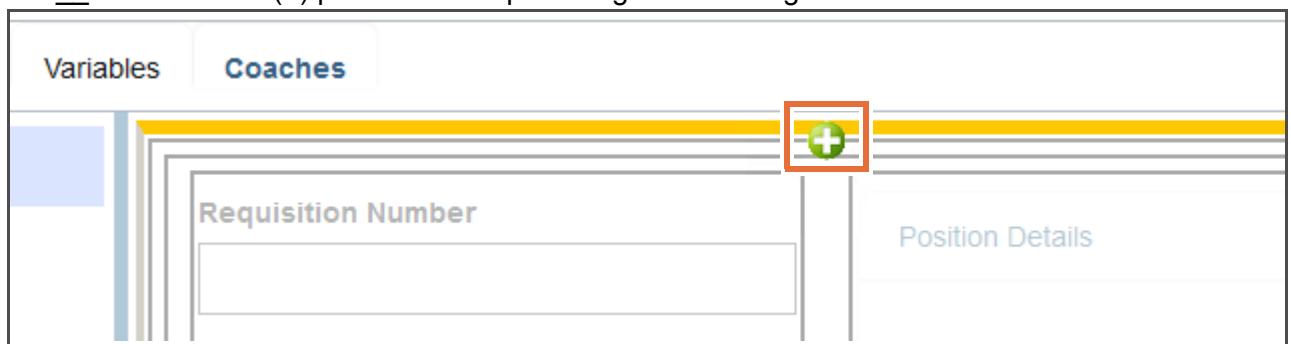


— 11. Add a grid in the coach to realign the sections.

— a. Click **Grid** on the right side above the palette tray to enable it.



— b. Click the (+) plus icon on top of the grid to add a grid at this location.



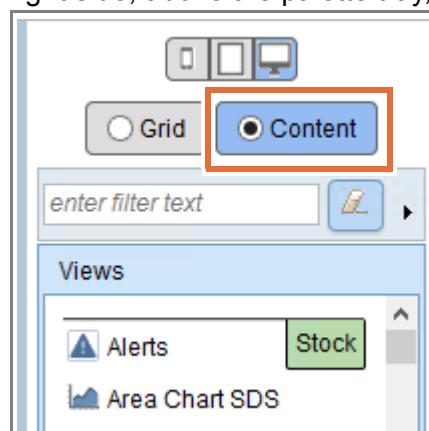


Information

You added a grid at the top.

The screenshot shows a user interface with a large orange rectangular area at the top. Below it are two smaller rectangular input fields. The left field is labeled "Requisition Number" and contains a single-line input box. The right field is labeled "Position Details" and contains a single-line input box labeled "Job Title".

- ___ c. Click **Content** on the right side, above the palette tray, to enable it.



- ___ d. Drag the **requisition Number** input control that is in the left grid to the new grid at the top.

The screenshot shows the user interface after the "Requisition Number" input control has been moved. The "Requisition Number" input control is now located in the large orange grid at the top. The other input controls ("Requester", "Position Details", "Job Title", and "Date Details") remain in their original positions below the main grid.

- ___ e. Click the **Requester** input control and move in the new grid below **Requisition Number**.

Requisition Number <input type="text"/>	Position Details Job Title <input type="text"/>
Requester <input type="text"/>	Date Details <input type="text"/>



Information

Requisition Number and Requester are now part of this new grid.

Requisition Number <input type="text"/>	Position Details Job Title <input type="text"/>
Requester <input type="text"/>	Date Details <input type="text"/>
Date Of Request <input type="text"/>	

- ___ f. Save your work.



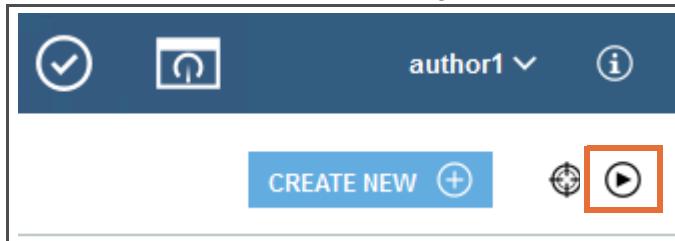
Important

Now the coach layout looks similar to the following coach.

Requisition Number <input type="text"/>	
Requester <input type="text"/>	
Date Details	Position Details
Date Of Request <input type="text"/>	Job Title <input type="text"/>
Date Position Available <input type="text"/>	Job Description <input type="text"/>
Recruiting Details	Job Level <input type="text"/>
<input type="checkbox"/> Multiple Employees Needed Num Employees Needed <input type="text"/>	Num Direct Reports <input type="text"/>
Department Details	Compensation Details
Department <input type="text"/> Division <input type="text"/>	Salary To Offer <input type="text"/> Bonus Amount <input type="text"/>
Hiring Manager Comments <input type="text"/>	
<input type="button" value="Submit"/>	

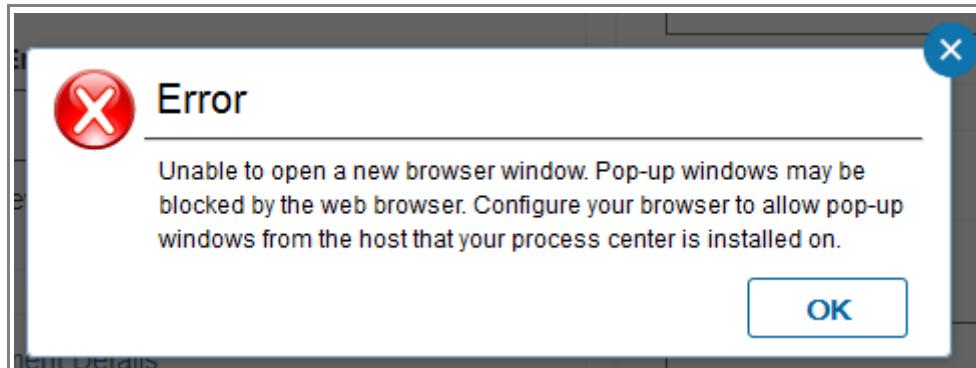
— 12. Run the coach.

- a. Now that a working coach has variables that are bound to the fields, run the coach and view it in a browser. Click **Run** at the upper-right corner of the window.

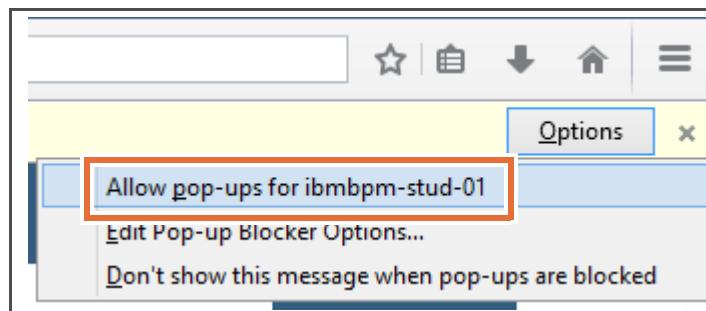


Note

The inspector attempts to open the task inside a new window, but Firefox prevents the site from opening the pop-up window. Click **OK**.



To correct the problem, in the yellow bar at the top of the browser, click **Options > Allow pop-ups for ibmbpmstud01**.



The pop-up window that appears shows the coach designer. Close this window and rerun the coach.

- ___ b. Another browser window opens and displays the coach.

Requisition Number	
<input type="text"/>	
Requester	
<input type="text"/>	
Date Details	
Date Of Request	<input type="text"/>
Date Position Available	<input type="text"/>
Recruiting Details	
<input type="checkbox"/> Multiple Employees Needed Num Employees Needed <input type="text" value="0"/>	
<input type="checkbox"/> New Position	
Department Details	
Department	<input type="text"/>
Division	<input type="text"/>
Hiring Manager Comments	
<input type="text"/>	
Position Details	
Job Title	<input type="text"/>
Job Description	<input type="text"/>
Job Level	<input type="text"/>
Num Direct Reports	<input type="text" value="0"/>
Compensation Details	
Salary To Offer	<input type="text" value="0"/>
Bonus Amount	<input type="text" value="0"/>



Important

The coach looks exactly like the coach that you created in the designer window. Most of the design and functional elements of a coach work the same in the designer as they do after you run the coach. The test run gives you an idea of the final page for the business users, but it usually looks the same as the coach in the designer view.

**Note**

This coach is not meant to be an attractive, fully functioning front end; you add more functional and visual features later on in this playback. The intent of this playback is to start data flow.

- ___ c. Click inside the **Date Of Request** input control to select a date, insert some data in other fields, and when you are ready, scroll down to click **Submit**.

Date Details Date Of Request <input type="text"/> Date Position Available <input type="text"/> Recruiting Details	Position Details Job Title <input type="text"/> Job Description <input type="text"/> Job Level <input type="text"/>
--	---

- ___ d. The task is complete; close the browser window that you ran the coach in.
___ e. Close the **Hiring Form** client-side human service window.

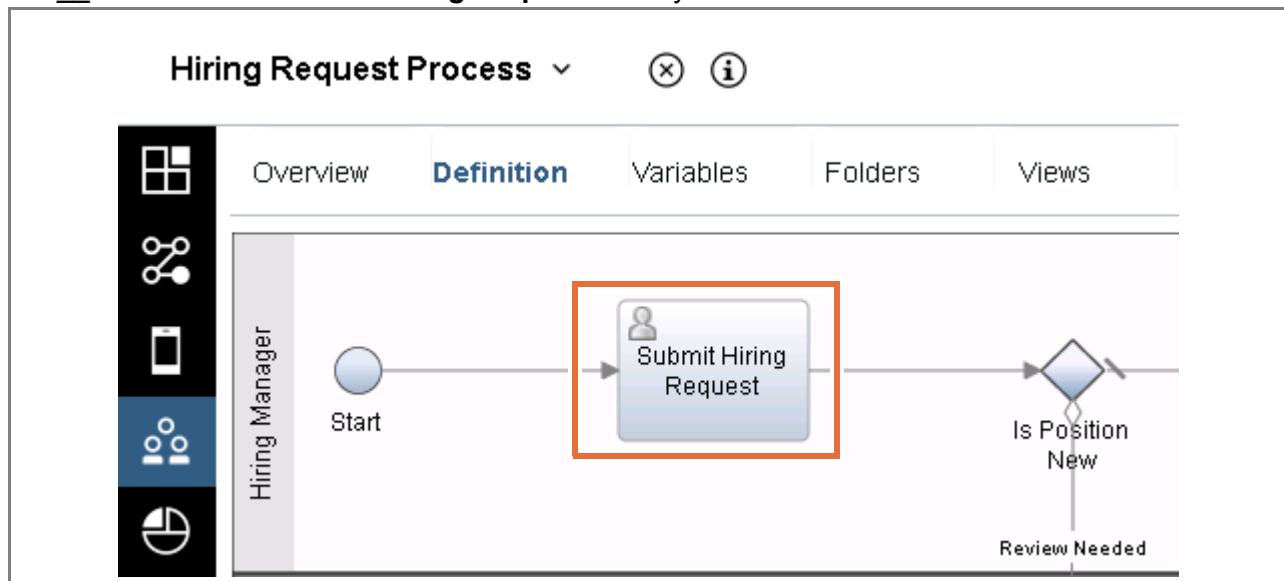
The screenshot shows a browser window titled "DESIGNER INSPECTOR". Inside the window, there is a toolbar with a "Hiring Form" icon, a red-bordered "X" button, an info icon (i), and a link icon (link). At the bottom of the window, there is a navigation bar with tabs: "Overview", "Diagram", "Variables", and "Coaches".

Part 3: Create human services

Because the **Approve New Hire Request** activity in the **Hiring Request Process** reviews the same data that is entered in the last human service, you can reuse the **Hiring Form** human service. Add a service to an activity.

In the **Hiring Request Process** modeling effort, many of the activities are placed on the diagram canvas with default human services. It is time to implement the services that were created to the appropriate activities in the process. In this section, you replace the human services in the **Submit Hiring Request** and **Approve New Hire Request** activities in the **Hiring Request Process** business process with the appropriate task services.

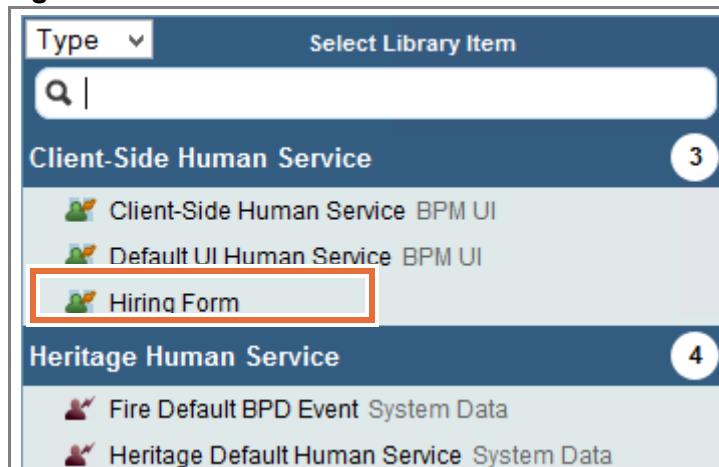
- 1. Implement the **Submit Hiring Request** activity service.
 - a. In the Process Designer, open the **Hiring Request Process** process.
 - b. Click the **Submit Hiring Request** activity.



- c. In the **Properties > Implementation > Implementation** section, click **Select** next to Implementation to select the service that implements the user task.

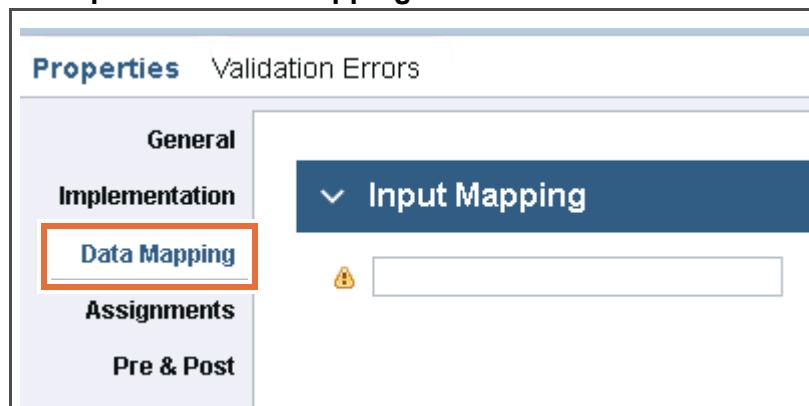
Properties	Validation Errors
<div style="background-color: #2e6b8d; color: white; padding: 5px; margin-bottom: 5px;">General</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9; margin-bottom: 5px;">Implementation</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9; margin-bottom: 5px;">Data Mapping</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9; margin-bottom: 5px;">Assignments</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9; margin-bottom: 5px;">Pre & Post</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9; margin-bottom: 5px;">Tracking</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9; margin-bottom: 5px;">Conditions</div>	
<div style="background-color: #2e6b8d; color: white; padding: 5px; margin-bottom: 5px;">Activity Type</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9; margin-bottom: 5px;">Type: User Task</div> <div style="background-color: #2e6b8d; color: white; padding: 5px; margin-bottom: 5px;">Implementation</div> <div style="border: 1px solid #ccc; padding: 5px; background-color: #f9f9f9; margin-bottom: 5px;">Implementation: Default UI Human Service BPM U Select... New...</div>	

- __ d. Select **Hiring Form** from the list.



- __ 2. Map the Submit Hiring Request input and output.

- __ a. Click the **Properties > Data Mapping** menu.

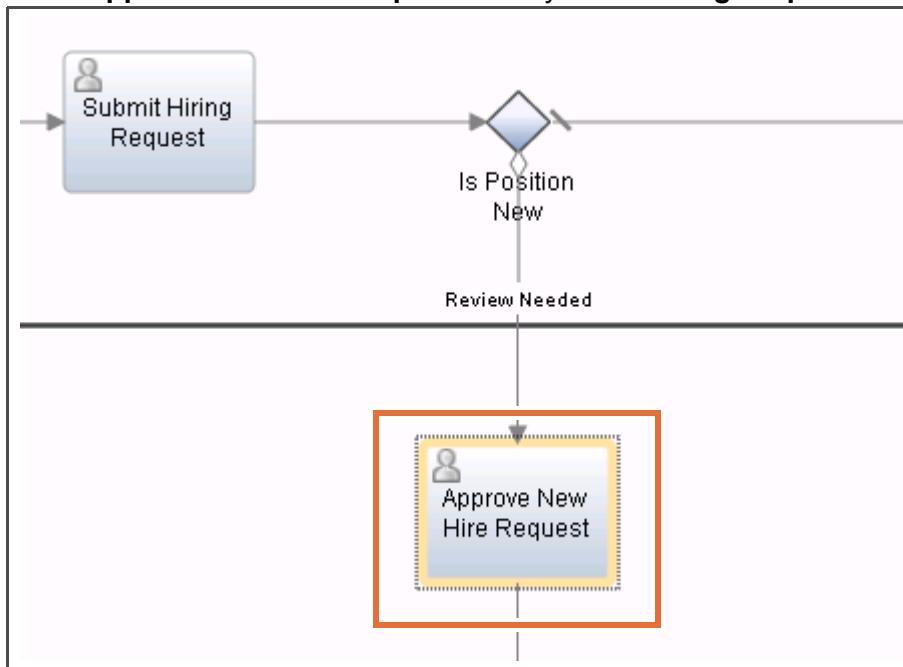


- __ b. Map `tw.local.requisitionDetails` to both **Input Mapping** and **Output Mapping**.

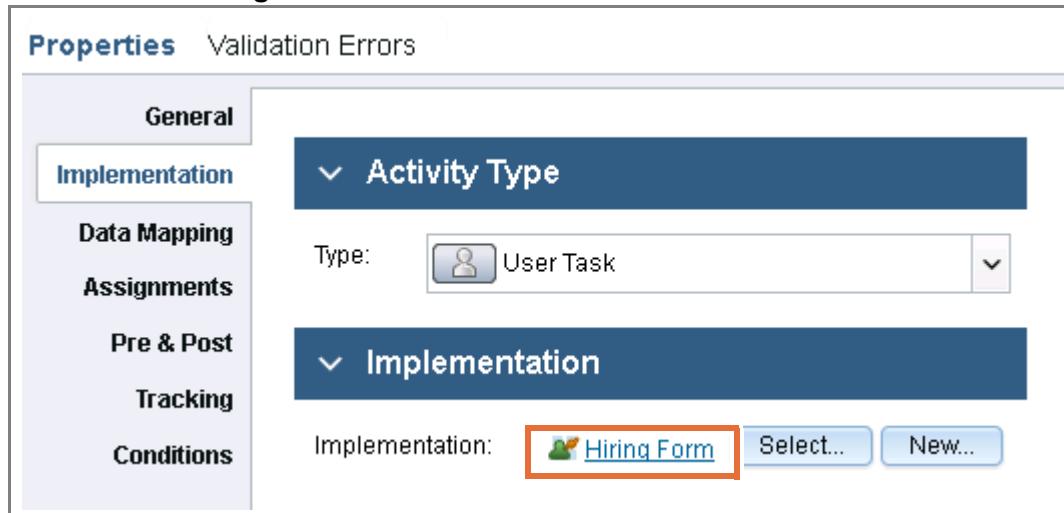


- __ c. Save your work.

- ___ 3. Implement the **Approve New Hire Request** activity.
- ___ a. Click the **Approve New Hire Request** activity in the **Hiring Request Process** process.



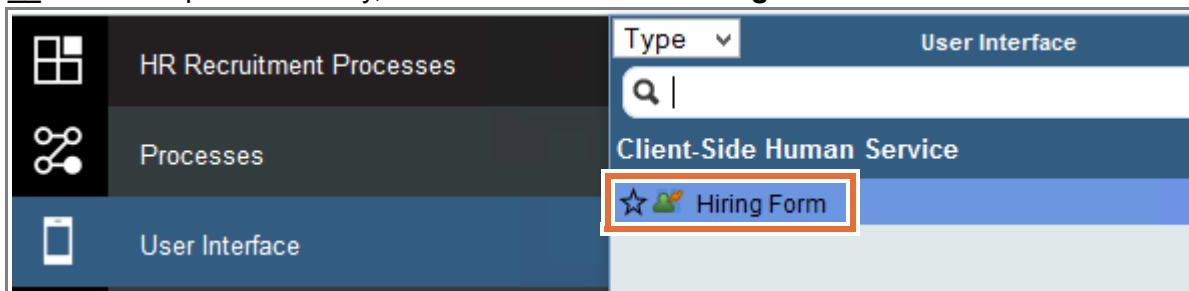
- ___ b. Click **Select** in the **Properties > Implementation > Implementation** section.
- ___ c. Select the **Hiring Form** client-side human service from the list.



- ___ 4. Map the **Approve New Hire Request** inputs and outputs.
- ___ a. Open the **Properties > Data Mapping** menu for the **Approve New Hire Request** activity.

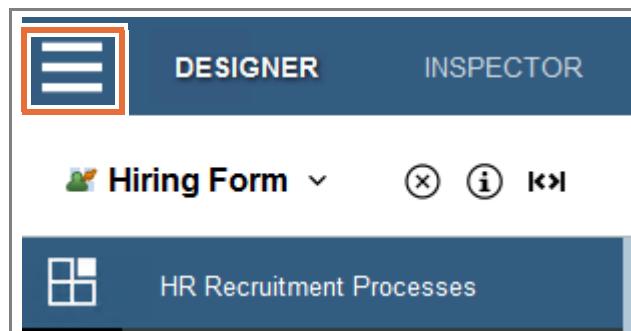
- __ b. Map `tw.local.requisitionDetails` to both **Input Mapping** and **Output Mapping**.

- __ c. Save your work.
- __ 5. To change the process flow, you can assign business data to a process flow object. Map the `NewPosition` variable to the `IsNewPosition` variable.
- __ a. In the process library, click **User Interface > Hiring Form**.



Hint

Expand the **Library** on the left side by clicking the library icon (three horizontal lines in the upper-left side) to view the menu.



- __ b. Click the **Variables** tab.

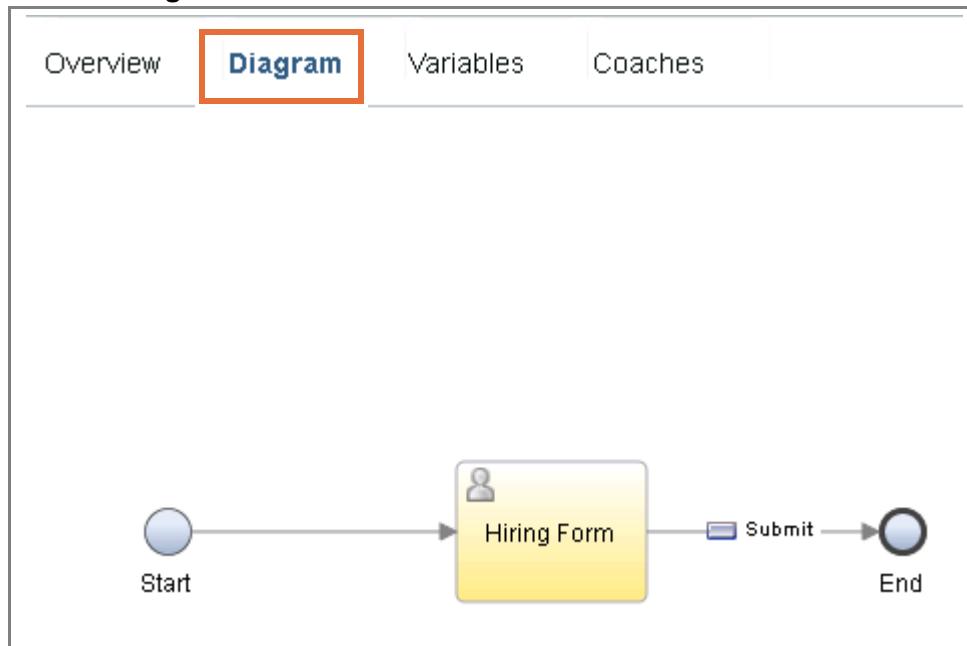


- __ c. Click the (+) plus sign next to Output.

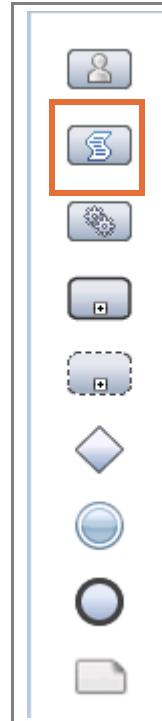
- __ d. Add the variable: `isNewPosition (String)`

The screenshot shows the 'Variables' dialog box. On the left, there are categories: 'Input' (with 'requisitionDetails (HiringRequisition)'), 'Output' (with 'requisitionDetails (HiringRequisition)' and 'isNewPosition (String)', where 'isNewPosition (String)' is highlighted with a red box), 'Private', 'Exposed Process Variables', 'Environment Variables', and 'Localization Resources'. On the right, there are buttons for adding (+), moving up (↑), moving down (↓), and deleting (X).

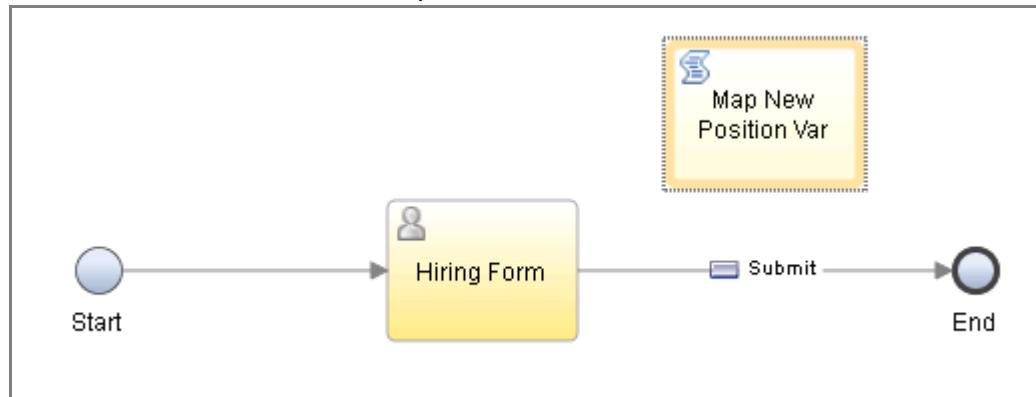
- __ e. Click the **Diagram** tab.



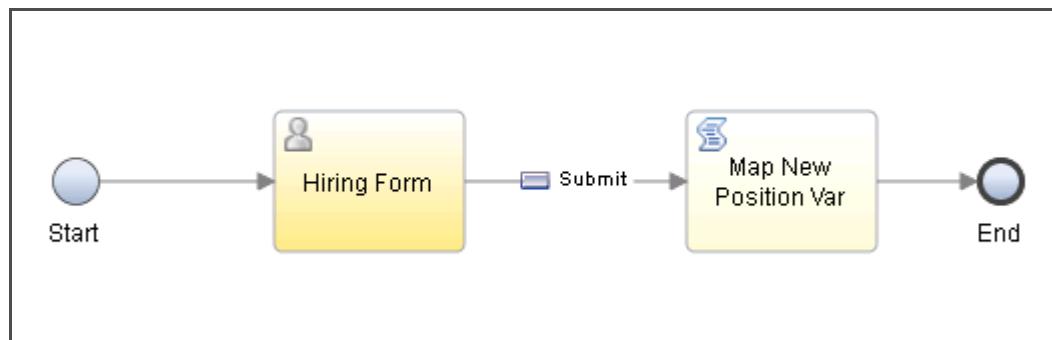
- ___ f. Drag a **Client-Side script** between the **Hiring Form** coach and the **End** event.



- ___ g. Name the new client-side-script: **Map New Position Var**

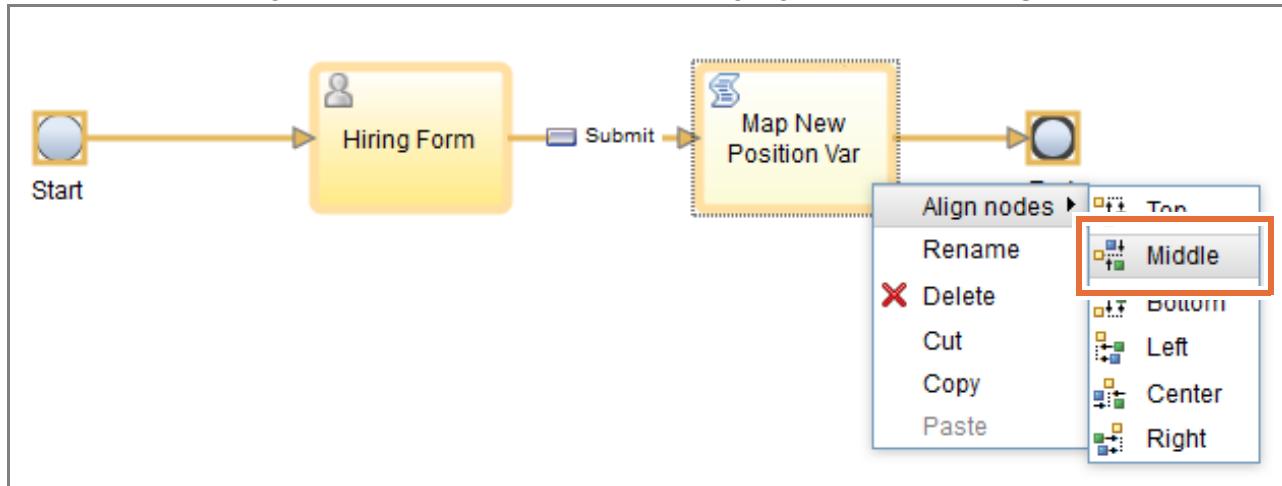


- ___ h. Connect the flow between the **Hiring Form** and **Map New Position Var** script and then to the **End** event.



**Hint**

To straighten the lines when building a service, press Ctrl+A on the keyboard to select all the items on the canvas. Right-click one of the items that are highlighted and click **Align Nodes > Middle**.



- i. Click the **Map New Position Var** server script object that you placed on the canvas.
- j. Open the **Properties > Script** menu.

Properties Validation Errors

General

Script

Pre & Post

This editor uses **standard JavaScript syntax**. Press **Ctrl-space** while typing to receive assistance

```

1 // This JavaScript will run within the browser and must use the client-side JavaScript syntax such as:
2 //      - To instantiate a complex object:           - To instantiate a list:
3 //          tw.local.customer = {};
4 //          tw.local.customer.name = "Jane";
5 //          tw.local.addresses = [];
6 //          tw.local.addresses[0] = {};
  
```

- ___ k. Copy and paste the following code in the **Script** field. You can copy the code from the file `Script1.txt` at the location:

`C:\labfiles\Exercise_Support_Files\Exercise06`

```
if (tw.local.requisitionDetails.recruitingDetails newPosition == true)
tw.local.isNewPosition = "1";
else
tw.local.isNewPosition = "0";
```

The screenshot shows the 'Properties' panel with the 'Script' tab selected. A note states: 'This editor uses standard JavaScript syntax. Press Ctrl-space while typing to receive assistance with script syntax and contents.' Below is the script code:

```
1 // This JavaScript will run within the browser and must use the client-side JavaScript syntax supported by the browser.
2 // - To instantiate a complex object: tw.local.customer = {};
3 // - To instantiate a list: tw.local.addresses = [];
4 // tw.local.customer.name = "Jane"; tw.local.addresses[0] = {};
5 if (tw.local.requisitionDetails.recruitingDetails newPosition == true)
6 tw.local.isNewPosition = "1";
7 else
8 tw.local.isNewPosition = "0";
```

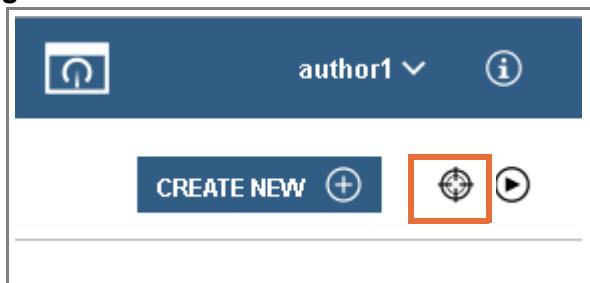
Optional

The client-side script box has some example code commented out. You can erase the code, or you can add the code below the commented code.

Note

The first time that you input the script into the field, the browser might try to point out JavaScript errors. If you click somewhere away from the script and then select it again, the errors disappear.

- ___ l. Save your work.
 ___ 6. Debug the **Hiring Form** client-side human service and verify that data is flowing.
 ___ a. Click the **Debug** icon.



- ___ b. A new browser window is displayed, but this time it is in debug mode.

This window is hosting your debug session, and must remain open while you are debugging.

The running client-side human service is paused at the first activity. To proceed with the debugging of the client-side human service, use the actions on the sidebar of the Inspector.

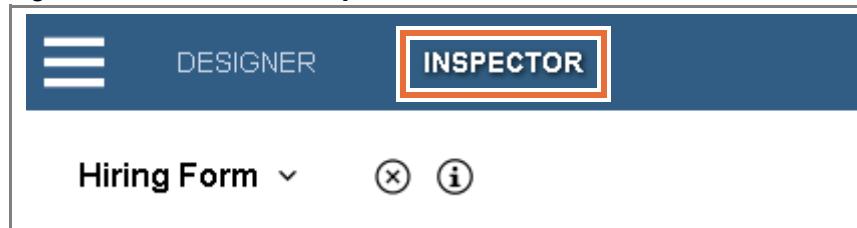
If this window is closed, use the 'Show Playback Window' action to reopen it.

- ___ c. Minimize this new browser window to view the client-side human service designer window.

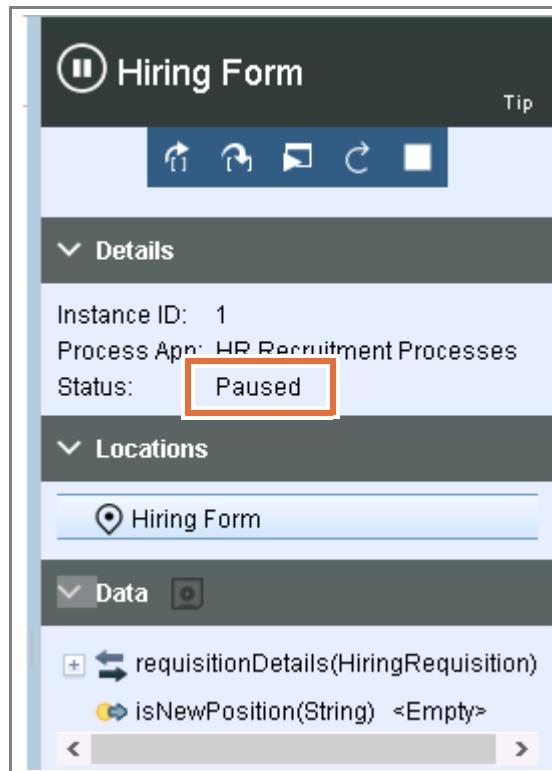


Information

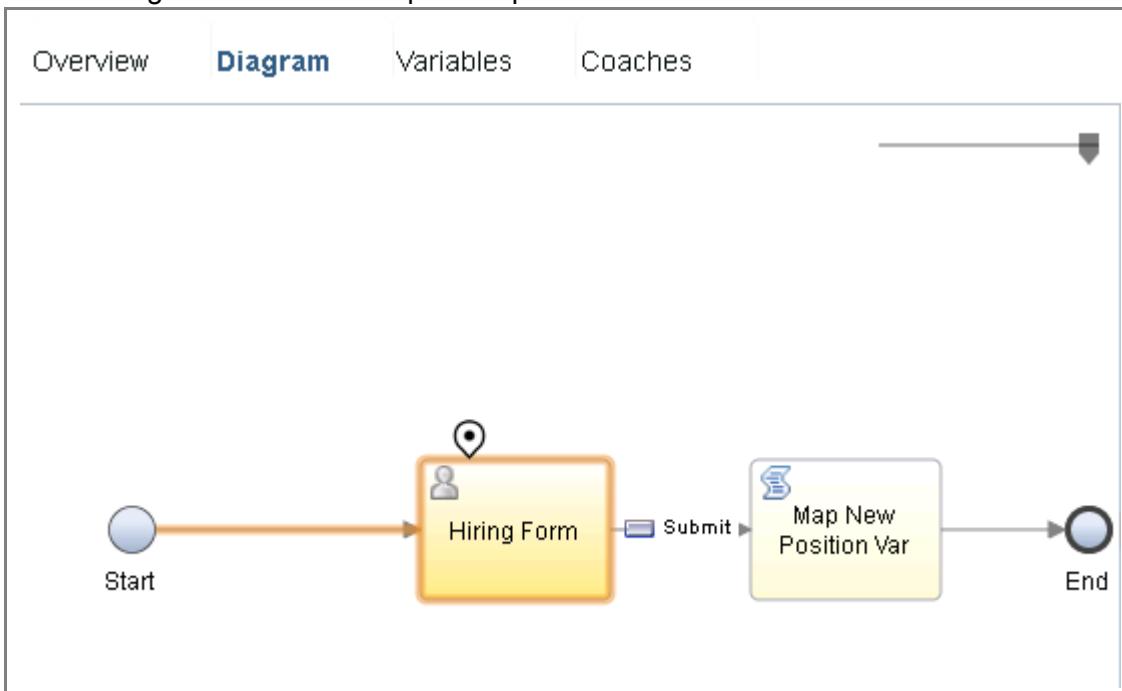
The coach designer switches to the **Inspector** tab.



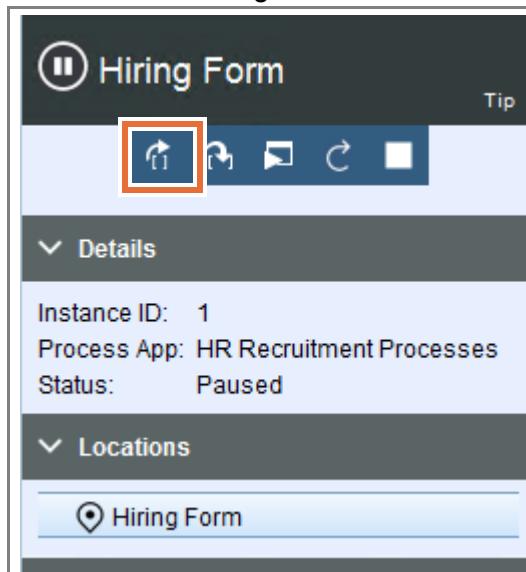
- ___ d. The inspector section is displayed on the right side. The status shows that the execution is Paused.



- __ e. If not already selected, click the **Diagram** tab. The inspector shows a token that is waiting to execute this step in the process.



- __ f. Click the **Step over** icon for the Hiring Form coach.



- __ g. The second browser window renders the coach. Enter 12345 as the Requisition Number. Ensure that the **NewPosition** check box is not selected.

Requisition Number	
12345	
Requester	
Date Details	
Date Of Request	
Date Position Available	
Recruiting Details	
<input type="checkbox"/> Multiple Employees Needed	
Num Employees Needed	
0	
<input type="checkbox"/> New Position	
Department Details	
Position Details	
Job Title	
Job Description	
Job Level	
Num Direct Reports	
0	
Compensation Details	
Salary To Offer	

h. Click **Submit**.

Department	Bonus Amount
<input type="text"/>	<input type="text"/>
Division	<input type="text"/>
Hiring Manager Comments	
<input type="button" value="Submit"/>	

i. Minimize the coach browser window and return to the designer browser window.

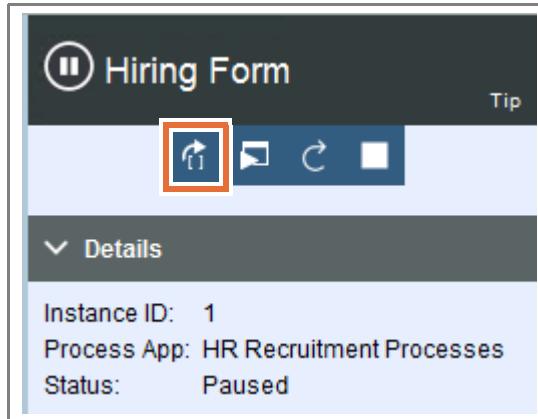


Information

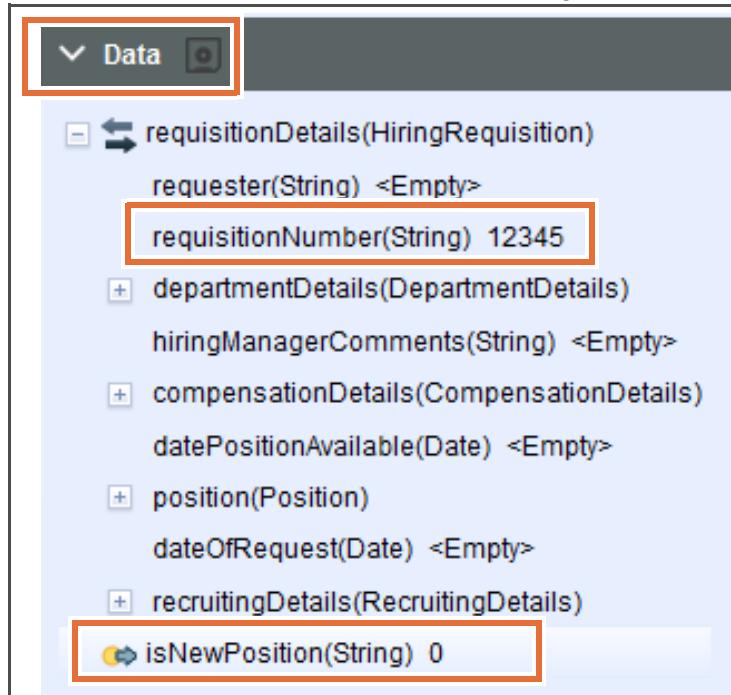
The Hiring Form coach is complete, and the token moves to **Map New Position Var**.



- __ j. The status is **Paused**, waiting for the next step in the service. Click **Step over** to run the Map New Position Var server script.



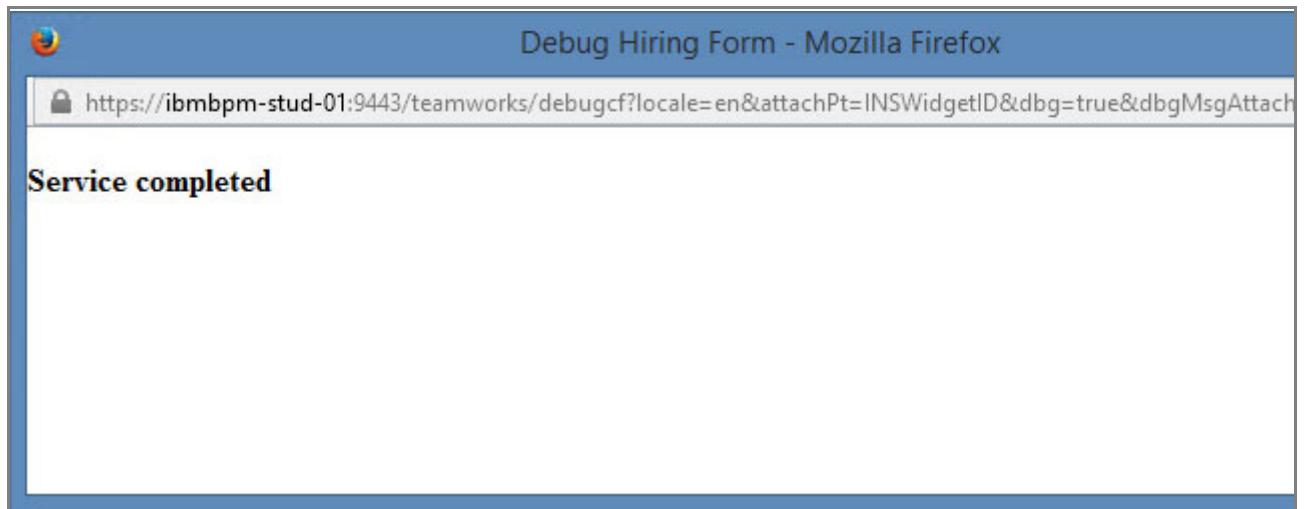
- __ k. The status changes to **Finished** and the token disappears because it reached the end event. On the right, expand the **Data** section and click the (+) plus sign next to **requisitionDetails** to see the updated value for the `requisitionDetails.requisitionNumber` variable. Verify that the updated value for the `requisitionDetails.requisitionNumber` variable is 12345. The `tw.local.isNewPosition` variable is set to the String "0".



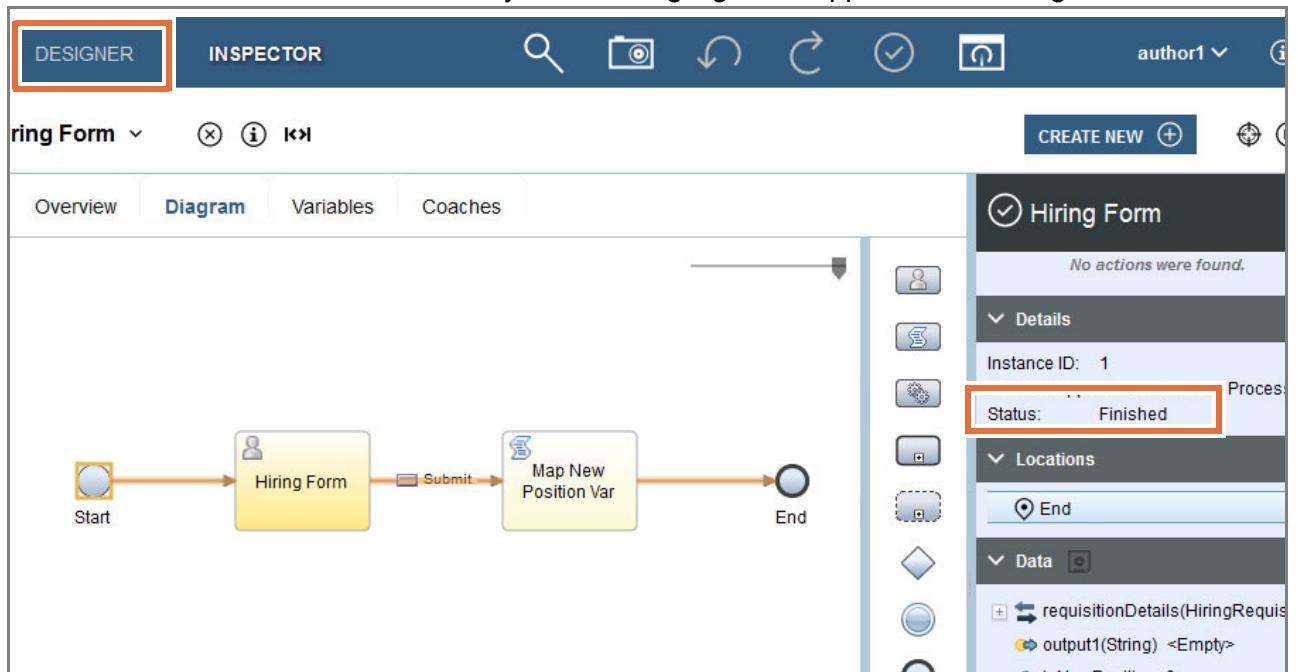
Optional

Run the service again, and this time select the **New Position** check box on the coach. The `isNewPosition` value should be set to 1.

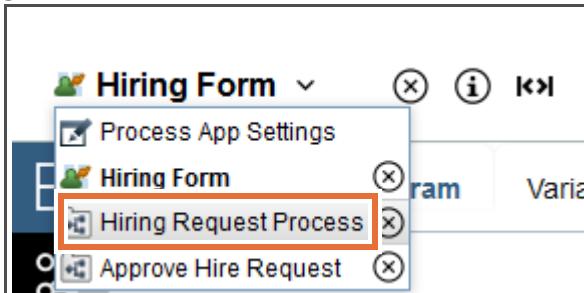
- __ I. Return to the Debug Hiring Form and verify that the service is completed. Close this browser window.



- __ m. The Status is **Finished**, and you are currently on the **INSPECTOR** tab. Return to the **DESIGNER** tab. The activity and flow highlights disappear in the Designer mode.



- __ 7. Map the new variable in the process.
__ a. Open the **Hiring Request Process** process.



- ___ b. Click the **Submit Hiring Request** activity.
- ___ c. Open the **Properties > Data Mapping** menu and map the `isNewPosition` output variable to `tw.local.isNewPosition`.



- ___ d. Click the **Approve New Hire Request** activity.
- ___ e. Open the **Properties > Data Mapping** menu and map the `isNewPosition` output variable to `tw.local.isNewPosition`.



- ___ f. Save your work.



Important

Playback 1: Business data and services is complete.

As part of the development process, you now review the playback and examine its functions in the Process Portal.

Test the following functions:

- When you create an instance of the Hiring Request Process, does it display your coach?
- Are all the fields present on the coach?

End of exercise

Exercise review and wrap-up

This exercise looked at creating the assets that are required for a coach. These assets include complex business objects, services, a coach form, and mapping.

Exercise 7. Playback 1: User interface design and implementation

Estimated time

01:00

Overview

In this exercise, you use group controls into tabs on a coach and change the appearance of the coach by applying a theme.

Objectives

After completing this exercise, you should be able to:

- Create tabs on a coach
- Change the appearance of a coach by applying a custom theme
- Change the coach layout for a mobile format

Introduction

In Playback 1: User interface design and implementation, the enhancements that the business wants for process application user interfaces happen at this stage. Often, the initial development efforts are marred with requests to begin with full fidelity user interfaces. It is not uncommon to have this request because it is driven from a desire to impress executive level stakeholders with prototypical user interfaces that can be manipulated. It is better to reserve this type of development for this playback stage. You ensure that the data model is in place, the functions of the user interface are approved, and all the remaining process application interactions and integrations are complete. Now the enhancements can be done without fear that rework would be needed later because of a change in requirements to the items in the prior playbacks. Changes can happen, but the likelihood is that those functional requirement changes are handled in Optimization and not within the development cycle because consensus is reached to move to this stage of development.

Requirements

Successful completion of the previous exercise is required.

Exercise instructions

Part 1: Group controls into tabs on a coach

In one of the previous playbacks, you provided a functional look at the coaches in the Hiring Request Process. Enhance the coaches with more features and functions. Start with creating a polished look to the coach by grouping similar data into tabs.

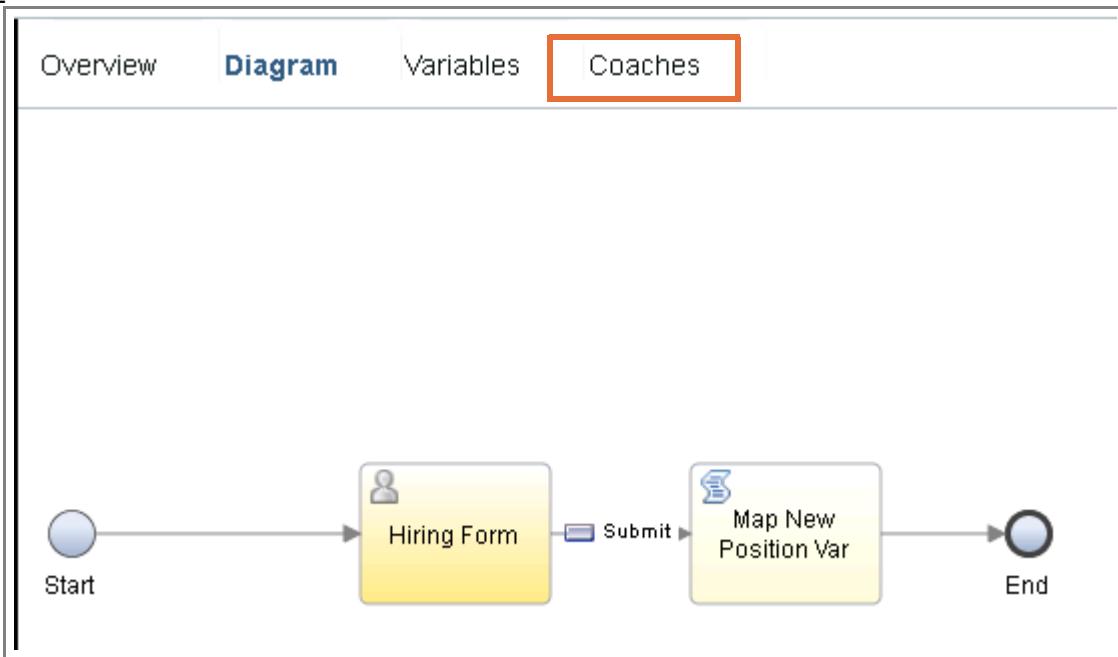


Reminder

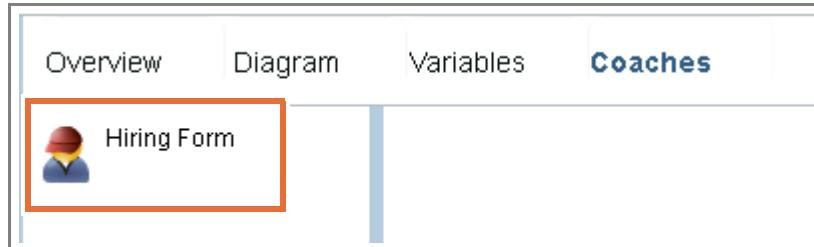
At certain times when working inside the web-based Process Designer, the session times out due to inactivity. If at any time during this course you see an expired session error, insert the same credentials that are shown (author1 and author01) and click **Log In**.

- 1. Create tabs to hold the grouped sections on the coach.
 - a. In the Process Designer library, open the **User Interface > Hiring Form**.

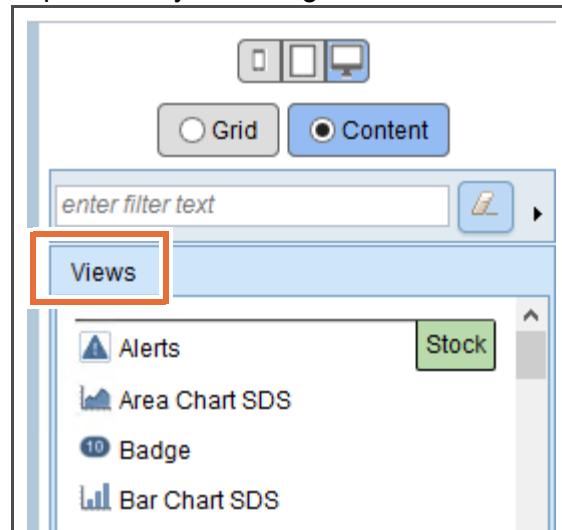
- b. Click the **Coaches** tab.



- c. Click the **Hiring Form** coach.



- d. Expand the **Views** palette tray on the right.



- __ e. Drag a **Tab Section** control onto the canvas in the top grid of the coach.

The screenshot shows the coach interface with a tab section control. The tab section has two tabs: "Date Details" and "Position Details". Under "Date Details", there is a field labeled "Date Of Request". Under "Position Details", there are fields labeled "Job Title" and "Job Description". On the right side, there is a toolbar with icons for Grid, Content, enter filter text, Views, Step Chart SDS, CSS Style, Switch, Tab Section, Table, Table Layout, Table Layout Cell, and Table Layout Row. The "Tab Section" icon is highlighted with a red border.

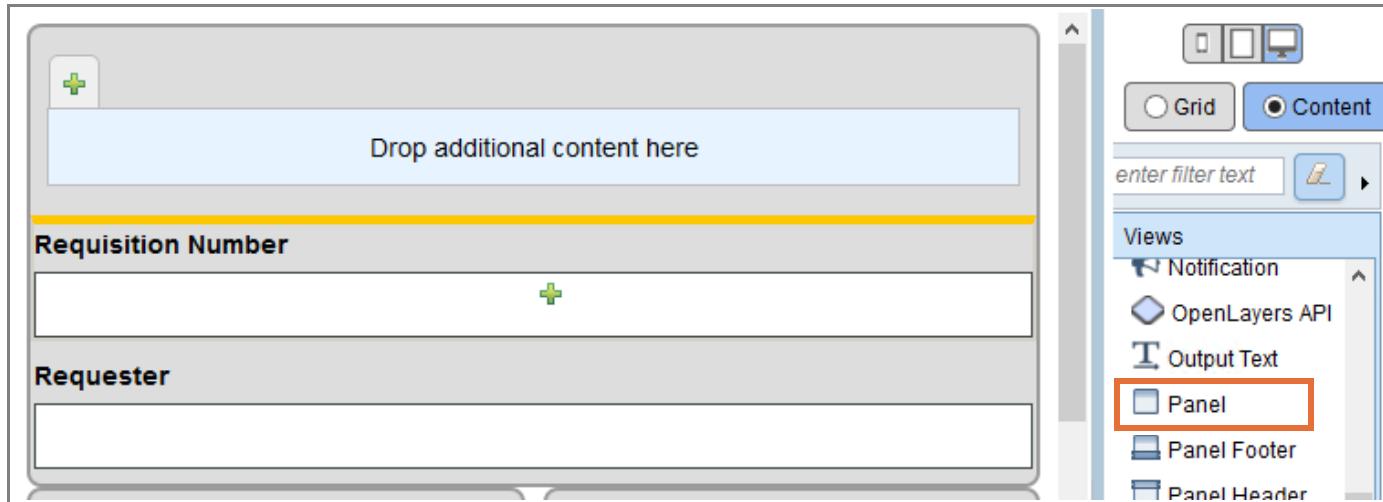


Information

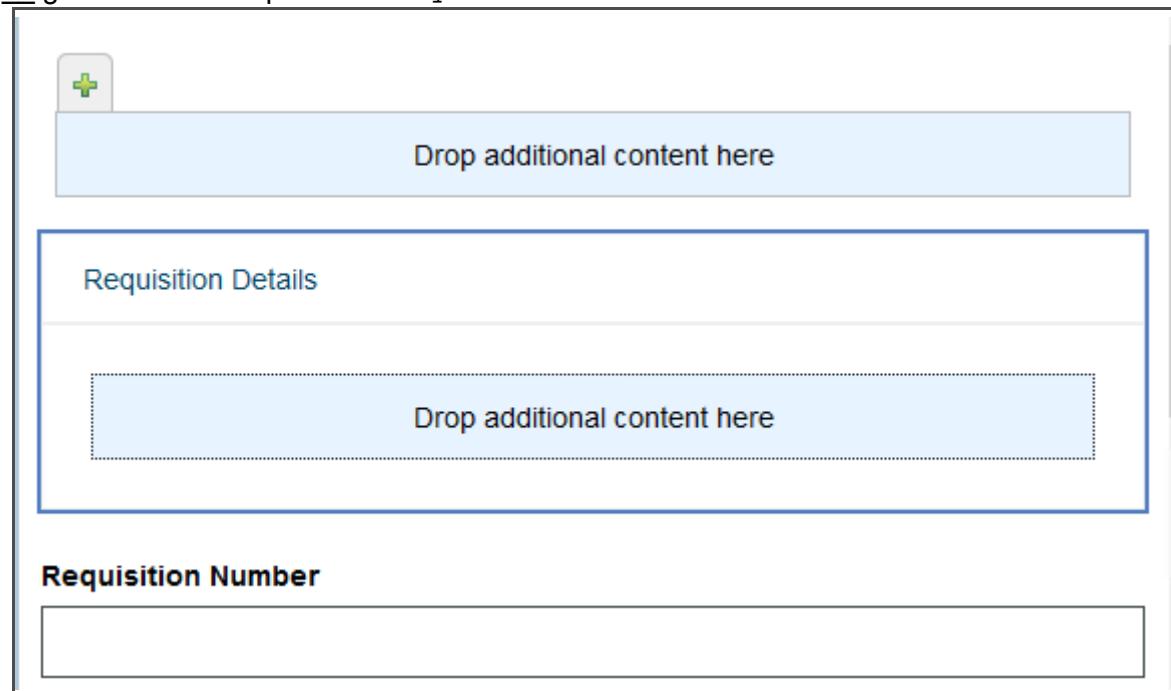
The tab section should look like this image at the top grid of the coach.

The screenshot shows the coach interface with the tab section control placed in the top grid. The tab section has two tabs: "Requisition Number" and "Requester". A placeholder text "Drop additional content here" is visible in the top grid area. On the right side, there is a toolbar with icons for Grid, Content, enter filter text, Views, Step Chart SDS, CSS Style, Switch, Tab Section, Table, Table Layout, and Table Layout Row. The "Tab Section" icon is highlighted with a red border.

__ f. Drag a **Panel** control above the Requisition Number input control.



__ g. Rename the panel to: Requisition Details



- h. Drag the **Requisition Number** and **Requester** controls from the coach onto the area that is labeled **Drop additional content here** under the Requisition Details panel.

Requisition Details

Requisition Number

Requester

Data Details



Hint

You can switch to the Grid view to verify the position of the input controls. The Requisition Number and Requester controls are part of Requisition Details section.

Drop additional content here

Requisition Details

Requisition Number

Requester

Grid

Content

enter filter text

Grid

- Single Column
- Two Columns
- Three Columns
- Mosaic
- Header Footer

- i. Drag the **Date Details** panel that contains the **Date of Request** and the **Date Position Available** inputs into the editable area of the panel under **Requester**.

Requisition Details

Requisition Number

Requester

Date Details

Date Of Request

Date Position Available

Position Details

Job Title

Job Description

Job Level



Information

Your coach should look like this image after you add the Date Details panel.

The screenshot shows a user interface for 'Requisition Details'. It consists of several input fields:

- Requisition Number:** An input field with a light gray background.
- Requester:** An input field with a light gray background.
- Date Details:** A section header followed by two input fields. This entire section is highlighted with a thick orange border.
- Date Of Request:** An input field with a light gray background.
- Date Position Available:** An input field with a light gray background.

- __ j. Place the **Hiring Manager Comments** control underneath the **Date Details** panel in the **Requisition Details** panel.

The screenshot shows a user interface with a main panel titled "Requisition Details". Inside this panel, there are two input fields: "Date Of Request" and "Date Position Available". Below these is a section labeled "Hiring Manager Comments", which is enclosed in a red rectangular border, indicating it is the control to be moved.



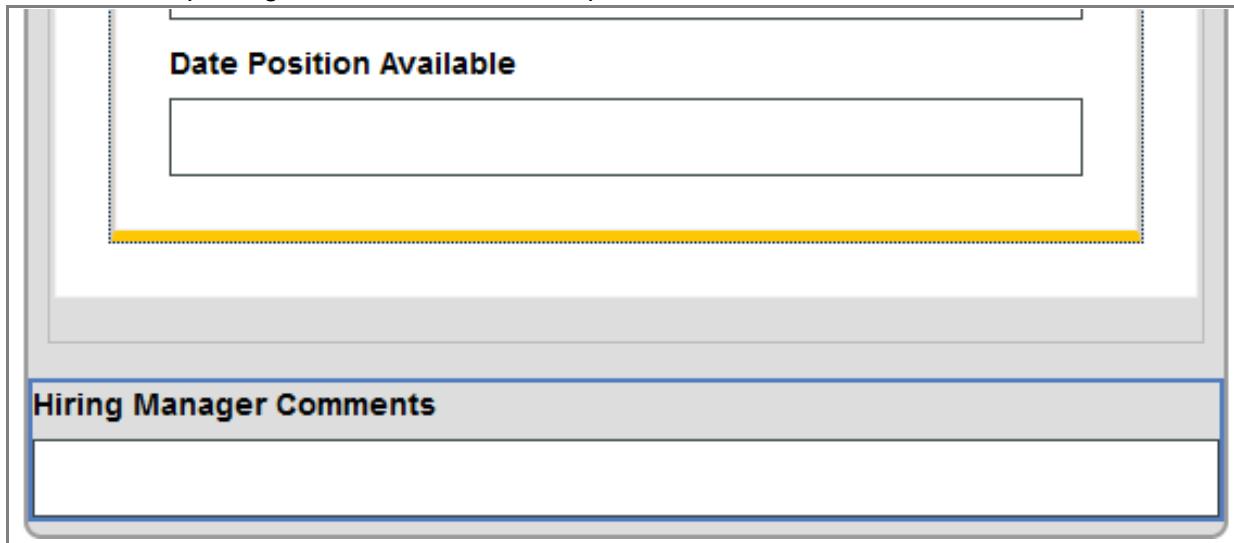
Note

It is easy to place a control nested in a section that it should not be a part of. It is best to try to find the border of the section or grid that you want to place it in. Then, carefully find the border between the areas where you want to place your object.

- __ k. Click the **Requisition Details** header section and drag the entire section into the tabs area that is labeled **Drop additional content here**. The label on the tab changes to match the label that is applied to the section, and a second tab opens.

The screenshot shows the same "Requisition Details" panel as before. The "Requisition Details" header is now highlighted with a red border. To its right is a tab labeled "Drop additional content here". Below the tabs, there is a section titled "Date Details" containing two input fields: "Date Of Request" and "Date Position Available".

- I. Click to view the **Requisition Details** tab. If they aren't already in the section, drag both the **Date Details** panel and the **Hiring manager Comments** into the Requisition Details tab, placing each control under the previous control.





Information

Your Requisition Details tab should look like this image.

The screenshot shows a user interface for entering requisition details. At the top, there is a tab labeled "Requisition Details" with a green plus sign icon to its right. Below the tabs, there are several input fields:

- Requisition Number:** An empty text input field.
- Requester:** An empty text input field.
- Date Details:** A section header followed by an empty text input field.
- Date Of Request:** An empty text input field.
- Date Position Available:** An empty text input field.
- Hiring Manager Comments:** An empty text input field.

- m. Click the (+) plus sign on the tab next to the **Requisition Details** tab.

The screenshot shows the same user interface as above, but with a red box highlighting the green plus sign icon on the "Requisition Details" tab. Below the tabs, there is a light blue rectangular area containing the text "Drop additional content here".

**Note**

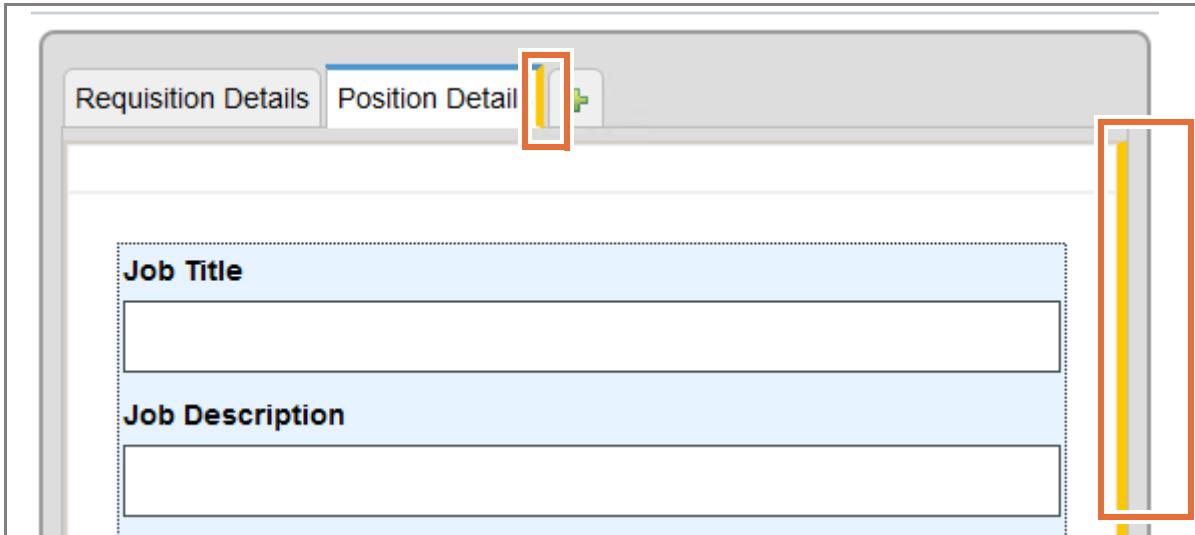
The section that you added in the Requisition Details tab is still there, but that disappears as soon as you click other tab. The tab contents appear again when you return to view the Requisition Details tab.

- n. Move the **Position Details** section onto the tab. This one is easier because all the controls are already in a single section.

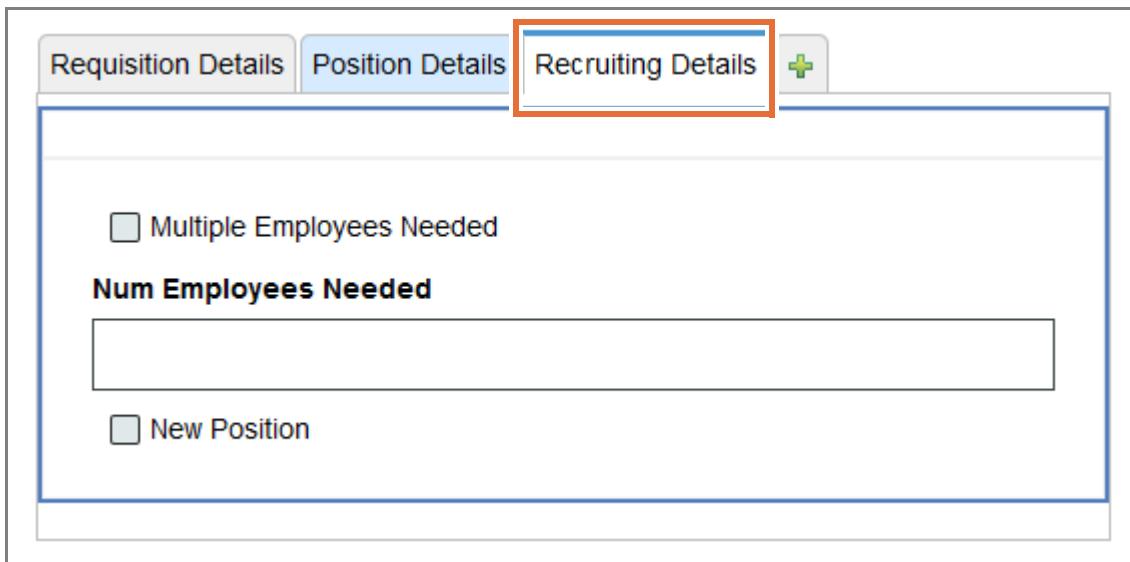
The screenshot shows a user interface with a tabbed panel. The 'Position Details' tab is selected and highlighted with a red border. Below the tabs, there are four input fields: 'Job Title', 'Job Description', 'Job Level', and 'Num Direct Reports'. Each field has a corresponding empty input box below it.

Section	Input Box
Job Title	[Empty Input Box]
Job Description	[Empty Input Box]
Job Level	[Empty Input Box]
Num Direct Reports	[Empty Input Box]

- o. Create a tab for the **Recruiting Details**. This time, drag the header for the **Recruiting Details** panel to the right of the **Position Details** tabs. Notice that an orange line indicates where the new tab is placed when you release the mouse button.



- p. Release the mouse button, and the section becomes the Recruiting Details tab next to the Position Details tab.

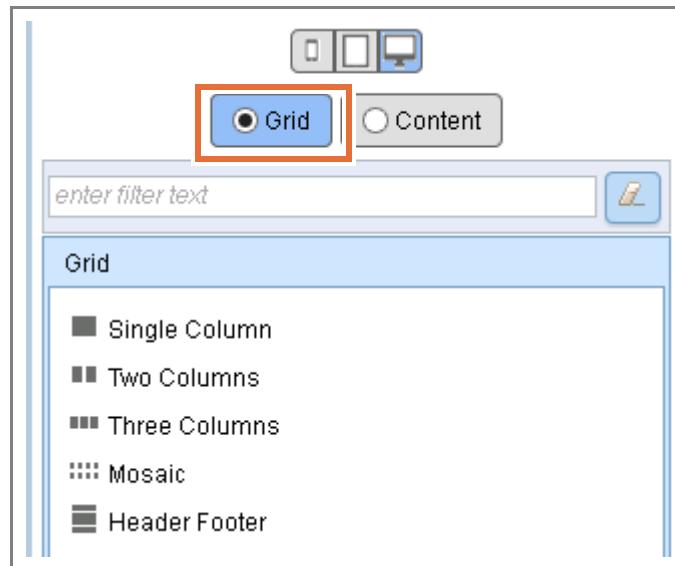


- __ q. Continue to create tabs for the **Compensation Details** and **Department Details**. Drag each of the respective sections onto their tabs.

The screenshot shows a horizontal navigation bar with five tabs: Requisition Details, Position Details, Recruiting Details, Compensation Details, and Department Details. The Department Details tab is currently selected, indicated by a blue border around its text. Below the tabs, there is a large rectangular container with a thin blue border. Inside this container, there are two text input fields. The top field is labeled "Department" and the bottom field is labeled "Division". Both fields have a placeholder text area below them.

- __ 2. Rearrange the empty grids.

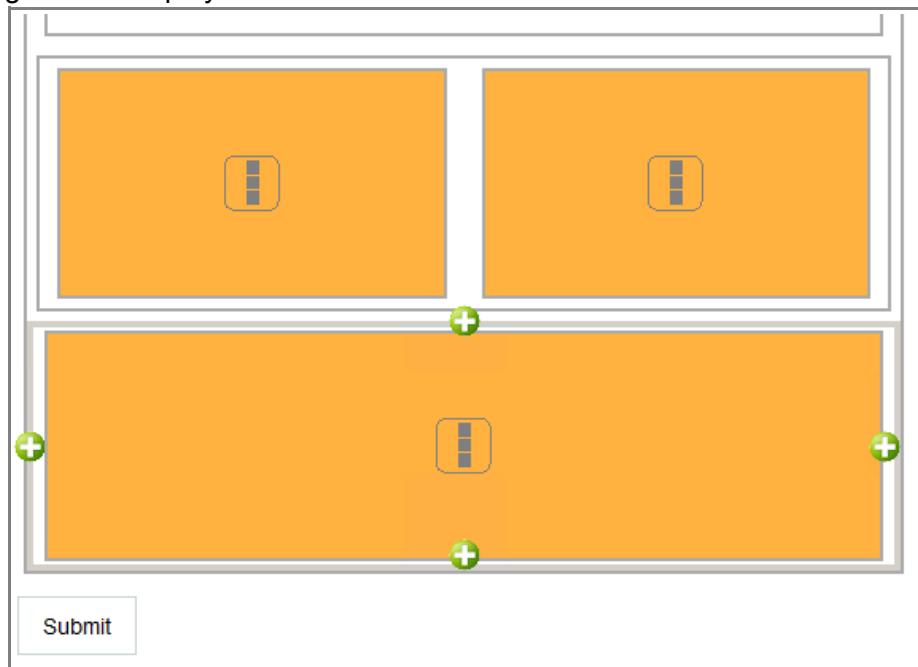
- __ a. Return to the **Grid** view.



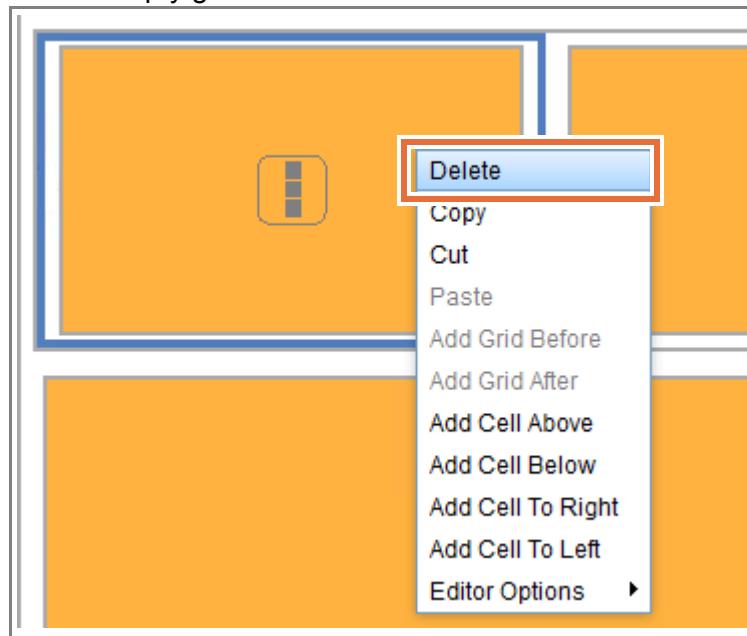


Information

The empty grids are displayed.



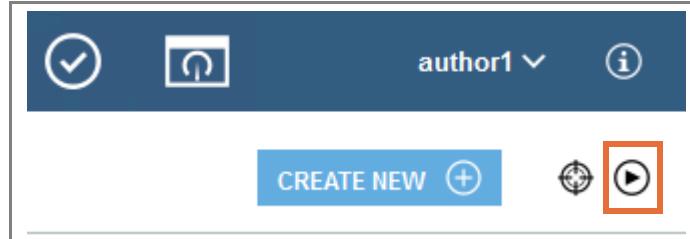
- __ b. Right-click each empty grid and click **Delete**.



- c. Now only the grid with the tabs is left with the Submit button at the bottom.

The screenshot shows a user interface with a grid layout. At the top, there are five tabs: 'Requisition Details', 'Position Details', 'Recruiting Details', 'Compensation Details', and 'Deparment Details'. The 'Deparment Details' tab is currently selected. Below the tabs is a large text input field with the placeholder 'Drop additional content here'. There are four green plus icons with '+' signs, one in each corner of the grid area. At the bottom left is a 'Submit' button.

- ___ d. Save your work.
 ___ 3. View the coach.
 ___ a. Click **Run** at the top of the coach.



- ___ b. The coach now has multiple tabs. Click the tabs to see what is contained on each tab.

The screenshot shows a coach interface with five tabs at the top: 'Requisition Details', 'Position Details', 'Recruiting Details', 'Compensation Details', and 'Deparment Details'. The 'Recruiting Details' tab is selected. The main content area displays a form for 'Recruiting Details'. It includes a checkbox for 'Multiple Employees Needed', a text input field for 'Num Employees Needed' containing '0', and a checkbox for 'New Position'. At the bottom left is a 'Submit' button.

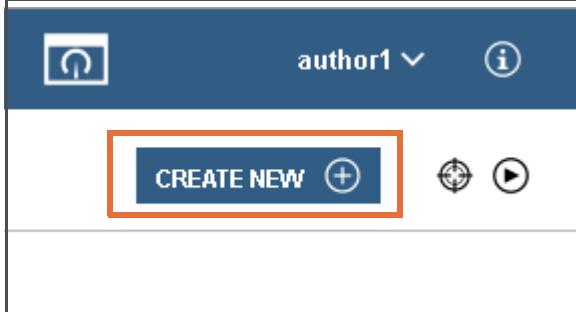
- ___ c. Close the browser that displayed the form.

Part 2: Change the appearance of the coach by applying a custom theme

Creating a theme consists of assigning values to a set of theme variables. By default, a new process application uses the BPM Theme because it contains all the variables that the controls in the Responsive Coaches toolkit use. To create a custom theme and use any of these responsive controls in your process application, you use the BPM Theme as your starting point. You can then extend your theme by adding custom variables and modifying the BPM variables.

— 1. Create a custom theme for the HR Recruitment Processes process application.

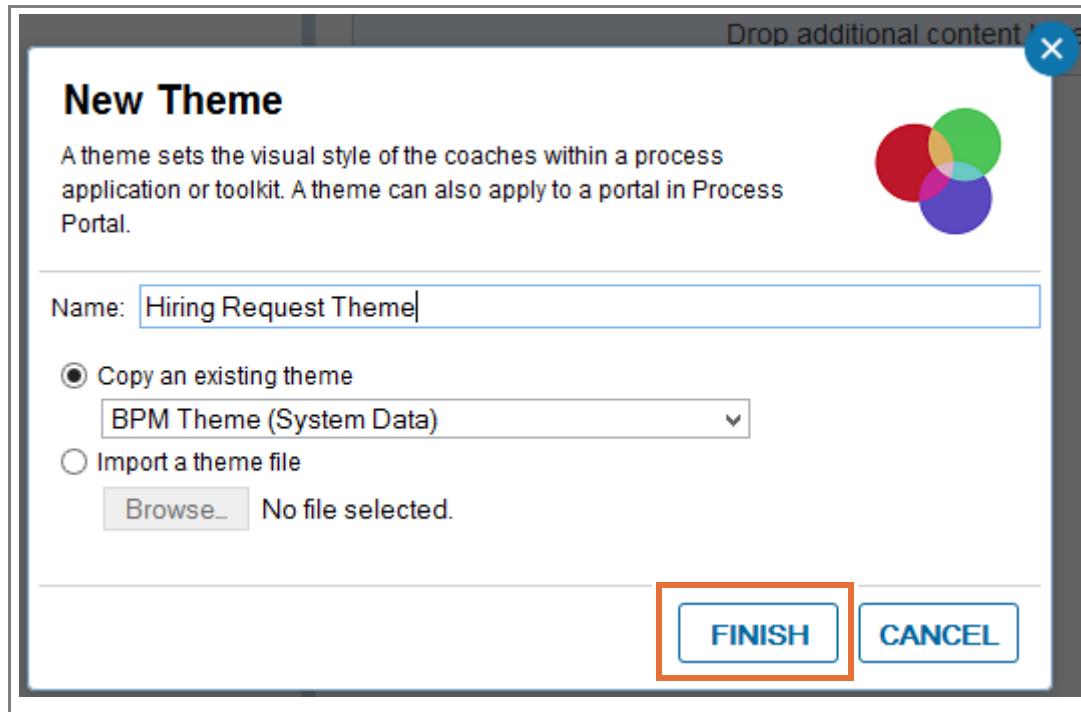
— a. Click **Create New** on the upper-right corner of the Process Designer.



— b. Click **Theme**.



- c. Enter the name: Hiring Request Theme
Leave the option **Copy an existing theme: BPM Theme (System Data)** selected and click **Finish**.



Important

If you are importing a theme and you are using responsive controls, ensure that it contains all of the BPM variables, the comments, and the metadata. If the theme does not contain these variables and you use BPM controls, an error occurs when the system generates CSS for the process application. For this reason, base your theme on the BPM Theme or another theme that you know has all of the BPM variables. The BPM variables start with `bpm` and they are reserved.

- __ d. The editor opens the new theme.

Variable	Value
@bpm-neutral-darkest	#0d1111
@bpm-neutral-darker	#2d3737
@bpm-neutral-dark	#3c4646
@bpm-neutral	#586464
@bpm-neutral-light	#6d7777
@bpm-neutral-lighter	#c8d2d2
@bpm-neutral-lightest	#dfe9e9
@bpm-color-primary	#325C80
@bpm-color-info	#c0e6ff
@bpm-color-success	#c8f08f
@bpm-color-warning	#fde876
@bpm-color-alert	#ad1625
@bpm-color-alert-light	#ffd2dd
@bpm-body-bg	#fff
@bpm-text-color	#000203
@bpm-link-color	@bpm-color-primary
@bpm-font-family-sans-serif	"Helvetica Neue", Helvetica, Arial, sans-serif
@bpm-font-size-base	14px
@bpm-line-height-base	1.5
@bpm-line-height-small	1.6
@bpm-padding-base-vertical	6px

- __ 2. Change the primary color of the theme

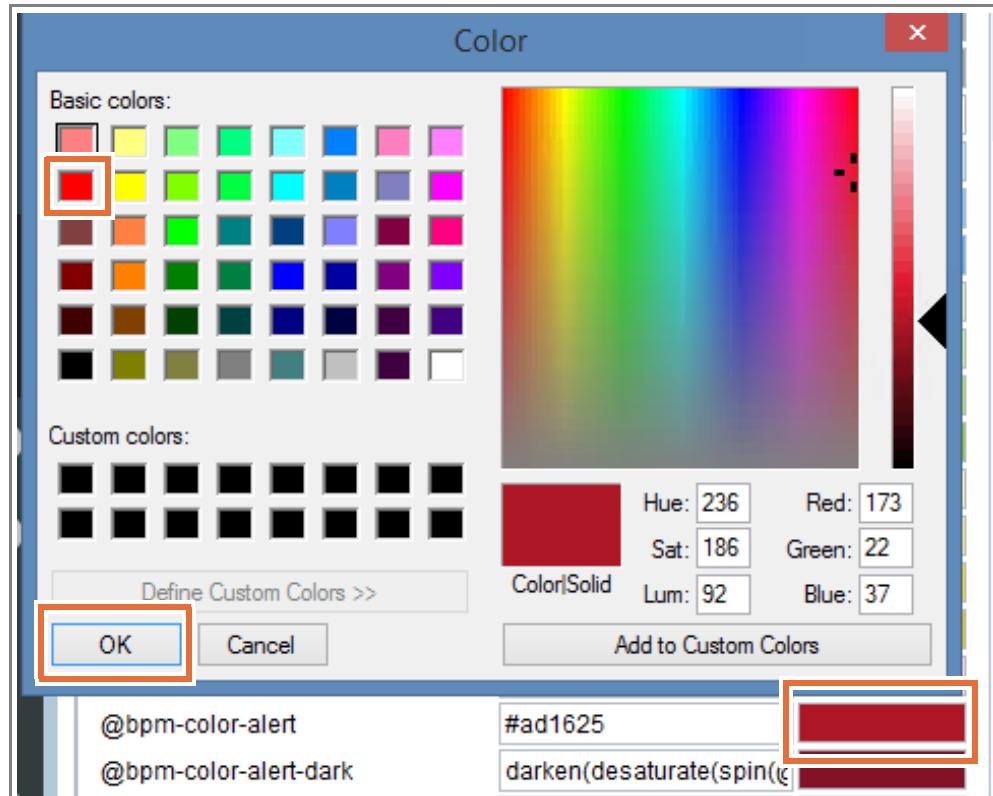
Management wants to change the color scheme to match the corporate color scheme. Change the responsive controls of the default BPM Theme and apply the new theme to the process.

- __ a. In the Design page of the theme editor, change the variable `@bpm-color-primary` in the **Base Settings** section from `#325C80` to: `#bcc2c5`

Variable	Value
@bpm-neutral-darkest	#0d1111
@bpm-neutral-darker	#2d3737
@bpm-neutral-dark	#3c4646
@bpm-neutral	#586464
@bpm-neutral-light	#6d7777
@bpm-neutral-lighter	#c8d2d2
@bpm-neutral-lightest	#dfe9e9
@bpm-color-primary-light	lighten(desaturate(spin(6)))
@bpm-color-primary	#bcc2c5
@bpm-color-primary-dark	darker(desaturate(spin(6)))

**Note**

Many variables also have a swatch that you can click, then choose a value directly from a picker, and click **OK**.



- b. When you “blur” or click away from the color input field, you immediately see the color change for the affected controls on the right. Click any control on the right to see the CSS that affects the appearance of that object on the page.

@bpm-color-primary	#bcc2c5	
@bpm-color-primary-dark	darker(desaturate(spin(@bpm-color-primary, 0, 100), 10))	
@bpm-color-primary-darker	darker(desaturate(spin(@bpm-color-primary-dark, 0, 100), 10))	
@bpm-color-info-light	lighten(@bpm-color-info, 10%)	
@bpm-color-info	#c0e6ff	
@bpm-color-info-dark	darker(spin(@bpm-color-primary, 0, 100), 10%)	
@bpm-color-info-darker	darker(desaturate(spin(@bpm-color-primary, 0, 100), 10), 10%)	
@bpm-color-success-light	lighten(@bpm-color-success, 10%)	
@bpm-color-success	#c8f08f	

Radio Button Group

 Option 1
 Option 2

Date Time Picker

Tuesday, May 9, 2017

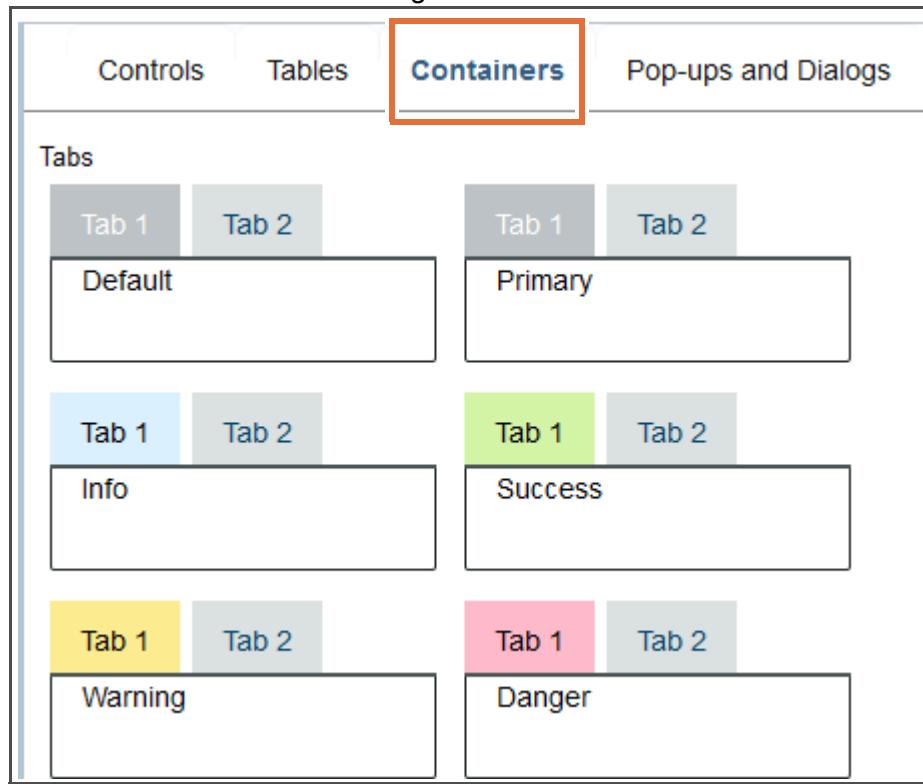
Font family: @bpm-font-family-base
 Font size: @bpm-font-size-base
 Font color: @bpm-fill-normal
 Background color: @bpm-color-primary
 Border color: @bpm-neutral

Check Box Group

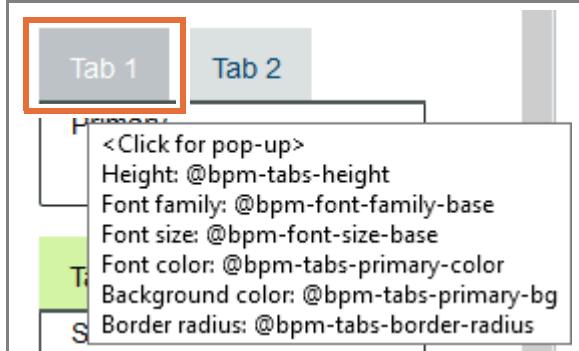
 Option 1
 Option 2

— 3. Change the font color of the tabs.

— a. Click the **Containers** tab on the right.



— b. Hover over the Primary Tabs control in the first row in the right column. It shows all the variables that are associated with this control. For example, to change the font color of the tabs, you need to change the value of the `@bpm-tabs-primary-color` variable.



- __ c. To change the font color, collapse the **Base Settings** section and expand the **Tabs** section.

The screenshot shows the 'Design' tab selected in a UI builder interface. On the left, there's a navigation sidebar with the following items:

- ▶ Base Settings (highlighted with an orange box)
- ▶ Fonts
- ▶ Forms
- ▶ Buttons
- ▶ Checkbox : Switch
- ▶ Section
- ▶ Table
- ▼ Tabs (highlighted with an orange box)

Below the sidebar, under the 'Tabs' section, are several CSS variables and their corresponding values and color swatches:

@bpm-nav-link-padding	10px 15px
@bpm-nav-tabs-border-color	@bpm-neutral-dark
@bpm-tabs-default-color	@bpm-text-color-negative
@bpm-tabs-default-bg	@bpm-color-primary

- __ d. Look for the variable `@bpm-tabs-primary-color` and change the color to: #008000

The screenshot shows a table of CSS variables and their values. One variable, `@bpm-tabs-primary-color`, is highlighted with an orange box and has its value changed to `#008000`.

@bpm-tabs-default-accent	desaturate(lighten(@bpm-
@bpm-tabs-default-content-ba	@bpm-fill-normal
@bpm-tabs-primary-color	#008000
@bpm-tabs-primary-simple-color	@bpm-text-color
@bpm-tabs-primary-bg	@bpm-color-primary
@bpm-tabs-primary-border	darker(@bpm-tabs-prima



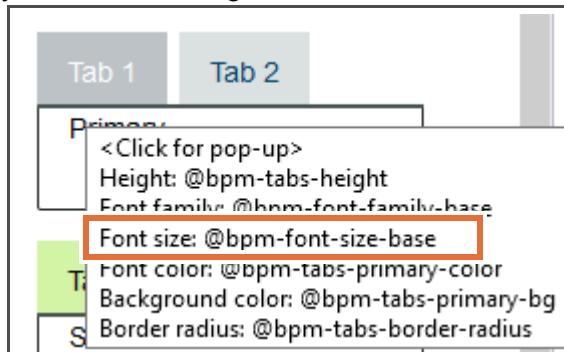
Hint

You can choose to use any color of your choice. If you run into any problem, return to the step and follow that exactly, and then you can customize the color.

- __ 4. Change the font size.

**Hint**

To change the font size, you need to change the value of the `@bpm-font-size-base` variable.



- ___ a. Return to the **Base Settings** section.
- ___ b. Look for the variable `@bpm-font-size-base` and change the value to: 16px

<code>@bpm-link-color</code>	<code>@bpm-color-primary</code>	
<code>@bpm-text-color</code>	<code>#000203</code>	
<code>@bpm-font-family-sans-serif</code>	"Helvetica Neue", Helvetica	Lorem Ipsum
<code>@bpm-font-size-base</code>	16px	
<code>@bpm-line-height-base</code>	1.5	
<code>@bpm-line-height-small</code>	1.6	

- ___ 5. Change the background color of the input controls.

- ___ a. Click the **Controls** tab on the right to view the input fields.

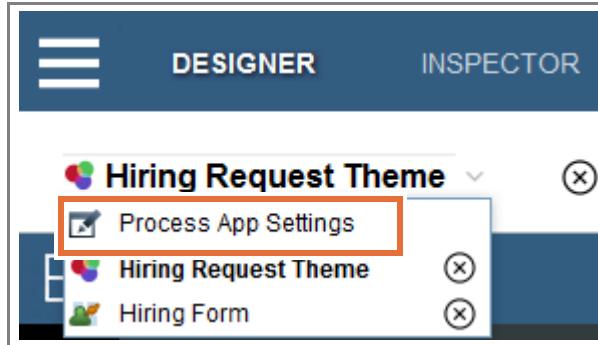
The screenshot shows the 'Controls' tab selected in a UI design application. Below the tabs, there are three examples of input fields: 'Input' (a standard text input), 'Input Small' (a smaller text input), and 'Input Large' (a large text input). The 'Controls' tab is highlighted with a red box.

- ___ b. Expand the **Forms** section on the left.

- __ c. Change the variable `@bpm-input-bg` to: `#d9ece3`

Variable	Value	Preview
<code>@bpm-label-font-family</code>	<code>@bpm-font-family-base</code>	Loem Ipsum
<code>@bpm-label-font-weight</code>	<code>bold</code>	Loem Ipsum
<code>@bpm-label-text-color</code>	<code>inherit</code>	
<code>@bpm-input-bg</code>	<code>#d9ece3</code>	Light Green Box
<code>@bpm-input-bg-disabled</code>	<code>@bpm-neutral-lightest</code>	
<code>@bpm-input-color</code>	<code>@bpm-text-color</code>	Black Box

- __ 6. Save your new theme.
 __ 7. Apply the new theme to the HR Recruitment Processes process application.
 __ a. Open the **Process App Settings** from the History menu on the top.



- __ b. In the Coach Designer Settings section, click **Select**.

Coach Designer Settings

Theme: **Select...** **New...** **Clear**

- __ c. Select **Hiring Request Theme**. The theme is now changed to the new theme.

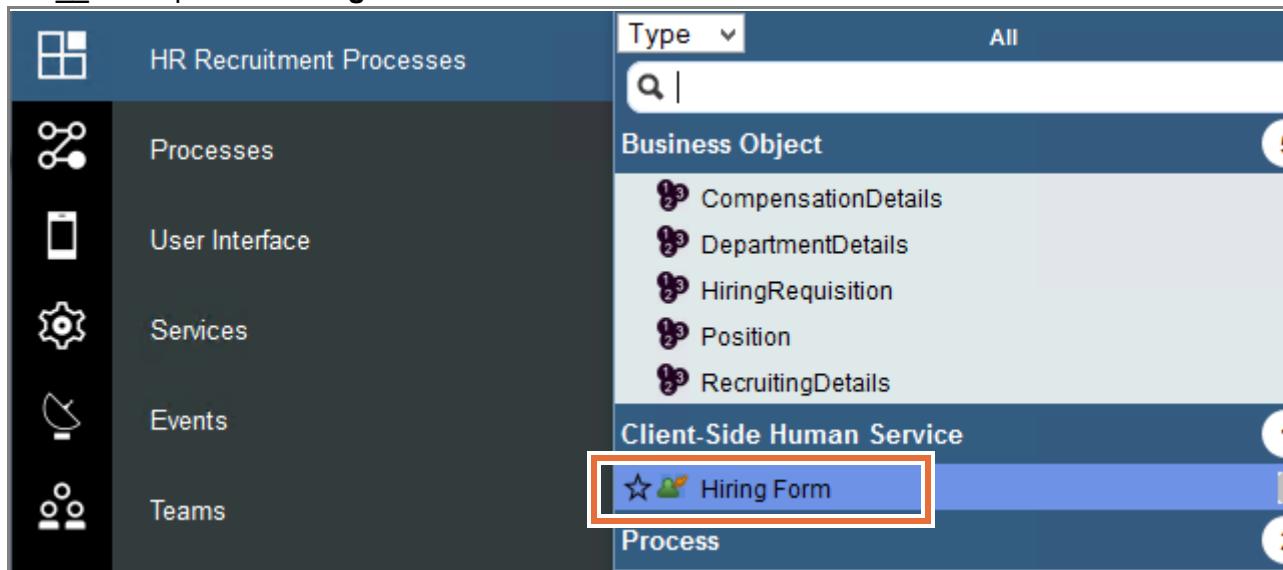
Coach Designer Settings

Theme: **Hiring Request Theme** **Select...** **New...** **Clear**

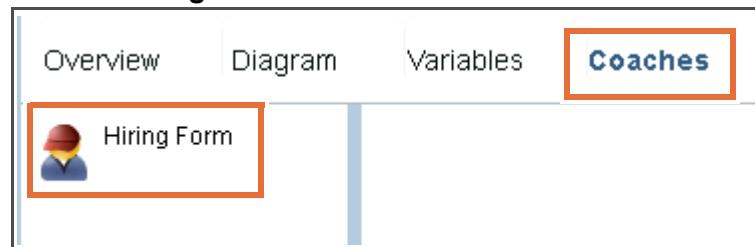
- __ d. Save your changes. The save generates the CSS that the browser uses to display the controls in the layout.

__ 8. Verify the theme changes

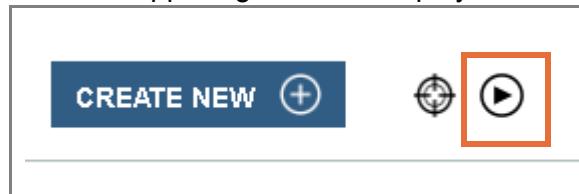
__ a. Open the **Hiring Form** Client-Side Human Service.



__ b. Click **Coaches > Hiring Form**.



__ c. Click the **Run** icon at the upper-right side to display the coach with the custom theme.



The coach has the changes that you made in the Hiring Request Theme.

Requisition Details	Position Details	Recruiting Details	Compensation Details	Department Details
Requisition Details				
Requisition Number				
<input type="text"/>				
Requester				
<input type="text"/>				
Date Details				
Date Of Request				
<input type="text"/>				
Date Position Available				
<input type="text"/>				
Hiring Manager Comments				
<input type="text"/>				
Submit				



Note

Initially when you view the coach, the server must generate the CSS and render the new coach on the screen, so a delay might occur before you see the effect. You must wait until the CSS is applied to the coach in the web Coach Designer.

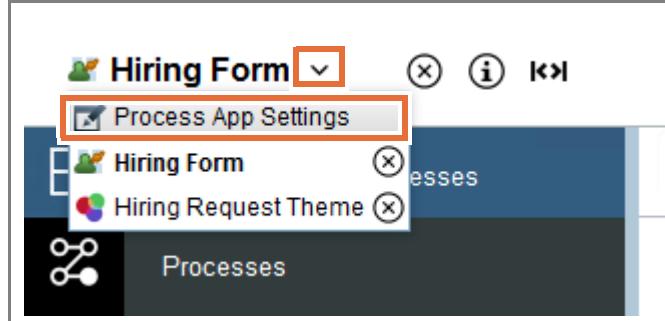
-
- ___ d. Close the browser window.

Part 3: Apply the Spark UI theme

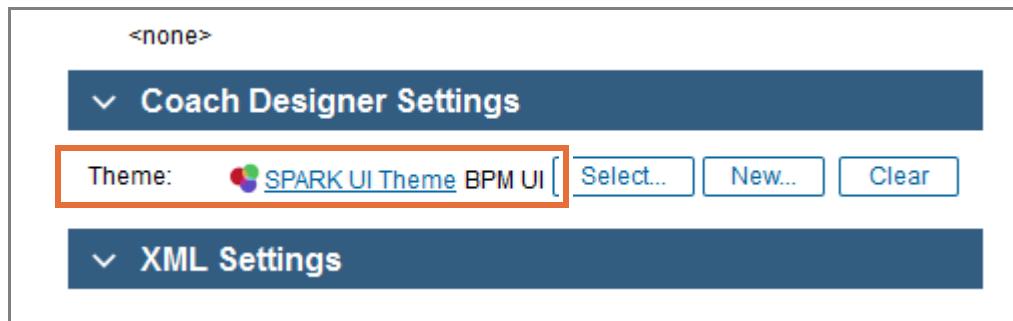
The Spark UI theme adds features that are not found in the default BPM theme. This part of your exercise shows some of the features of this unique theme.

- ___ 1. Change the theme to Spark UI.

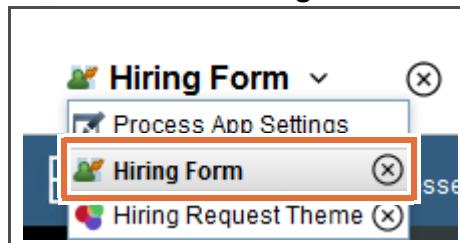
- ___ a. Expand the history menu and select the **Process App Settings**.



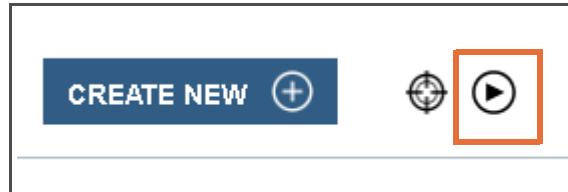
- ___ b. Under the **Coach Designer Settings > Theme** selection, change the theme to **Spark UI Theme**.



- ___ c. Using the history menu, return to the **Hiring Form**.



- ___ d. Click **Coaches > Hiring Form** to view the coach.
 - ___ e. Save your work.
 - ___ f. Click the **Run** button to apply the new theme to the coach.



- ___ g. Changing the process application theme changes all the settings that you made in the last part of this exercise. When you click **Run**, the first thing you notice is that the green backgrounds are now missing.

The screenshot shows a requisition coach with the following tabs: Requisition Details (selected), Position Details, Recruiting Details, and Compensation Details. The Requisition Details panel contains fields for Requisition Number and Requester.

- ___ h. Close the new browser window when you finish viewing the new coach.
 ___ 2. Add an icon to the coach.
 ___ a. Select the **Requisition Details** tab on the coach.



Note

You might need to select **Content** from the right if **Grid** is selected.

- ___ b. Click the **Date Details** panel.
 ___ c. Expand the **Properties > Configuration > Appearance** settings.

The screenshot shows the Properties panel with the following sections: General, Positioning, Configuration (highlighted with a red box), Events, Visibility, and HTML Attributes. Under Configuration, the Appearance section is expanded, showing settings for Icon, Color style, and Light color. The Date Details panel is also highlighted with a red box.

- __ d. In the **Icon** field, type in calendar.



Hint

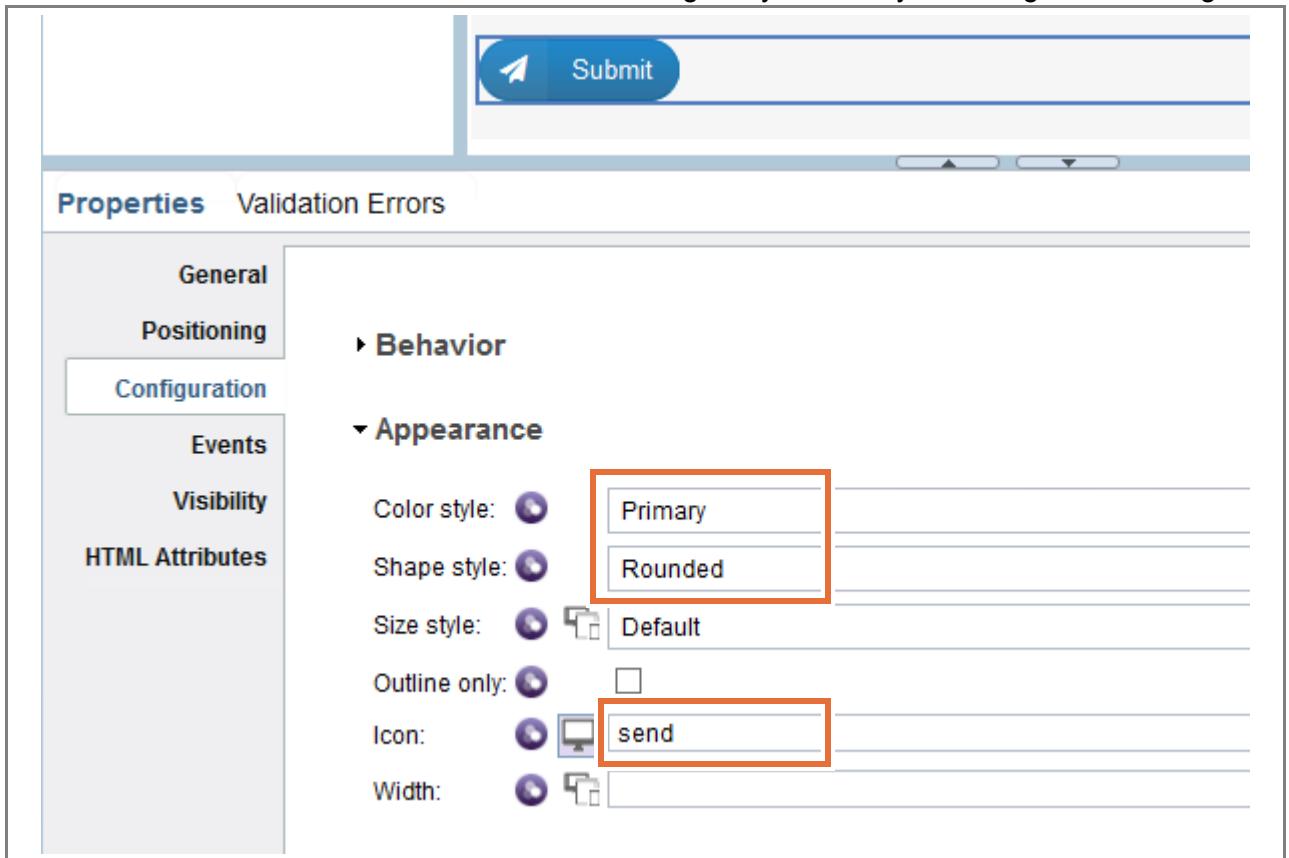
The icon shows on the right of the panel in the designer window when you enter a valid icon name. You can use any of the icons that are found on the <http://www/fontawesome.io> website.

The screenshot shows the IBM Designer interface. At the top, there's a panel titled "Date Details" with a "Date Of Request" input field. To the right of the input field is a small calendar icon, which is highlighted with a red box. Below this panel, there's a section labeled "on Errors". Underneath, there's a "Appearance" configuration section. It contains two rows: "Icon:" followed by a radio button selected for "calendar", and "Color style:" followed by a radio button selected for "Default".

- __ 3. Change the appearance of the submit button.
- __ a. Select the **Submit** button on the canvas.
 - __ b. Expand the **Properties > Configuration > Appearance** menu.

The screenshot shows the IBM Designer interface with the "Properties" tab selected. On the left, there's a sidebar with tabs: General, Positioning, Configuration (which is selected), Events, Visibility, and HTML Attributes. In the main area, under the "Configuration" tab, there's a "Behavior" section with a "Appearance" subsection. This subsection is highlighted with a red box. It contains three rows: "Color style:" with "Default" selected, "Shape style:" with "Default" selected, and "Size style:" with "Default" selected. Above the "Appearance" section, there's a "Submit" button on the canvas, which is also highlighted with a red box.

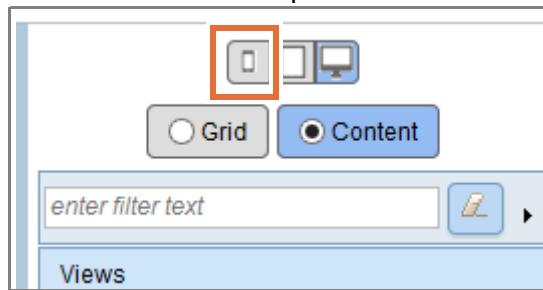
- c. Set the Color style to **Primary**, the Shape Style to **Rounded**, and the Icon to send. The button on the canvas reflects these settings as you make your configuration changes.



Part 4: Modify the mobile format presentation

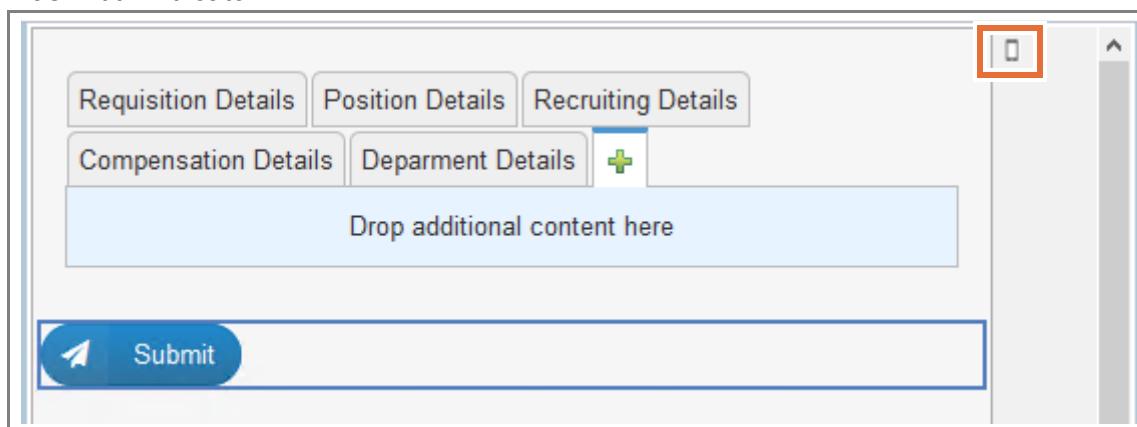
You are going to change the layout of the coach to mobile layout so that the coach is displayed correctly on a mobile browser. You learn to make few sections collapsible and change the width of a few input controls based on the width of the browser window.

- 1. Edit the coach for the small screen size.
- a. Click the **small screen** icon above the palette.



Note

The designer view canvas adjusts to show what the elements look like on the small format. If you run this coach, it renders depending on the width of the browser. If you shrink the browser window, it snaps to the new format as soon as it passes the width threshold for the small, medium, or large formats. This utility shows what the coach looks like if you open it with a mobile device. The medium view is designed for a tablet format. All sizes are designed for a portrait orientation (vertical, not horizontal) of the device. For the small screen format, the canvas reduces to the approximate width of a smartphone browser window in portrait mode. The cell phone icon is shown to the right of the canvas width indicator.



- b. Click the **Submit** button. In the **Properties > Configuration > Appearance** menu, set the **Width** to **100%**.

**Note**

The icon next to the Width value changes to the small screen icon.

Properties Validation Errors

- General
- Positioning
- Configuration**
- Events
- Visibility
- HTML Attributes

Appearance

Color style:	<input checked="" type="radio"/> Primary
Shape style:	<input checked="" type="radio"/> Rounded
Size style:	<input checked="" type="radio"/> Default
Outline only:	<input type="checkbox"/>
Icon:	send
Width:	<input checked="" type="radio"/> 100%

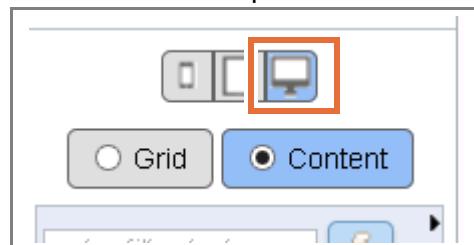
- c. On the **Requisition Details** tab, select the header of the **Requisition Details > Date Details > Date Details** panel.

Date Details

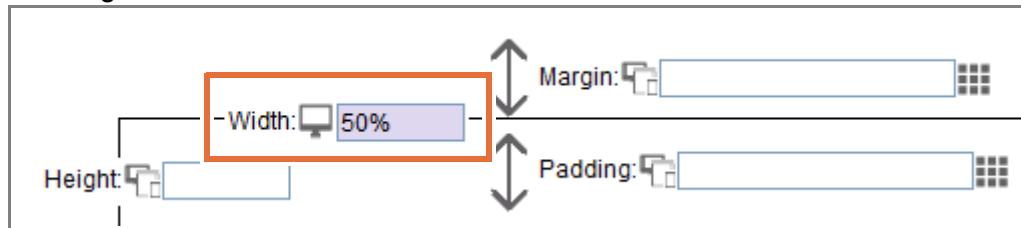
Date Of Request

Date Position Available

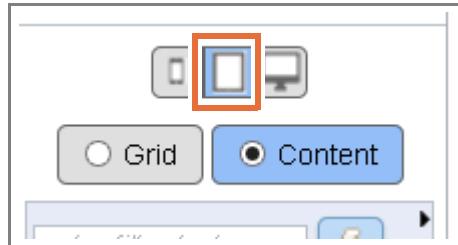
- d. Click the **large screen** icon above the palette.



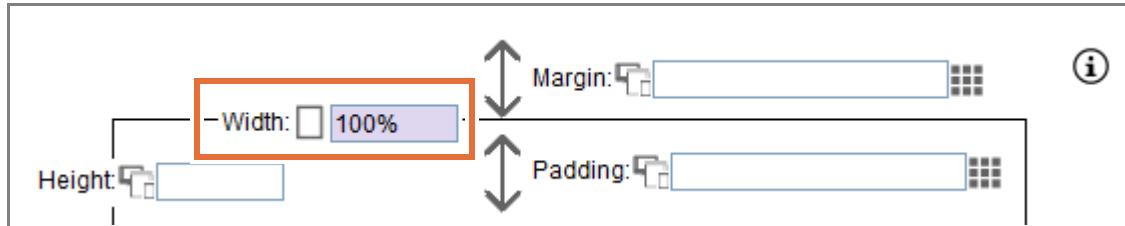
- __ e. Set the **Properties > Positioning > Width** to 50%. The entered width is applicable only for large screen devices.



- __ f. Click the **medium screen** icon above the palette.



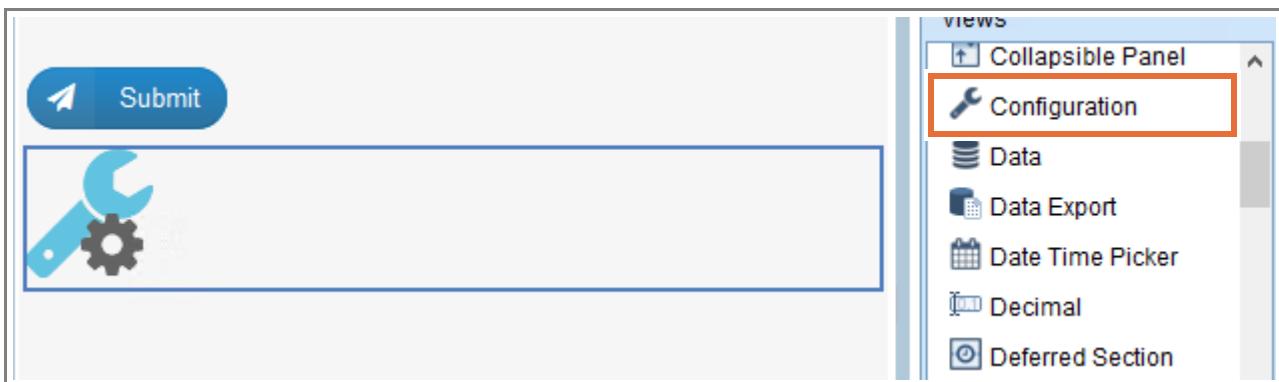
- __ g. Set the **Properties > Positioning > Width** to 100%.



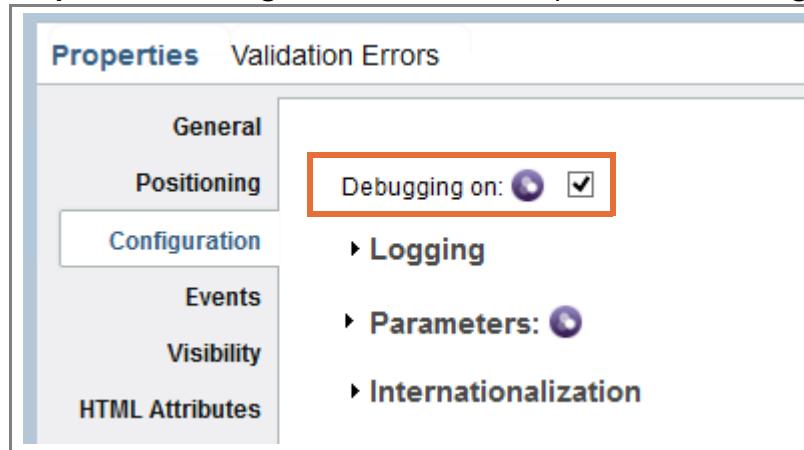
Note

When you set the tablet setting (medium screen), the small screen automatically is set to 100%. You see that this setting is inherited from the tablet format setting, so you do not have to set it to 100%.

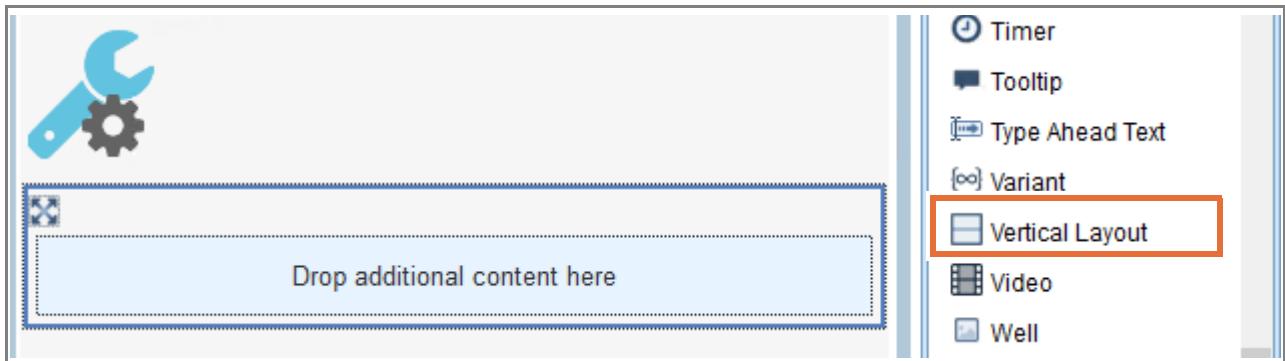
-
- __ h. Save your work.
- __ 2. Verify the coach display for different screen sizes.
- __ a. Drag a **Configuration** coach view from the palette and place it on the canvas below the Submit button.



- __ b. Under **Properties > Configuration**, enable the option to turn **Debugging On**.



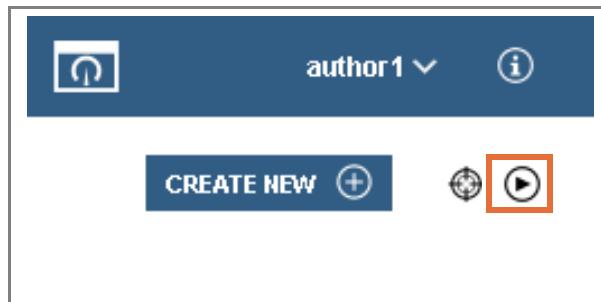
- __ c. Drag a **Responsive Sensor** below the Configuration on the canvas.
 __ d. Drag a **Vertical Layout** to the area inside the Responsive Sensor that is labeled Drop additional content here.



Information

You can drag any object inside the Responsive Sensor, but it requires an object to correctly operate. When the coach is run, the browser width is displayed in this Responsive Sensor as soon as the browser is resized to test the responsive settings.

- __ e. Save your work.
 __ f. Click the **Run** button.



- g. A new browser window opens with the coach displayed, and the browser width is shown at the bottom. The Date Details panel in this image spans the entire browser window because the initial width is set to a medium screen setting.

The screenshot shows a user interface for a requisition form. At the top, there are tabs for 'Requisition Details' (which is active), 'Position Details', 'Recruiting Details', and 'Compensation Details'. Below the tabs is a section titled 'Department Details'. The main content area contains several input fields and sections:

- 'Requisition Details' section with a text input field.
- 'Requisition Number' section with a text input field.
- 'Requester' section with a dropdown menu.
- 'Date Details' section, which is highlighted with a thick orange border. It contains three input fields: 'Date Of Request' (with a calendar icon), 'Date Position Available', and a large 'Hiring Manager Comments' text area.
- 'Submit' button with a paper airplane icon.
- A blue box at the bottom right of the page contains the text 'Width: 798px'.

- h. Slowly reduce the width of the browser. When you pass the 640 px threshold, the Submit button expands to 100% of the screen width.

The screenshot shows a user interface for a requisition form. At the top, there are tabs for 'Requisition Details', 'Position Details', and 'Recruiting Details'. Below these are two more tabs: 'Compensation Details' and 'Department Details'. The main area contains several input fields and sections:

- Requisition Details**: A section header.
- Requisition Number**: An input field.
- Requester**: An input field.
- Date Details**: A section header with a calendar icon.
- Date Of Request**: An input field.
- Date Position Available**: An input field.
- Hiring Manager Comments**: A large input field.

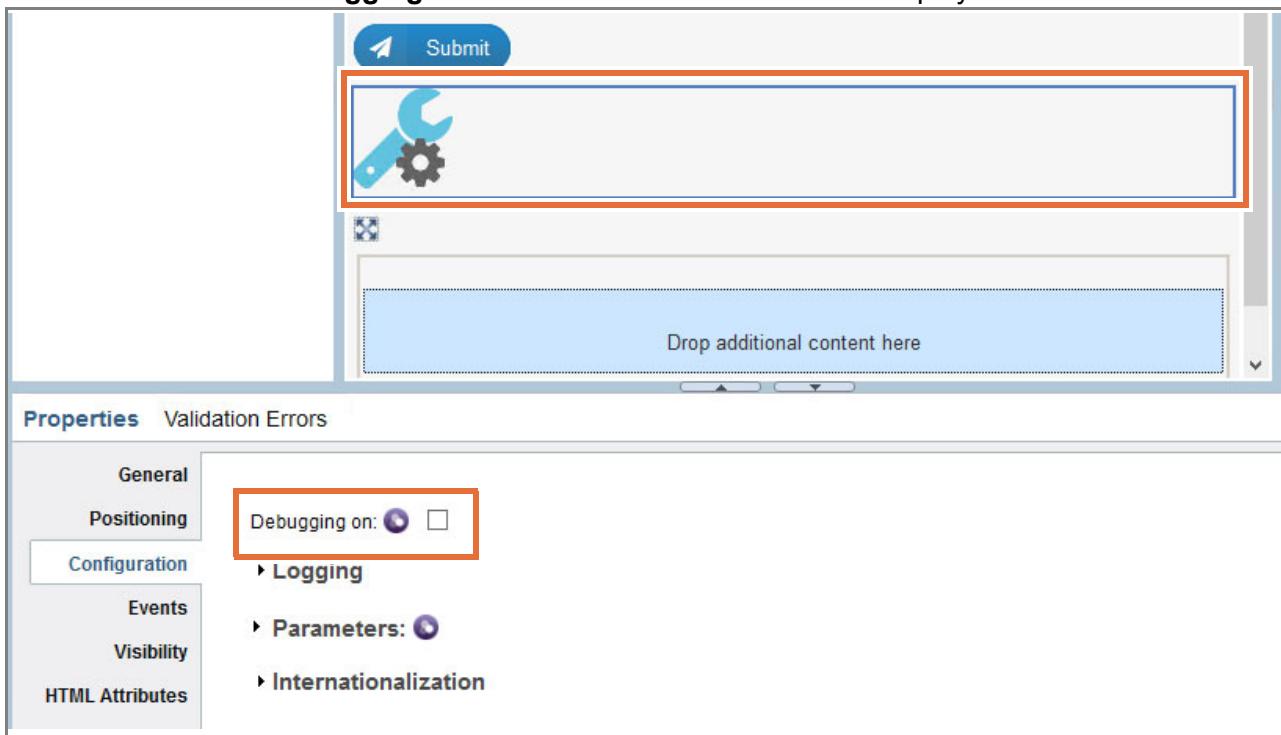
At the bottom is a blue 'Submit' button with a white arrow icon on the left. The entire 'Submit' button is highlighted with a thick orange border. Below the button, a tooltip-like box says 'Width: 640px'.

- i. Now expand the browser window. When the width passes 1024 px, the Date Details panel contracts to 50% of the browser width because of the responsive control settings based on the large screen width setting.

The screenshot shows a web application interface for a requisition. At the top, there are tabs: Requisition Details (which is active), Position Details, Recruiting Details, Compensation Details, and Department Details. Below the tabs, there are several input fields: Requisition Number, Requester, and a large Date Details panel. The Date Details panel is highlighted with a red border and contains three sub-fields: Date Of Request and Date Position Available, each with a calendar icon. Below these is a field for Hiring Manager Comments. At the bottom left is a blue 'Submit' button with a white arrow icon. A tooltip at the bottom right of the Date Details panel says 'Width: 1025px'. The entire form is contained within a light gray box.

- j. Close the browser window.

- ___ k. Click the Configuration on the canvas. On the Properties > Configuration menu, clear the Debugging on check box to remove the width display on the coach.



- ___ l. Save your work.
___ m. Minimize your browser.



Important

Playback 1: User interface design and implementation is now complete.

As part of the development process, you now review the playback and examine its functions in the Process Portal.

Test the following functions:

- Does the coach display all the tabs?
- Are all the fields on the correct tabs of the coach?
- Do the mobile features meet the requirements?

For more information about conducting the playback, see the Appendix A: Conducting playbacks.

End of exercise

Exercise review and wrap-up

This exercise looked at Playback 1: User interface design and implementation, how to create group controls into tabs on a coach, and how to change the appearance of the coach.

Exercise 8. Playback 1: Conducting the playback session

Estimated time

01:00

Overview

This exercise covers how to conduct a playback of your process. The exercise demonstrates the process, following various paths that flow from the exclusive gateways in the process and demonstrate tasks that are assigned. It also describes the task that is created in the Process Portal inbox, depending on the swimlane and routing settings for an activity. You also create a toolkit to store and share these assets.

Objectives

After completing this exercise, you should be able to:

- Log on to the Process Portal
- Create an instance of a process
- Demonstrate that the process follows the various paths modeled
- Create a toolkit
- Create a dependency on a toolkit

Introduction

The following list provides a quick view of what to accomplish in this exercise:

1. Log on to the Process Portal and create a process instance.
2. Log on to the Process Portal as two different teams to show that the correct process team is being assigned the correct task.
3. Complete the human activities.
4. Demonstrate that the correct path is followed.
5. Change variable values and demonstrate following a different path.
6. A toolkit is a collection of assets that are shared between process apps or other toolkits during development. Attentive application of toolkits is an efficient method for managing the reusability and compartmentalization of project assets.

Requirements

Successful completion of the previous exercise is required.

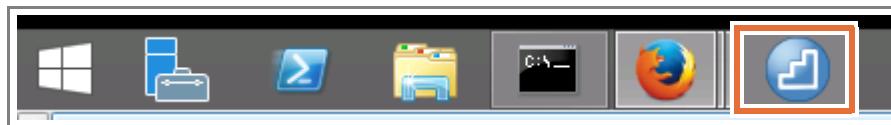
Exercise instructions

Part 1: Demonstrate the Review Needed path

During this playback session, you demonstrate the process, following various paths that flow from the exclusive gateways in the processes, and demonstrate tasks that are assigned.

You accomplish a process playback session just as you would do when you seek consensus to move the project to the next playback phase. You log in as different users of the process to demonstrate tasks that different users accomplish in the process, demonstrating coaches that you created in the current playback phase.

- 1. Log on to the Process Portal.
 - a. Maximize the **IBM Business Process Manager Quick Start** browser window from the Windows taskbar.



- b. Click the **Process Portal** link.

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.

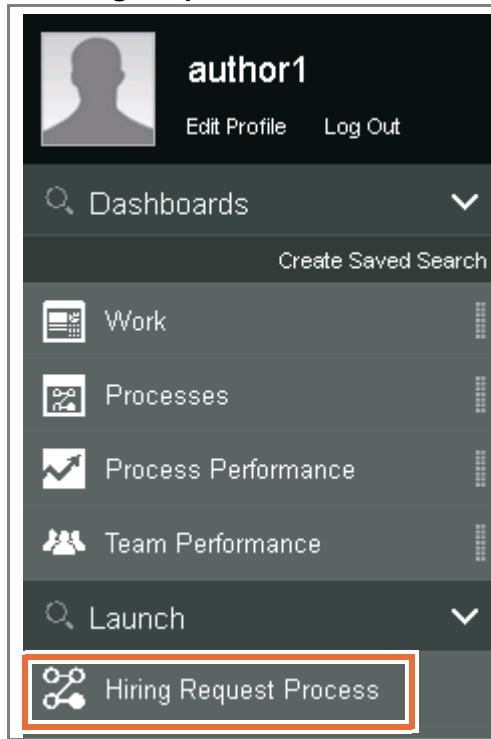
[Process Portal](#)



Note

If prompted with the login screen, log in with `author1` as the user ID and `author01` as the password.

- __ 2. Complete the first activity, Submit Hiring Request, in the Process Portal.
- __ a. In the left frame, click **Hiring Request Process** to start an instance of the process.



Troubleshooting

If the library is not displayed, click the **Main Menu** icon on the left side of the Work tab.



- __ b. A notice is displayed to indicate that the process was started.

You received a new task titled Step: Submit Hiring Request



Troubleshooting

The first task starts automatically in the portal because you assigned the first task to Lane and Last User. If the task does not start automatically, return to the steps in Exercise 5, Part 5.1 to configure the first activity to start automatically.

You can also click the Work tab in the Process Portal. The process instance is displayed, and it is paused on the Submit Hiring Request activity. Click the task that is labeled Step: Submit Hiring Request to work on it.

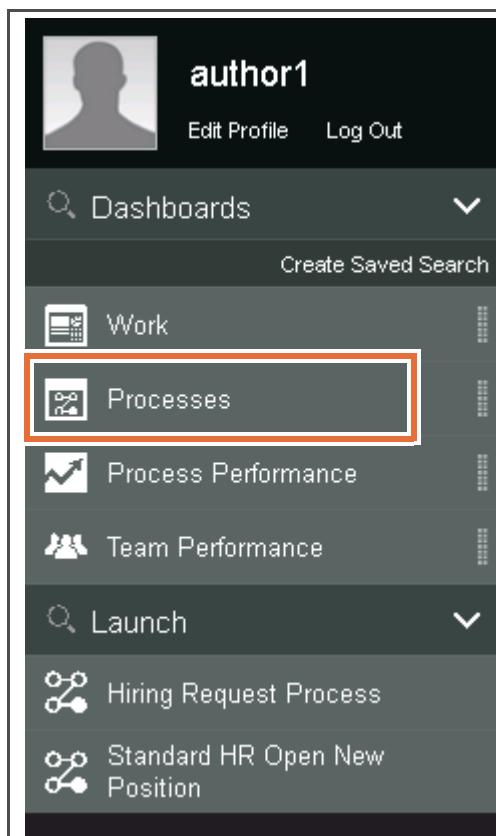
A window is displayed with the human service coach.

Requisition Details	Position Details	Recruiting Details	Compensation Details	Department Details
Requisition Details				
Requisition Number <input type="text"/>				
Requester <input type="text"/>				
Date Details Calendar icon				
Date Of Request <input type="text"/>				
Date Position Available <input type="text"/>				
Hiring Manager Comments <input type="text"/>				
<input type="button" value="Submit"/>				

- c. Click the **Recruiting Details** tab, and select the **New Position** check box so that it follows the Review Needed path. Click **Submit**.

The screenshot shows a software interface for managing requisitions. At the top, there are tabs: Requisition Details, Position Details, Recruiting Details (which is highlighted with a red box), and Compensation. Below the tabs is a section titled "Recruiting Details". Inside this section, there is a checkbox labeled "Multiple Employees Needed" which is unchecked. Below it is a field labeled "Num Employees Needed" with the value "0". Underneath these fields is a checkbox labeled "New Position" which is checked and highlighted with a red box. At the bottom of the page is a blue "Submit" button, which is also highlighted with a red box.

- 3. View the task that is assigned to the next activity, **Approve New Hire Request**, in the process.
— a. In the Library click **Processes**.



- __ b. The Processes page opens. Click **Hiring Request Process**.

The screenshot shows the 'Processes' interface. The title 'Processes' is at the top right, with a three-line menu icon on the left. Below the title is the heading 'Process Instances'. A search bar with a magnifying glass icon and the placeholder 'Type a search filter' is present. To its right are a clear button (with an 'X') and a help button (with a question mark). The main area displays a single process instance card. The card features a circular icon with a double-headed arrow, the text 'Hiring Request Process:11' in blue, and a small star icon. The entire card is enclosed in an orange rectangular border. At the bottom of the list area, the text 'Showing 1 out of 1 results' is visible.



Note

The process instance ID can be different in your environment. If you have more than one instance, click the one with the highest instance number (the number after the colon).

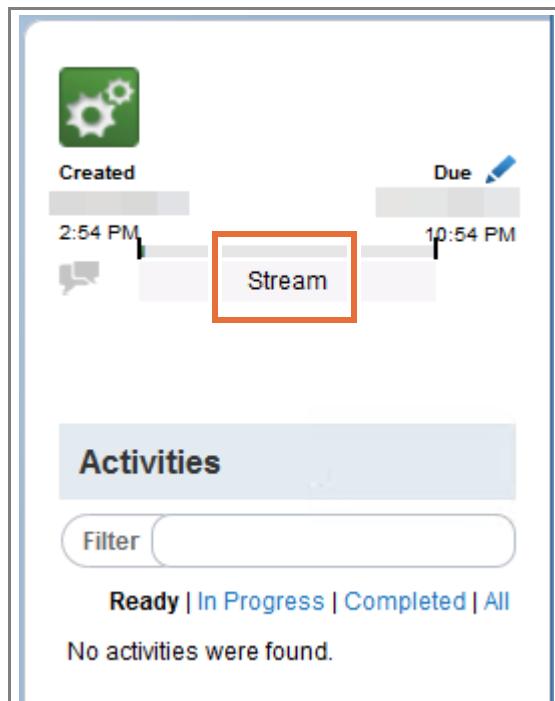
- __ c. The process tab displays data about the instance.

**Important**

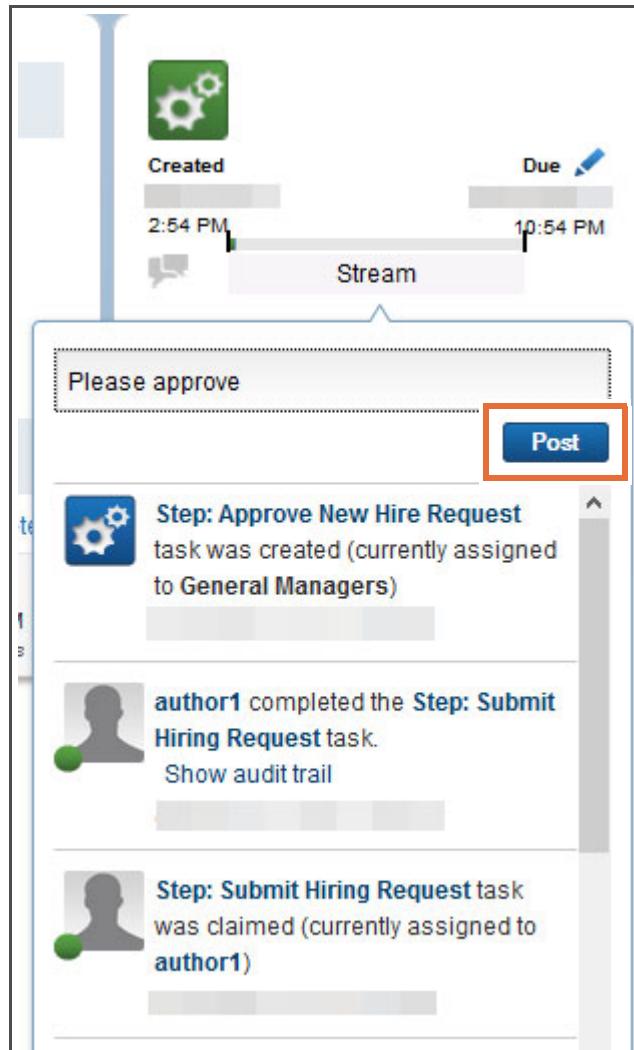
The bottom section that is labeled **Tasks** shows the next incomplete task in the process. Do **NOT** click the task. This task is assigned to **General Managers** team, and **user1** is a member of this team. In the next part of this exercise, you log in as **user1** to complete this task.

The screenshot shows a process instance interface. At the top, there are sections for **Data**, **Department: [not defined]**, **Job Title: [not defined]**, **Requisition Number: [not defined]**, **Salary: [not defined]**. Below this is a **Documents** section with a folder icon labeled **Hiring Requisition**, **author1**, **2:54 PM**, and **Retrieved items: 1**. To the right is a **Tasks** section with a green gear icon labeled **Step: Approve New Hire Request**, **Due:** (with a timeline bar), and **General Managers**. A red box highlights the **Step: Approve New Hire Request** task.

- ___ d. On the right, the creation data, and the process due date are shown, and a timeline that shows the current duration of the instance. Click the **Stream** area to view the actions that occur for this instance.



- e. To add to the stream, type Please approve into the stream field and click **Post**. Your comment appears as the most recent entry in the stream either immediately or when you click outside the stream.





Reminder

Click somewhere outside the Stream section to view the full **Activities** area below the stream. The lower-right section that is labeled Activities shows all the ad hoc activities in the process. If the process contains an ad hoc activity, users can start the activity from this section.

The screenshot shows a user interface for managing activities. At the top, there is a header labeled 'Activities'. Below it is a 'Filter' button. Underneath the filter is a row of four status links: 'Ready', 'In Progress', 'Completed', and 'All'. A message 'No activities were found.' is displayed below these links. The entire section is enclosed in a light gray border.

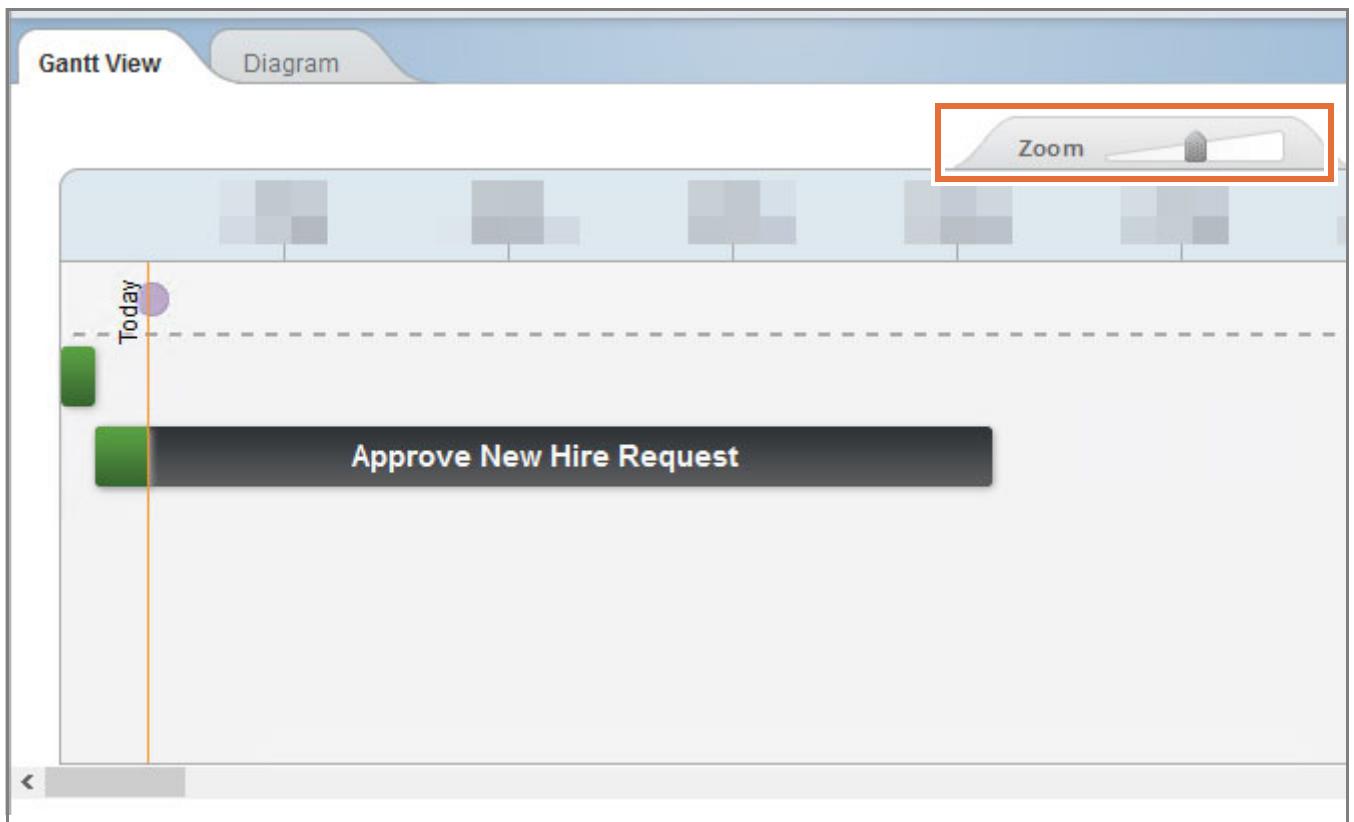
- ___ f. Click the **GANTT CHART** icon in the upper-right corner to go to the Gantt chart view.



Note

The Gantt chart is part of the Process Performance tab. When you click the link, you open the **Process Performance > Instance Details** page for the process instance. Wait for few seconds as the loading of the page might take some time.

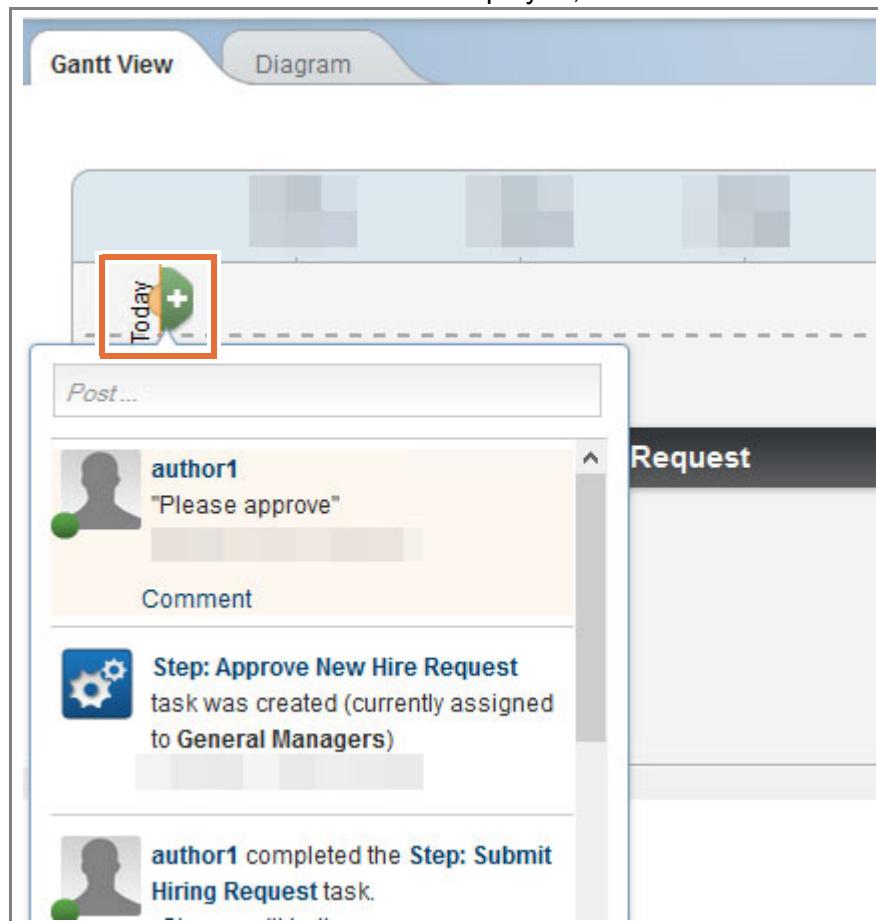
- __ g. When the Process Performance page opens, move the slider in the **Zoom** to zoom in on the Gantt chart view data.



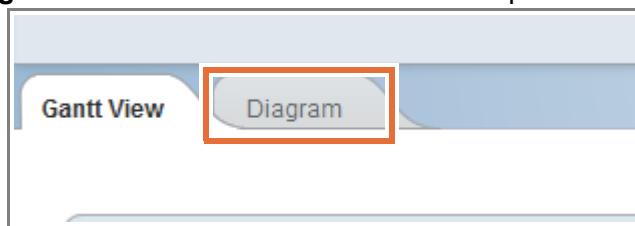
Important

The Gantt chart lists all the instance tasks that are part of the process and the expected durations of the activities as defined by the due dates of the activities. The next task is assigned to the General Managers team. This result is because you set the default value of the first gateway to "1" (Review Needed). This value takes the runtime token to the Approve New Hire Request task. You view the process and the paths later in this exercise.

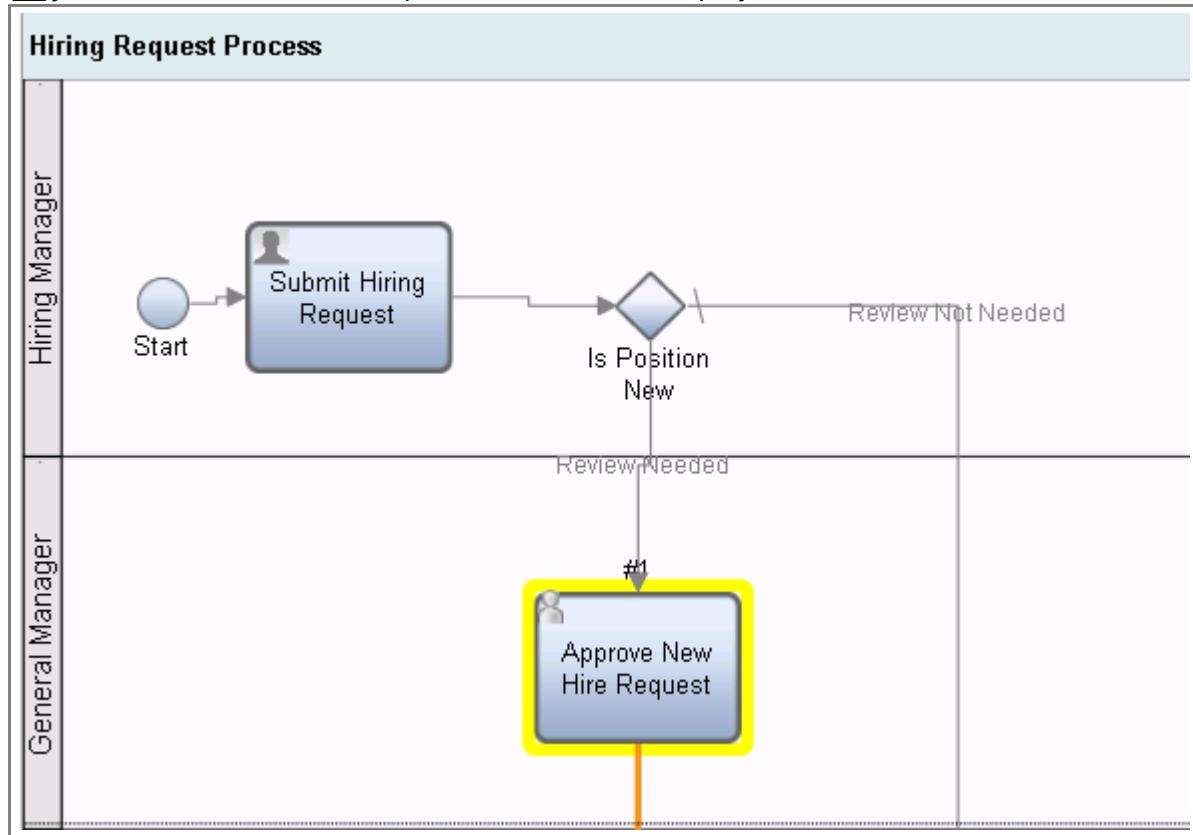
- h. Hover the cursor over the **purple circle** next to the **Today** label, and it turns yellow and shows the stream. If the stream is not displayed, then zoom in a little more.



- i. Click the **Diagram** tab for a model view of the same process instance.



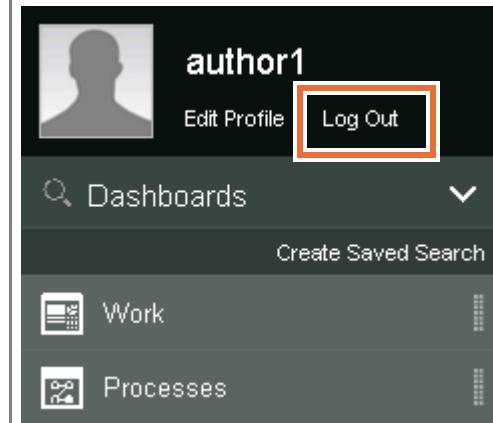
- ___ j. The model view of the process instance is displayed.



Important

This view shows the process with the Approve New Hire Request task that is highlighted for the **Review needed** path, and in the next part of this exercise you work on the **Review not needed** path.

- ___ 4. Complete the Approve New Hire Request activity.
 ___ a. In the library, click **Log Out** to log out of the Process Portal.



- __ b. Log in to the Process Portal with the **User name** user1, **Password** user01, and click **Continue**.



Sign in to BPM

Username
user1

Password

Continue →

Licensed Materials - Property of IBM. © Copyright 2008, 2010 IBM Corporation. IBM, the IBM logo, and WebSphere are trademarks of IBM Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies.

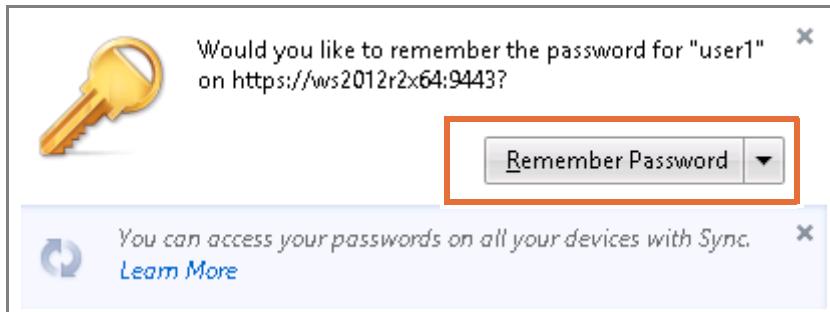
IBM

The 'Username' and 'Password' fields are highlighted with a red box. The 'Continue' button is also highlighted with a red box.

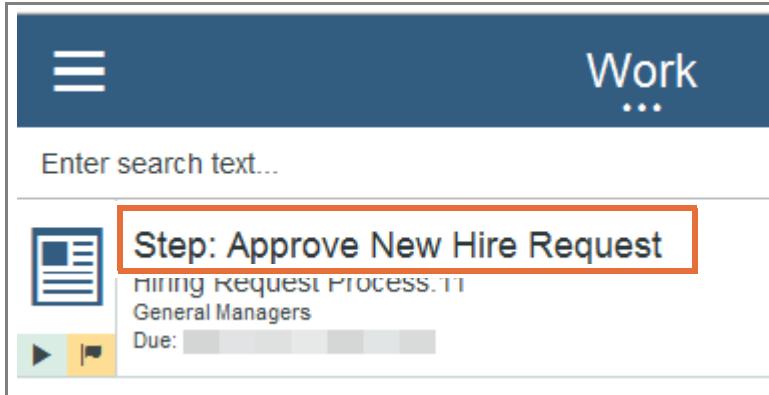


Note

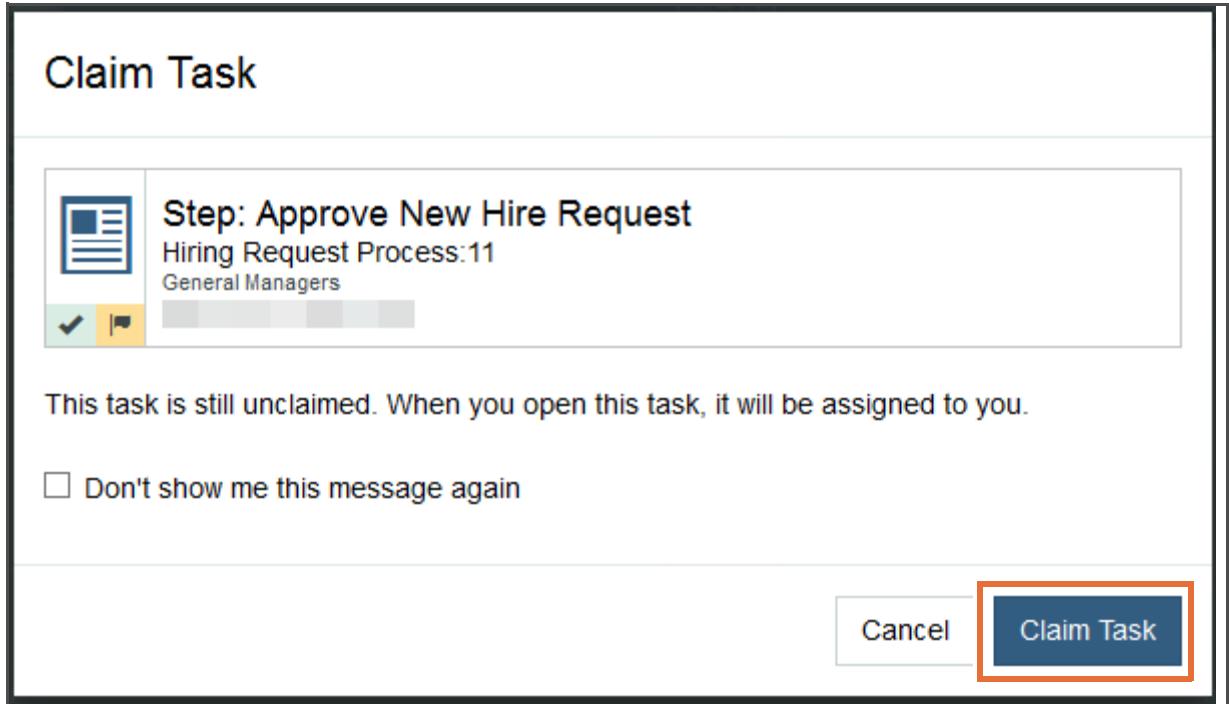
If prompted for Remember Password, click **Remember Password**.



- ___ c. A task is in the inbox for the Hiring Request Process process. The task stopped at the Approve New Hire Request activity. Notice that the task is assigned to the General Managers team, of which user1 is a member. Click the **Step: Approve New Hire Request** task.



- ___ d. Click **Claim Task**, and the task is automatically assigned to you.



- ___ e. A human service coach is displayed for the Approve New Hire Request activity.

The screenshot shows a web-based form titled "Requisition Details". At the top, there are tabs for "Requisition Details", "Position Details", "Recruiting Details", "Compensation Details", and "Department Details". The "Requisition Details" tab is active. Below the tabs, there are several input fields: "Requisition Number" (empty), "Requester" (empty), "Date Details" (with a calendar icon), "Date Of Request" (empty), "Date Position Available" (empty), "Hiring Manager Comments" (empty), and a "Submit" button at the bottom left.

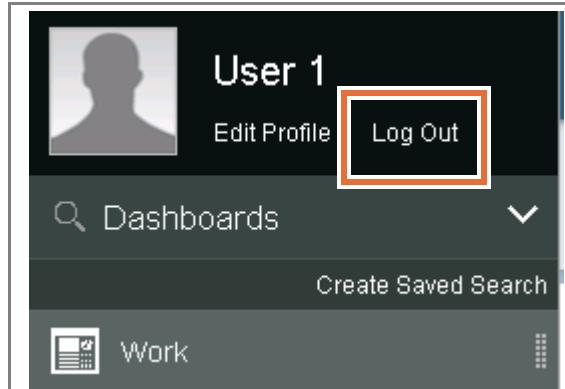
- ___ f. Enter any data in the fields and complete this task by clicking **Submit** in the human task service.
 ___ g. Click **Processes** in the library.
 ___ h. Click the **Hiring Request Process** instance with the same instance ID you worked on in the last steps.

Important

The next task is assigned to author1. Do **NOT** click this task.

The screenshot shows a user interface with two main sections: "Documents" and "Tasks". The "Documents" section on the left shows a folder named "Hiring Requisition" created by "author1" at 8:00 AM, with 1 item retrieved. The "Tasks" section on the right shows a task titled "Step: Complete Hire Request" with a status of "Open | Completed". The task is assigned to "author1" and has a due date. A red box highlights this task.

- i. Log out from the Process Portal.



Note

If Log Out is not displayed, click the **Main Menu** icon on the left side.



Part 2: Demonstrate the Review Not Needed path

Now that you demonstrated the `Review Needed` path, this next part of the playback demonstrates the exception path and shows the process that is traversing down the `Review Not Needed` path. It follows the same approach where you simulate the real-world experience of the different users inside the process as they log on to the portal and complete their tasks inside the process.

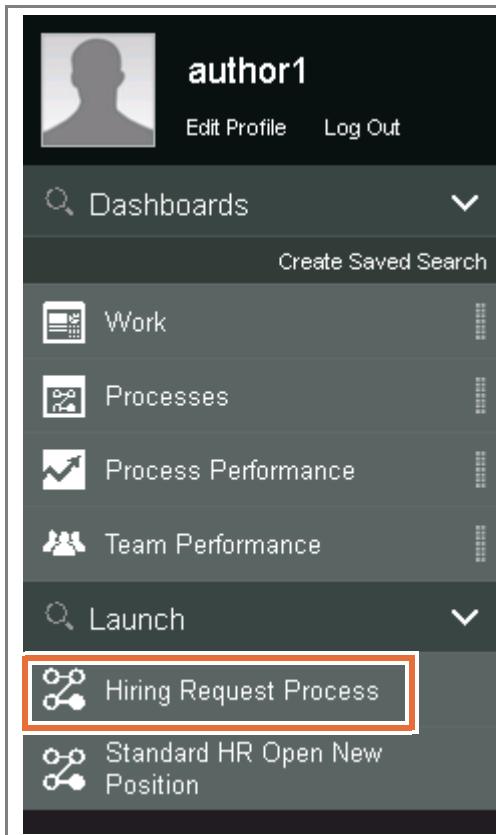
- 1. Show the process as it follows the `Review Not Needed` path, and create an instance to test the “existing position” scenario.
 - a. Log in to the Process Portal with `author1` as the **User name** and `author01` as the **Password**.



Hint

If you already closed the window, use the IBM Business Process Manager Quick Start browser window to start the Process Portal.

- __ b. Create an instance of the **Hiring Request Process** by clicking the link in the Launch section.



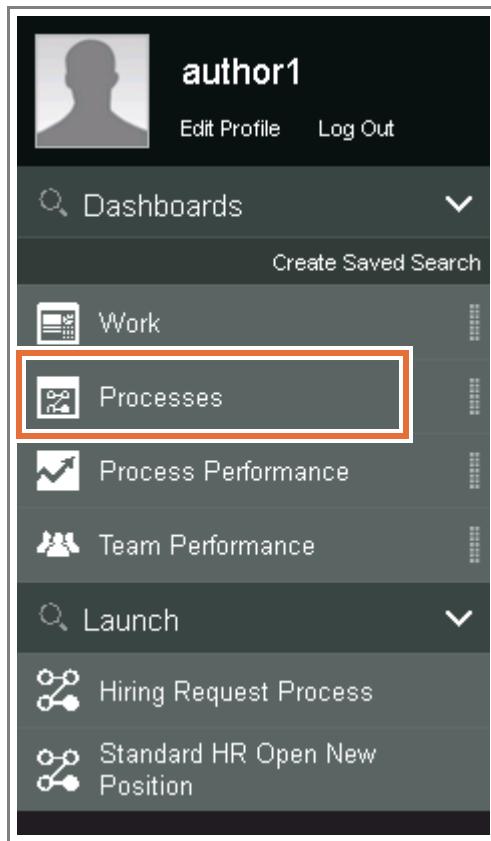
- __ c. You immediately see in a dialog box that you received a new task. A window is displayed with the human service coach.
 __ d. Click the **Recruiting Details** tab, and ensure that the **New Position** check box is cleared so that it follows the Review Not Needed path.

Requisition Details	Position Details	Recruiting Details
Recruiting Details		
<input type="checkbox"/> Multiple Employees Needed		
Num Employees Needed		
<input type="text" value="0"/>		
<input type="checkbox"/> New Position		

- __ e. Enter any data for the other fields and click **Submit**.

__ 2. Complete the process.

__ a. In the library, click **Processes**.



__ b. Click the process with the highest process instance ID.



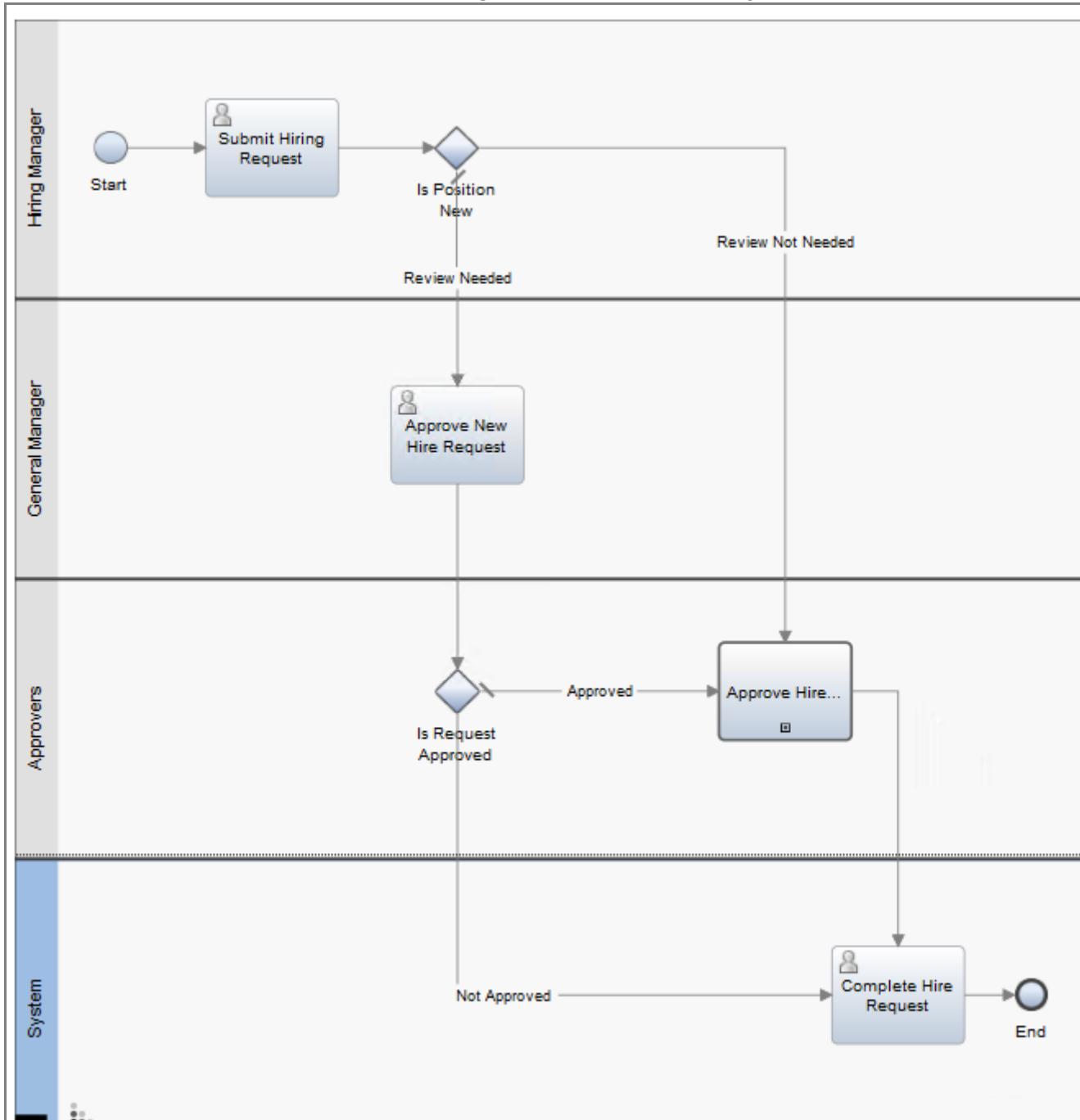
Note

The task with the highest instance ID is the newest one you created.



Information

This time, the process follows the **Review Not Needed** path. Instead of passing to **Approve New Hire Request**, the process stepped through the **Approve Hire Request** linked process. You can click **GANTT CHART**, and then click **Diagram** to view the flow diagram.



- __ c. Click the **Step: Check Hire Request** task link.

The screenshot shows the application's interface with two main panes: 'Documents' and 'Tasks'. The 'Documents' pane on the left lists two items: 'Approve Hire Requ...' and 'Hiring Requisition', both assigned to 'author1'. The 'Tasks' pane on the right shows a single task named 'Step: Check Hire Request' with a status of 'Open | Completed'. This task is also highlighted with a red box. Below the panes, it says 'Retrieved items: 2'.

- __ d. Click **Done** to complete the task.

The screenshot shows the 'Details' view for the task 'Step: Check Hire Request'. It includes sections for 'This is a sample coach for the following activity:' and the task itself. At the bottom, there are two buttons: 'Done' (highlighted with a red box) and 'Complete Later'.

- __ e. Now the process moves to the next task. Click **Step: Override Hire Request**.

The screenshot shows the application's interface with the same 'Documents' and 'Tasks' panes as before. The 'Tasks' pane now shows a different task titled 'Step: Override Hire Request' (highlighted with a red box), indicating the process has moved to the next step.

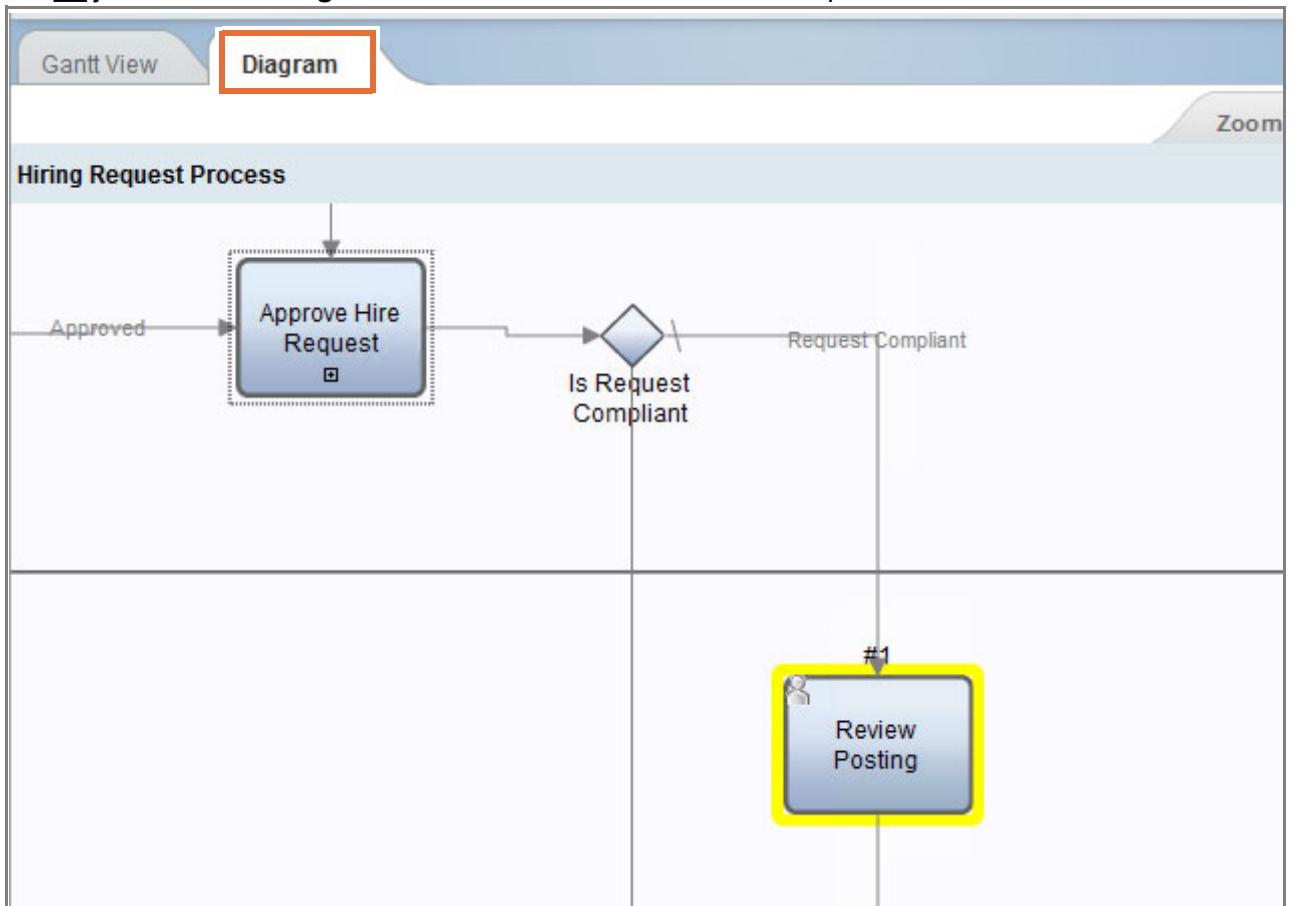
- __ f. Click **Claim Task**.

The screenshot shows a 'Claim Task' dialog box. It contains a message stating 'This task is still unclaimed. When you open this task, it will be assigned to you.' There is a checkbox labeled 'Don't show me this message again' and a large blue 'Claim Task' button at the bottom, which is highlighted with a red box.

- __ g. Click **Done** to complete the task.
- __ h. The process now moves to **Step: Review Posting**.
- __ i. Click the **GANTT CHART** icon on the upper-right corner of the window. Wait for few seconds to load the page.

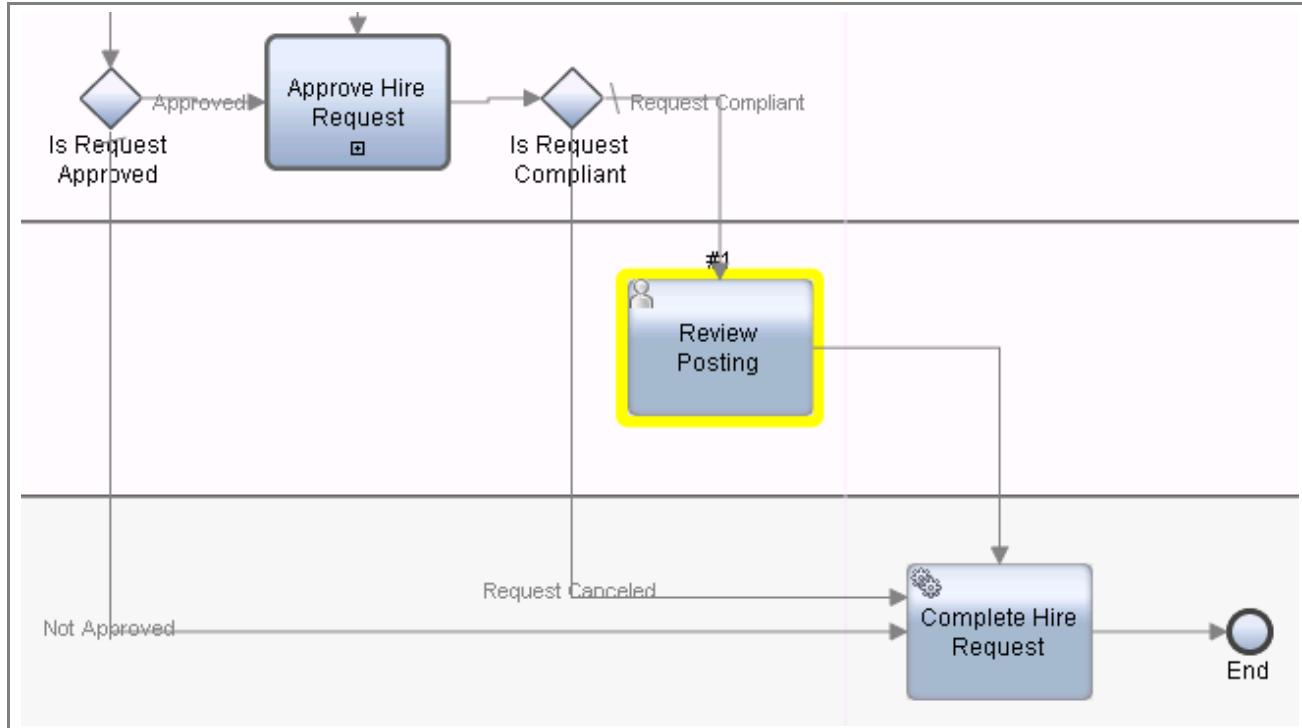


- __ j. Click the **Diagram** tab for a model view of the same process instance.

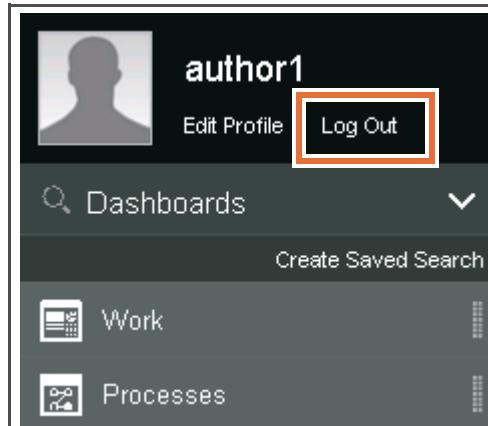


**Note**

This time, the process follows the Review Not Needed path. Instead of passing to Approve New Hire Request, the process stepped through the Approve Hire Request linked process, through the Is Request Compliant gateway, and then to the Review Posting activity. The process followed a different path from the first instance, which demonstrates that the Is Position New gateway and process flow data are working correctly when driven by business flow data on a coach.



- ___ 3. Log out of the Process Portal and close the tab.





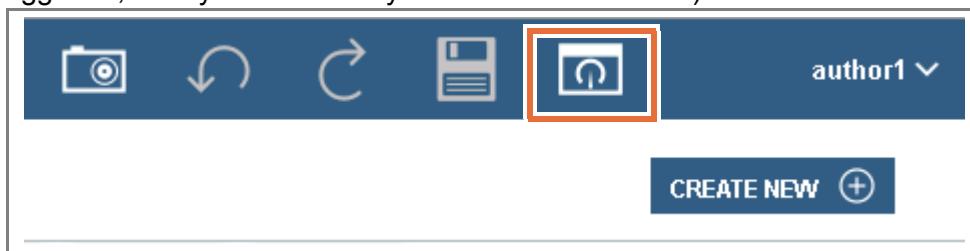
Reminder

In this course, you did not implement all the coaches for the sake of brevity. But one of the requirements to move to Playback 3 is that all the coaches are implemented in the process and all of them have the finalized presentation elements in place (CSS and JS, for example).

Part 3: Create the Hiring Requisition Toolkit

Now, you have numerous assets, some of which might be useful to other developers. Create a toolkit from the assets you created as part of the process application HR Recruitment Processes.

- 1. Return to the IBM Process Designer with user account author1 and password author01.
- a. Click the **Process Center** icon in the upper-right corner of the Designer. (If you just logged in, then you are already at the Process Center).

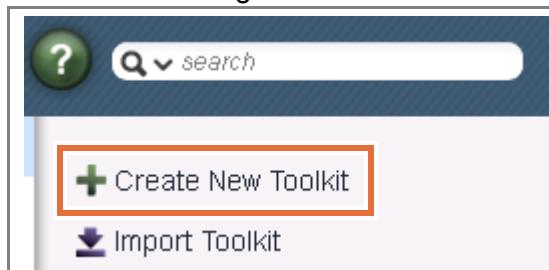


- 2. Click the **Toolkits** tab.

Toolkit Name	Last Updated	Action
BPM UI toolkit (SYSBPMUI)	Last updated on [redacted] by author1	Open
SAP Guided Workflow (deprecated) (SGW)	Last updated on [redacted] by author1	Open
System Governance (TWSYSG)	Last updated on [redacted] by author1	Open
Dashboards (SYSD)	Last updated on [redacted] by author1	Open
Content Management (SYSCM)	Last updated on [redacted] by author1	Open

___ 3. Create a toolkit.

___ a. Click **Create New Toolkit** on the right side of the Process Center.



- ___ b. Enter the name for the toolkit: Hiring Requisition Toolkit
- ___ c. Enter the toolkit acronym: HRT
- ___ d. Enter a brief description: This toolkit contains all common artifacts that are associated with the HR Recruitment Processes.
- ___ e. Click **Create**.

A screenshot of a "Create New Toolkit" dialog box. The title bar says "Create New Toolkit" and has a close button. The dialog contains the following fields:

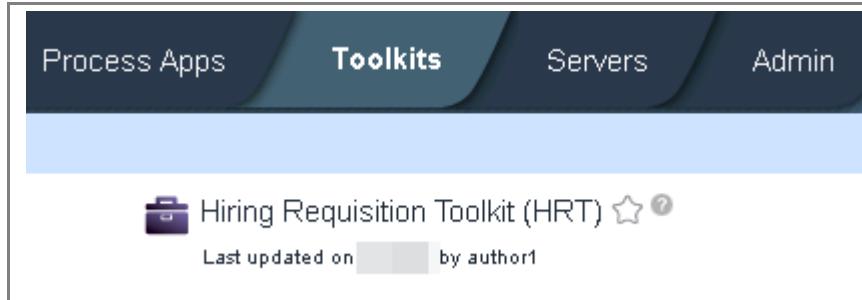
- Toolkit Name:
- Acronym: ?
- Documentation:

This toolkit contains all common artifacts that are associated with the HR Recruitment Processes

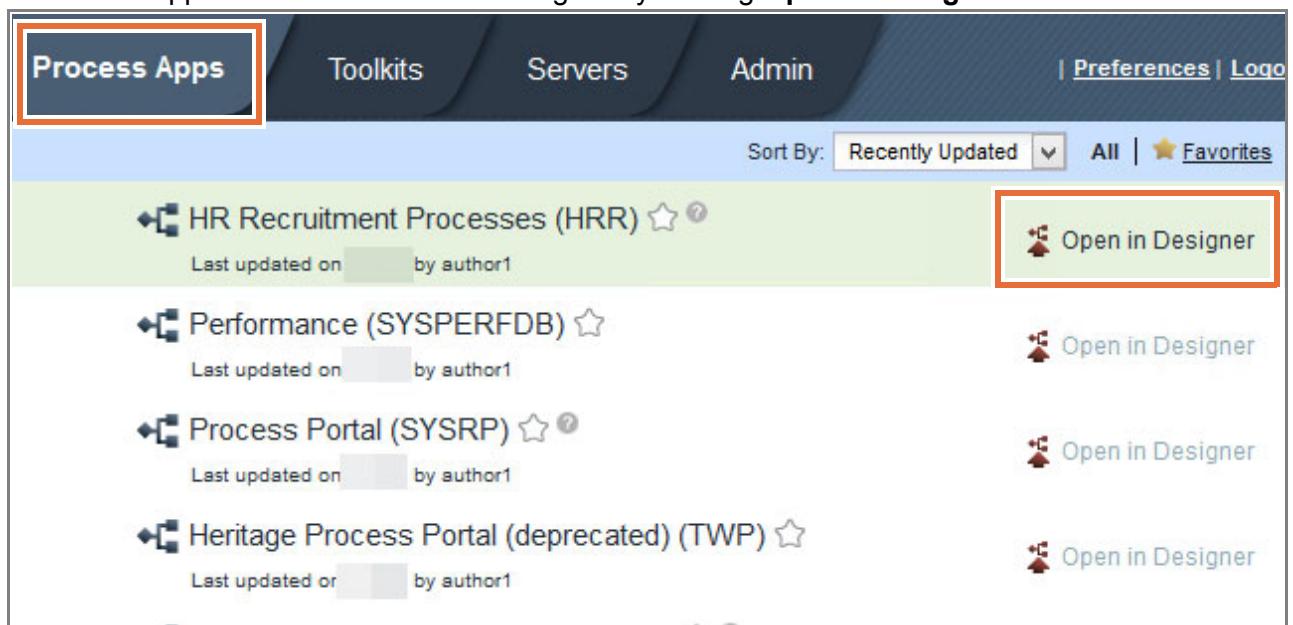
Formatting tools: B I U 12pt

In the bottom right corner of the dialog, there is a "Create" button, which is highlighted with a red rectangular border.

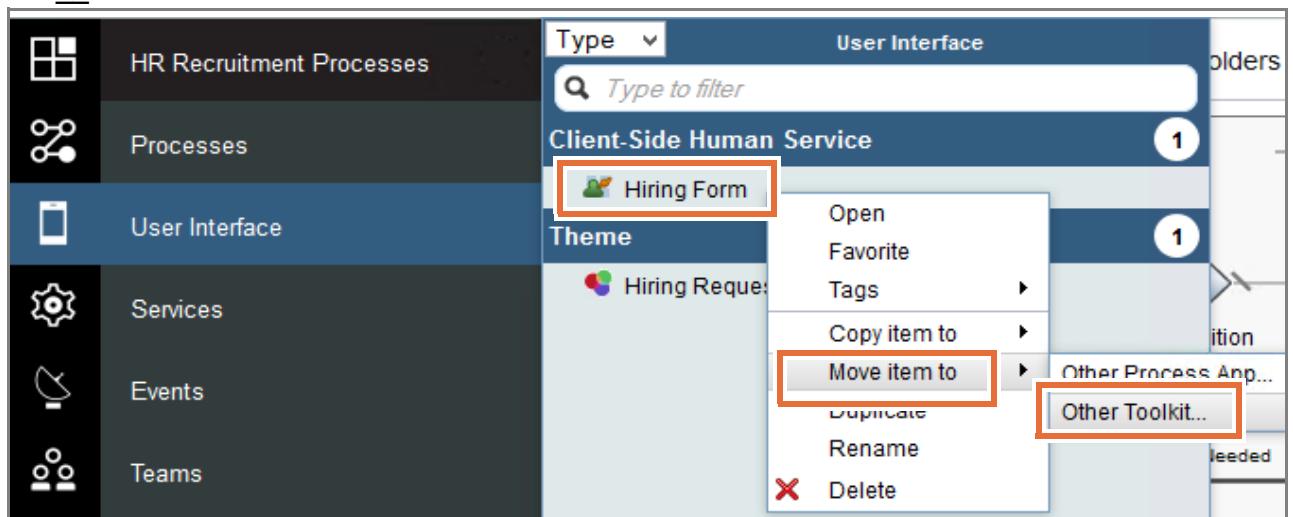
- __ f. The Hiring Requisition Toolkit (HRT) shows up under the **Toolkits** tab.



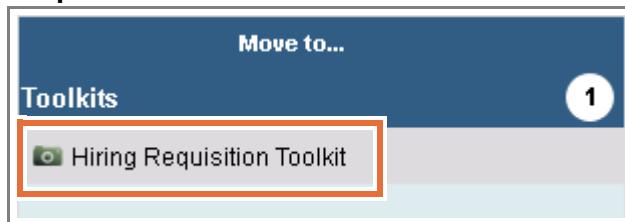
- __ 4. Move the client-side human service into the toolkit and create a baseline snapshot for the test environment
- __ a. Click the **Process Apps** tab. Open the **HR Recruitment Processes (HRR)** process application in the Process Designer by clicking **Open in Designer**.



- __ b. In the library, right-click the **User Interface > Hiring Form** client-side human service.
- __ c. Click **Move Item to > Other Toolkit**.



- __ d. Select **Hiring Requisition Toolkit**.



- __ e. Keep the defaults and click **Move**.

Moving items to Toolkit: Hiring Requisition Toolkit

Items to be moved
Checked items will be moved.

Hiring Form

Dependent items to be moved
Checked items will be moved. Items that are not moved might cause broken references.

HiringRequisition

Position

CompensationDetails

DepartmentDetails

RecruitingDetails

Other items that depend on items being moved
Selected items will be moved. Items that are not moved might cause broken references.

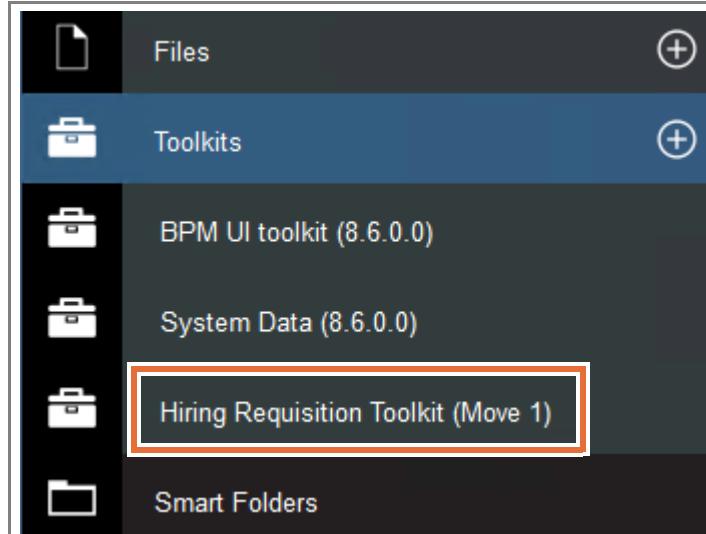
Hiring Request Process

Approve Hire Request

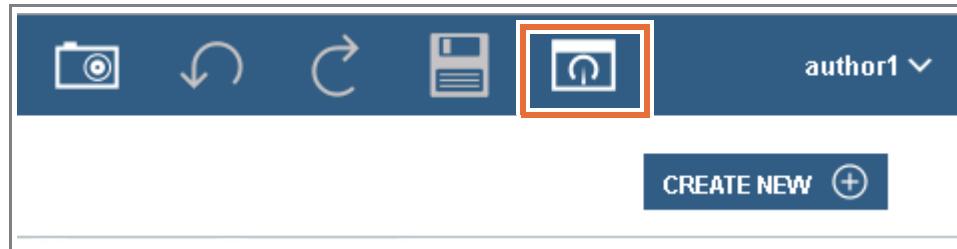
Move **Cancel**

**Note**

All the dependent assets are also moved into the toolkit to include the variables that the client-side human service depends on. A dependency on the toolkit is automatically created inside the process application. In the Library section, expand **Toolkits** to see the Hiring Requisition Toolkit.

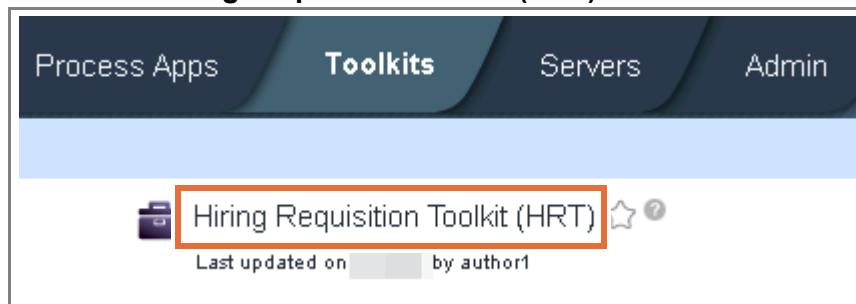


- ___ f. Return to the **Process Center**.

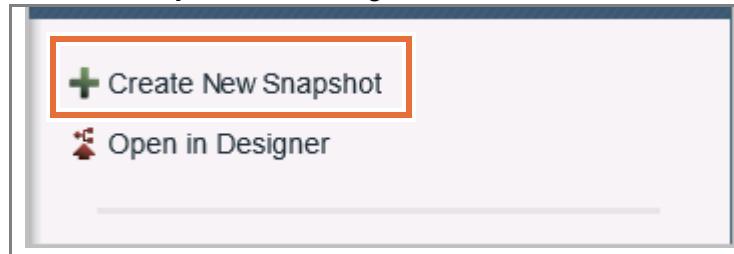


- ___ g. Click the **Toolkits** tab.

- ___ h. Click the toolkit **Hiring Requisition Toolkit (HRT)**.



- __ i. Click **Create New Snapshot** on the right side.



- __ j. Name the snapshot **Baseline** and add a description: This is the baseline snapshot of the Hiring Requisition Toolkit

Then, click **Create**.

A screenshot of a 'Create New Snapshot' dialog box. The title bar says 'Create New Snapshot' and has a close button. The main area has two sections: 'Snapshot Name:' with a text input containing 'Baseline' and 'Documentation:' with a rich text editor toolbar and a placeholder text area containing 'This is the baseline snapshot of the Hiring Requisition Toolkit'. At the bottom right is a 'Create' button, which is highlighted with a red box.

__ k. The snapshot **Baseline** is created.

Process Apps **Toolkits** Servers Admin

Hiring Requisition Toolkit (HRT) Snapshots History

Current

Last changed on by author1
(Not Used)

Baseline (B) (New)

Created on by author1
► Where used:

Move 1 (M1) (New)

Created on by author1
► Where used:

Part 4: Create or update dependency on a toolkit

- ___ 1. Update the toolkit version on a process application.
 - ___ a. Click the **Process Apps** tab and open the HR Recruitment Processes (HRR) process application in the designer by clicking **Open in Designer**.

The screenshot shows the Process Apps tab selected in the top navigation bar. Below it, a list of process applications is displayed. The first item, "HR Recruitment Processes (HRR)", is highlighted with a red box around its name. To the right of this item is a button labeled "Open in Designer", which is also highlighted with a red box.

Process Application	Last updated on	Author	Action
HR Recruitment Processes (HRR)	by author1		Open in Designer
Performance (SYSPERFDB)	by author1		Open in Designer
Process Portal (SYSPRP)	by author1		Open in Designer
Heritage Process Portal (deprecated) (TWP)	by author1		Open in Designer

- ___ b. When you return to the process application, Expand the **Toolkits** category in the Library section. The Hiring Request Toolkit shows a warning.

The screenshot shows the Library section with the "Toolkits" category expanded. A toolkit named "Hiring Requisition Toolkit (Move 1)" is highlighted with a red box and contains a warning icon (an exclamation mark inside a triangle).

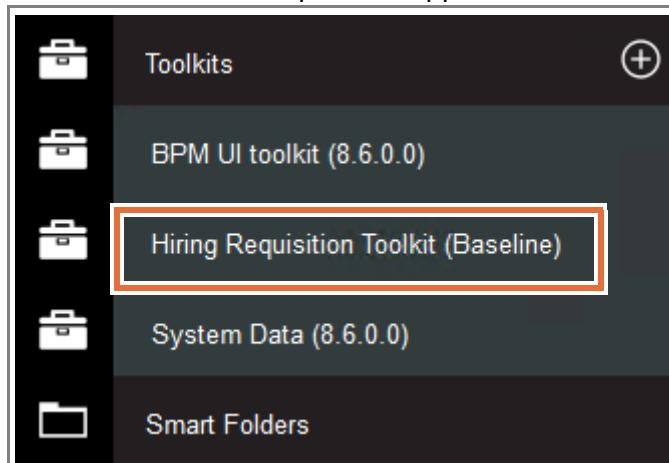
Toolkit
BPM UI toolkit (8.6.0.0)
Hiring Requisition Toolkit (Move 1)
System Data (8.6.0.0)

- ___ c. Click the arrow next to the Toolkit and select **Upgrade Dependency to (Baseline)**.

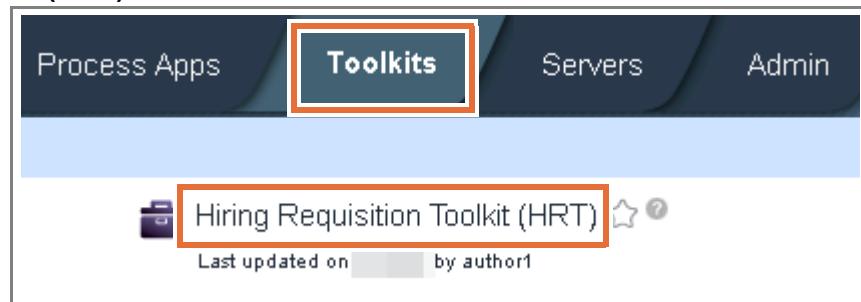
The screenshot shows a context menu for the "Hiring Requisition Toolkit (Move 1)" toolkit. The "Upgrade Dependency to (Baseline)" option is highlighted with a red box.

- [Upgrade Dependency to \(Baseline\)](#)
- [Ignore Version \(Baseline\) of Dependency](#)
- [Change Version of Dependency](#)
- [Remove Dependency](#)

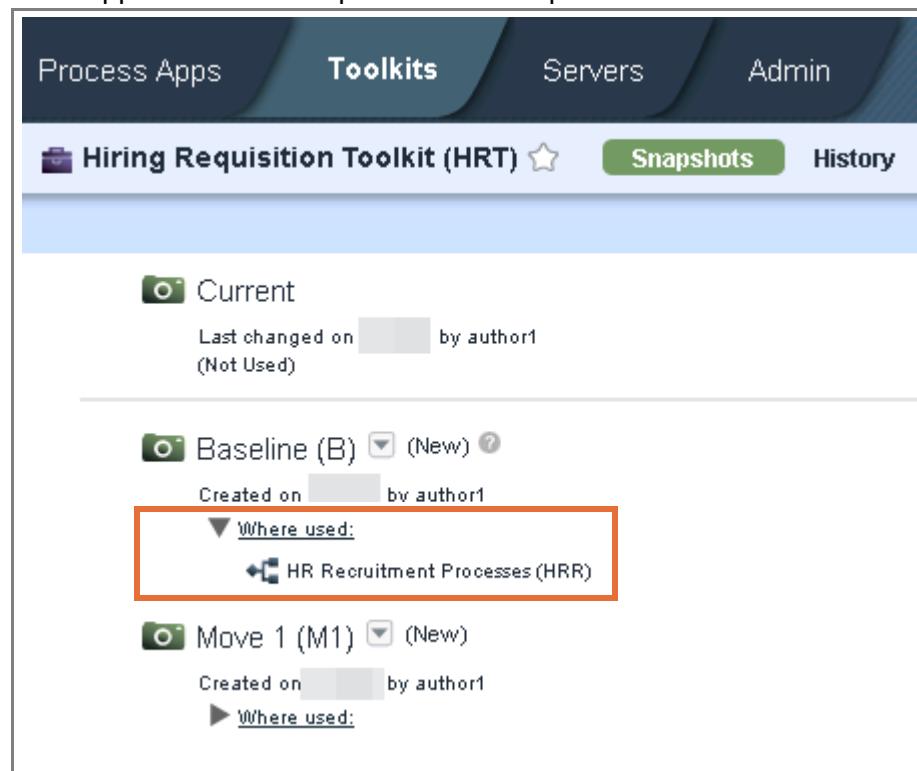
- ___ d. In the Toolkits category, verify that the Hiring Requisition Toolkit (Baseline) is now linked to the HR Recruitment Processes process application.



- ___ e. Return to the **Process Center > Toolkits** tab, and select the **Hiring Requisition Toolkit (HRT)**.



- ___ f. Expand the **Where used** collapsible section of the **Baseline (B)** snapshot to view any process applications that depend on the snapshot.



__ g. Minimize the browser window for later use.

End of exercise

Exercise review and wrap-up

In this exercise, you looked at Playback 1: Conducting the playback session, how to expose a process application for a team to start, and how to control the process flow by using coaches. You also created the Hiring Requisition Toolkit.

Exercise 9. Playback 2: Integrations

Estimated time

01:30

Overview

This exercise covers how to create implementation assets to support Playback 2: Integrations.

Objectives

After completing this exercise, you should be able to:

- Create a decision service
- Create and configure a UCA
- Start a process with a message start event
- Use tagging to organize assets
- Query a database to obtain information and populate a list variable
- Create environment variables (ENVs) and exposed process variables (EPVs)
- Change a text control to a single select control

Introduction

In Playback 2: Integrations, you implement the process interactions and integrations so the process can have all the functions that are needed to complete any process activity. Not all the functions are developed within the process application when it comes to user interactions. Some of the business data can be found, for instance, in systems within an organization. So it is important to be able to retrieve the data, and that is why integrations play a vital role in having a full and robust process application. Other process interactions involve events within the process model. In this stage of Playback 2: Integrations, you implement those system activities that set business data and process flow data with rules and automations. These intermediate events or even start events might need unique event handlers, such as listeners for messages that trigger an event. It would be the appropriate time to implement, test, and finalize all remaining process application interactions.

Requirements

Successful completion of the previous exercise is required.

Exercise instructions

Part 1: Create a decision service

The Hire Request business process is a candidate to include a decision service in the process flow. Decision services allow the business process to make routine decisions that are based on real business rules to speed up the process and eliminate user error.

If the salary is not compliant, the criteria that they must use is to route hiring requisitions for administrator override. The salary compliance depends on the job level. Different job levels have different compliance ceilings. In the as-is process, a user decides. In the to-be process, a user was going to decide, but now they want the system to decide based on business rules.

Your task is to implement that decision service in a process. You created the process model during Playbacks 0 and 1, and in Playback 2, you implement all the services.

This exercise is long, so a break is included in the middle of the instructions. The instructions include some swapping back and forth and moving around different assets in the library. The sequence is intentionally set to simulate the steps that a developer takes when creating these assets, and reflects real-world development efforts.

- 1. Create a decision service.
 - a. Click the **Process Apps** tab.
 - b. Click the **Open in Designer** link next to the HR Recruitment Process (HRR) process application.

The screenshot shows the Process Apps interface. The top navigation bar includes tabs for Process Apps, Toolkits, Servers, Admin, and links for Preferences and Logout. Below the tabs, there is a search bar and a sorting dropdown set to 'Recently Updated'. A list of applications is displayed:

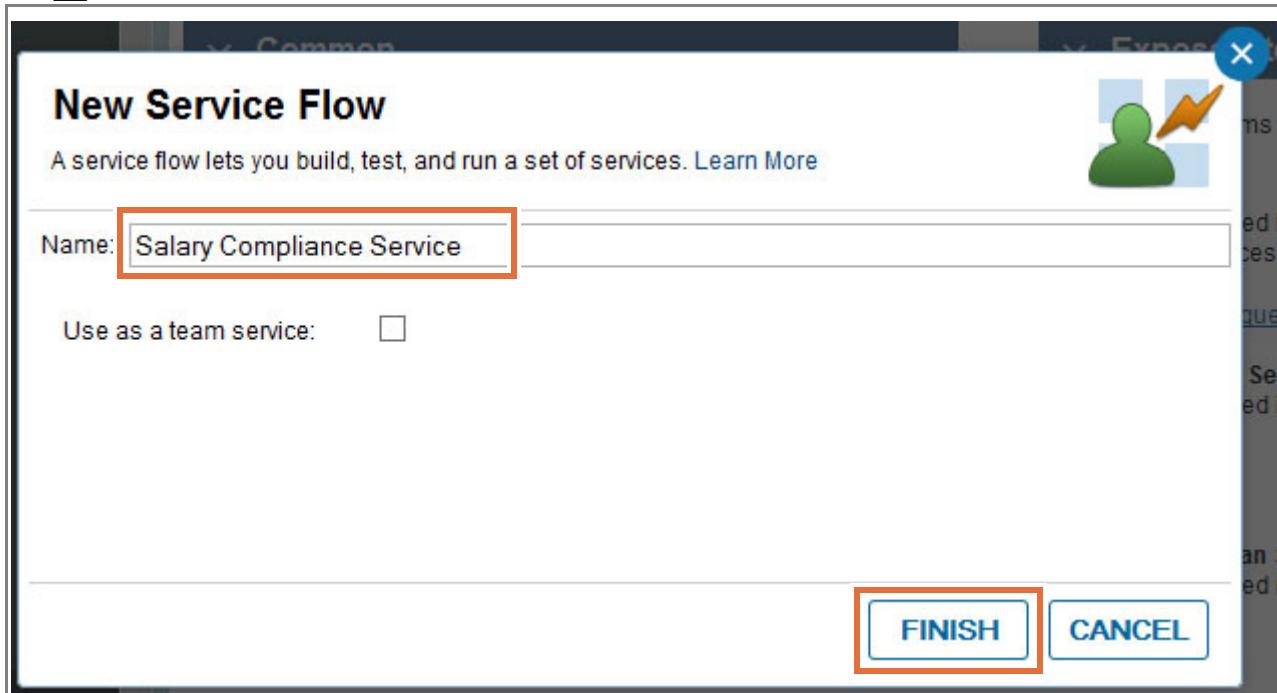
- HR Recruitment Processes (HRR)** (highlighted with a red box): Last updated on [redacted] by author1. To its right is an 'Open in Designer' button (also highlighted with a red box).
- Procurement Sample (STPPS1)**: Last updated on [redacted] by author1. To its right is an 'Open in Designer' button.
- Hiring Sample Advanced (HSBV1)**: Last updated on [redacted] by author1. To its right is an 'Open in Designer' button.

- c. In the library, click the (+) plus sign next to Services. Click **Service Flow** to create a service.

The screenshot shows the Services library interface. On the left, there are categories: User Interface, Services (highlighted with a red box), Events, and Teams. Each category has a '+' icon to its right. On the right, a detailed view for 'Services' is shown:

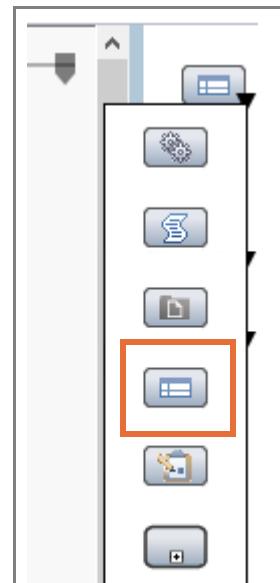
- Name:** HR Recruitment Processes
- Type:** New
- Service Flow** (highlighted with a red box): This option is selected under the 'External Service' section.
- Web Service**: Another service type listed.

- __ d. Name the service Salary Compliance Service and click **Finish**.

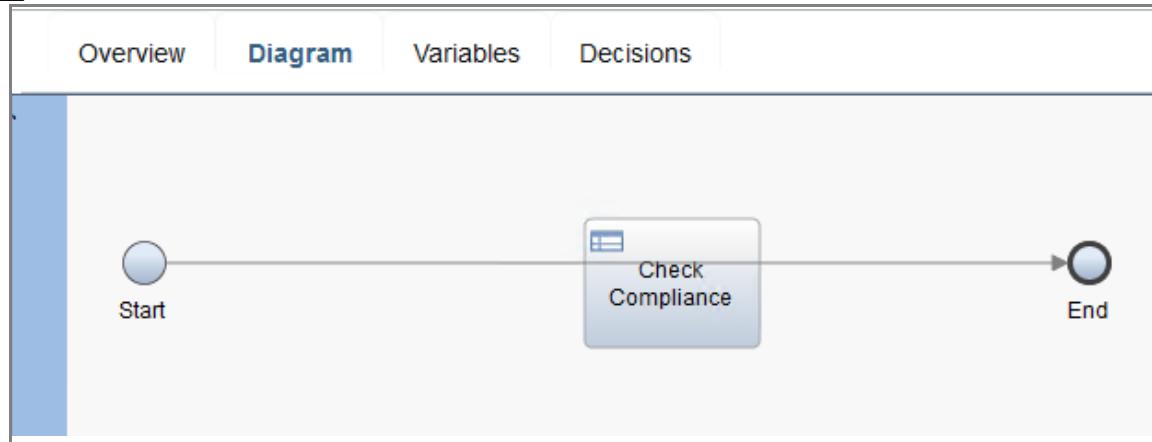


- __ 2. Add a Business Action Language rule.

- __ a. You are now in the new service. Expand the top activity and drag a **Decision** step from the service palette onto the canvas between the Start and End events.



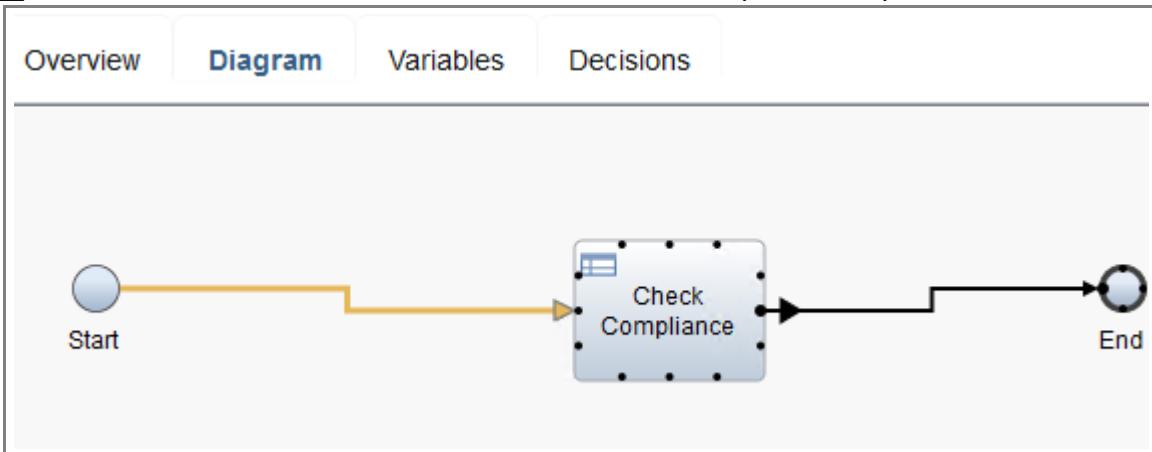
- b. Name the Decision: Check Compliance



Reminder

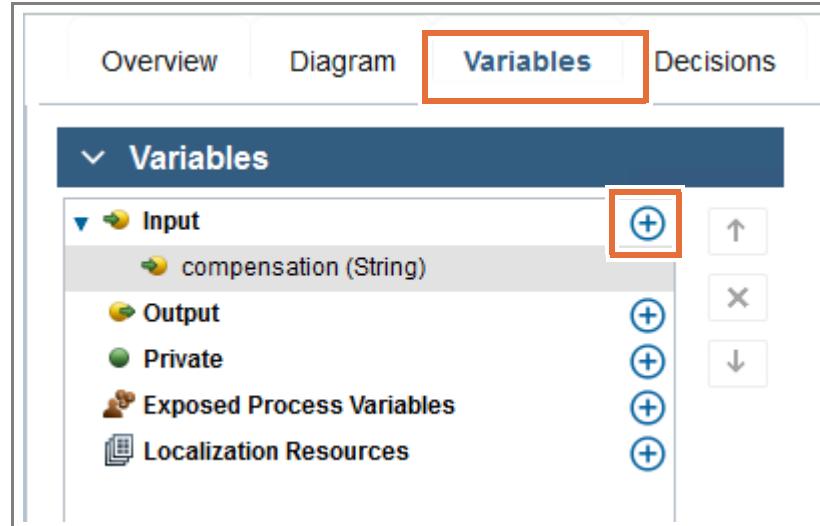
You can rename artifacts two ways: either through the **Properties > General > Common** section, or by double-clicking the artifact and renaming it on the canvas.

- c. Connect the start and end flows to the Check Compliance step.

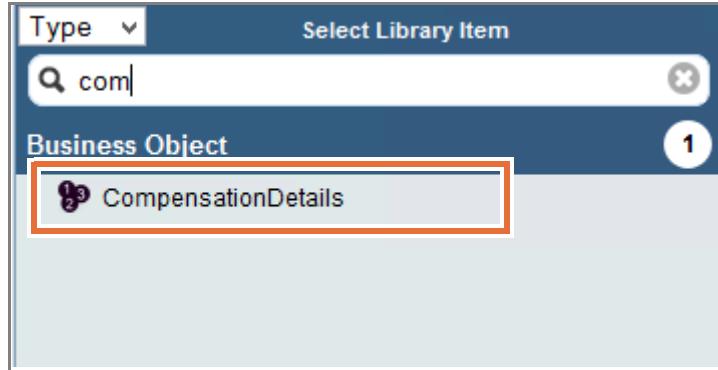


- ___ d. Straighten the flows.
 ___ 3. Add variables to the service.
 ___ a. Click the **Variables** tab.

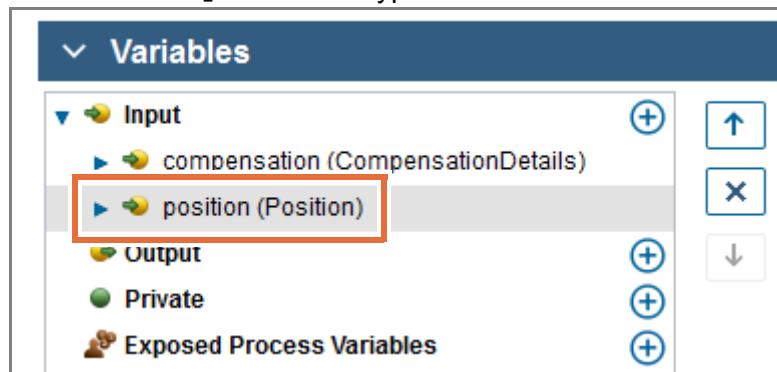
- __ b. Add an input and name the variable: compensation



- __ c. Click **Select** next to the Variable type. To filter the variable type list, type: Com
Click the variable type CompensationDetails.



- __ d. Add a second variable position of type Position.



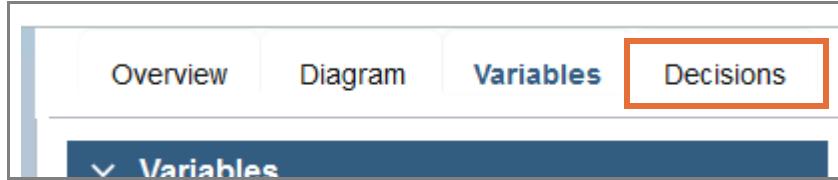
__ e. Add an output variable: isCompliant (String) .



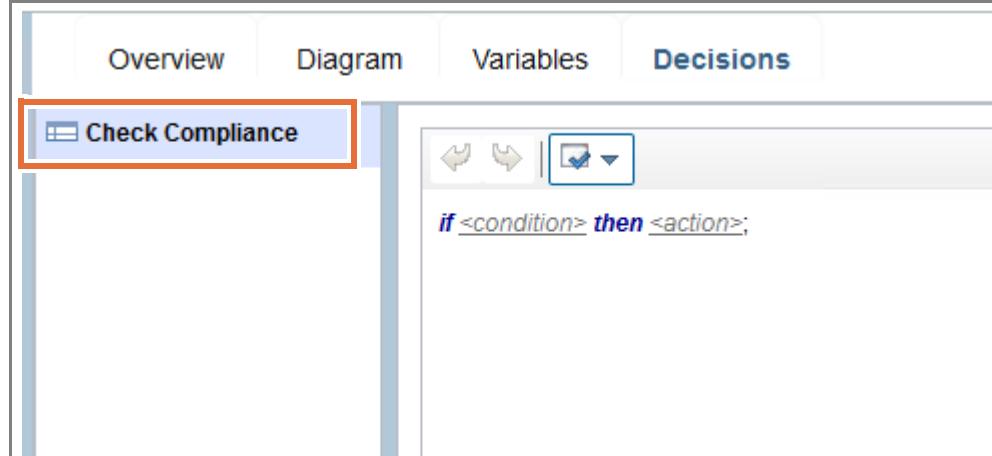
__ f. Save your changes.

__ 4. Add the rules for the service.

__ a. Click the **Decisions** tab.



__ b. Select the **Check Compliance** BAL rule on the left.



A default rule is added to the BAL rule editor. The condition and action options are editable. If you click either option, an editor is displayed to assist you with writing the rule. As is the case with a Java or JavaScript editor, you can also use content assist (Ctrl+Space).

The BAL rule is a simple set of if-then statements. In this example, if the salary is over or under a threshold, it is not compliant. Otherwise, it is compliant. The threshold levels are:

- Associate. Minimum salary: 40,000, maximum: 60,000
- Manager. Minimum salary: 50,000, maximum: 75,000
- Director. Minimum salary: 70,000, maximum: 95,000
- President. Minimum salary: 100,000, maximum: 150,000



Troubleshooting

The virtual machine settings occasionally cause unexpected behavior when using content assist. If the content assist window ever disappears, press the Delete key and then press the spacebar to see the content assist window display again. Use content assist as much as possible when building rules.



Note

To avoid any typographical errors, the rules text is provided in the file `BAL Rules.txt` at the location: `C:\labfiles\Exercise_Support_Files\Exercise09\`

The text provided between the angle brackets `<>` are variables and are included for your understanding. Do not include it in the script.

- ___ c. For the first rule, select the `<condition>` statement for your rule, and select the following rule pieces:

the salary to offer of `<a compensation details>`

compensation

is between `<min>`

`<number>`

`40000 <space>`

and `<max>`

`<number>`

`60000 <space>`

and

the job level of `<a position>`

position

contains `<a string>`

`"Associate"`

- ___ d. Select the `<action>` statement of your rule, and select the following rule pieces:

set `<variable>`

isCompliant

to

`<string>`

`"1";`

`else <space>`

```
set <variable>
  isCompliant
  to <variable value>
<string>
"0";
```

The screenshot shows a software interface for defining rules. At the top, there are navigation icons (back, forward, search) and a toolbar icon. Below the toolbar, the rule is defined in a natural language-like syntax:

```
if the salary to offer of compensation
is between 40000 and 60000 and
the job level of position contains "Associate"
then set isCompliant to "1";  

else set isCompliant to "0";|
```

The BAL rule reads much like a natural sentence. You assign a value in the first rule, and the other rules below it evaluate. If one of the rules evaluate to true, then it overwrites the variable.

- ___ e. Click the (+) plus symbol to create a second rule.

The screenshot shows the same software interface as before, but with a red box highlighting the '+' button in the toolbar on the right side. This button is used to add a new rule to the list.

The rule definition remains the same as in the previous screenshot:

```
if the salary to offer of compensation
is between 40000 and 60000 and
the job level of position contains "Associate"
then set isCompliant to "1";  

else set isCompliant to "0";|
```

- ___ f. Build the next rule by using the previous method to flag the salary as non-compliant when it does not fall within the same range. Build the rule to read:

```
if the job level of position contains "Manager"
and all of the following conditions are true:
- the salary to offer of compensation is more than 50000
- the salary to offer of compensation is less than 75000
then set isCompliant to "1";
```

```

if the job level of position contains "Manager"
and all of the following conditions are true:
- the salary to offer of compensation is more than 50000
- the salary to offer of compensation is less than 75000
then set isCompliant to "1";

```

- g. Continue on with the third BAL rule. Copy and paste the prior rule and change the variables as required.
- ```

if the job level of position contains "Director"
and all of the following conditions are true:
- the salary to offer of compensation is more than 70000
- the salary to offer of compensation is less than 95000
then set isCompliant to "1";

```
- h. Add the final rule. Copy and paste the prior rule and change the variables as required.
- ```

if the job level of position contains "President"
and all of the following conditions are true:
- the salary to offer of compensation is more than 100000
- the salary to offer of compensation is less than 150000
then set isCompliant to "1";

```



Reminder

To avoid any typographical errors, the rules text is provided in the file `BAL Rules.txt` at the location: `C:\labfiles\Exercise_Support_Files\Exercise09\`

- i. Change the position of the different rules to make the first rule you created the top rule, and each subsequent rule underneath the prior one by using the arrows on the right of each rule.

```

if the salary to offer of compensation
is between 40000 and 60000 and
the job level of position contains "Associate"
then set isCompliant to "1";
else set isCompliant to "0";

```

**Important**

All the four rules are created and are placed in the correct order.

The screenshot shows a rule editor interface with four stacked rule definitions:

- Rule 1:** *if the salary to offer of compensation is between 40000 and 60000 and the job level of position contains "Associate" then set isCompliant to "1"; else set isCompliant to "0";*
- Rule 2:** *if the job level of position contains "Manager" and all of the following conditions are true: - the salary to offer of compensation is more than 50000 - the salary to offer of compensation is less than 75000 then set isCompliant to "1";*
- Rule 3:** *if the job level of position contains "Director" and all of the following conditions are true: - the salary to offer of compensation is more than 70000 - the salary to offer of compensation is less than 95000 then set isCompliant to "1";*
- Rule 4:** *if the job level of position contains "President" and all of the following conditions are true: - the salary to offer of compensation is more than 100000 - the salary to offer of compensation is less than 150000 then set isCompliant to "1";*

- ___ 5. Save your work.
- ___ 6. Debug the service.

To make sure that the rules are working as expected, debug your service by providing default values to the variables.

- ___ a. Switch to the **Variables** tab.

- __ b. Click the input variable **compensation (CompensationDetails)**. Select the **Has Default** check box under the Default Value section on the right.

The screenshot shows the 'Variables' and 'Details' panels. In the 'Variables' panel, 'compensation (CompensationDetails)' is selected under 'Input'. In the 'Details' panel, the 'Name' field is 'compensation' and the 'Documentation' field is empty. Under 'Default Value', the 'Has default' checkbox is checked. Below it, a code editor shows the following Java code:

```

1 var autoObject = new tw.object.CompensationDetails();
2 autoObject.salaryToOffer = 0.0;
3 autoObject.bonusAmount = 0.0;
4 autoObject

```

- __ c. Change the value for the variable **salaryToOffer** to **45000.0**

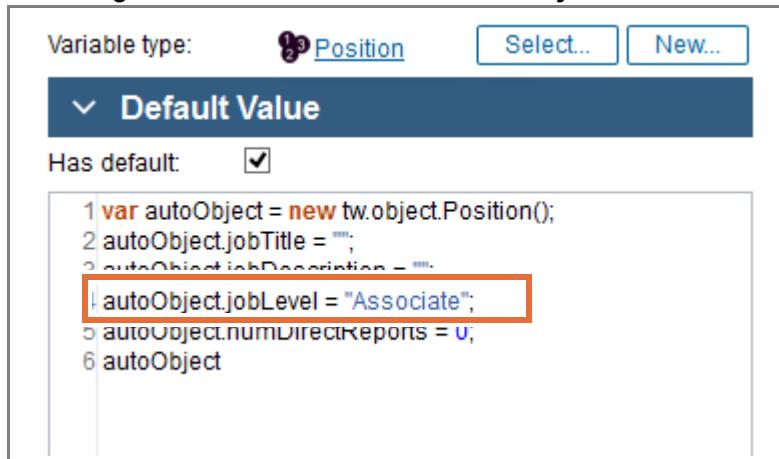
The screenshot shows the 'Default Value' configuration for the 'salaryToOffer' variable. The 'Has default' checkbox is checked. The code editor displays the following Java code, with the line 'autoObject.salaryToOffer = 45000.0;' highlighted by a red box:

```

1 var autoObject = new tw.object.CompensationDetails();
2 autoObject.salaryToOffer = 45000.0;
3 autoObject.bonusAmount = 0.0;
4 autoObject

```

- __ d. Click the input variable **position (Position)**. Select **Has Default** under the Default Value section on the right. Add the value for the variable **jobLevel** as: "Associate"



- __ e. Save the service and click the **Debug** button at the top.

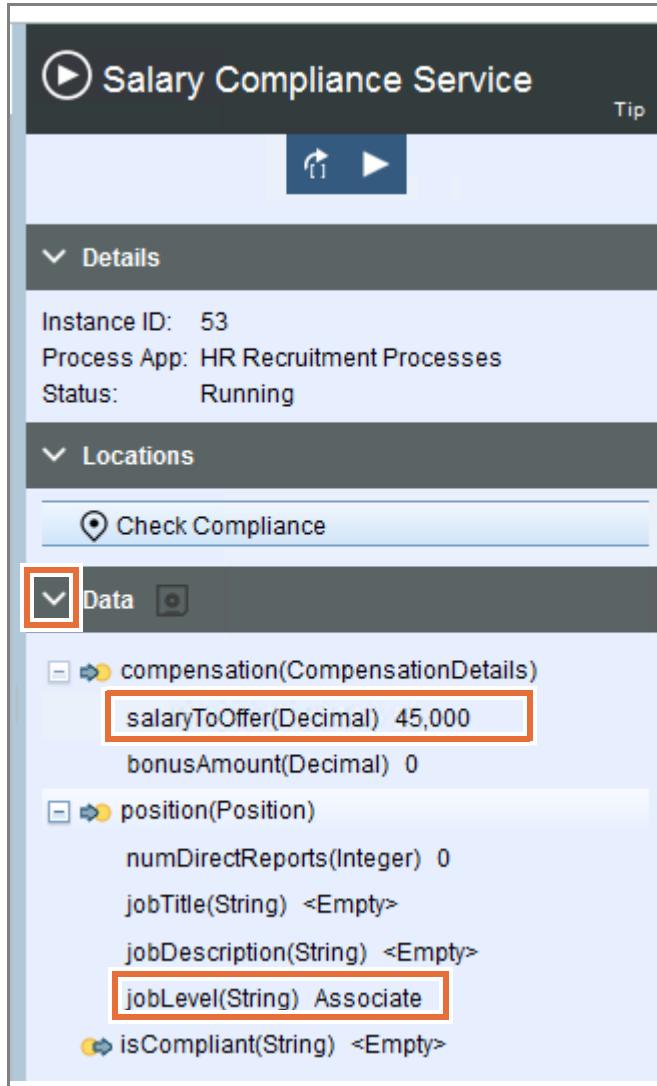


Note

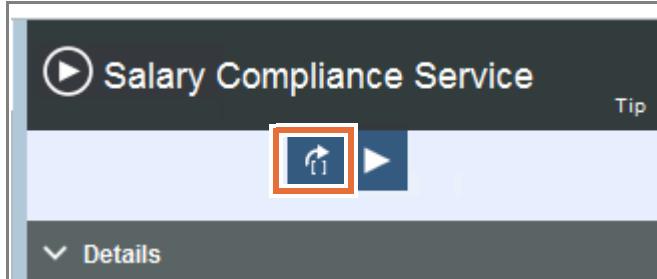
The view changes from the designer view to the inspector view.



- f. The debug window opens on the right to show information about the service. Expand the **Data** section on the bottom and verify that the default variables are assigned correctly.



- g. Click the **Step Over** icon on the top to test the first decision rule with the value for variable **salaryToOffer** as **45000** and the value for variable **jobLevel** as **Associate**.





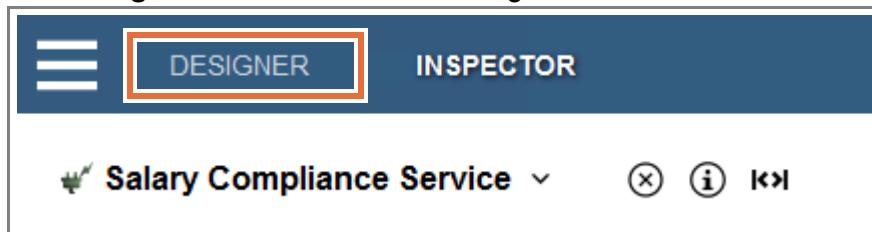
Information

The service executes the BAL rule, the service completes, and the status changes to Finished. The correct functionality with this set of inputs is that the request is compliant. You get an isCompliant variable value of 1 because for the Associate position the salary is in the range 40000 – 60000 and you tested with the value 45000, which makes this request compliant.

The screenshot shows the 'Salary Compliance Service' instance details. The status is 'Finished'. The data section includes compensation details (salaryToOffer: 45,000) and a position (Associate). The isCompliant variable is set to 1.

Variable	Type	Value
compensation	CompensationDetails	salaryToOffer(Decimal) 45,000 bonusAmount(Decimal) 0
position	Position	numDirectReports(Integer) 0 jobTitle(String) <Empty> jobDescription(String) <Empty> jobLevel(String) Associate
isCompliant	String	1

- h. Click the **Designer** tab to return to the Designer view.



- i. On the **Variables** tab, select the **position (Position)** input variable.

- __ j. Change the value of the **jobLevel** variable to "Manager" and save the service. Click the **Debug** button on the top.

The screenshot shows the 'Details' panel for a service variable named 'position'. The 'Variable type' is set to 'Position'. In the 'Default Value' section, the code is as follows:

```
1 var autoObject = new tw.object.Position();
2 autoObject.jobTitle = "";
3 autoObject.jobDescription = "";
4 autoObject.jobLevel = "Manager";
5 autoObject.numberOfReports = 0;
6 autoObject
```

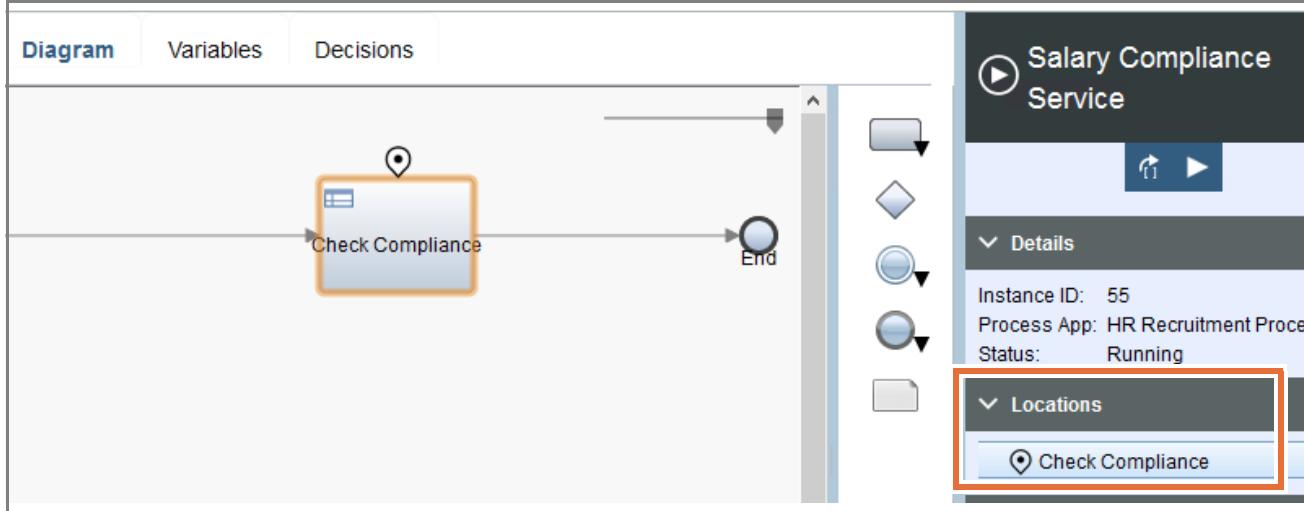
The line `4 autoObject.jobLevel = "Manager";` is highlighted with a red box.

- __ k. The information window opens on the right. Click the **Diagram** tab to view the service.

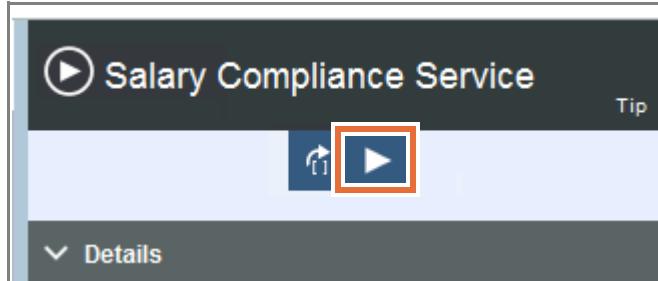


Information

The token on the Check Compliance service step shows the next step in the service to be run. The token matches the step that is shown in the Locations section in the information window.



- I. Expand the **Data** section and click the **Run** button on the top to run the service to completion.



Note

The Step Over button runs a single step in the service where the token is located. The Run button runs the service until a coach is displayed or until the service completes.

- ___ m. The service request is not compliant. You get an **isCompliant** variable value of 0 because the salary for the Manager position is in the range 50000 – 75000 and you tested with salary 45000, which makes this request non-compliant.

The screenshot shows the 'Salary Compliance Service' interface. At the top, there's a 'Tip' section with the text 'No actions were found.' Below it, the 'Details' section shows 'Instance ID: 55', 'Process App: HR Recruitment Processes', and 'Status: Finished'. The 'Locations' section also indicates 'No locations were found.' The 'Data' section contains two entries: 'compensation(CompensationDetails)' with 'salaryToOffer(Decimal) 45,000' and 'bonusAmount(Decimal) 0', and 'position(Position)' with 'numDirectReports(Integer) 0', 'jobTitle(String) <Empty>', 'jobDescription(String) <Empty>', and 'jobLevel(String) Manager'. The 'isCompliant(String) 0' entry is highlighted with a red box.

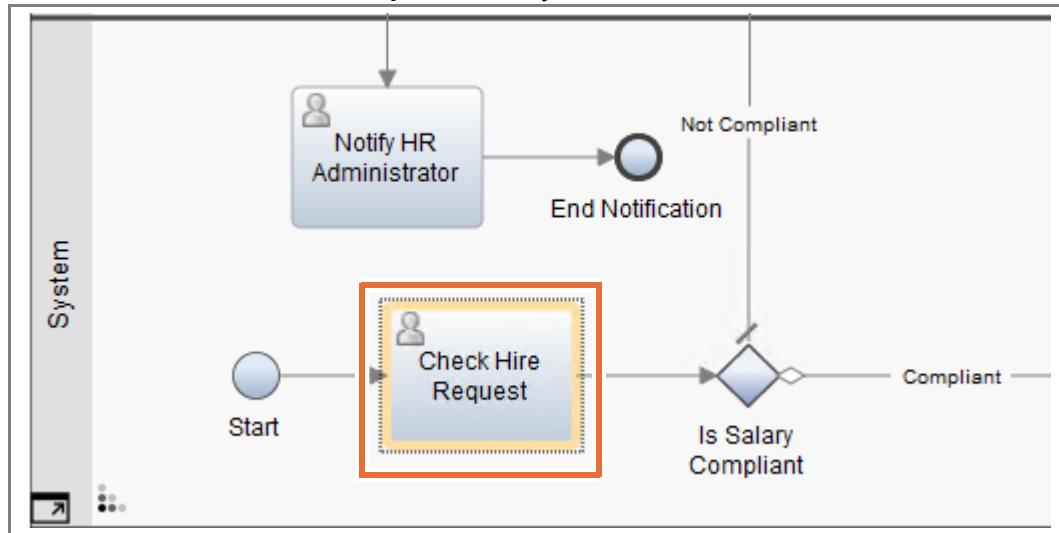
- ___ n. Return to the **Designer** view of the Process Designer.
- ___ o. Click the **Variables** tab and clear the **Has Default** check box under the **Default Value** section for the input variables **compensation (CompensationDetails)** and **position (Position)**.
- ___ p. Save your changes.
- ___ 7. Add the rule to the process.
- ___ a. Open the **Approve Hire Request** process.



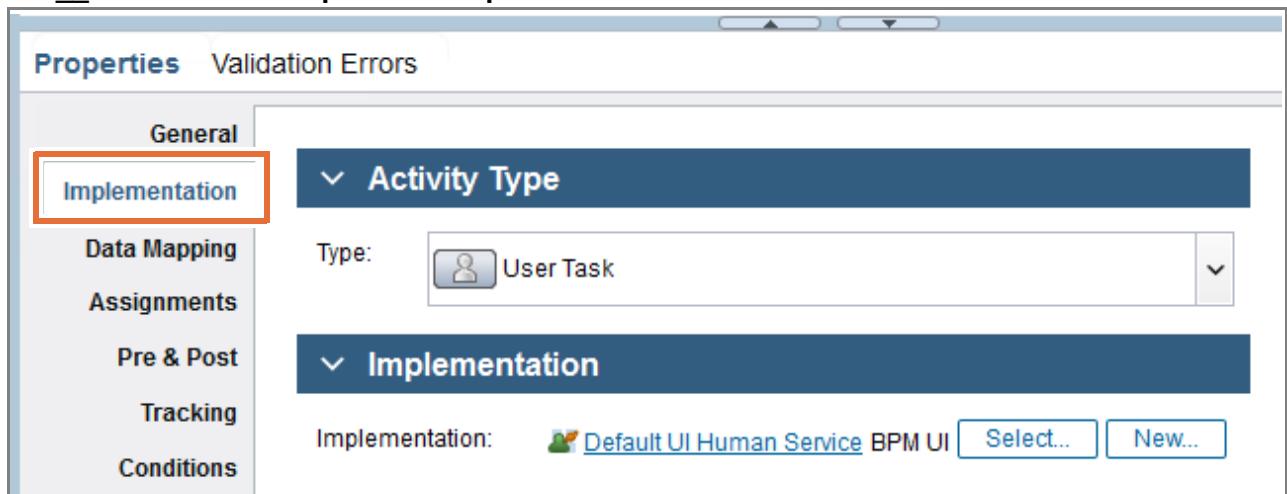
Note

You might need to click the menu shown by three horizontal lines, then click **Processes > Approve Hire Request**.

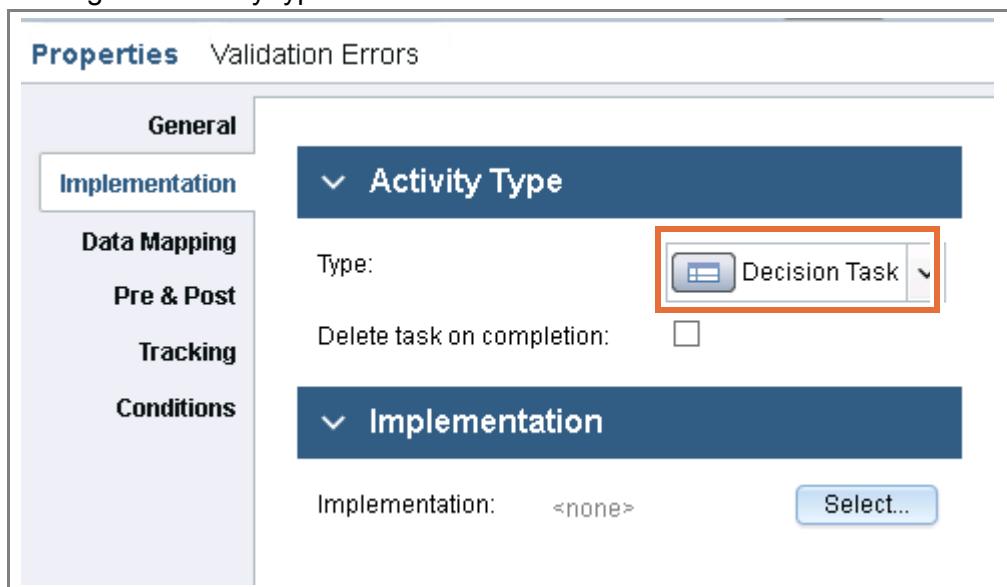
- __ b. Select the **Check Hire Request** activity in the **System** lane.



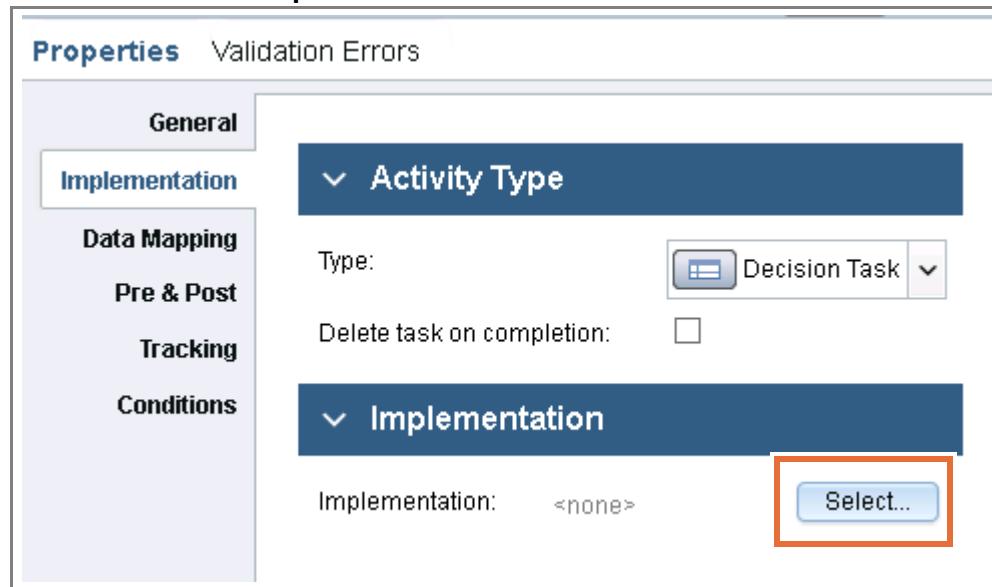
- __ c. Click the **Properties > Implementation** menu.



- __ d. Change the activity type to **Decision Task**.



- __ e. Click **Select** in the **Implementation** section.

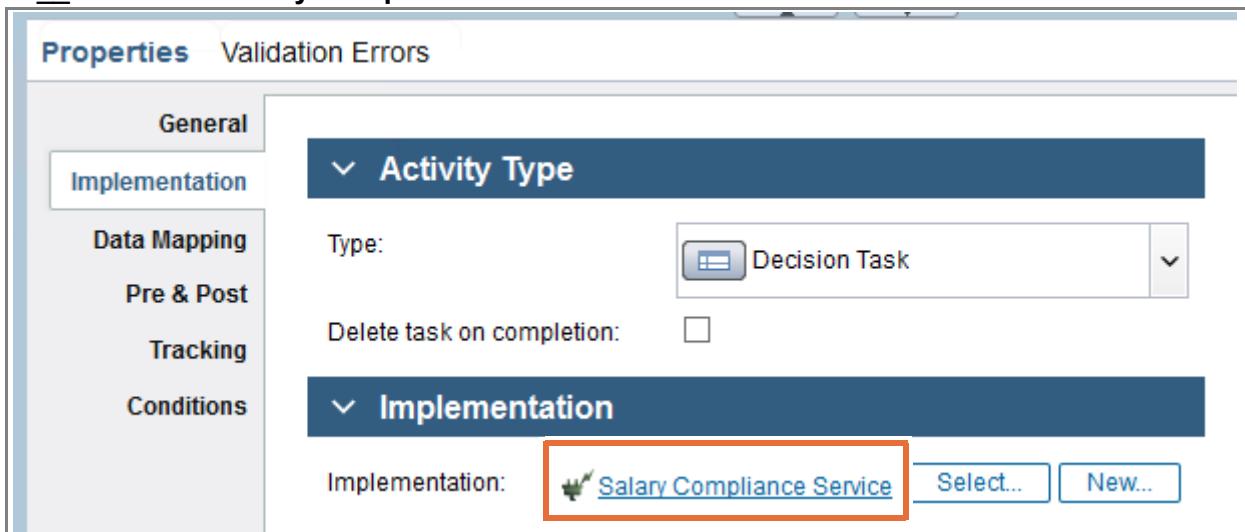


Information

If you want to run an automated service that does not require routing, you must select the **Delete task on completion** check box. This option deletes the task from the system and the process inspector after the task completes. When you select this check box, the Process Server does not retain audit data for the task. By default, this check box is not selected.

Deleting the task on completion from the task table in the database provides some performance advantages. However, in this case you do not select this option because you want to avoid losing some auditing data, including the details of when that system task executed and how long it took.

- __ f. Select **Salary Compliance Service**.



- __ 8. Map the variables for the **Check Hire Request** activity.

- __ a. Open the **Properties > Data Mapping** menu.

- __ b. Enter `tw.local.requisitionDetails.compensationDetails` for the first input to map it with **compensation**.
- __ c. Enter `tw.local.requisitionDetails.position` for the second input to map it with **position**.

Input Mapping

<code>tw.local.requisitionDetails.compensationDetails</code>			compensation (Compensation...)
<code>tw.local.requisitionDetails.position</code>			position (Position)

- __ d. Enter `tw.local.isCompliant` for the output to map it with **isCompliant**.

Output Mapping

isCompliant (String)		<code>tw.local.isCompliant</code>
--------------------------------------	--	-----------------------------------

Information

Notice that you do not need to send the full `requisitionDetails` object into every activity. If a subprocess or an activity needs only a small amount of data from the larger object, send in what is necessary. This approach helps the performance of your processes at run time.

If you know that variables hold the same data, it also helps to name those variables the same across your processes because it helps with maintenance and troubleshooting.

- __ e. Save your work.

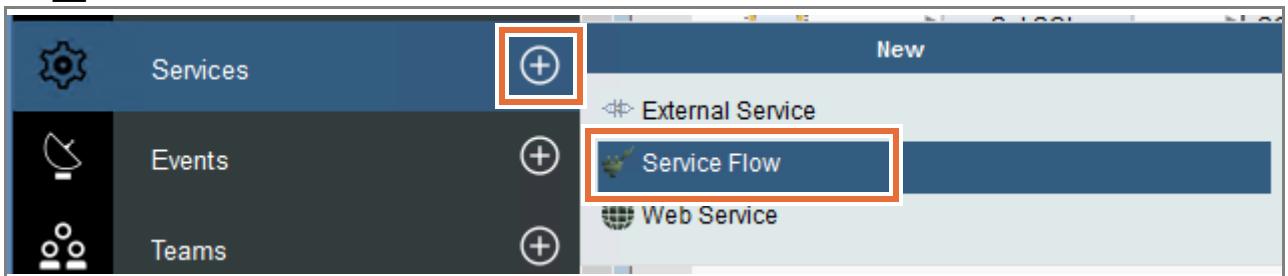
Part 2: Implement a message start event

In the Hiring Request Process modeling, you created a process that communicated how a process instance is started with an online data entry. Either customers or customer representatives complete the entry, or the entry is submitted by using older green screen systems. Tight integration is planned with this process and the older systems until they can be replaced with the new IBM Business Process Manager servers.

The older systems would create a special hire request that would purge obsolete records from the older database system. Because the new IBM Business Process Manager system must replace the older systems, this job must be initiated from the process. The good thing is that the Complete Hire Request service that is already included in the process is the same service that was used to process this special request. When a hire request is sent to this service with a blank hire request number, it initiates the purge service in the older system.

To satisfy the green screen system's start of the process, a Start Message event must be added to the process. Now it is time to implement the process so an instance of the process can be initiated monthly.

- 1. Create a **General System Service** to enable the undercover agent (UCA).
 - a. In the library, click the (+) plus sign next to the **Services** menu.
 - b. Click **Service Flow**.

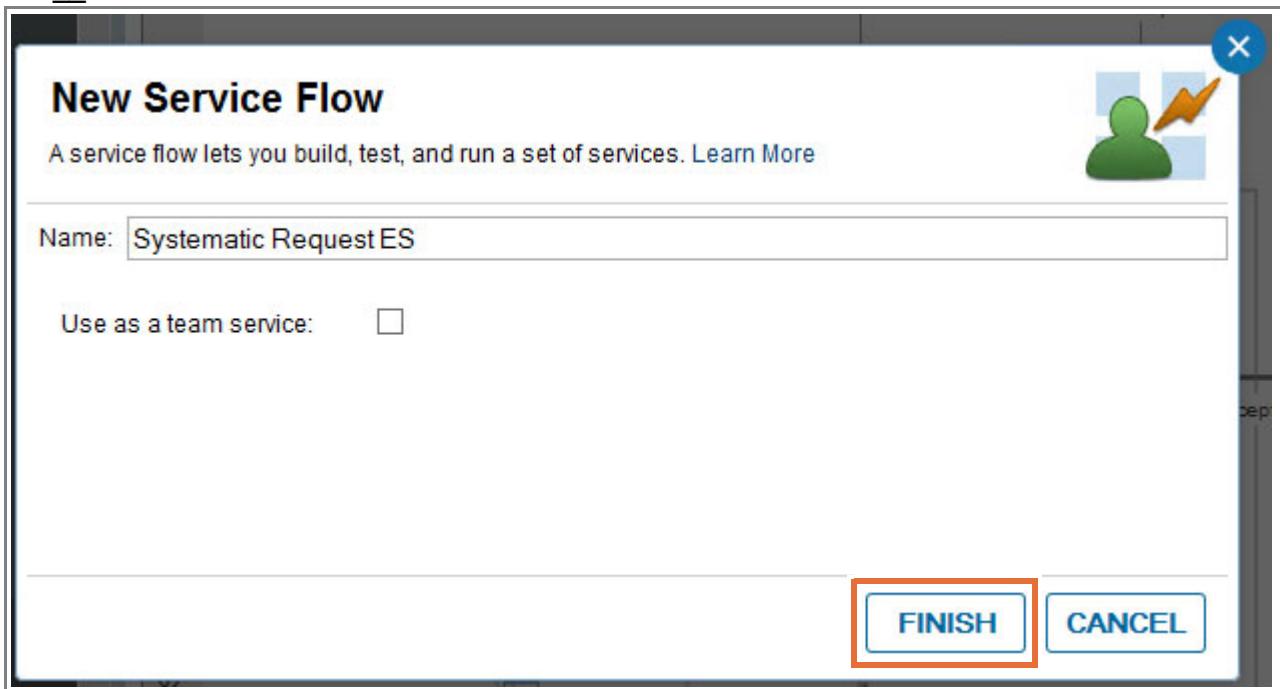


- c. Name the service: Systematic Request ES

 i
Information

Generally, the enabling service for a UCA is suffixed with `ES` for `Enabling Service`.

__ d. Click **Finish**.



__ e. Click the **Variables** tab. Continuing from Part 1 of this exercise, create a `requisitionDetails (HiringRequisition)` input and output variable.

Variables	
Input	(+)
↳ requisitionDetails (HiringRequisition)	(+)
Output	(+)
↳ requisitionDetails (HiringRequisition)	(+)
Private	(+)
Exposed Process Variables	(+)
Localization Resources	(+)

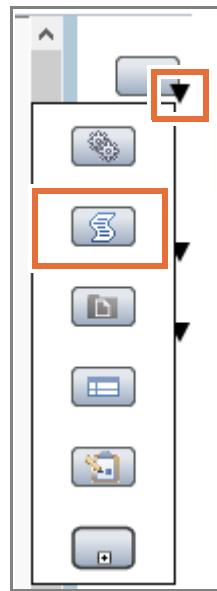


CAUTION

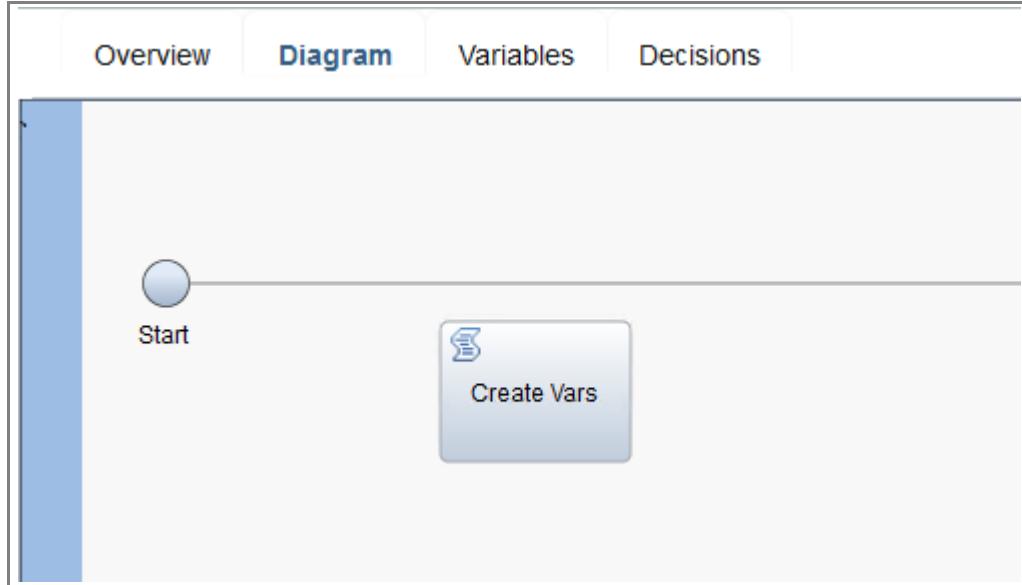
Make sure any single variable that is passed in and out of a service is spelled the exact same for both fields. Remember, variable names ARE case-sensitive. If you have any differences in the variable names, then the system considers them two different variables. This error frustrates many developers when troubleshooting why your variable is not passing out the expected data. Copying and pasting variable names ensures that your variables are the exact same, and reduces problems that result from typographical errors.

__ f. Save your work.

- 2. Create a server script to initialize the variables and pass it out of the service.
- a. Return to the **Diagram** tab. Expand the **Activity** selection and drag a server script from the palette onto the canvas.



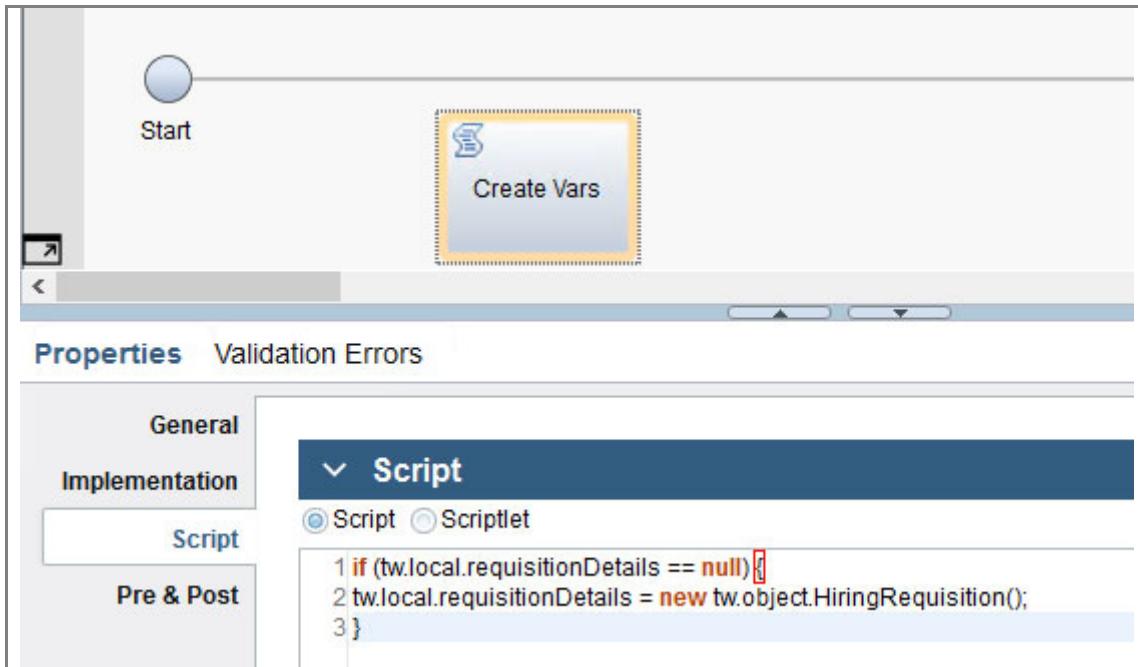
- b. Name the server script: Create Vars



- c. Create a server script to initialize the variables and pass it out of the service. In the **Properties > Script** menu, enter the following script in the box:

```

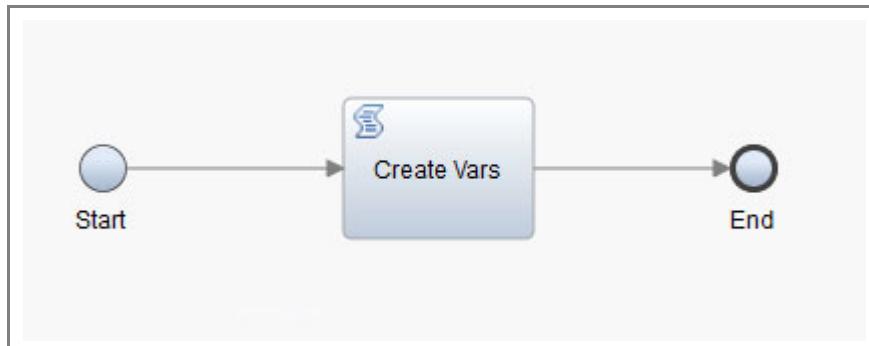
if (tw.local.requisitionDetails == null) {
    tw.local.requisitionDetails = new tw.object.HiringRequisition();
}
  
```



Note

To avoid any typographical errors, the script code is provided in the file `Script1.txt` at the location: `C:\labfiles\Exercise_Support_Files\Exercise09\`.

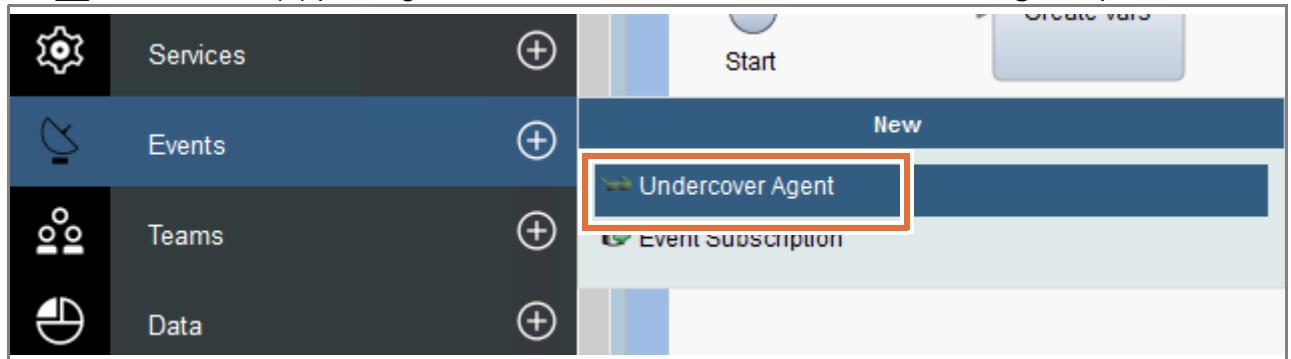
- ___ d. Connect the flows.



- ___ e. Save the service.

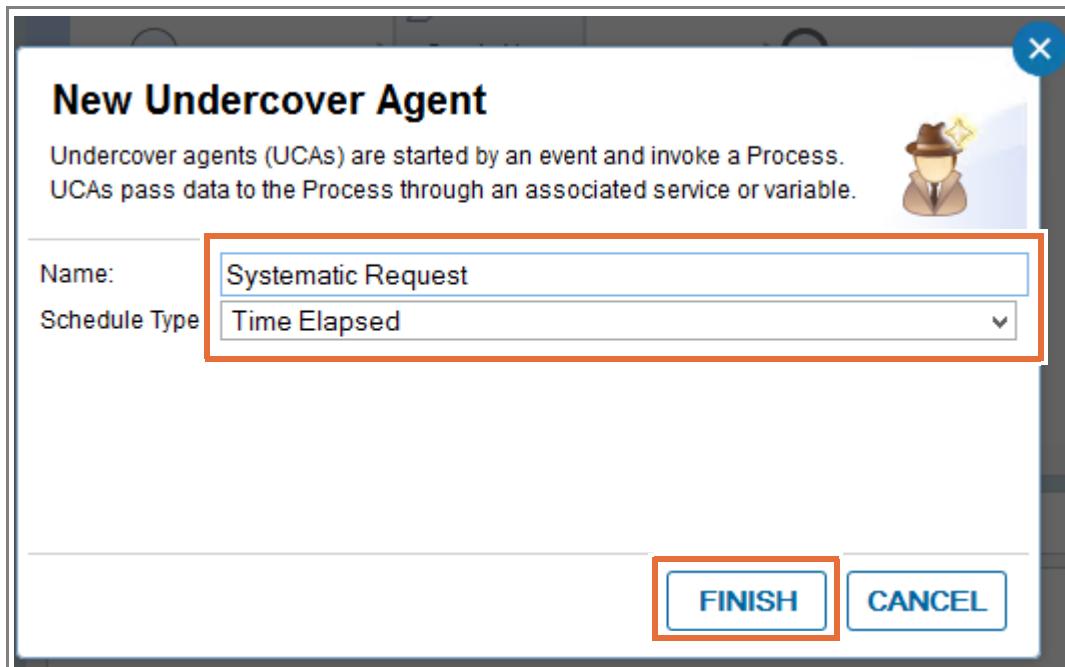
— 3. Create and enable an Undercover Agent.

— a. Click the (+) plus sign next to **Events**, and click the **Undercover Agent** option.



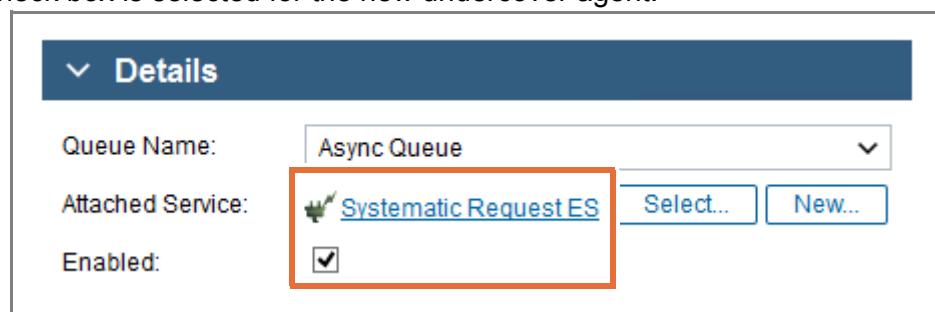
— b. Name the UCA: Systematic Request

Make sure that the **Time Elapsed** schedule type is selected. Click **Finish** to complete the creation of the UCA.



— c. In the Details section of the UCA settings page, click **Select** next to Attached Service.

Select the **Systematic Request ES** general system service. Verify that the **Enabled** check box is selected for the new undercover agent.



— d. Create the schedule according to the following script:

Every month (all months) on the first Saturday at Midnight on the hour.

Select all the months, then the **First**, **Saturday**, **Midnight**, and **On the hour** options.

This setting generally provides time for the purging service to complete over the weekend.

The screenshot shows the Undercover Agent configuration interface. In the 'Time Schedule' section, the following options are selected:

- Months:** January, February, March, April, May, June, July, August, September, October
- Days:** Every ..., First ..., Last ...
- Weekdays:** Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday
- Midnight:** 1:00 am, 2:00 am, 3:00 am, 4:00 am, 5:00 am, 6:00 am, 7:00 am, 8:00 am, 9:00 am
- On the hour:** 5 past the hour, 10 past the hour, 15 past the hour, 20 past the hour, 25 past the hour, 30 past the hour, 35 past the hour, 40 past the hour, 45 past the hour

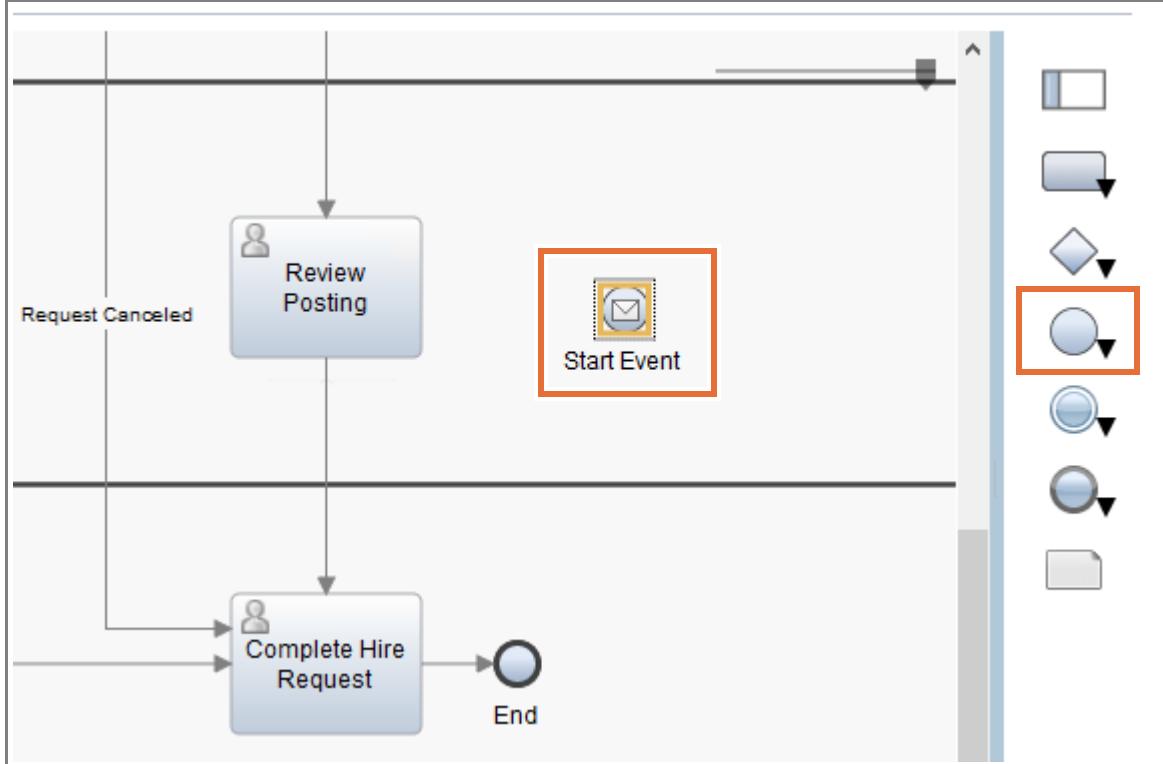
- ___ e. Save your work.
- ___ 4. Implement the **Systematic Request** start message event on the **Hiring Request Process** process.
 - ___ a. Open the **Hiring Request Process** process.



Note

You might need to click the menu with the three horizontal lines, and then click **Processes > Hiring Request Process**.

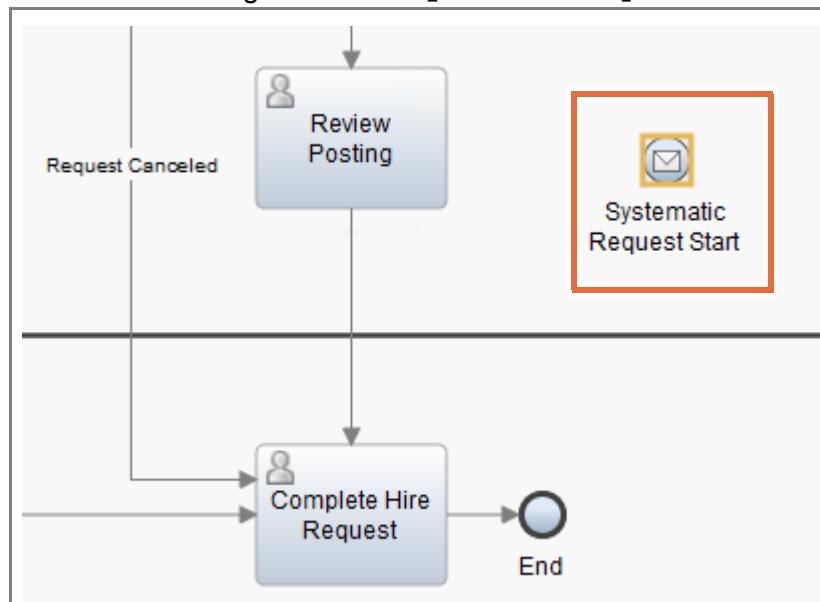
- b. Drag a **Start** event from the palette to the process near the **End** event.



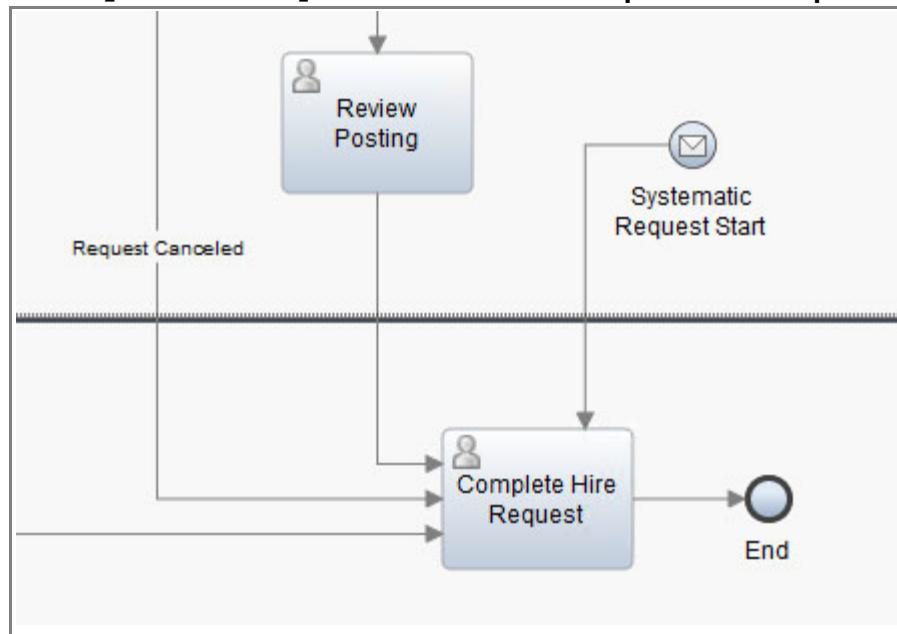
Information

Because a start event is already on the canvas, subsequent start events added to the canvas default to a Start message event. You can also expand the start event in the palette and drag a Start message event to the canvas to get the same result.

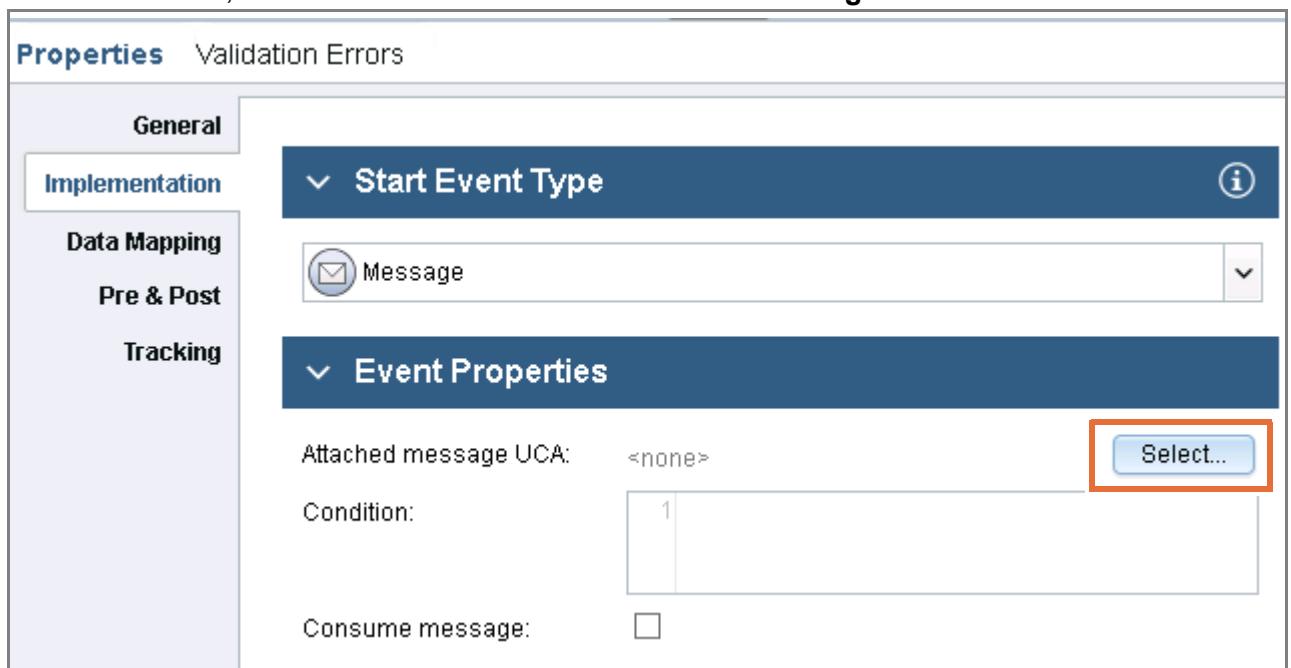
- c. Rename the start message event to: Systematic Request Start



- __ d. Connect Systematic Request Start to the **Complete Hire Request** activity.



- __ e. Click the **Systematic Request Start** event, click the **Properties > Implementation** menu, and then click **Select** next to **Attached message UCA**.



- ___ f. Select the **Systematic Request** UCA that you created for the activity.

The screenshot shows the 'Properties' dialog for a 'Start Event Type'. On the left, there are tabs for 'General', 'Implementation', 'Data Mapping', 'Pre & Post', and 'Tracking'. The 'Implementation' tab is selected. Under 'Implementation', there is a section titled 'Start Event Type' with a dropdown menu labeled 'Message'. Below this is another section titled 'Event Properties' with fields for 'Attached message UCA' (containing 'Systematic Request'), 'Condition' (containing '1'), and 'Consume message' (with an unchecked checkbox). A 'Select...' button is also present.

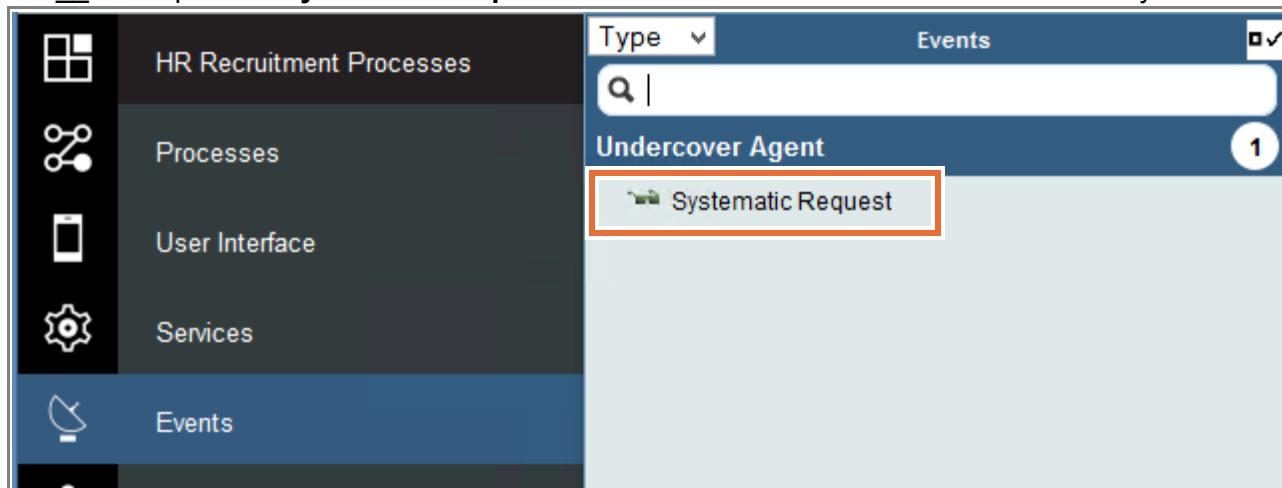
- ___ g. Save your work.
- ___ 5. Map the Systematic Request start event UCA outputs to the process variables.
- ___ a. Click the **Properties > Data Mapping** menu.
- ___ b. In the **Output Mapping** section, map the requisitionDetails output variable to the requisitionDetails (HiringRequisition) variable.

The screenshot shows the 'Properties' dialog for a 'Data Mapping' section. On the left, there are tabs for 'General', 'Implementation', 'Data Mapping', and 'Pre & Post'. The 'Data Mapping' tab is selected. Under 'Data Mapping', there is a section titled 'Output Mapping' with a mapping rule: 'requisitionDetails (Hi...)' is mapped to 'tw.local.requisitionDetails'. The 'tw.local.requisitionDetails' part is highlighted with a red box.

- ___ c. Save your work.

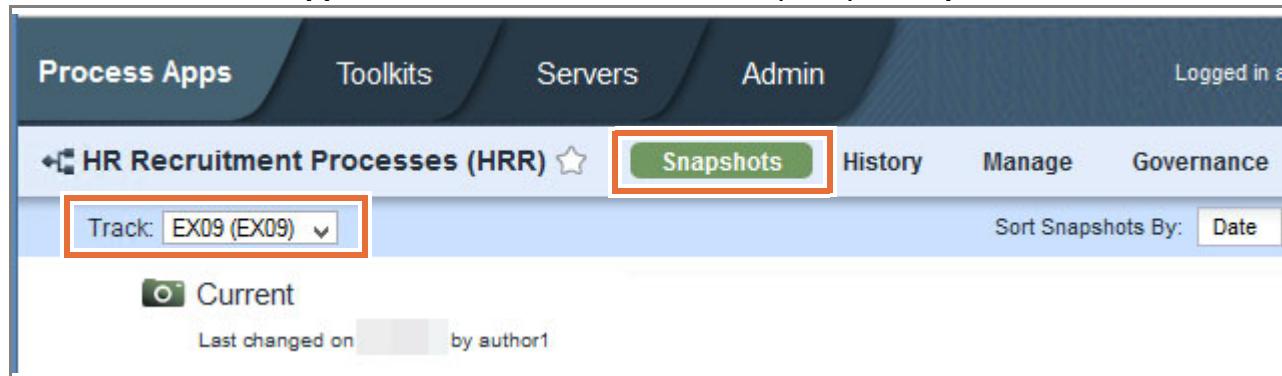
6. Test the UCA.

- a. Open the **Systematic Request** UCA listed in the Events section of the library.

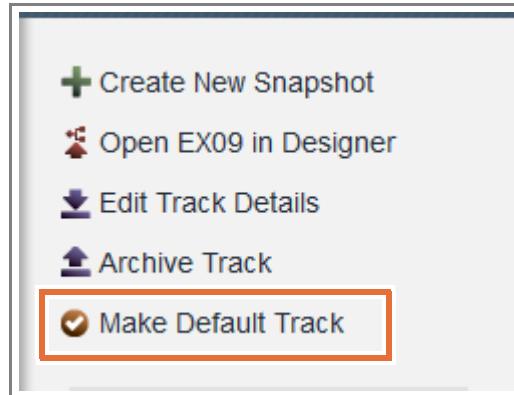


Important

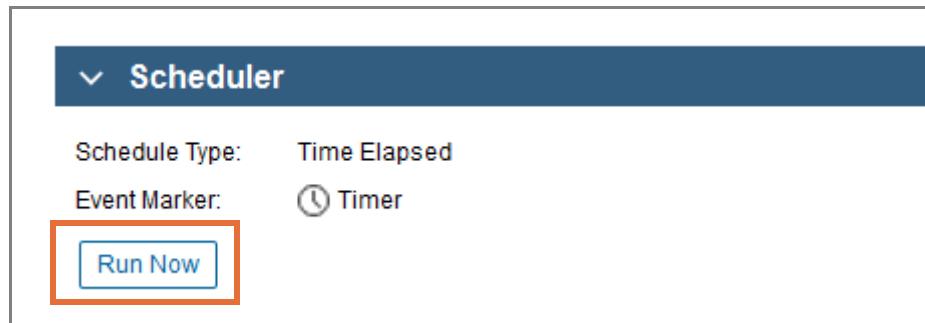
If you imported one of the solution files, you imported a track to your process application. UCAs can run only on the default track, which is usually the main track (tip) of the development. Before you run the time-based UCA, from the main window, click the process app name and select your track. Click the **Process Apps > HR Recruitment Processes (HRR) > Snapshots** menu.



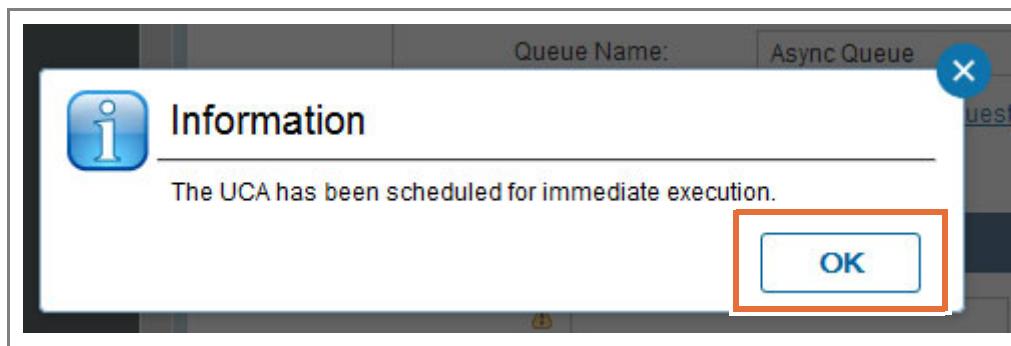
On the right, click **Make Default Track**.



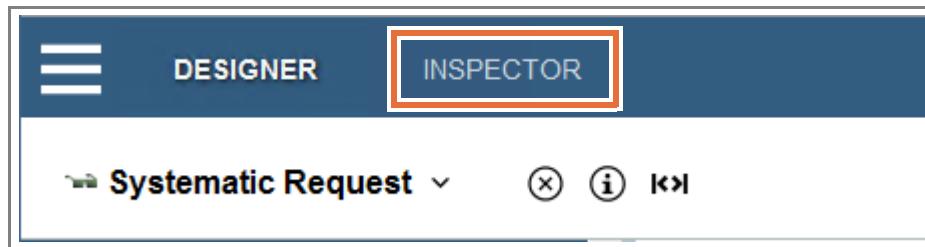
- __ b. Click **Run now** in the Scheduler section.



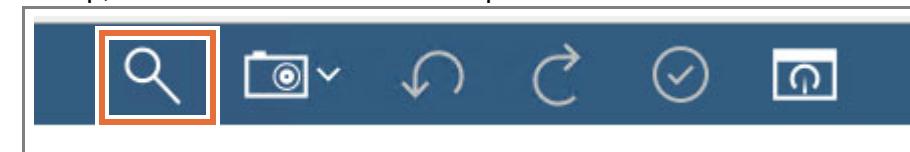
- __ c. A dialog box is displayed, which indicates that the message is scheduled for immediate execution. Click **OK**.



- __ d. Click the **INSPECTOR** view tab.



- __ e. At the top, click the first icon to **search** process instances.



- f. On the left, underneath the search filter criteria, click **Search** to view all the instances of all the processes in the Process Center Server.

The screenshot shows a search interface with the following elements:

- Search:** A title bar with a magnifying glass icon.
- Filter process instances by:**
 - Status:** A section with checkboxes for Active, Completed, Failed, Suspended, and Terminated.
 - Severity type:** A section with a plus sign and the text "Severity type".
- Person:** A field labeled "Name or user name".
- Last modified date:** Two sets of date and time input fields labeled "From" and "To", each with a calendar and clock icon.
- Search:** A large blue button with the word "Search" in white text, which is highlighted with a thick orange border.



Information

If everything was successful, the UCA and Message Start event created a **Hiring Request Process** instance. The instance that you created when you ran the test is the highest numbered instance that is shown in the right frame. This window shows all the instances that ran in the Process Designer and their status, although your environment might not match what you see in this screen capture.

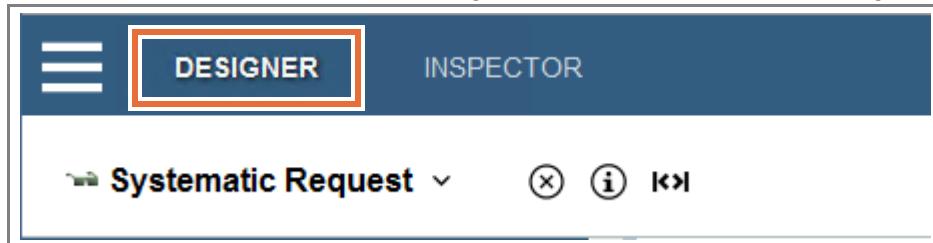
The screenshot shows a user interface for managing process instances. On the left, there is a sidebar with filtering options: 'Status' (Active: 1, Completed: 1, Failed: 0, Suspended: 0, Terminated: 0), 'Severity type' (selected), 'Person' (Name or user name input field), and 'Last modified date' (From and To date/time inputs). A 'Refresh' button is at the bottom of the sidebar. On the right, a main panel displays a list of instances. At the top of this panel are three buttons: 'Select shown instances', 'Select all instances', and 'Clear selection'. Below these buttons, two instances are listed:

- Hiring Request Process:53** (highlighted with a blue arrow) - Last modified [redacted]
- Hiring Request Process:3** (highlighted with a green checkmark) - Last modified [redacted]

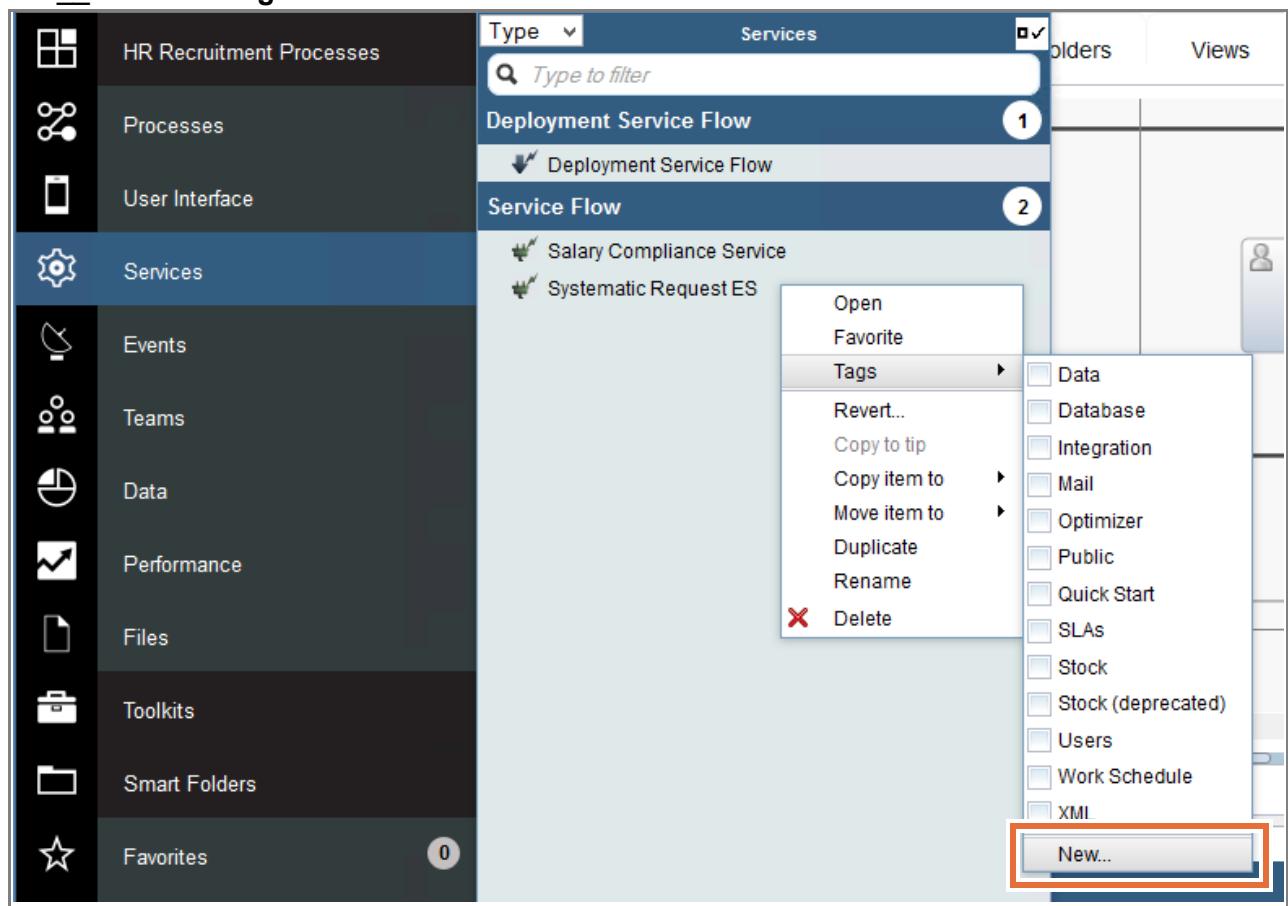
Part 3: Apply asset tags

The Designer library now has a robust number of assets for the **Hiring Request Process** process. You can use asset tagging in IBM Business Process Manager to organize your existing process application assets. Asset tagging allows developers to accomplish tasks such as associating a UCA with its enabling service.

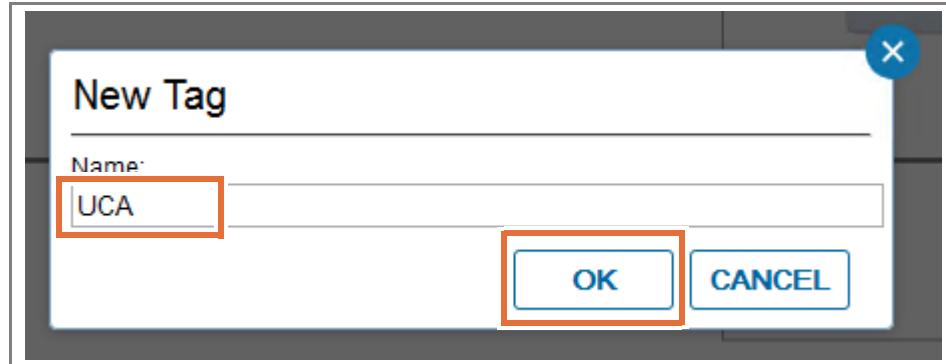
- 1. Tag the **Systematic Request Start** service.
 - a. Click **DESIGNER** to return to the Designer mode of the Process Designer.



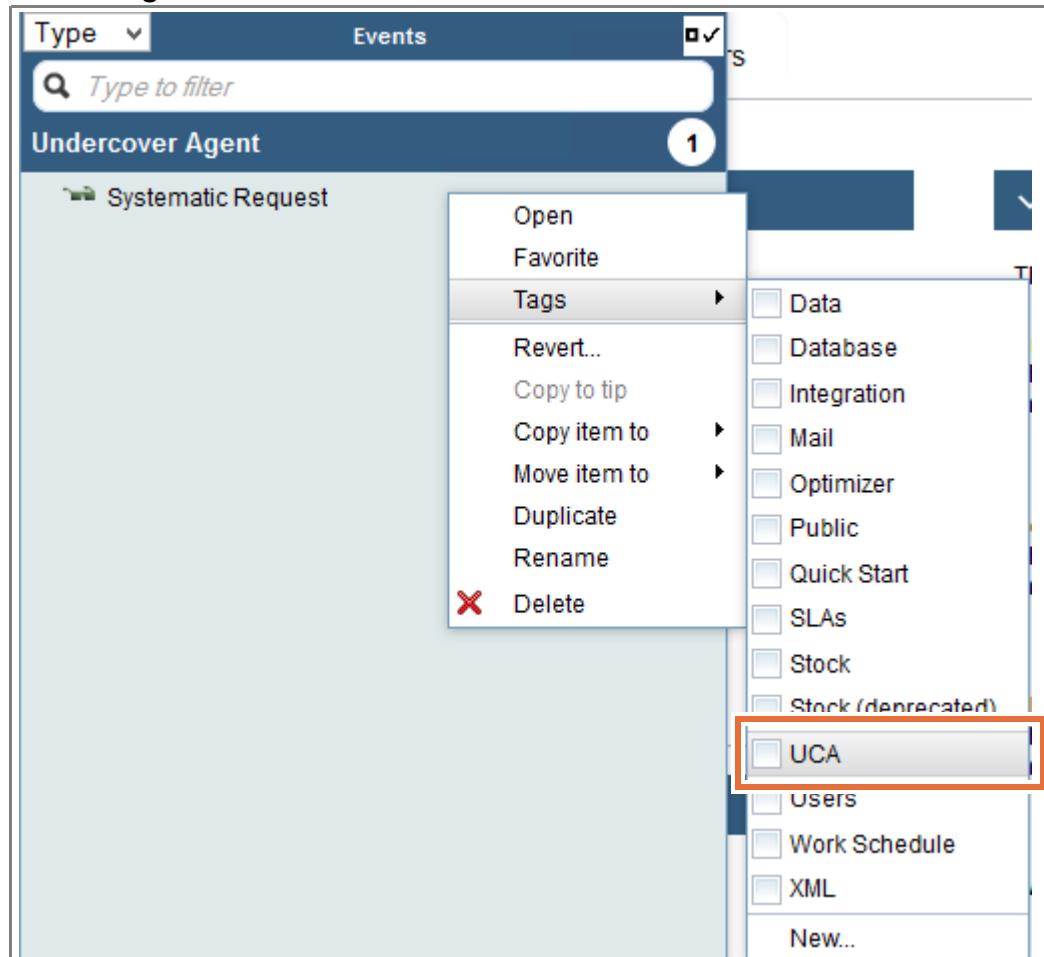
- b. In the Designer library, open the **Services** category.
- c. Right-click the **Systematic Request ES** general system service.
- d. Click **Tags > New**.



- __ e. Name the tag **UCA** and then click **OK**.



- __ f. Return to the library and open the **Events** category.
__ g. Right-click the **Systematic Request UCA**.
__ h. Click **Tags > UCA**.



- i. Click the **HR Recruitment Processes** header at the top. This category lists all the artifacts in the library. The two artifacts are now tagged with UCA in (parenthesis) after the artifact name.

The screenshot shows a library interface with the following categories and their contents:

- Deployment Service Flow** (1 item):
 - Deployment Service Flow
- Service Flow** (2 items):
 - Salary Compliance Service
 - Systematic Request ES (UCA)
- Team** (1 item):
 - General Managers
- Undercover Agent** (1 item):
 - Systematic Request (UCA)
- Theme** (1 item):
 - Hiring Request Theme

- 2. Group by tag.
- a. In the Process Designer library, click the Process Application name **HR Recruitment Processes** to view all the assets in the library. At the top of the list, the default grouping is by Type.

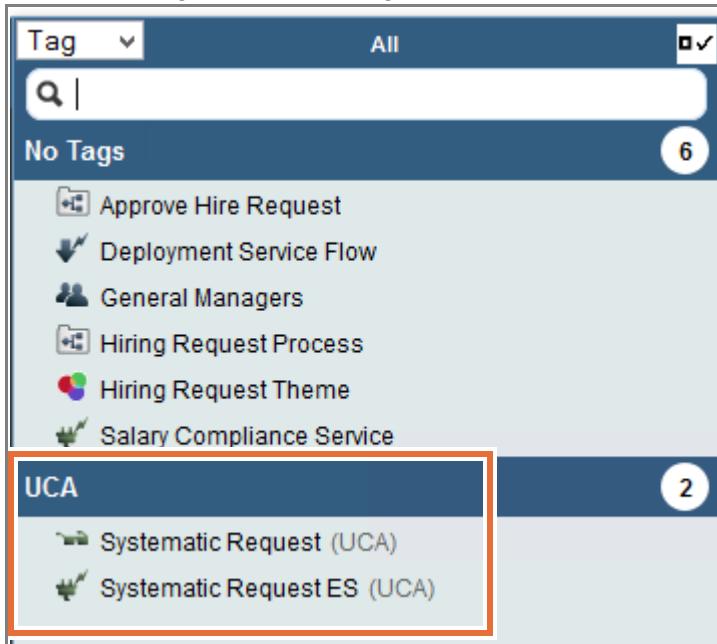
The screenshot shows the Process Designer library with the following structure:

- HR Recruitment Processes** (highlighted with an orange box):
 - Processes**
 - User Interface
 - Services
 - Events
 - Teams
- Type** dropdown: All (checked)
 - Tag**: *to filter*
 - Name**:
 - Type**: Object (highlighted with a blue box) (5 items):
 - CompensationDetails
 - DepartmentDetails
 - HiringRequisition
 - Position
 - RecruitingDetails
- Client-Side Human Service** (1 item):
 - Hiring Form
- Process** (2 items):
 - ...

- ___ b. Click the arrow to the right of **Type**. Select **Tag** from the list.



- ___ c. The artifacts with the tag UCA are now grouped in the list.



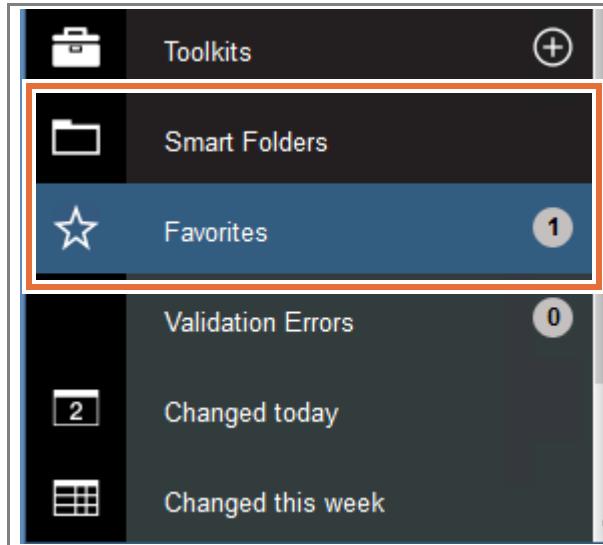
- ___ d. After viewing the **Name** grouping type, return the grouping type back to **Type**.



- ___ 3. Create a favorite.

- ___ a. Click the **Processes** category.
- ___ b. Hover to the left of the **Hiring Request Process**. The outline of a star appears next to the process name.
- ___ c. Click the **star** next to the **Hiring Request Process** to designate it a favorite.

- ___ d. If it is not already expanded, open the **Smart Folders** category at the bottom of the library. Click the **Favorites** category to verify that the process is now part of the **Favorites** smart folder.



Information

You can arrange library items in smart folders for quick and easy access. The Process Designer includes several default smart folders such as the **Changed today** folder, which includes all library items in the current process application that were changed on the current day. The **Changed today** and **Changed this week** smart folders include any changed library items by any users who have access to the current process application.



Optional

You completed part 3 of the 5, so you are now half done with this exercise. Now would be a good time to take a 10-minute break.

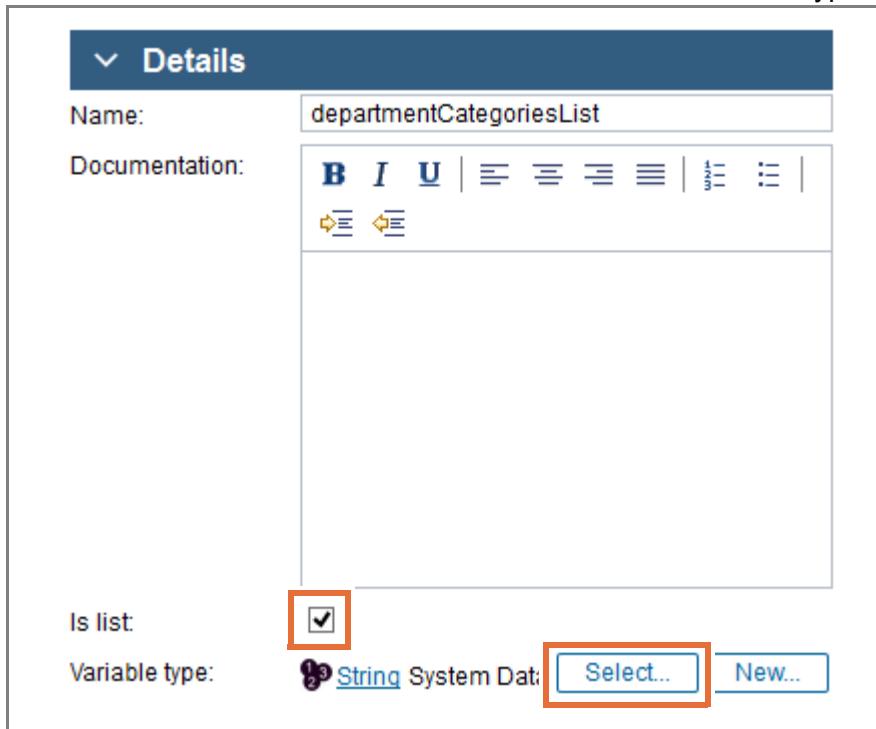
Part 4: Create a service to query a database and populate a list

In this part of the exercise, you create a service to query a database and populate a list. The current systems require employees to look up the different codes and enter them into the system. Because customers use this system, a better approach is to offer a menu with readable options that correspond to the codes in the database. The Job Level and Department Details inputs make up the menus.

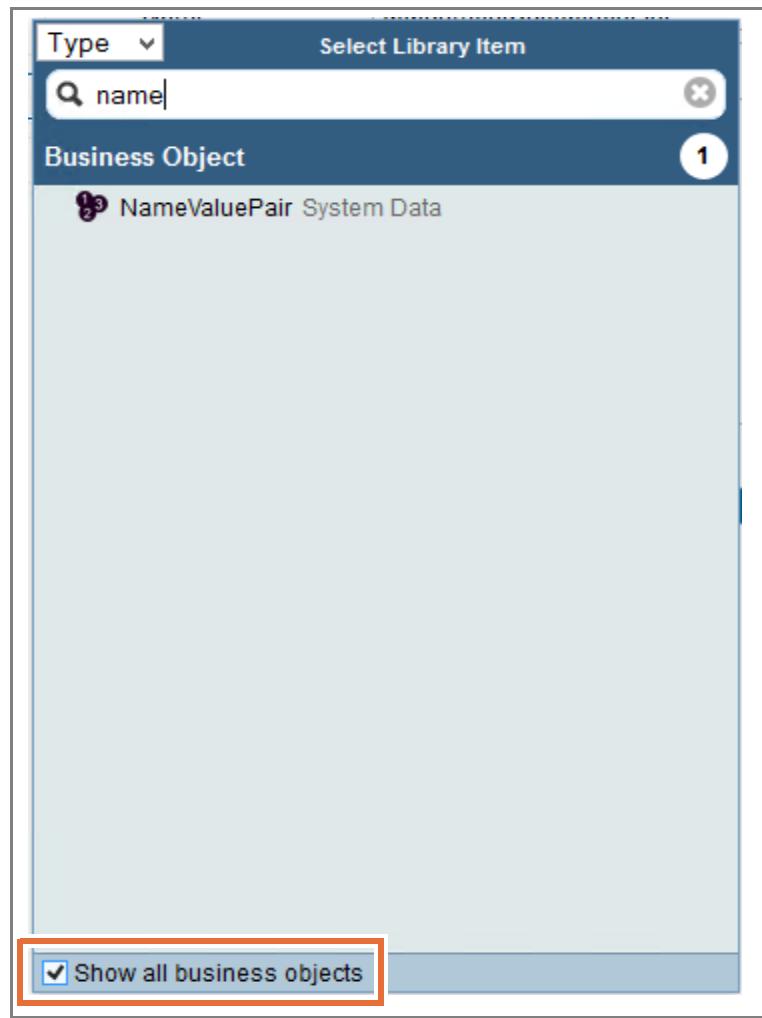
The tables are already established in the TRAINING database as JOBLEVELS. You build a new data object called DepartmentDetails that contains the other fields: Division and Department. These fields match the DIVISIONS and DEPARTMENTS tables in the TRAINING database.

Use an environmental variable to hold database information for one query, and an exposed process variable (EPV) to hold the database information for the other.

- 1. Create a **General System Service** to retrieve the incident categories.
 - a. In the Process Designer library, click the (+) plus sign next to the **Services** category and click the **Service Flow** option.
 - b. Name the service `Retrieve Department Categories` and click **Finish**.
- 2. Add variables to the service. These variables hold the list data that is retrieved from the database.
 - a. Click the **Variables** tab.
 - b. Add an output variable named: `departmentCategoriesList`
 - c. Select the **Is List** check box and then click **Select** for the variable type.



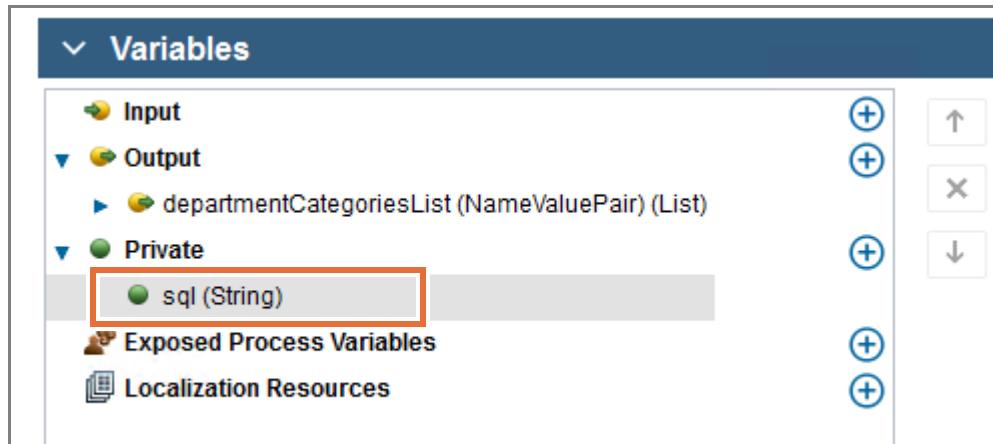
- __ d. Select the **Show all business objects** check box at the bottom, and select **NameValuePair** from the list.



Hint

This variable is used to populate the list of options in a select menu on the coach.

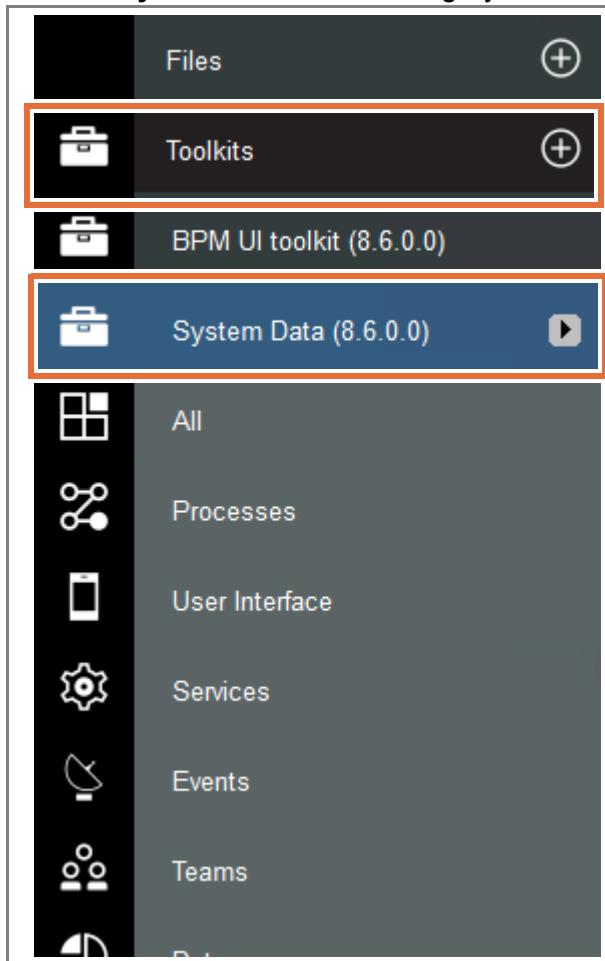
- __ e. Now add a **Private** variable: `sql (String)`



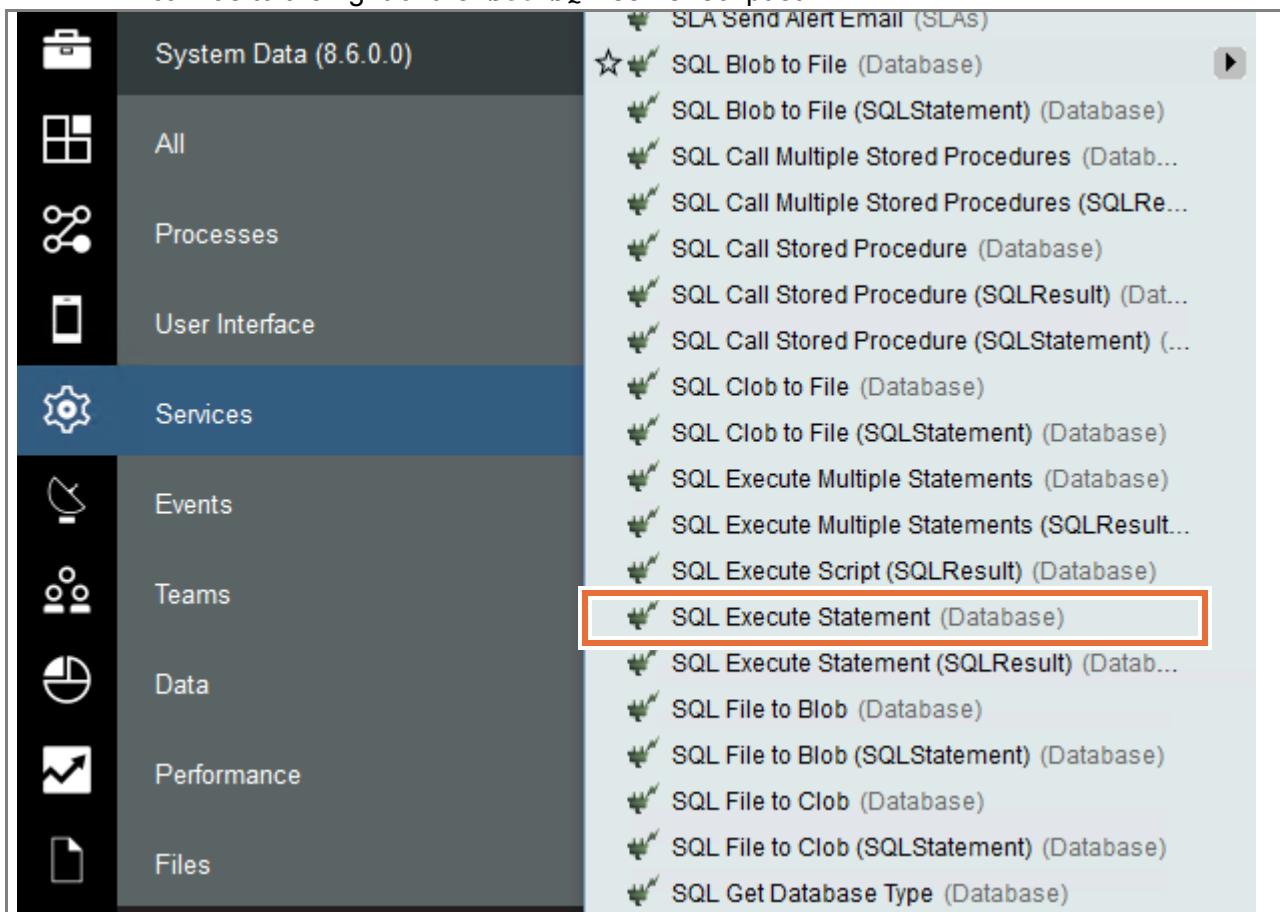
- ___ 3. Add the elements onto the canvas.
- ___ a. Click the **Diagram** tab of the service.
- ___ b. Drag a **Server Script** from the service palette to the canvas.
- ___ c. Rename the server script to: Set SQL



- ___ d. Expand the **Toolkits > System Data** toolkit category in the Process Designer library.



- __ e. Select the **Services** category. Drag the **SQL Execute Statement** service onto the canvas to the right of the `Set SQL server scriptlet`.

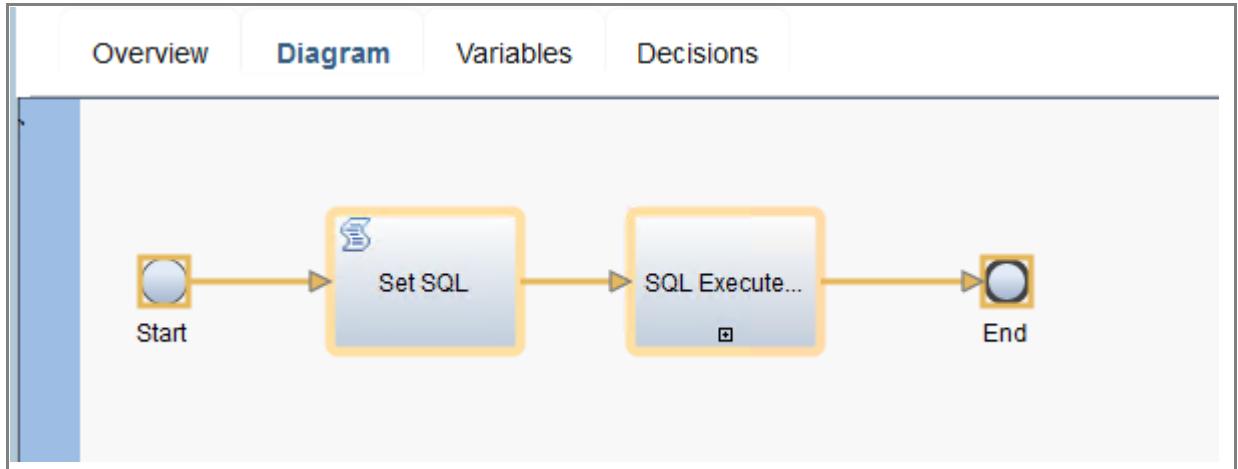


Hint

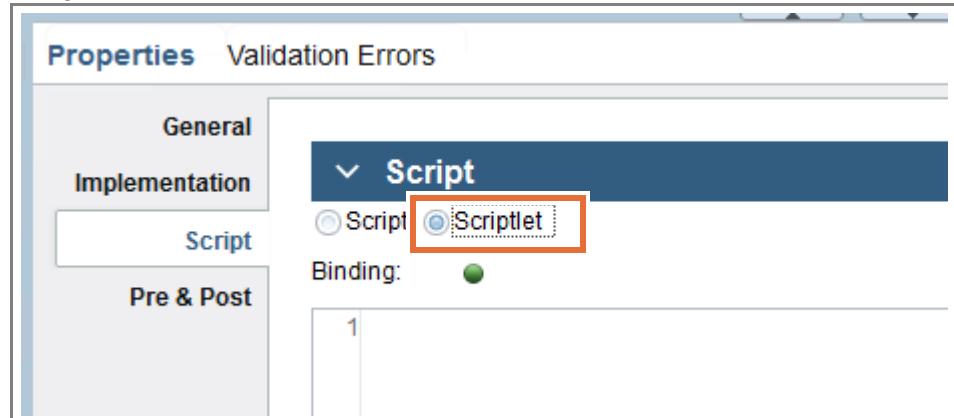
You can find the service that is grouped in with the services that are the Service Flow type, or change to sort by tag and view those items that contain the Database tag. You can also type `SQL Execute` to filter the list to find the service.

If you have problems with dragging the service from the library to the canvas, you can drag a Service Task activity from the palette to the canvas. Then, implement the service by using the SQL Execute service from the System Data toolkit.

- __ f. Connect the objects on the canvas from left to right. Connect the **Start** event to the `Set SQL server scriptlet`. Then, connect the `SQL Execute Statement`. Finally, connect the **End** event.



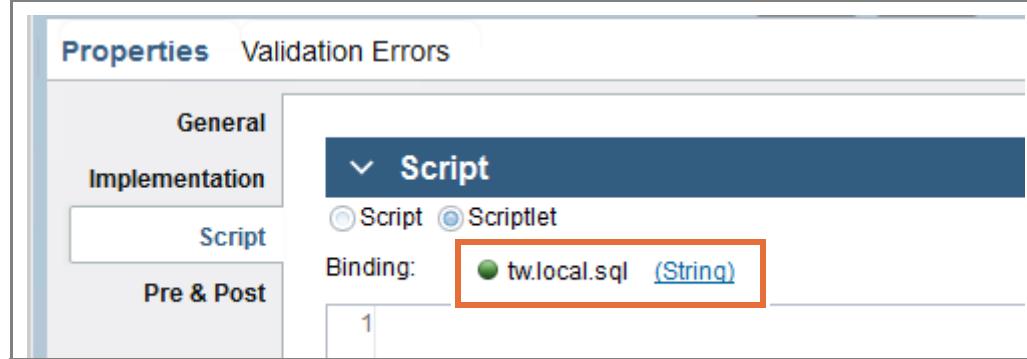
- __ 4. Implement the `Set SQL` server scriptlet.
- __ a. Click the `Set SQL` server script.
 - __ b. Click the **Properties > Script** menu option.
 - __ c. Change the service from **Script** to **Scriptlet**.



Hint

Use a server scriptlet when assigning a value to a single variable.

- __ d. Click **Select** next to **Binding:** and click the **sql (String)** variable.



- __ e. Enter the following SQL query in the script field. The intent is to map the results directly into a **NameValuePair** type object, so you must rename the results to enable that direct mapping.

```
SELECT DEPARTMENTCODE as name, DEPARTMENTNAME as value
FROM TWKS.DEPARTMENTS
```

```
1 SELECT DEPARTMENTCODE as name, DEPARTMENTNAME as value
2 FROM TWKS.DEPARTMENTS
```



Note

To avoid any typographical errors, the script code is provided in the file `Script2.txt` at the location: `C:\labfiles\Exercise_Support_Files\Exercise09\`.

-
- __ 5. Map the inputs and outputs of the **SQL Execute Statement** service.
- __ a. Click the **SQL Execute Statement** service step.
 - __ b. Click the **Properties > Data Mapping** menu option.

- __ c. In the **Input Mapping** section, map the `tw.local.sql` variable to the `sql (String)` variable.

The screenshot shows the 'Properties' dialog with the 'Implementation' tab selected. Under 'Data Mapping', the 'Input Mapping' section is expanded. A red box highlights the first input field, which contains the value 'tw.local.sql'. To the right of this field is a list of available variables with their types: `sql (String)`, `parameters (List o...)`, `maxRows (Integer)`, `returnType (String)`, and `dataSourceName ...`.

- __ d. For the `returnType (String)` variable mapping, type (include the quotation marks):
"NameValuePair"
__ e. A JDBC data source was already created for you. In the **dataSourceName** field, type
"jdbc/TrainingDB" (include the quotation marks).

The screenshot shows the 'Properties' dialog with the 'Implementation' tab selected. Under 'Data Mapping', the 'Input Mapping' section is expanded. The 'tw.local.sql' field now contains 'sql (String)'. The 'returnType (String)' field contains the value 'NameValuePair'. The 'dataSourceName' field contains the value 'jdbc/TrainingDB'. The other fields remain empty or show their original names.

- __ f. In the **Output Mapping**, map the `results (ANY)` variable to the
`tw.local.departmentCategoriesList` variable.

The screenshot shows the 'Properties' dialog with the 'Implementation' tab selected. Under 'Data Mapping', the 'Output Mapping' section is expanded. It displays a mapping from 'results (ANY)' to 'tw.local.departmentCategoriesList'. A note at the top says 'Converted Script. Click here to see original <#>.' Below the mapping, there is a small icon with a double-headed arrow.

- __ g. Save your work.
__ h. Click the **debug** icon to debug the service.
__ i. Make sure that you click the **Diagram** tab so you see the token move between the steps in the service. Expand the **Data** section on the right and click **Step over**.

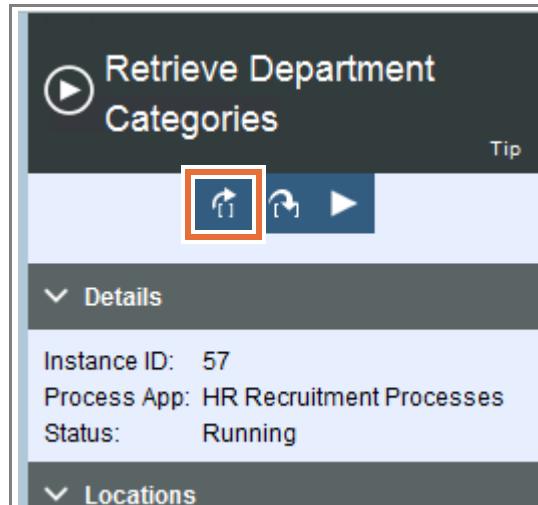
**Note**

You might need to click the Search icon from the top to close the search window.

**Reminder**

The first step assigns the string to the sql variable for the database call.

- ___ j. For the SQL Execute step, click **Step Over** until the status changes to Finished.

**Important**

Two commands in the debugger are for stepping through a nested service: **Step Over** and **Step Into**. When a step in your service is a nested service that contains more than one step, you can use **Step Into** to debug the steps inside the nested service. If you click **Step Over**, the debugger runs all of the steps and moves the token to the end of the nested service, showing only the output after the service is complete.

- k. The `departmentCategoriesList` variable is populated with the values in the database. Notice that the `departmentCategoriesList` variable contains items 0 - 4 as part of the list. This result means that the database returned five rows, which were mapped to the `NameValuePair` object per the configuration options of the **SQL Execute** service.

The screenshot shows the Oracle BPM Studio interface with the process titled "Retrieve Department Categories". The "Data" panel is open, displaying a list of department categories. Item 0 is selected, showing its name ("Marketing") and value ("SELECT DEPARTMENTCODE as nan FROM TWKS.DEPARTMENTS").

```

[-] departmentCategoriesList(NameValuePair)(list)
  [-] item[0]
    name(String) 101
    value(String) Marketing
  [+]- item[1]
  [+]- item[2]
  [+]- item[3]
  [+]- item[4]
  sql(String) SELECT DEPARTMENTCODE as nan
  FROM TWKS.DEPARTMENTS
  < >

```

- l. Click **DESIGNER** to return to the Designer view.
 — 6. Add an exposed process variable (EPV) to manage the data source name.

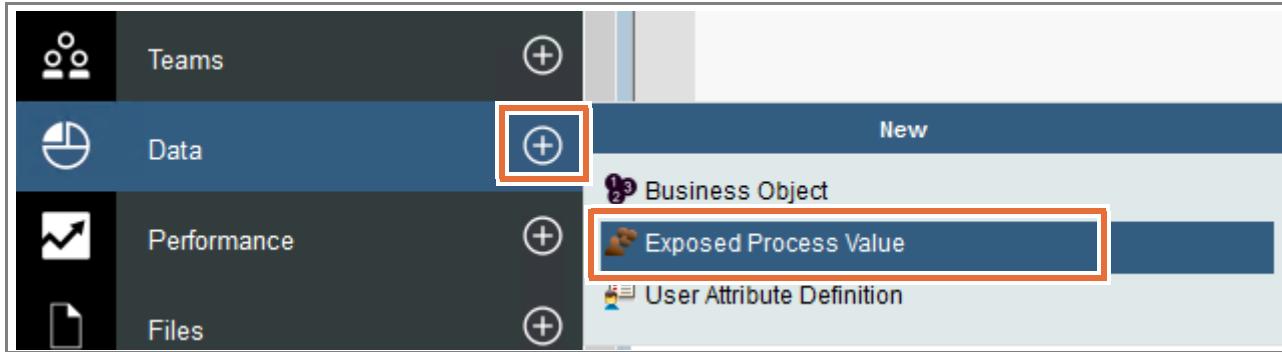


Note

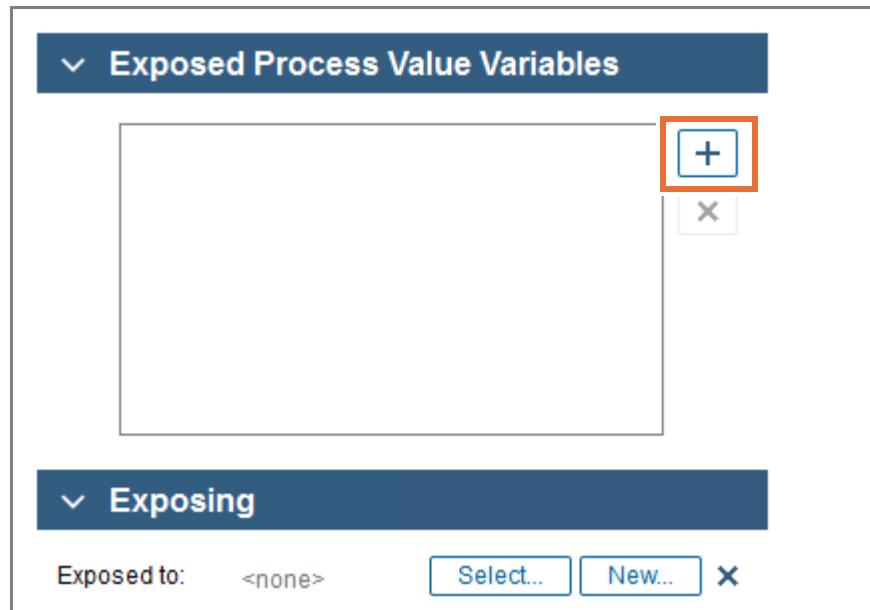
The data source name should normally be stored as an environment variable. EPVs are designed to store business data because they can be exposed to business users so they can set real-time process variables. In the next two steps, you practice the creation and use of both environment variables and EPVs.

- a. Click the (+) plus sign in the **Data** menu option in the Process Designer library.

- __ b. Select **Exposed Process Value**.



- __ c. Name your EPV DataSource and click **Finish**.
__ d. In the **Exposed Process Value** settings, click the (+) plus sign in the **Exposed Process Value Variables** section.



__ e. Set the following values in the **Variable Details** section:

- **External Name:** TrainingDatabase
- **Variable Name:** trainingDB
- **External Description:** This is the name of the JDBC data source for the training database.
- **Default Value:** jdbc/TrainingDB

Variable Details

External name:	TrainingDatabase
Variable name:	trainingDB
External description:	<p>B I U </p> <p>This is the name of the JDBC data source for the training database.</p>
Variable type:	String System Data Select...
Default value:	jdbc/TrainingDB
Use new values:	<input type="checkbox"/>



Note

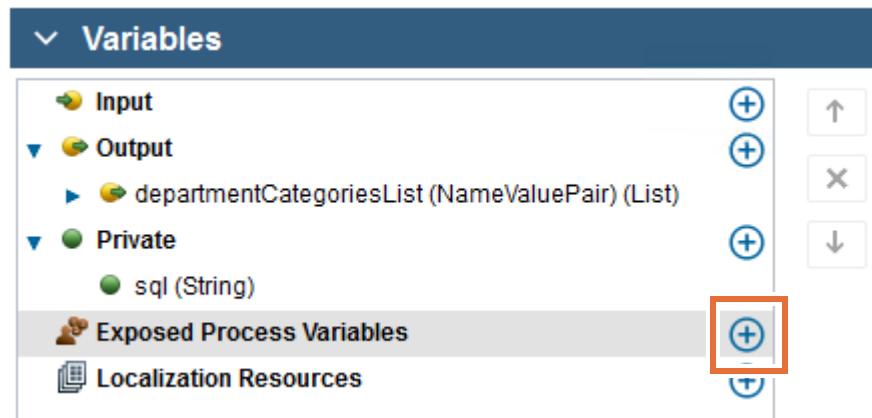
Because the variable type is String, you do not need to enclose the value in quotation marks.

__ f. Save your work.

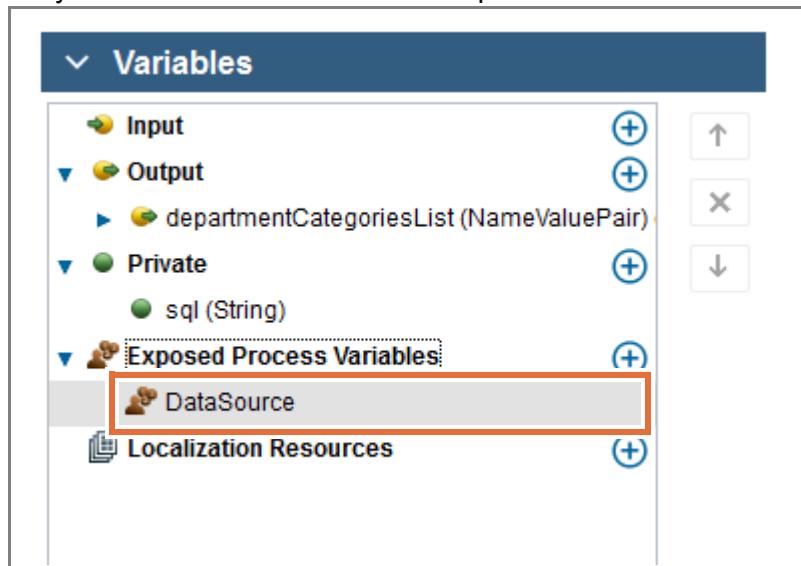
- 7. At the upper left of the designer, open the History menu and return to the **Retrieve Department Categories** service.



- a. Open the **Variables** tab.
- b. Click the (+) plus sign next to the **Exposed Process Variables** category in the Variables list, and then select the **DataSource** EPV.



- c. Verify that you added **DataSource** to the **Exposed Process Variables** variables list.



- __ d. Click the **Diagram** tab to return to the service objects. Click the **SQL Execute Statement** step.
- __ e. Click the **Properties > Data Mapping** menu option.
- __ f. Map the `trainingDB` EPV to the `dataSourceName` variable. Enter as the `dataSourceName: String(tw.epv.DataSource.trainingDB)`



Important

You must cast your EPV to a String. Replace the string that you previously mapped to the variable with `String(tw.epv.DataSource.trainingDB)`. Although you created the EPV as a String type, the system defines it as an EPV type when you map the variable. Therefore, you must cast the variable to String to use it.

The screenshot shows the 'Input Mapping' dialog box. It contains five entries:

- `tw.local.sql` → `sql (String)`
- An empty field → `parameters (List of Object)`
- An empty field → `maxRows (Integer)`
- `"NameValuePair"` → `returnType (String)`
- `String(tw.epv.DataSource.trainingD` → `dataSourceName ...`

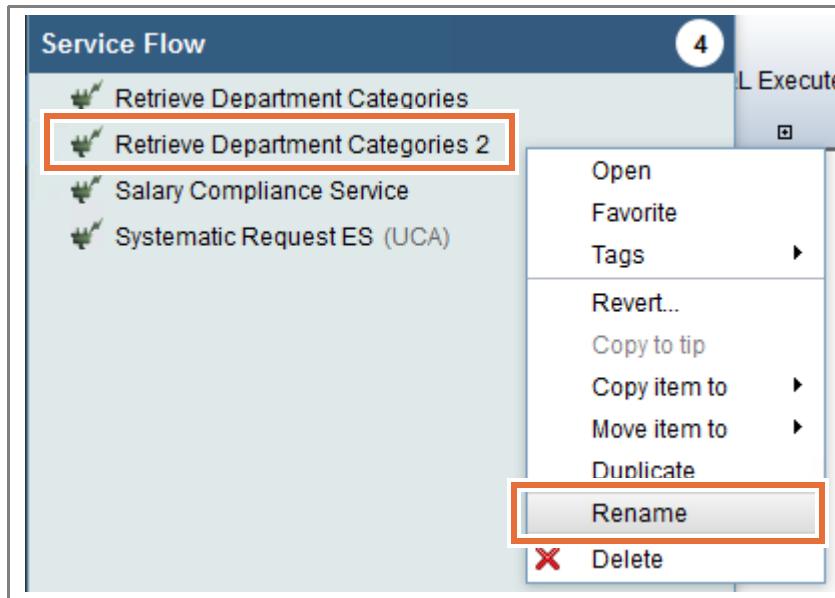
- __ g. Save your work.
- __ 8. Create a **Service Flow** to retrieve job levels; however, use an environment variable (ENV) to accomplish this implementation.
 - __ a. In the Designer library, click the **Services** category. Right-click the **Retrieve Department Categories** service flow and click **Duplicate**.

The screenshot shows the IBM Designer library interface. The left sidebar has categories: User Interface, Services (selected), Events, Teams, Data, and Performance. The main area shows a list of service flows under the 'Service Flow' category:

- Deployment Service Flow
- Retrieve Department Categories (highlighted with a red box)
- Salary Compliance Service
- Systematic Request ES (UCA)

A context menu is open over the 'Retrieve Department Categories' service flow, with the 'Duplicate' option highlighted (also with a red box). Other options in the menu include Open, Favorite, Tags, Revert..., Copy to tip, Copy item to, Move item to, Rename, and Delete.

- ___ b. Right-click the new service and click **Rename**. Name the new service **Retrieve Job Levels**. Click **Finish**.



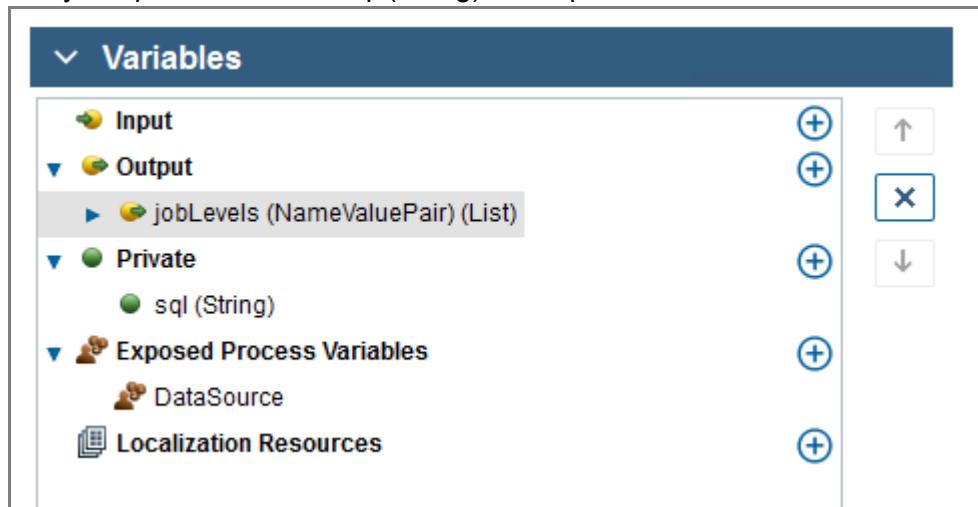
- ___ 9. Change the new service to retrieve job levels.
___ a. Return to the new **Services > Retrieve Job Levels** service flow.



Information

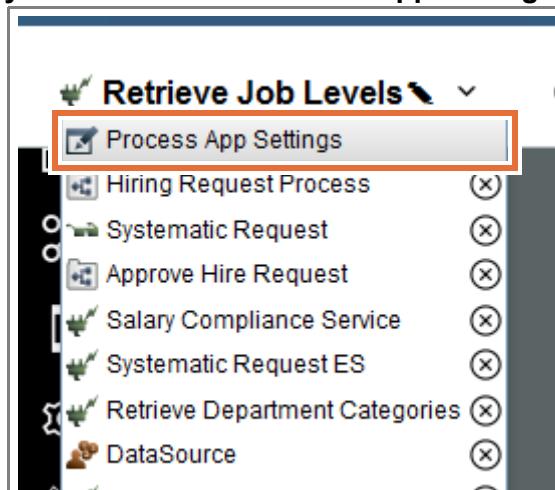
Because many of these database calls are similar, you need to change a few things about this service to retrieve the job levels from the database.

- ___ b. Click the **Variables** tab.
___ c. Click the departmentCategoriesList **Output** variable. Change the variable name to: **jobLevels**.
___ d. Ensure that the **Is List** check box is selected, and the Variable Type is **NameValuePair**.
___ e. Verify the private variable **sql (String)** is still present in the variables list.

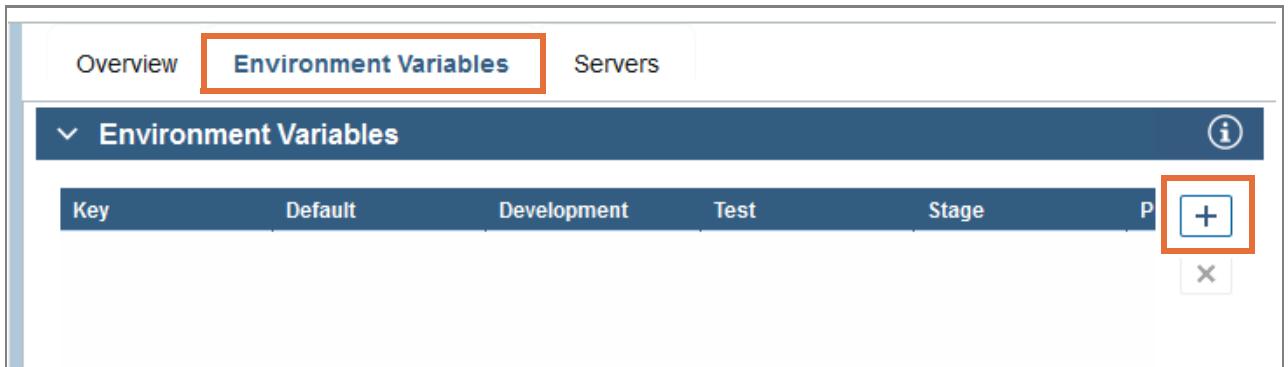


__ 10. Add an environment variable to your process application.

__ a. Open the **History** menu and select **Process App Settings**.



__ b. Click the **Environment Variables** tab and then click the (+) plus sign on the right side of the **Environment Variables** section.



__ c. Complete the following values for the environment variable:

- **Key:** TrainingDB
- **Default:** jdbc/TrainingDB

Key	Default	Development	Test	Stage
TrainingDB	jdbc/TrainingDB			

__ d. Save your work.

__ 11. Implement the `Set SQL` server scriptlet.

__ a. Return to the **Retrieve Job Levels** Service Flow.

__ b. Click the **Diagram** tab.

- __ c. Click the Set SQL server script.
- __ d. Click the **Properties > Script** menu option.
- __ e. Verify that the **Scriptlet** type is selected and the scriptlet is bound to the tw.local.sql (String) variable.
- __ f. Change the **Implementation** script field to:

```
SELECT JOBLEVELCODE as name, JOBLEVELNAME as value
FROM TWKS.JOBLEVELS
```

The screenshot shows the 'Script' configuration dialog. The 'Scriptlet' type is selected. The binding is set to 'tw.local.sql (String)'. The script code is:

```
1 SELECT JOBLEVELCODE as name, JOBLEVELNAME as value
2 FROM TWKS.JOBLEVELS
```



Note

To avoid any typographical errors, the script code is provided in the file `Script3.txt` at the location: `C:\labfiles\Exercise_Support_Files\Exercise09\`.

- __ g. Save your work.
12. Map the inputs and outputs of the SQL Execute Statement service.
- __ a. Click the **SQL Execute Statement** subservice on the canvas.
 - __ b. Click the **Properties > Data Mapping** menu option.
 - __ c. Under the **Input Mapping** section, verify the `sql (String)` input is mapped to the `tw.local.sql (String)` variable.
 - __ d. Verify the `returnType (String)` input is "NameValuePair" (include the quotation marks).
 - __ e. For the `dataSourceName (String)` input, enter: `tw.env.TrainingDB`

The screenshot shows the 'Input Mapping' configuration dialog. The 'Data Mapping' tab is selected. The mapping details are:

Input	Type	Mapping
<code>sql (String)</code>	<code>tw.local.sql</code>	<code>sql (String)</code>
<code>returnType (String)</code>	<code>"NameValuePair"</code>	<code>returnType (String)</code>
<code>dataSourceName (String)</code>	<code>tw.env.TrainingDB</code>	<code>dataSourceName (String)</code>

**Note**

You are not required to cast an ENV to a String.

- ___ f. Change the **Output Mapping** section to the `tw.local.jobLevels` variable.

The screenshot shows the 'Output Mapping' section of a configuration interface. It consists of two main parts: a left panel labeled 'results (ANY)' and a right panel labeled 'tw.local.jobLevels'. A blue rectangular box highlights the 'tw.local.jobLevels' input field. Above the panels are two icons: a left-pointing arrow and a right-pointing arrow, indicating a bidirectional mapping relationship.

**Optional**

Now you can remove the DataSource Exposed Process Variable from the Variables tab that was added to the original service and duplicated for this service.

The screenshot shows the 'Variables' tab in a configuration interface. It lists several variables under categories: Input, Output, Private, and Exposed Process Variables. Under 'Exposed Process Variables', the 'DataSource' item is highlighted with a red rectangular box. To its right, there is a column of icons: a plus sign, an upward arrow, a delete button (highlighted with a red box), a downward arrow, and another plus sign. The 'DataSource' item is also highlighted with a red box.

- ___ g. Save your work.
 ___ 13. Debug the service to determine whether the data is retrieved from the database correctly.
 ___ a. Click the **debug** icon above the palette.

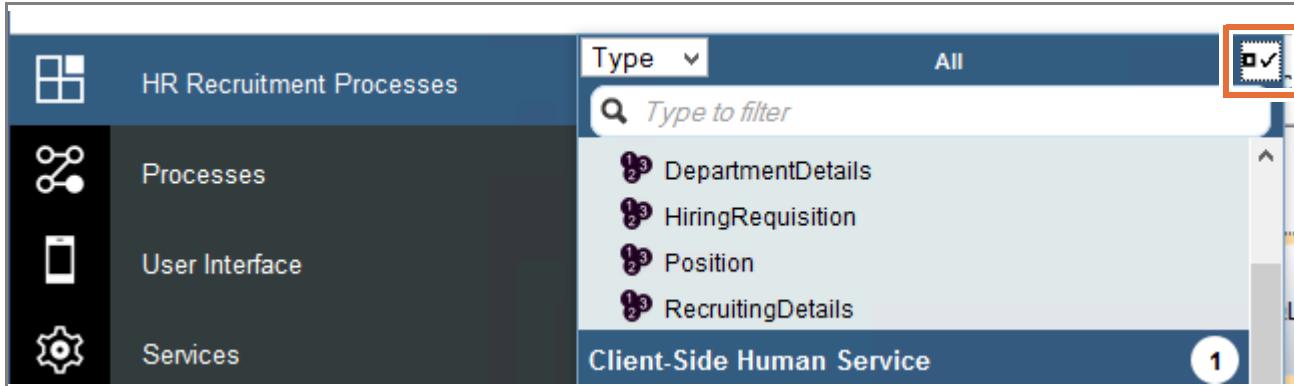
- ___ b. Step through the service and verify that the database call executes and returns the data from the database.

The screenshot shows the Oracle Service Catalog interface with the following details:

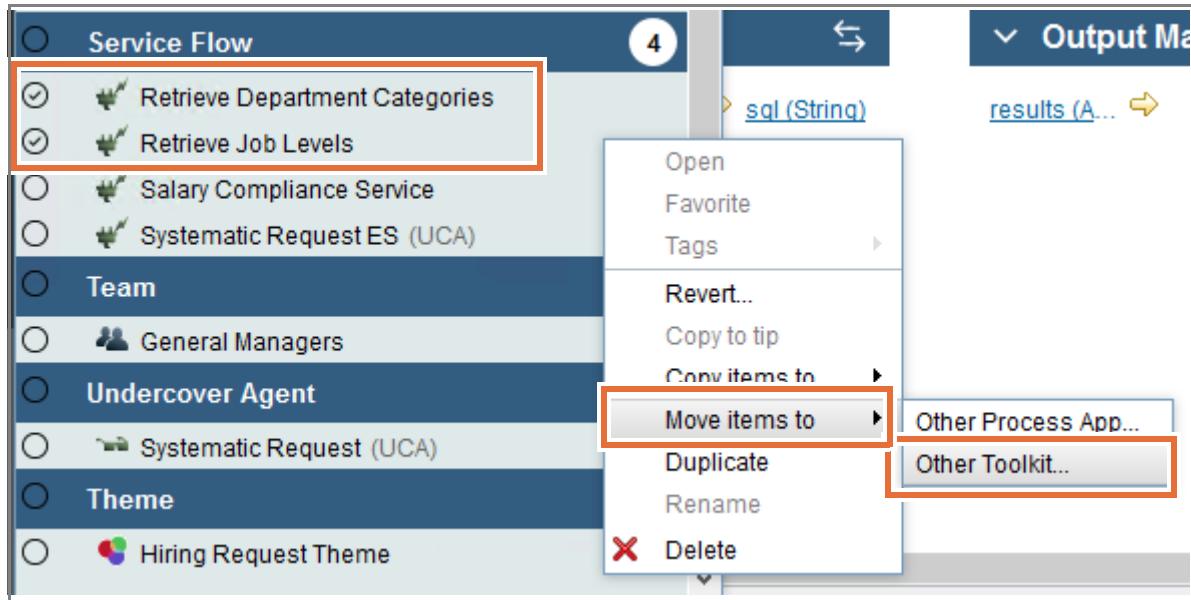
- Title:** Retrieve Job Levels
- Tip:** No actions were found.
- Details:** Instance ID: 58, Process App: Hiring Requisition Toolkit, Status: Finished
- Locations:** No locations were found.
- Data:**
 - jobLevels(NameValuePair)(list)
 - item[0]
 - name(String) 5001
 - value(String) Jr Associate
 - item[1]
 - item[2]
 - item[3]
 - item[4]
 - item[5]
 - item[6]
 - sql(String) SELECT JOBLEVELCODE as name, JOBLEV FROM TWKS.JOBLEVELS

- ___ c. Click **DESIGNER** to return to the Designer view.
- ___ 14. To make these services available to other process applications, move the services to the toolkit.
- ___ a. At the top of the library, click **HR Recruitment Processes** to view all the assets in your library.

- ___ b. On the top of the asset list, click the icon to change to **multi-selection mode**.

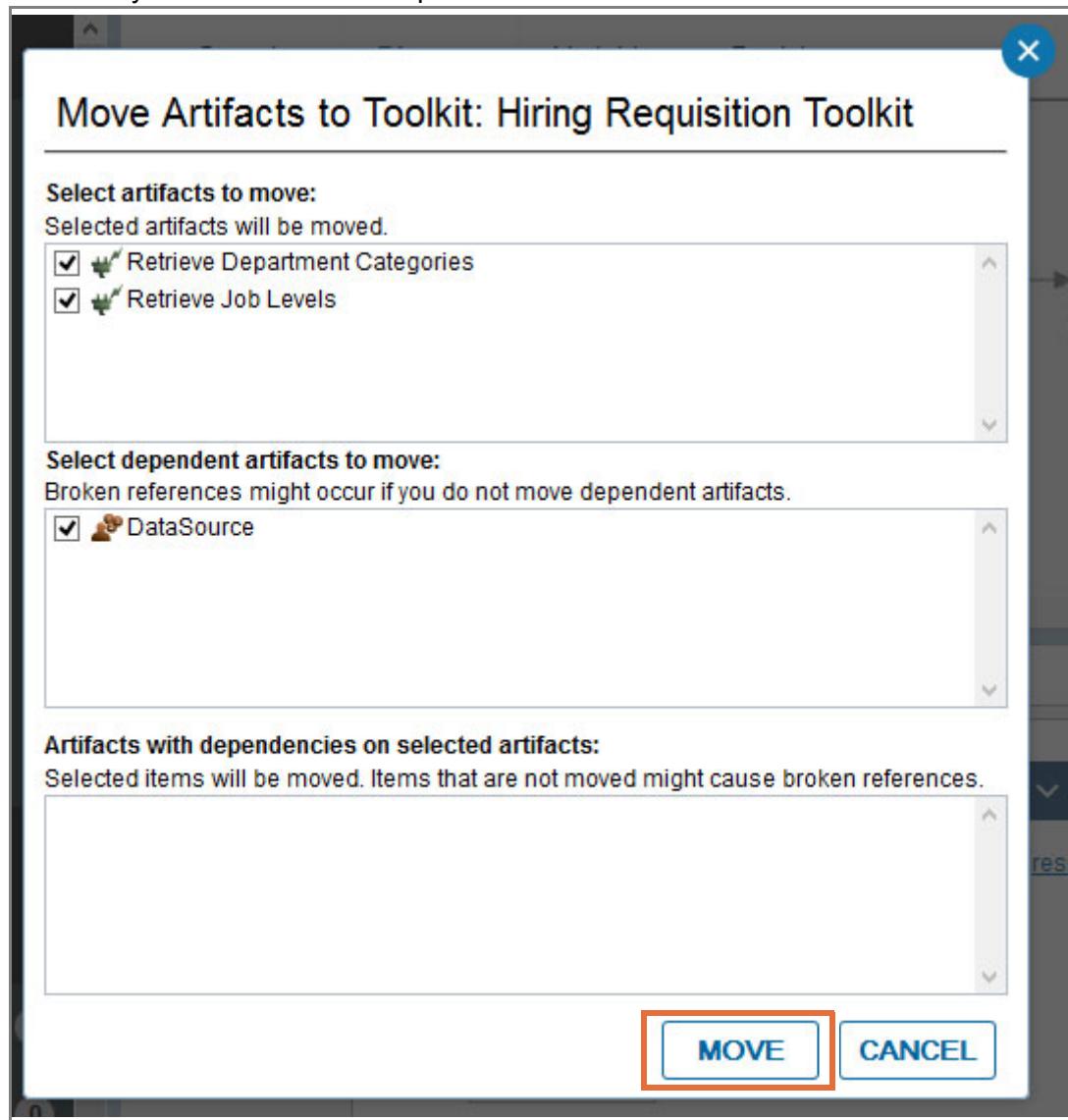


- ___ c. Radio buttons appear to the left of every asset in the library. Select both the **Retrieve Department Categories** and **Retrieve Job Levels** Service Flow services. Right-click and click **Move items to > Other Toolkit**.

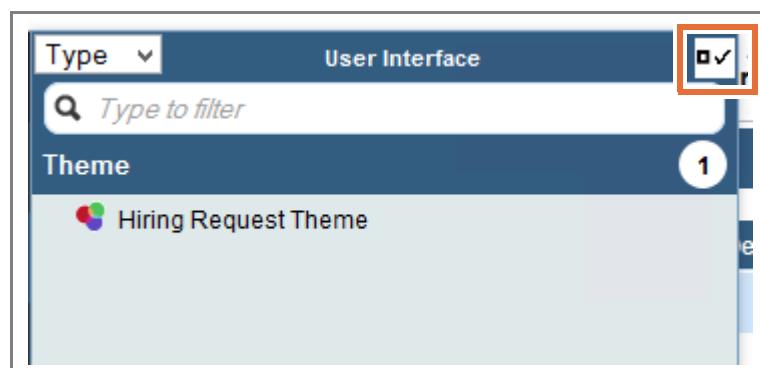


- ___ d. Click **Hiring Requisition Toolkit**.

- e. The system identifies the dependencies. Click **Move**.

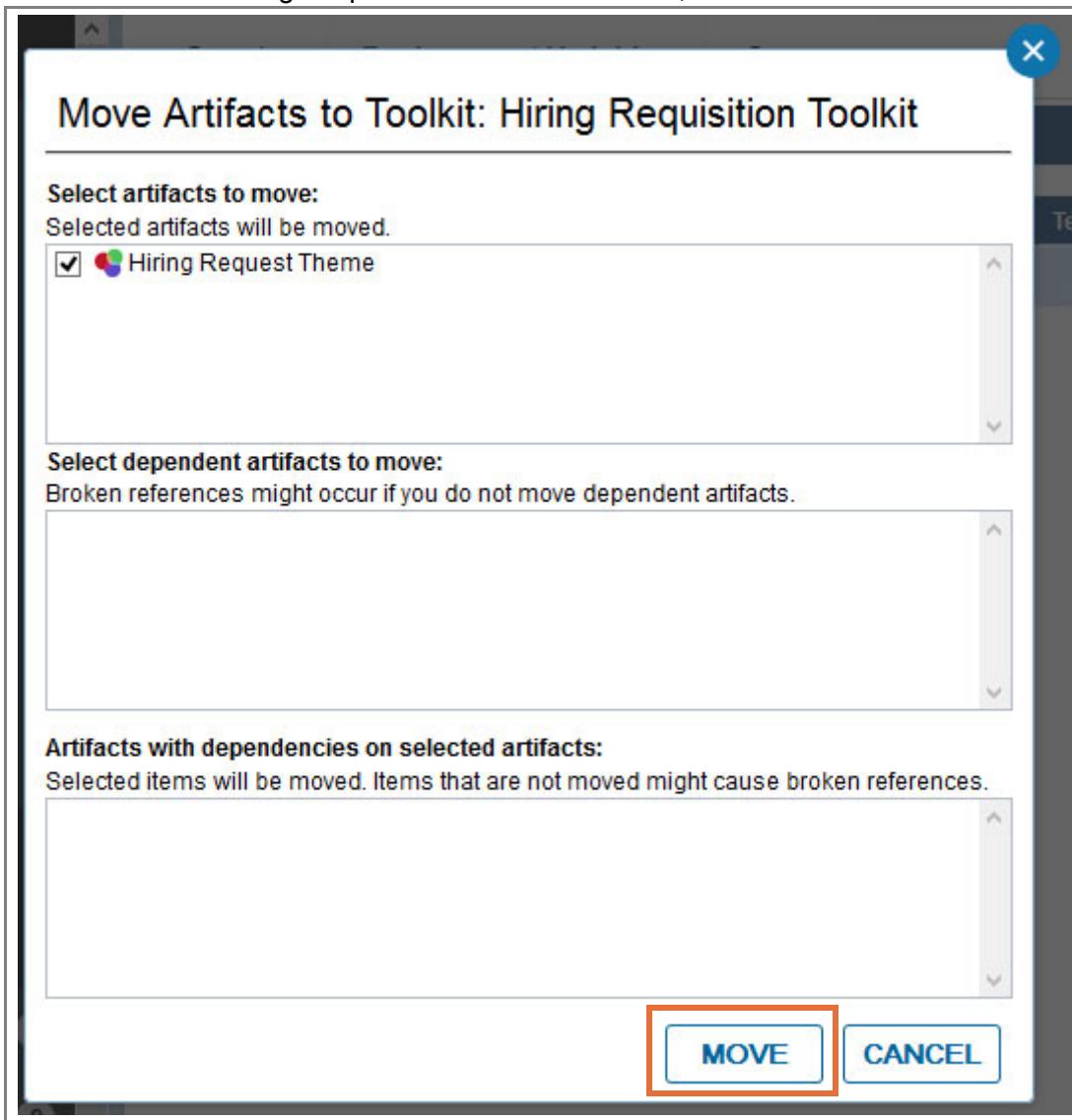


- 15. In the last exercise, you created a Hiring Request Theme. To make this theme available to other process applications, move the theme to the Hiring Requisition Toolkit.
- a. Open the **User Interface** category in the library. Turn the **multi-select** off for this next move.

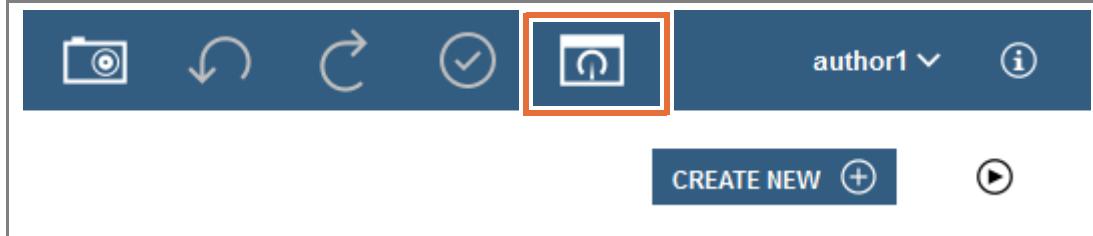


- b. Right-click **User Interface > Hiring Request Theme** in the library. Click **Move item to > Other Toolkit**, and select the **Hiring Requisition Toolkit**.

- c. To move the Hiring Request Theme to the Toolkit, click **Move**.



- d. Finally, because the ENV variables are specific to the toolkit or Process App, you need to re-create the environment variable for this toolkit. Click the **Process Center** link.



- e. Click the **Toolkits** tab.

- __ f. Click **Open in Designer** for the Hiring Requisition Toolkit.

The screenshot shows the SAP BAPI interface with the 'Toolkits' tab selected. Below it, three toolkit entries are listed:

- Hiring Requisition Toolkit (HRT)**: Last updated on [redacted] by author1. An 'Open in Designer' button is highlighted with a red box.
- BPM UI toolkit (SYSPBMUI)**: Last updated on [redacted] by author1. An 'Open in Designer' button is present but not highlighted.
- SAP Guided Workflow (deprecated) (SGW)**: Last updated on [redacted] by author1. An 'Open in Designer' button is present but not highlighted.

- __ g. At the top, click the **Environment Variables** tab.
 __ h. In the Environment Variables section, add the TrainingDB key with the default value of jdbc/TrainingDB. This setting is needed to run the Retrieve Job Levels service flow.

The screenshot shows the 'Environment Variables' tab selected. A table is displayed with the following data:

Key	Default	Development	Test	Staging
TrainingDB	jdbc/TrainingDB			



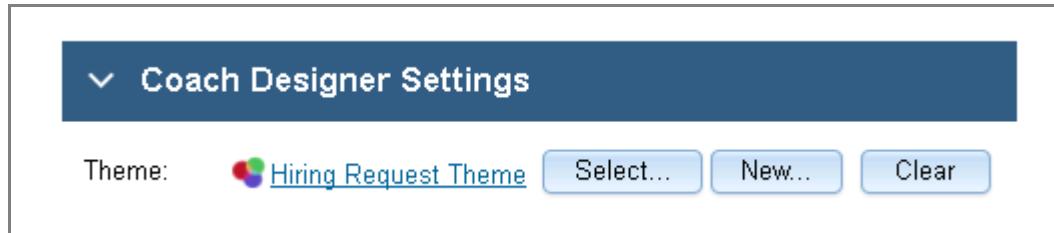
Hint

Click the (+) Add icon from the right to add a variable.

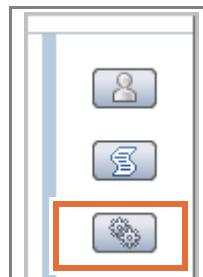
Part 5: Change an input to a single select on a coach

Until now, any input would be accepted for the Department input on the page. If only certain values are allowed, you must then approach the control in one of the following ways. The first way would be to limit the input to only acceptable values by using a selection like the single select on a coach. The second way is through validation of the input data. Many times the solution is both of these approaches. This section demonstrates how to create a single selection to limit the values that are allowed.

- ___ 1. Apply the Hiring Request Theme to the Hiring Requisition Toolkit.
 - ___ a. In the **Overview > Coach Designer Settings** section, click **Select** to select the Hiring Request Theme.



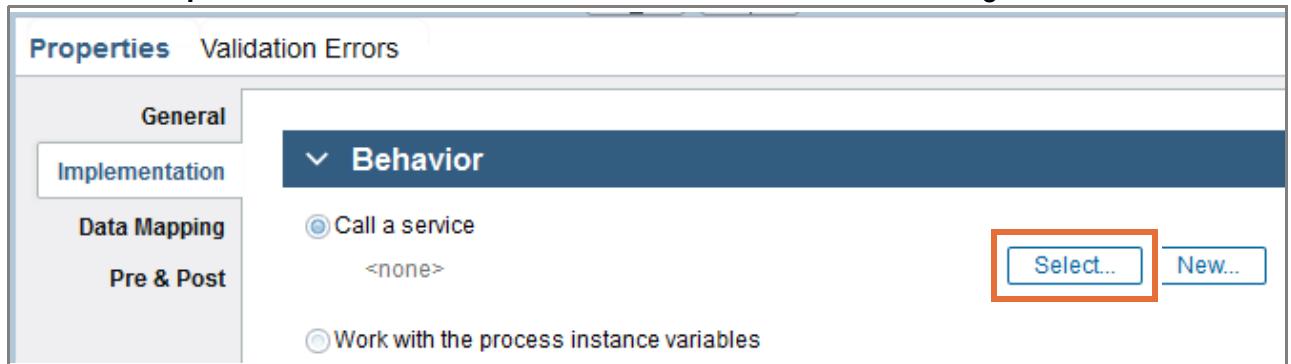
- ___ b. Save your changes.
- ___ 2. Add the **Retrieve Department Categories** service to the **Hiring Form** human service.
 - ___ a. Select **User Interface** from the library and click **Hiring Form**.
 - ___ b. The **Hiring Form** service opens. In the **Diagram** tab, drag a **Service** from the palette onto the **Hiring Form** canvas between the **Start** event and the **Hiring Form** coach.



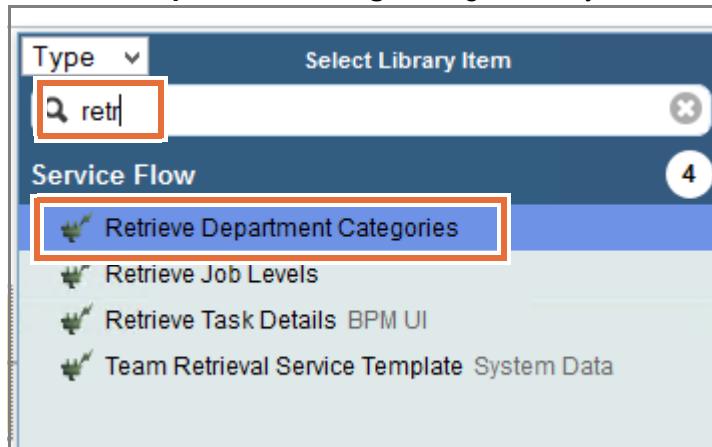
- ___ c. Reconnect the **Hiring Form** flow so that the new service is part of the flow.
- ___ d. Rename the new service step to: **Retrieve Department Categories**



- __ e. Select the **Retrieve Department Categories** service. In the **Properties > Implementation** menu, click **Select** for the **Call a service** setting.



- __ f. Select the **Retrieve Department Categories** general system service that you created.



- __ g. Save your work.

- 3. Map the output variable of the integration service and allow the system to create the private variable.



Note

In the **Properties > Data Mapping > Output Mapping** section, you see that the service requires you to map a variable to the `departmentCategoriesList` variable.

Properties Validation Errors

- General
- Implementation
- Data Mapping**
- Pre & Post

▼ Output Mapping

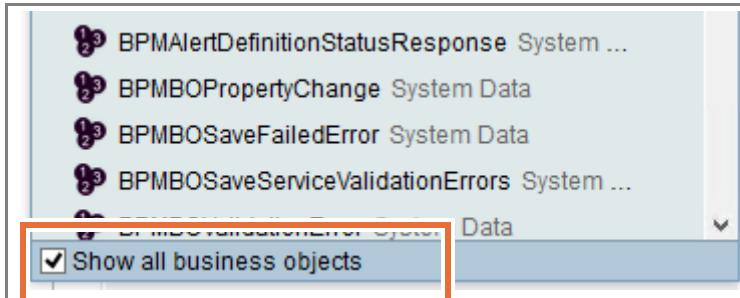
[departmentCategoriesList \(List o...\)](#) ➔

- a. Click the **Variables** tab. Create a private variable: `departmentCategoriesList (NameValuePair) (List)`

Variables	
Input	(+)
↳ requisitionDetails (HiringRequisition)	(+)
Output	(+)
↳ requisitionDetails (HiringRequisition)	(+)
↳ isNewPosition (String)	(+)
Private	(+)
↳ departmentCategoriesList (NameValuePair) (List)	(+)
Exposed Process variables	(+)
Environment Variables	(+)
Localization Resources	(+)

**Reminder**

Select **Show all business objects** to choose the `NameValuePair` variable type.



- ___ b. Return to the **Diagram** tab. Verify that the **Retrieve Department Categories** step on the canvas is selected. In the **Properties > Data Mapping** section, map the `departmentCategoriesList` variable to the output `tw.local.departmentCategoriesList`.
- ___ c. Save your work.
- ___ 4. Change the control on the **Hiring Form** coach to a single select.
- ___ a. Click the **Coaches** tab and the **Hiring Form** coach.

**Note**

Initially when you view the coach, the server must generate the CSS and render the new coach on the screen, so a delay might occur before you see the effect.

- ___ b. From the **Department Details** tab, select the control **Department** that is bound to the department variable.

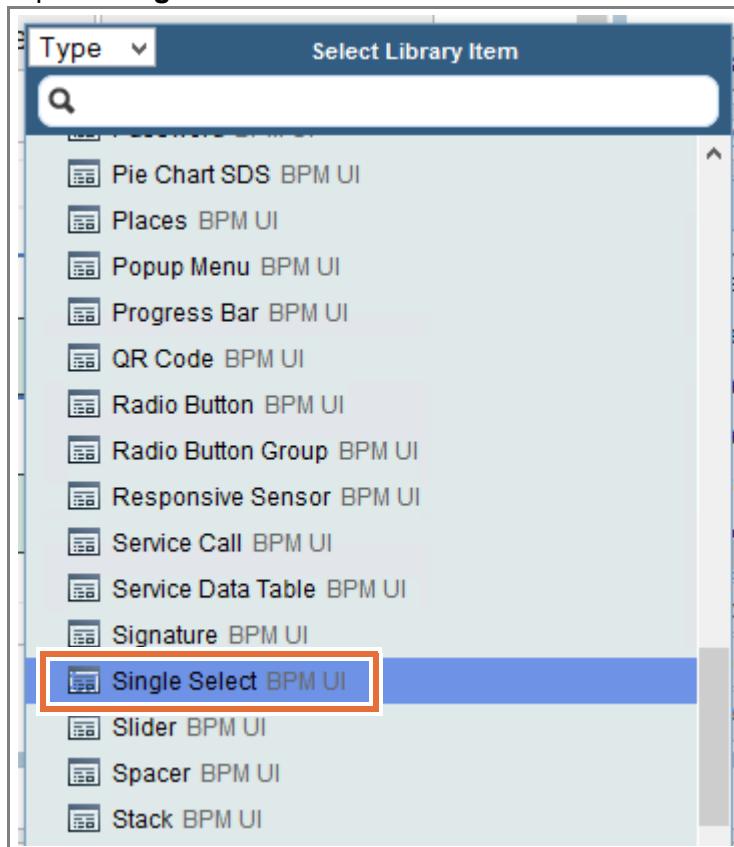
The screenshot shows a horizontal navigation bar with several tabs: Requisition Details, Position Details, Recruiting Details, Compensation Details, Department Details, and a plus sign icon. The Department Details tab is currently selected. Below the tabs, there are two input fields: 'Department' and 'Division', each with a light green background. A red rectangular box highlights the 'Department' input field. At the bottom of the screen is a large grey button labeled 'Submit'.

Change the view that the control is bound to. Right now, all the input boxes on the coach are text views. These controls are all variables that are bound to String data types.

- ___ c. In the **Properties > General > Behavior** section, click **Select** next to View.

The screenshot shows the 'Behavior' section of the properties panel. It includes fields for 'Binding' (radio button selected for 'requisitionDetails.departmentDetails.department'), 'View' (set to 'Text' with '(String)' type), and 'Label visibility' (set to 'Show'). The 'Select...' button in the 'View' row is highlighted with a red box.

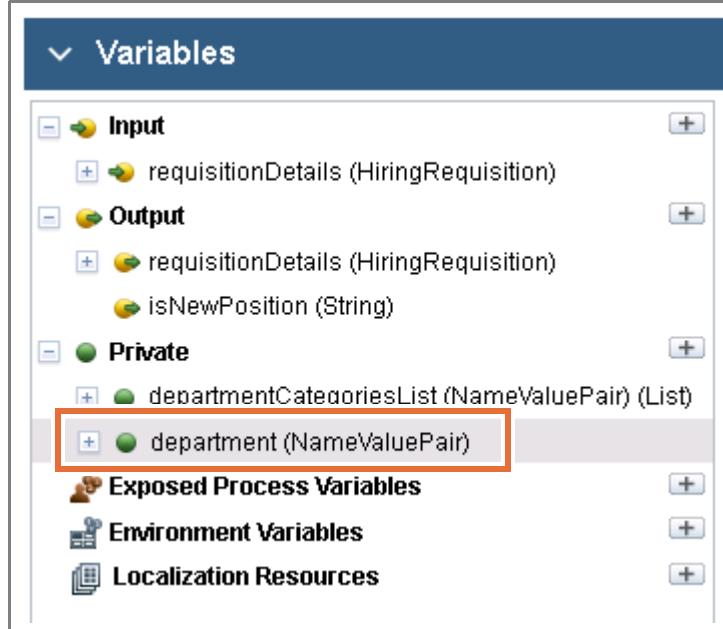
- __ d. Select the option **Single Select** coach view.



- __ e. After you change the view binding, the control on the palette has a different look. It is now a **Single Select** type of view control.

A screenshot of a user interface. At the top, there are several tabs: Requisition Details, Position Details, Recruiting Details, Compensation Details, Department Details, and a plus sign button. Below these tabs is a section labeled "Department" containing a single-line input field with a dropdown arrow at the end. This "Department" section is highlighted with a red rectangular border. Below it is another section labeled "DIVISION" containing a single-line input field. At the bottom of the screen is a large grey "Submit" button.

- ___ f. Save your work.
- ___ 5. The select control returns the single element that is chosen from the list, and you must create a simple variable of the list type to store the selection.
 - ___ a. Click the **Variables** tab of the **Hiring Form** service.
 - ___ b. Create a private variable: **department** (**NameValuePair**)



- ___ c. Save your work.
- ___ 6. Store the selected department to the department variable.
 - ___ a. Return to the **Coaches** tab.
 - ___ b. Select the **Department** field under the **Department Details** tab.
 - ___ c. In the **Properties > General > Behavior** section, click **Select** next to the **Binding** property.
 - ___ d. Click the new Private variable **department (NameValuePair)** from the variable list.



- ___ 7. Configure the setting that determines the list of selections available to the user on the **Department** select control.
 - ___ a. Click the **Properties > Configuration** menu. Expand the **Items** section.

- ___ b. Change the Item lookup mode to **Items from Config Option**. Click **Select** next to the Item input data setting and click the **departmentCategoriesList (NameValuePair)** variable.



Note

Because the control uses the `.name` variable for the **Display Name Property** and the `.value` of the **Value** property settings by default, you do not have to set those settings. If you are using a variable other than a `NameValuePair`, you must assign the values that populate the name and value of the control.

- ___ c. Save your work.
- ___ 8. The Department selection that is stored in the private `department (NameValuePair)` variable must be mapped to the pipeline variable business object. Bind the selection value back to the variable.
- ___ a. In the **Hiring Form** service, click the **Diagram** tab.
 - ___ b. Change the name of the **Map New Position Var** script to: `Map Vars`
 - ___ c. In the **Properties > Script** menu, add the following implementation code to the bottom of the script:

```
tw.local.requisitionDetails.departmentDetails.department =
tw.local.department.name;
```

```
1 // This JavaScript will run within the browser and must use the client-side JavaScript syntax
2 // - To instantiate a complex object:           - To instantiate a list:
3 //   tw.local.customer = {};
4 //   tw.local.addresses = [];
5 if (tw.local.requisitionDetails.recruitingDetails newPosition == true)
6 tw.local.isNewPosition = "1";
7 else
8 tw.local.isNewPosition = "0"
9 tw.local.requisitionDetails.departmentDetails.department = tw.local.department.name;
```

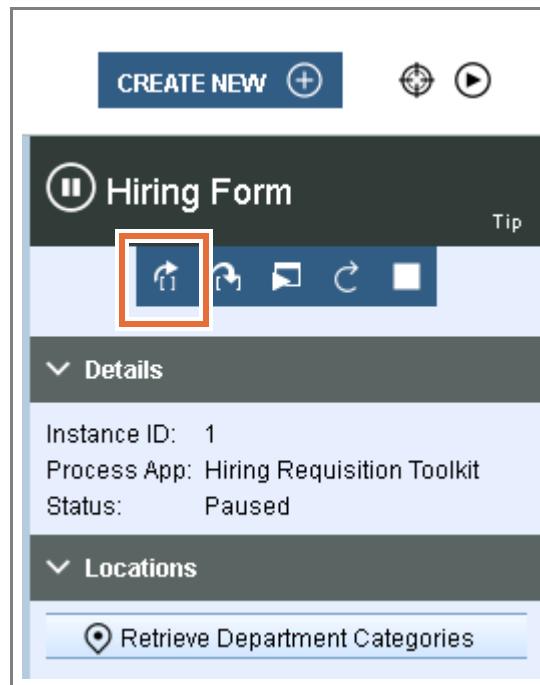
**Note**

The script code is provided in the file `Script4.txt` at the location:
`C:\labfiles\Exercise_Support_Files\Exercise09\`.

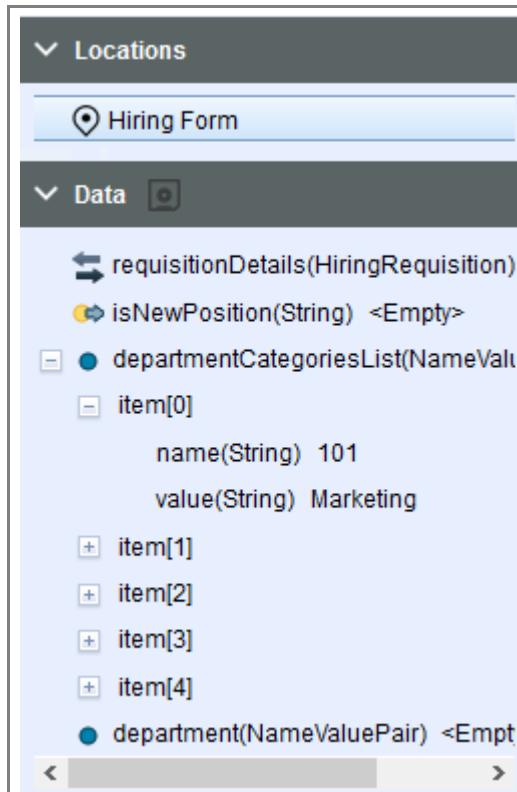
- ___ d. Save your work.
- ___ 9. Debug the coach, and verify that the **Department** single select works and binds the data to the variable.
 - ___ a. Click the **Debug** icon.



- ___ b. When the debug window opens, minimize the new window.
- ___ c. Click **Step over**.



- ___ d. The `departmentCategoriesList` variable is populated with data from the integration that you created in the last exercise.



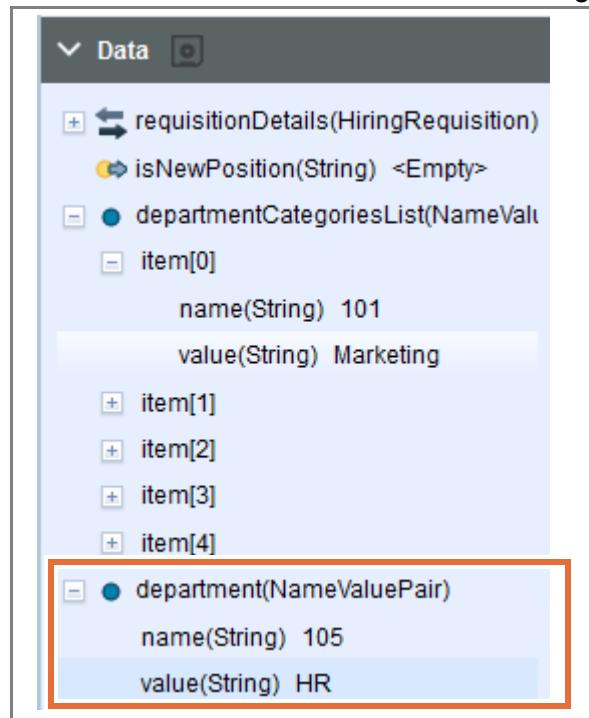
- ___ e. Click **Step over** to show the coach. The coach is displayed in the Playback window.
 ___ f. Click the **Department Details** tab. The database retrieval service now drives the selections that are presented to the user.

The screenshot shows the Playback window in IBM Worklight Studio. The tabs at the top are labeled: Requisition Details, Position Details, Recruiting Details, Compensation Details, and Department Details. The Department Details tab is highlighted with a red border. Below the tabs, there is a section titled "Department Details". Within this section, there is a table with a single row. The first column is labeled "Department" and contains a dropdown menu. The options listed in the dropdown are: Marketing, Finance, Engineering, Professional Services, and HR. The entire dropdown menu is also highlighted with a red border.

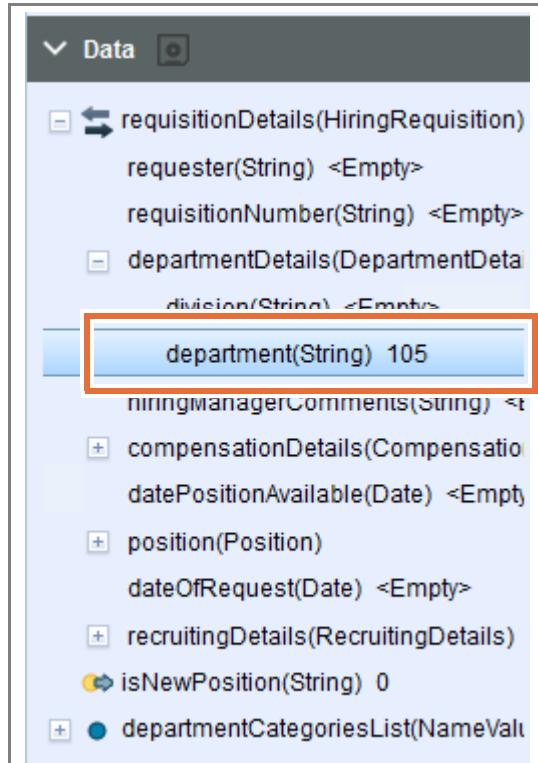
- __ g. In the **Department** field, select **HR** and click **Submit**.

The screenshot shows a user interface for a requisition form. At the top, there are three tabs: 'Requisition Details', 'Position Details', and 'Recruitin'. Below these tabs, the 'Requisition Details' section is active, titled 'Department Details'. Inside this section, there is a 'Department' field containing the value 'HR', which is highlighted with a red box. Below the department field is a 'Division' field, which is currently empty. At the bottom of the 'Requisition Details' section is a 'Submit' button, also highlighted with a red box.

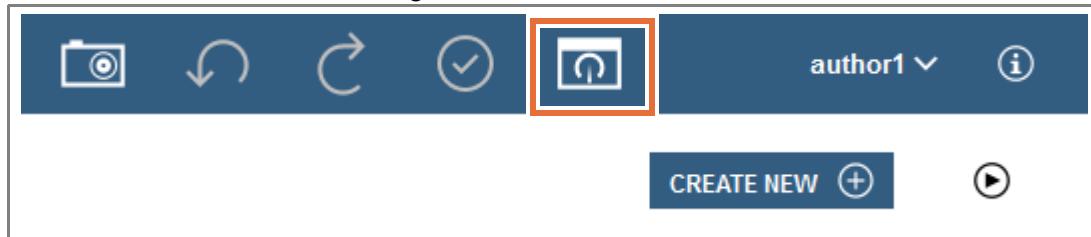
- __ h. Verify that the data is bound to the variable inside the debugger.



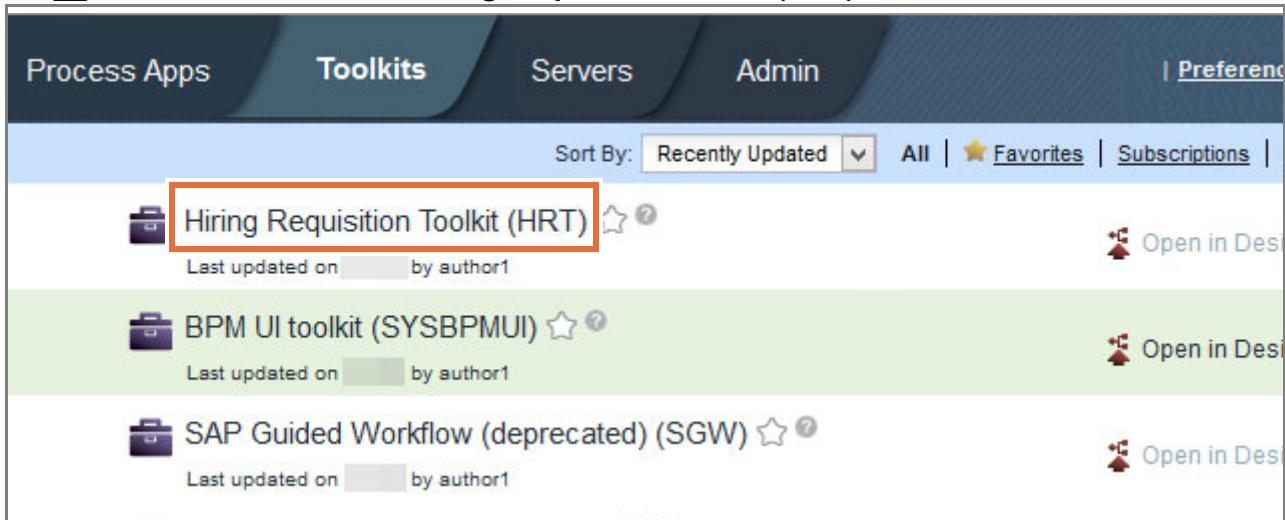
- __ i. Click **Step Over**. Verify that the data was mapped into the `requisitionDetails.departmentDetails.department` pipeline variable. The `Map Vars` script stores the key for the object, not the value.



- __ j. Close the **Debug Hiring Form** browser window.
- __ 10. Update the process application HR Recruitment Processes dependency to the most recent version of the Hiring Requisition Toolkit.
- __ a. Return to the Process Designer and click **Process Center** icon.

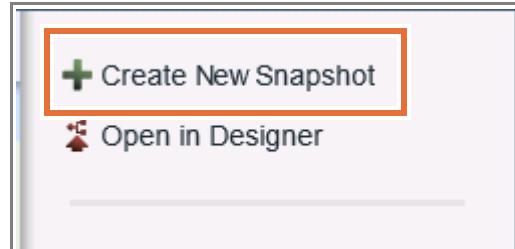


- __ b. Click the **Toolkits > Hiring Requisition Toolkit (HRT)** link.



The screenshot shows the SAP Fiori Launchpad interface. At the top, there are tabs for "Process Apps", "Toolkits" (which is the active tab), "Servers", and "Admin". Below the tabs, there is a search bar with the placeholder "Sort By: Recently Updated" and a dropdown arrow. To the right of the search bar are buttons for "All", "Favorites", and "Subscriptions". The main area displays a list of toolkits. The first item in the list, "Hiring Requisition Toolkit (HRT)", has a red box drawn around its name. Below it, there are two more toolkit entries: "BPM UI toolkit (SYSBPMUI)" and "SAP Guided Workflow (deprecated) (SGW)". Each toolkit entry includes a small icon, the toolkit name, a star rating, a question mark icon, and a "Last updated on" timestamp followed by "by author1". On the far right of each entry is a "Open in Designer" button with a small icon.

- __ c. On the right, click **Create New Snapshot**.



- ___ d. Name the snapshot **Playback 2** and provide the following documentation: This snapshot includes integrations. Click **Create**.

Create New Snapshot

Snapshot Name:
Playback 2

Documentation:

This snapshot includes integrations.

Create

- ___ e. Click the **Process Apps** tab. Open the **HR Recruitment Processes** process application.

Process Apps

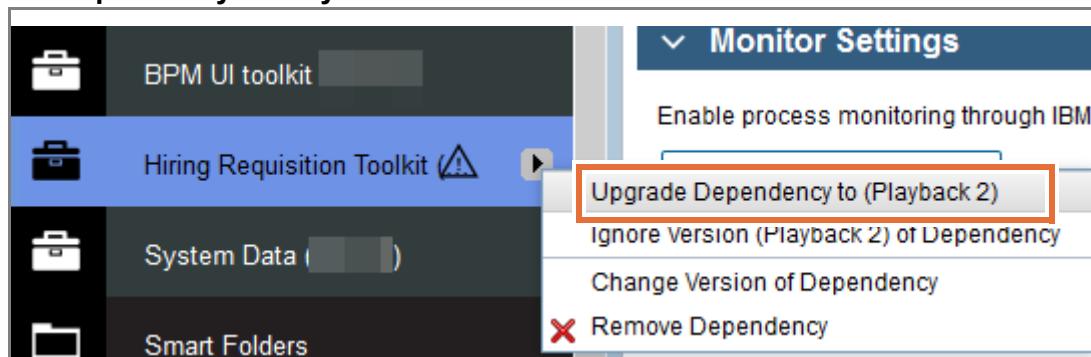
Toolkits Servers Admin

Sort By: Recently Updated | All | Favorites

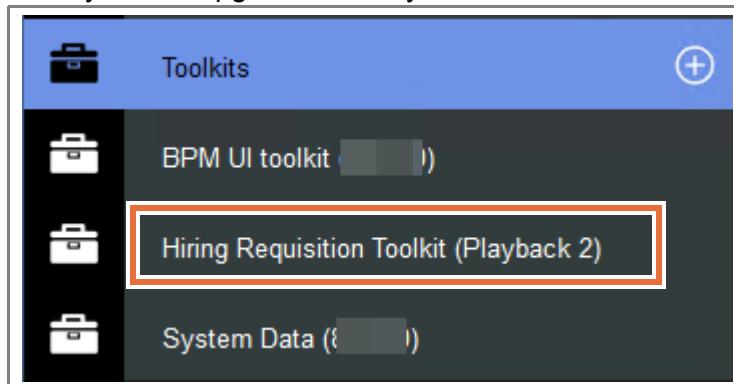
HR Recruitment Processes (HRR)	Open in Designer
Procurement Sample (STPPS1)	Open in Designer
Hiring Sample Advanced (HSAV1)	Open in Designer

- ___ f. Expand the **Toolkits** category in the library.

- __ g. Click the arrow next to **Hiring Requisition Toolkit (Baseline)** and click **Upgrade Dependency to Playback 2**.



- __ h. The dependency is now upgraded to Playback 2.



Important

Playback 2: Integrations, is now complete.

As part of the development process, you now review the playback and examine its functions in the Process Portal.

Does the decision service route the request correctly according to job type and salary?

You cannot test the undercover agent in the Process Portal.

You test the database query in a later exercise.

For more information about conducting the playback, see the Appendix A for playback.

End of exercise

Exercise review and wrap-up

The first part of this exercise looked at creating a decision service. You created rules in Business Action Language (BAL) to support the service and then implemented the new service in a process. Next, you created a message start event as a trigger to run the process. You organized assets with tags. Finally, you built a query service to read information from a database. The query service is built with environment variables and exposed process variables.

Exercise 10.Playback 3: Creating error handling for a service

Estimated time

00:15

Overview

This exercise covers how to implement exception handling in a service.

Objectives

After completing this exercise, you should be able to:

- Harden a service with a Catch Exception component

Introduction

1. Inside the Process Designer service diagram, drag a Catch Exception and attach it to a generic service or a human service.
2. Create a flow from the Catch Exception.

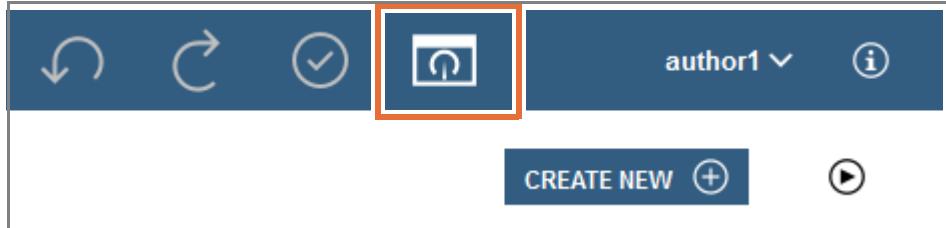
Requirements

Successful completion of the previous exercise is required.

Exercise instructions

Part 1: Implement exception handling in a service

- 1. Add a Catch Exception event to the Hiring Form service.
- a. Return to the Process Designer and click the **Process Center** icon.

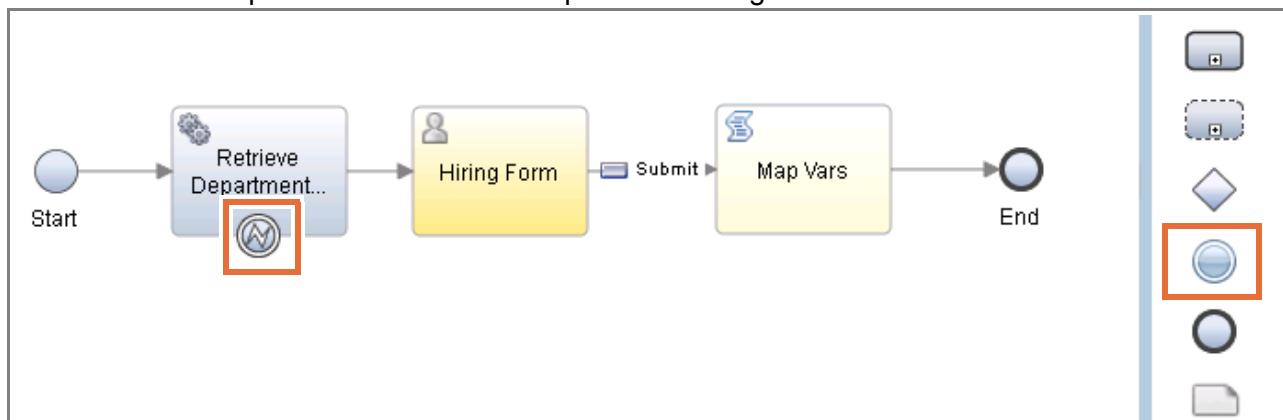


- b. Click the **Toolkits** tab.
- c. Click **Open in Designer** to open the **Hiring Requisition Toolkit (HRT)**.

A screenshot of the SAP Toolkits page. The 'Toolkits' tab is selected (highlighted with a red box). Below it, a list of toolkits is shown:

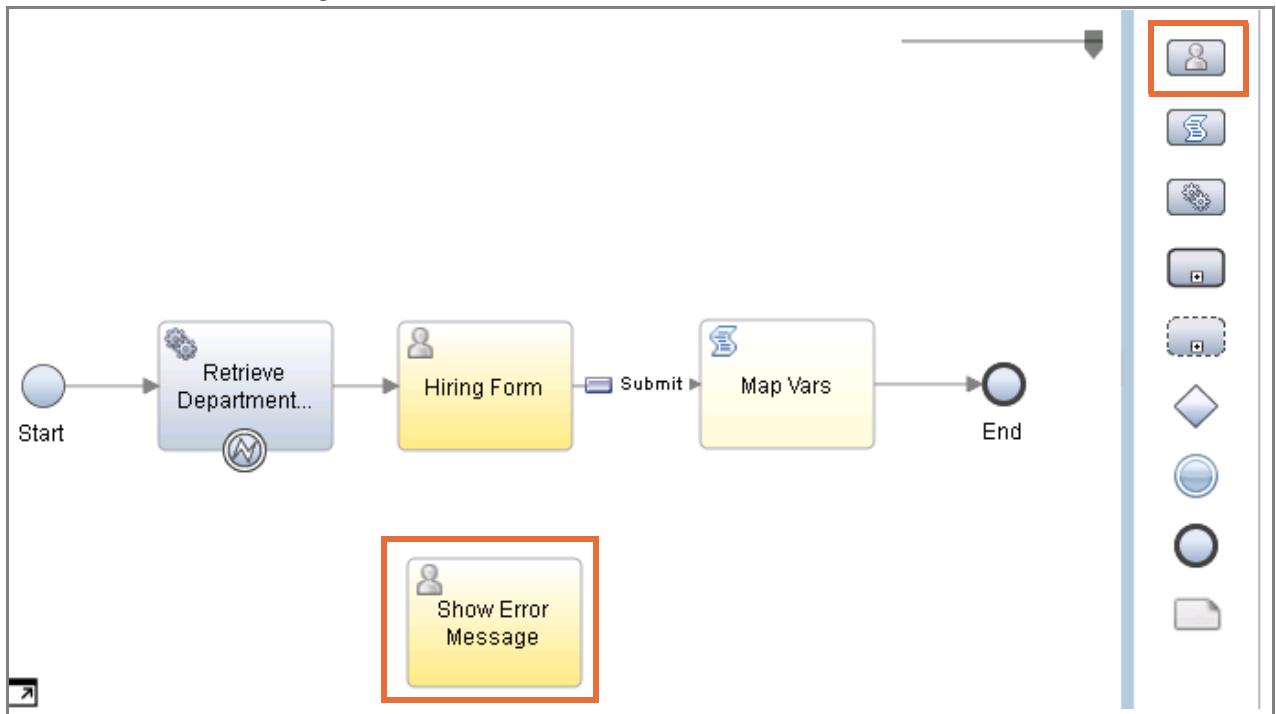
- Hiring Requisition Toolkit (HRT)** (highlighted with a red box) - Last updated on [redacted] by author1. Action: Open in Designer (highlighted with a red box).
- BPM UI toolkit (SYSBPMUI)** - Last updated on [redacted] by author1. Action: Open in Designer.
- SAP Guided Workflow (deprecated) (SGW)** - Last updated on [redacted] by author1. Action: Open in Designer.
- System Governance (TWSYSG)** - Last updated on [redacted] by author1. Action: Open in Designer.

- d. In the Hiring Requisition Toolkit, open the **User Interface > Hiring Form** human service.
- e. On the **Diagram** tab, drag an **Intermediate Event** from the service palette on to a control point on the Retrieve Department Categories service.



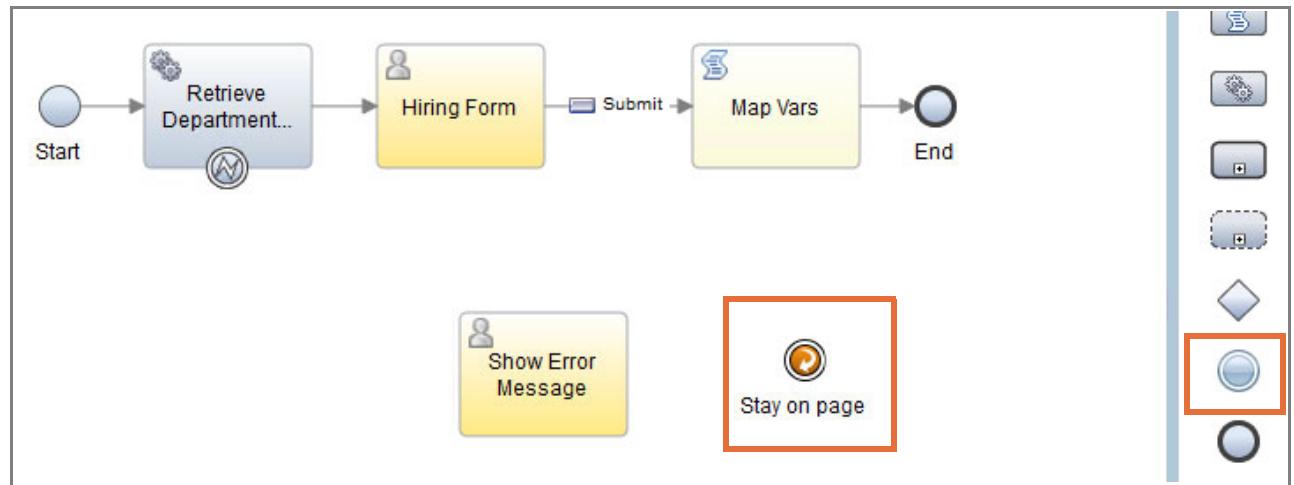
__ 2. Add a coach to display an error message.

__ a. Drag a **Coach** from the service palette onto the canvas. Rename the coach: Show Error Message

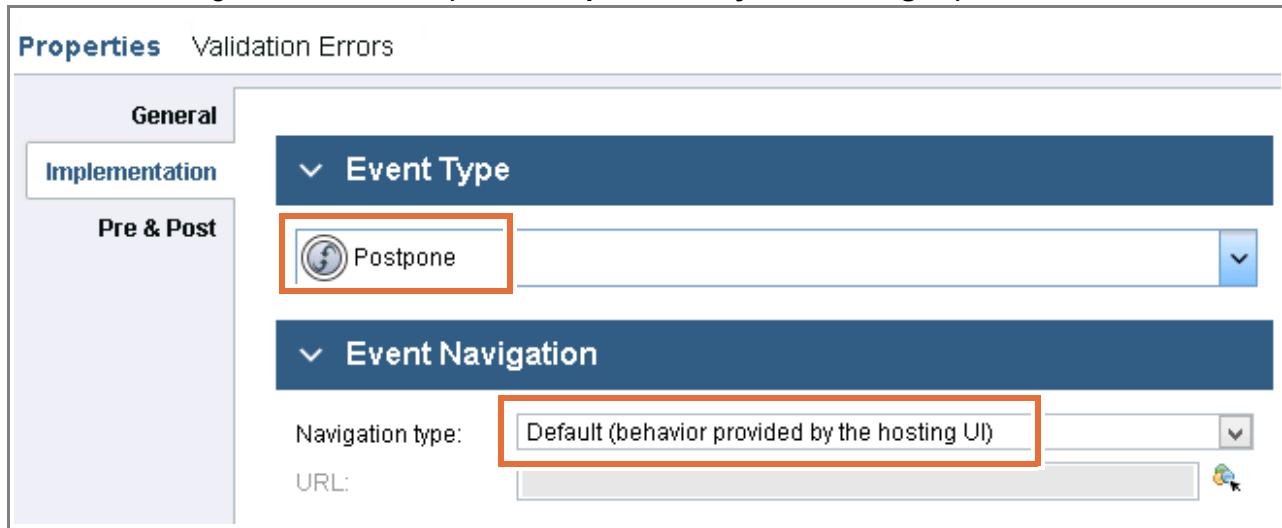


__ 3. Add a postpone task service.

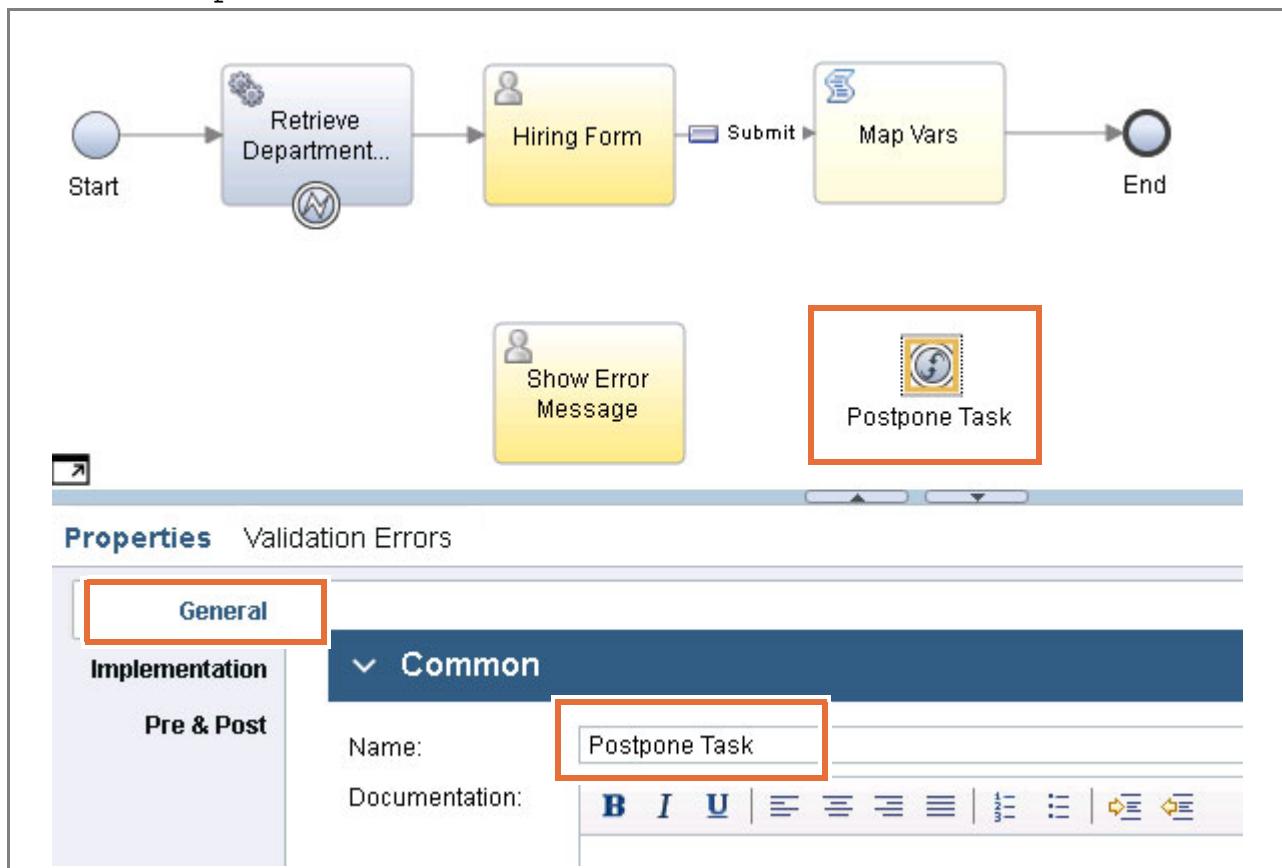
__ a. Drag an **Intermediate Event** from the palette to the right of the **Show Error Message** coach.



- ___ b. With the new Intermediate Event selected on the canvas, open the **Properties > Implementation** menu. Change the Event Type to **Postpone**. Leave the Event Navigation as **Default (behavior provided by the hosting UI)**.



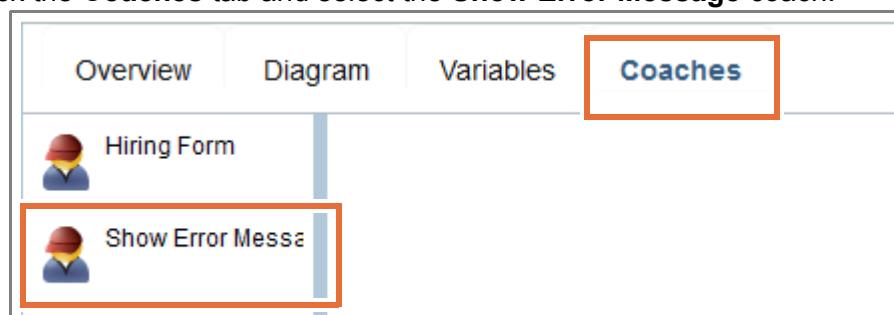
- ___ c. Open the **Properties > General > Common** section and rename the new event: Postpone Task



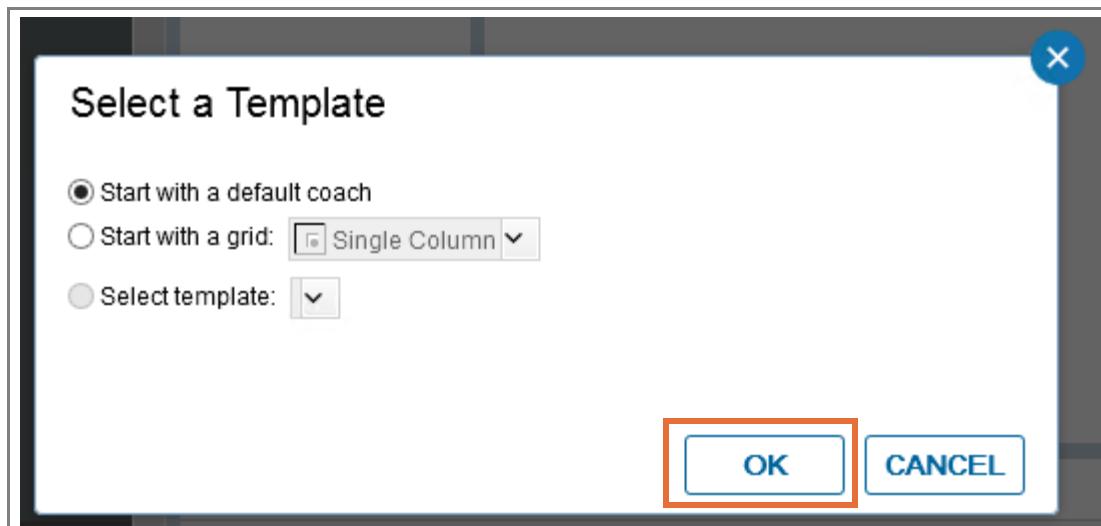
- ___ d. Save your work.

__ 4. Create the error message coach.

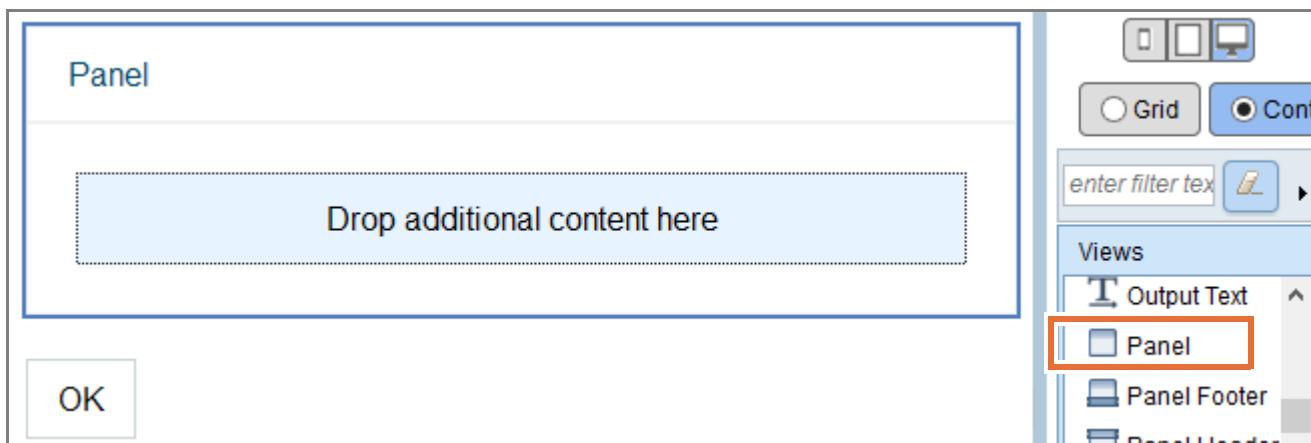
__ a. Click the **Coaches** tab and select the **Show Error Message** coach.



__ b. In the “Select a Template” window, ensure that **Start with a default coach** is selected, and click **OK**.



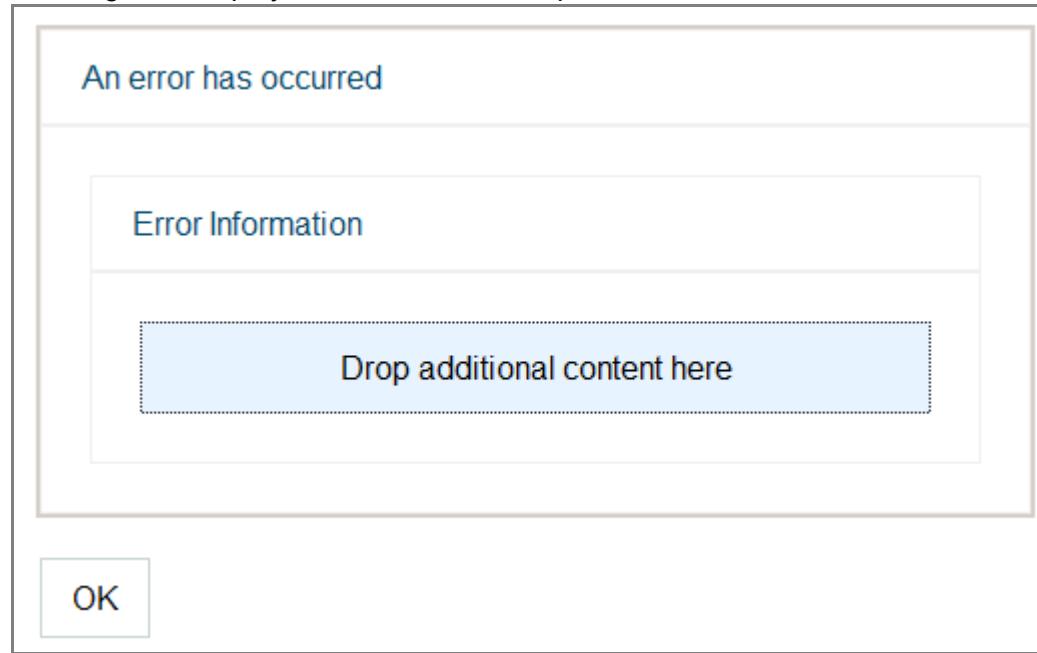
__ c. Drag a **Panel** from the Stock section in the palette to the top of the coach designer canvas.



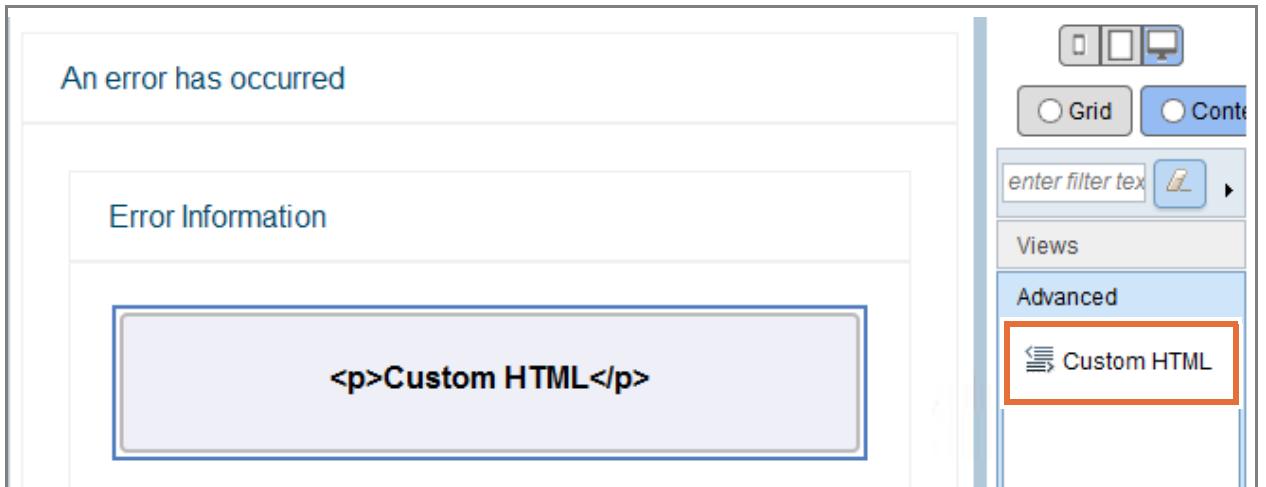
__ d. Change the **Properties > General > Common > label** on the panel to: An error has occurred

__ e. Drag another Panel inside the first one, into the section labeled **Drop additional content here**.

- ___ f. Change the display label on the internal panel to: Error Information

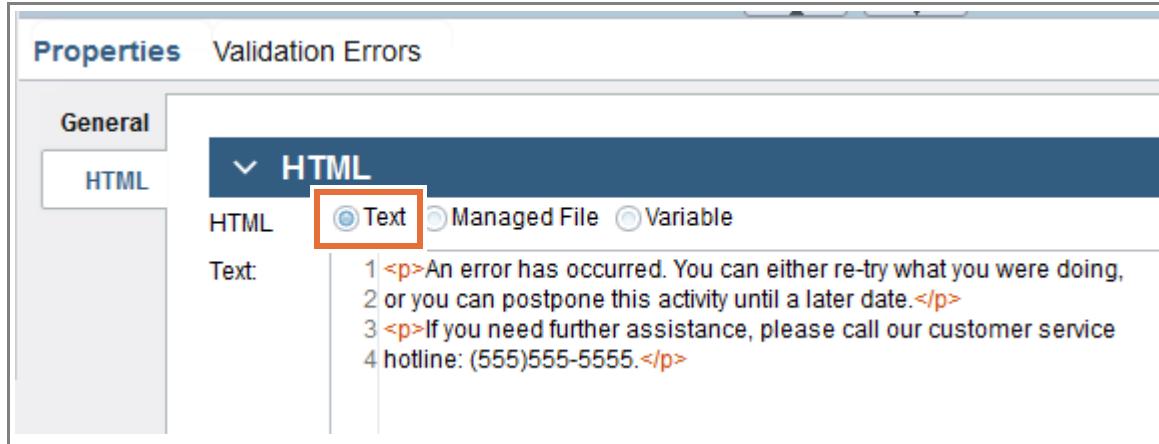


- ___ g. Drag a **Custom HTML** control from the **Advanced** palette tray into the Error Information additional content section.



- h. Under the **Properties > HTML** menu, select **Text** and add error information to the HTML text block that is provided to the user:

```
<p>An error has occurred. You can either re-try what you were doing, or  
you can postpone this activity until a later date.</p>  
<p>If you need further assistance, please call our customer service  
hotline: (555)555-5555.</p>
```

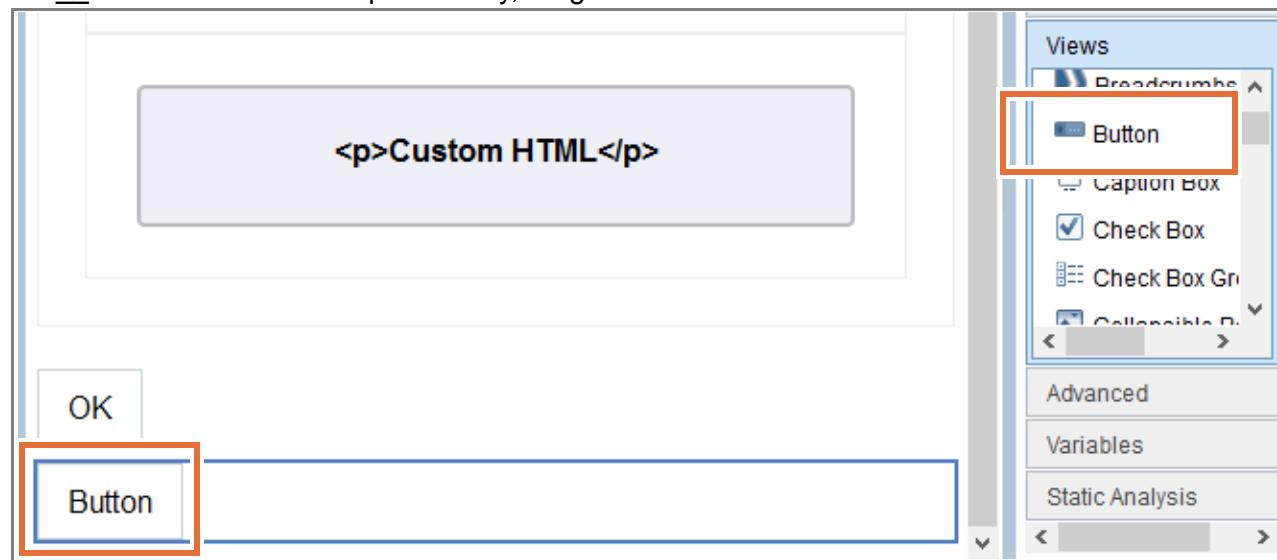


Hint

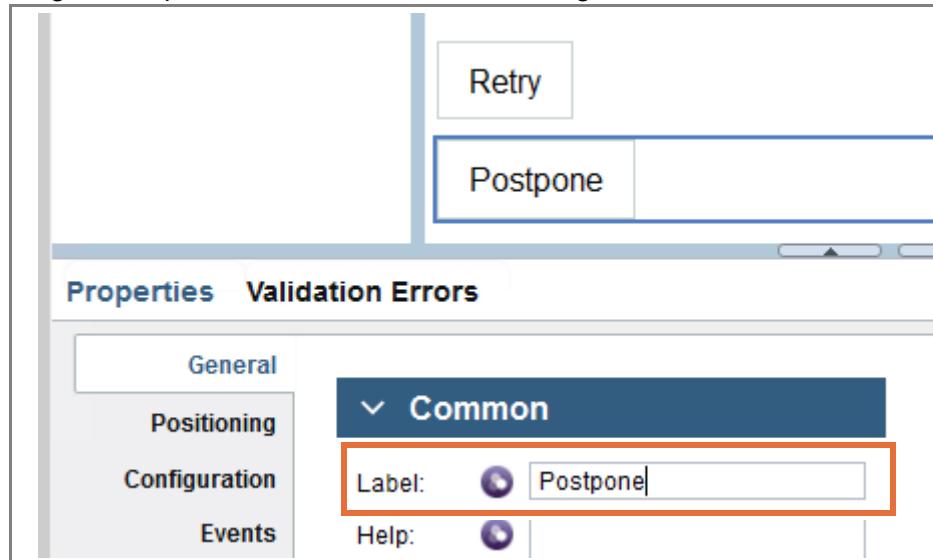
You can copy and paste the text from the following file that is saved to your environment:

C:\labfiles\Exercise_Support_Files\Exercise10\Script1.txt.

- i. From the **Views** palette tray, drag a **Button** below the **OK** button.

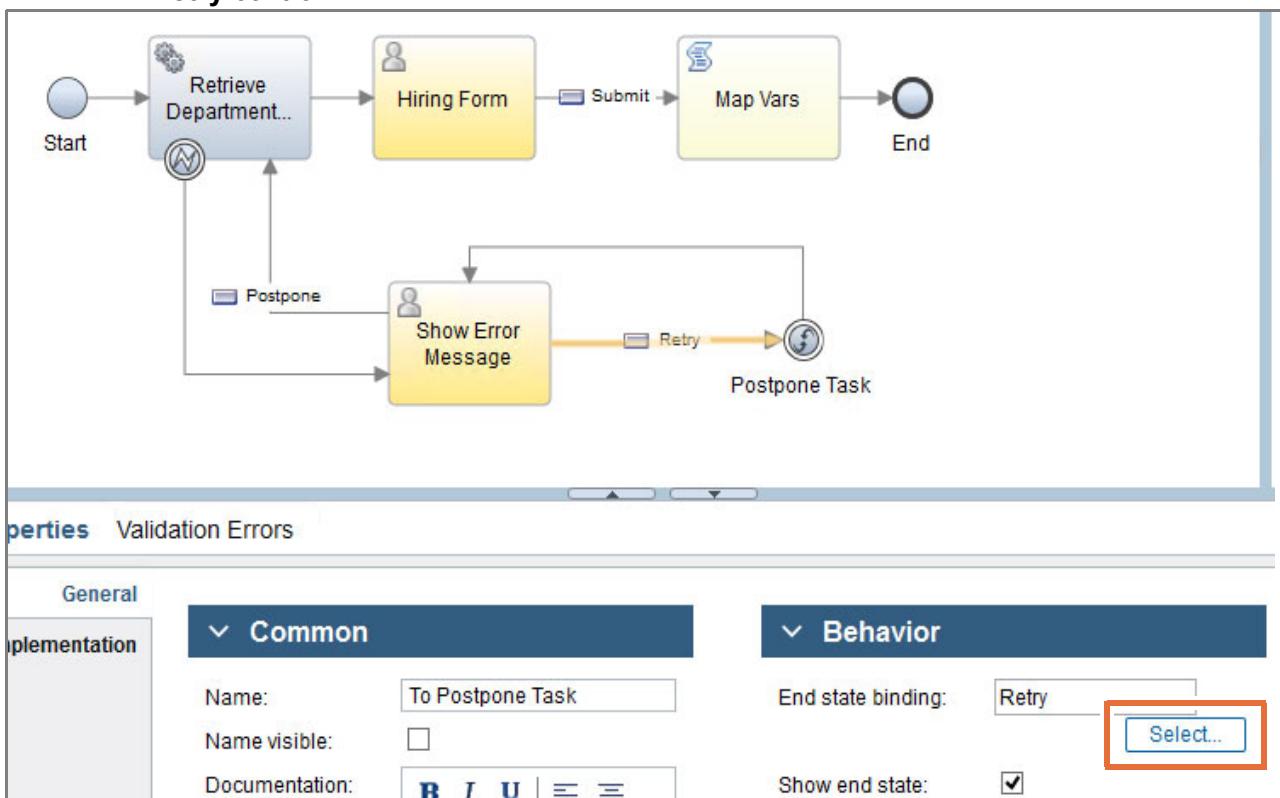


- __ j. Change the top button label to: **Retry**. Change the second button label to: **Postpone**.

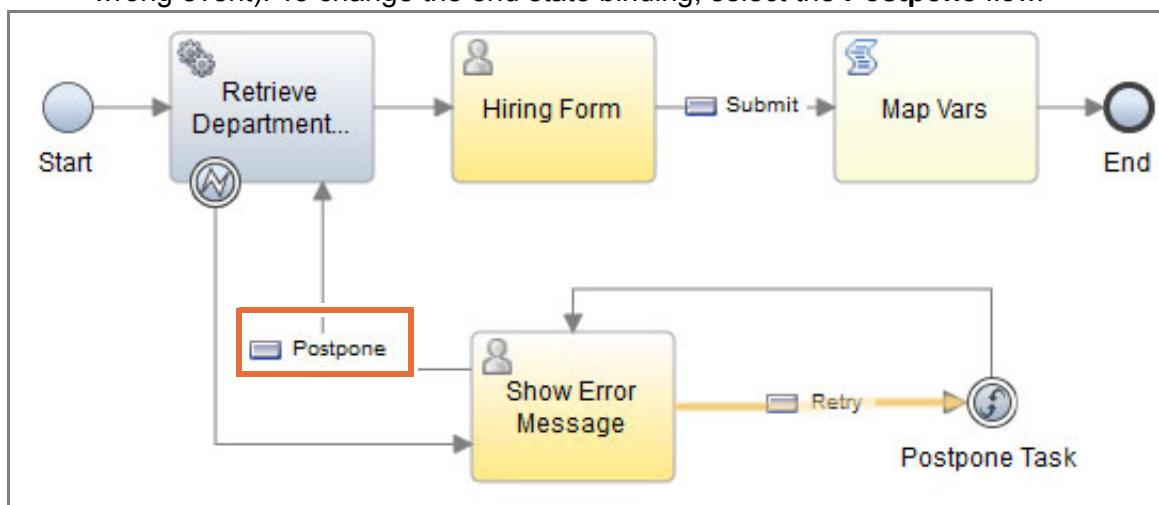


- __ k. Save your work.
- 5. Reconnect the flow on the **Hiring Form** canvas.
- __ a. Click the **Diagram** tab.
 - __ b. Connect the new elements on the canvas. First, connect the **Error Intermediate Event** event to the **Show Error Message** coach.
 - __ c. Next, connect the **Show Error Message** coach to the **Postpone Task** event.
 - __ d. Third, connect **Postpone Task** back to **Show Error Message**.

- __ e. Finally, connect **Show Error Message** to **Retrieve Department Categories** by the **Retry** control.



- __ f. The label on the flow is incorrect (the instructions intentionally wired the flows up to the wrong event). To change the end state binding, select the **Postpone** flow.



- __ g. In the **Properties > General > Behavior** section, click **Select...**.

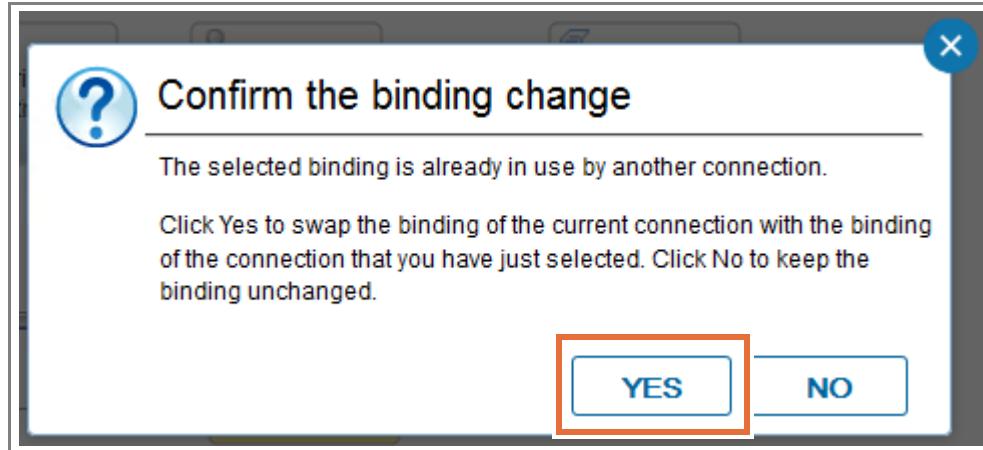
Behavior

End state binding: **Postpone** **Select...**
Show end state:

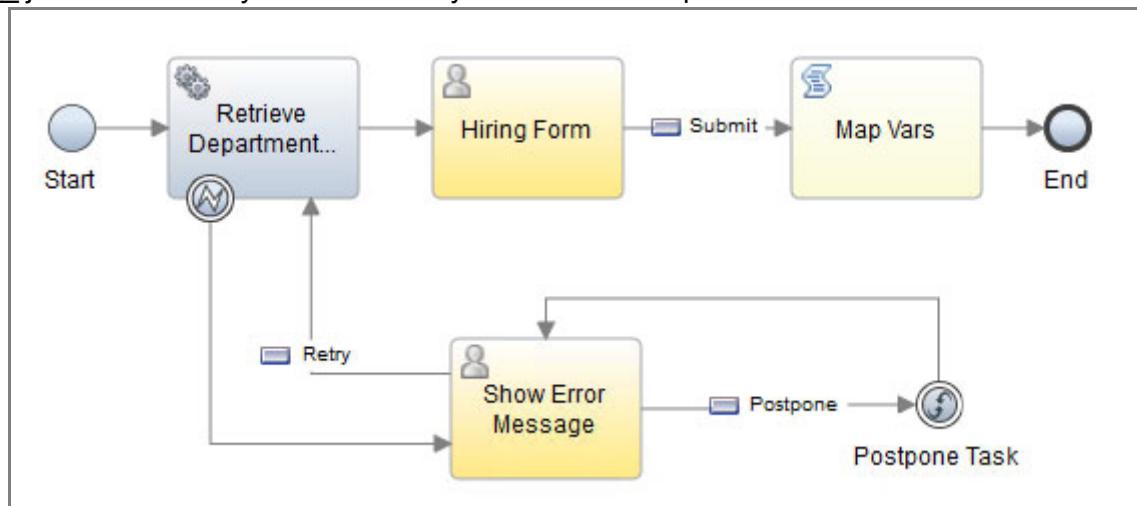
- __ h. From the coach dialog box, select the **Retry** button.



- __ i. Click **Yes** to confirm the binding change.



- __ j. Ensure that your buttons on your coach correspond to the correct flows on the coach.



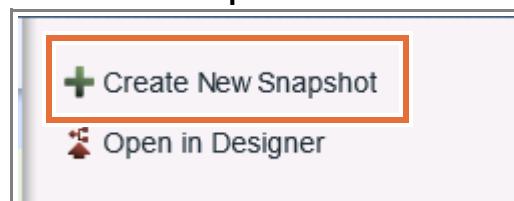
- __ 6. Save your work.
 __ 7. Create a snapshot of the new toolkit and upgrade the dependency to the new snapshot.
 __ a. Return to Process Center.



- __ b. Click the **Toolkits > Hiring Requisition Toolkit (HRT)** link.

The screenshot shows the SAP Fiori Launchpad interface. At the top, there are tabs for "Process Apps", "Toolkits" (which is highlighted with a red box), "Servers", and "Admin". Below the tabs, there is a search bar with "Sort By: Recently Updated" and filters for "All", "Favorites", and "Subscriptions". The main area displays a list of toolkits. The first toolkit, "Hiring Requisition Toolkit (HRT)", is highlighted with a red box. It has a brief description: "Last updated on [redacted] by author1". To the right of the toolkit name are two buttons: "Open in Designer" and a small question mark icon. Other toolkits listed are "BPM UI toolkit (SYSBPMUI)" and "SAP Guided Workflow (deprecated) (SGW)".

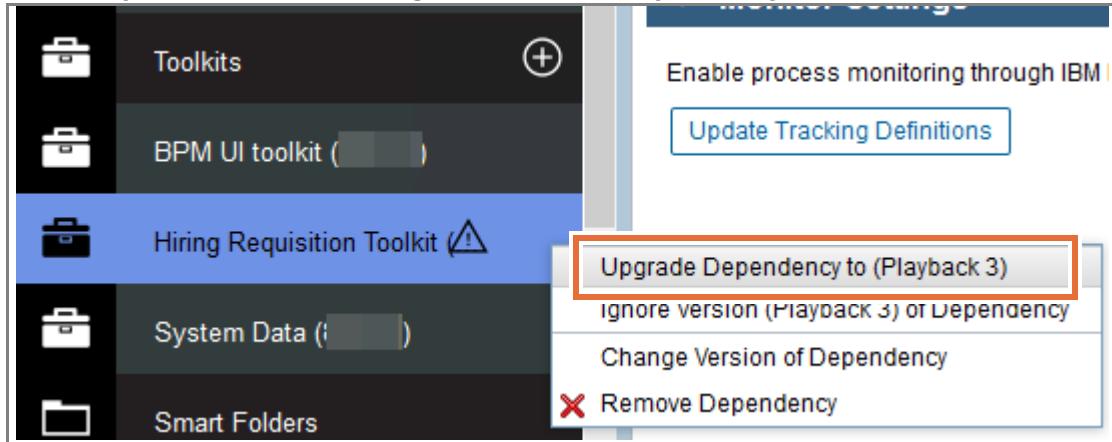
- __ c. On the right, click **Create New Snapshot**.



- __ d. Name the snapshot `Playback 3` and provide the following documentation: This snapshot includes error handling for a service. Click **Create**.

The screenshot shows the "Create New Snapshot" dialog. The "Snapshot Name:" field is filled with "Playback 3" and is highlighted with a red box. Below the field is a rich text editor toolbar with various buttons for bold, italic, underline, font size (12pt), and alignment. The documentation area contains the text "This snapshot includes error handling for a service.". In the bottom right corner of the dialog, there is a "Create" button which is highlighted with a red box.

- __ e. Open the **Process Apps > HR Recruitment Processes** process application in the process designer.
- __ f. Click **Toolkits** in the library. Click the arrow next to **Hiring Requisition Toolkit (Playback 2)** and click **Upgrade Dependency to (Playback 3)**.



Important

Playback 3: Creating error handling for a service is complete.

To test the error handling capability of the service, see the Appendix A: Conducting playbacks, Part 6.

As part of the development process, you can now review the playback and examine its functions in the Process Portal.

End of exercise

Exercise review and wrap-up

In this exercise, you added a Catch Exception component to capture and manage errors in the process.

Exercise 11. Playback 3: Creating a snapshot for deployment

Estimated time

00:15

Overview

This exercise covers how to create a snapshot for deployment.

Objectives

After completing this exercise, you should be able to:

- Create a snapshot for deployment

Introduction

Create a snapshot from inside the IBM Process Designer:

1. Inside of IBM Process Designer, click **Snapshot**.
2. Give the snapshot a descriptive name.

Requirements

Successful completion of the previous exercise is required.

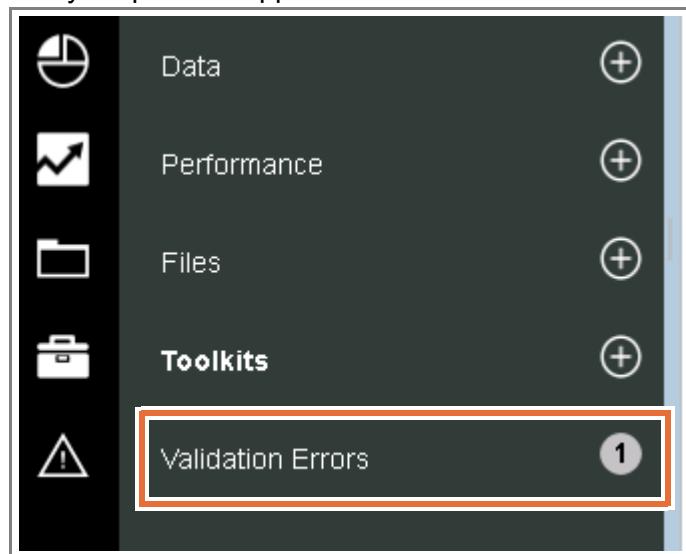
Exercise instructions

Part 1: Prepare for final snapshot

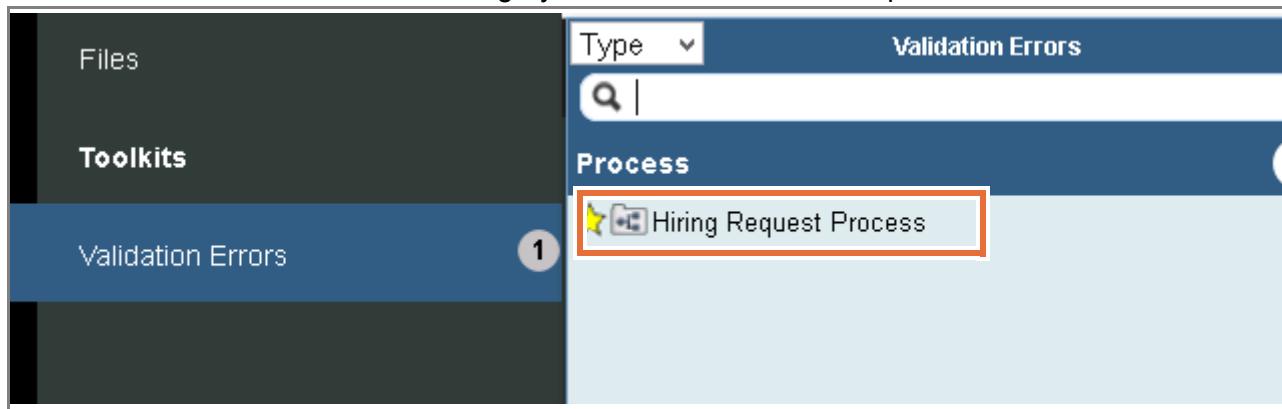
- 1. Check for any validation errors in the process application.

Before taking the final snapshot of the process application for deployment, it is a good practice to check for any validation errors in the process application.

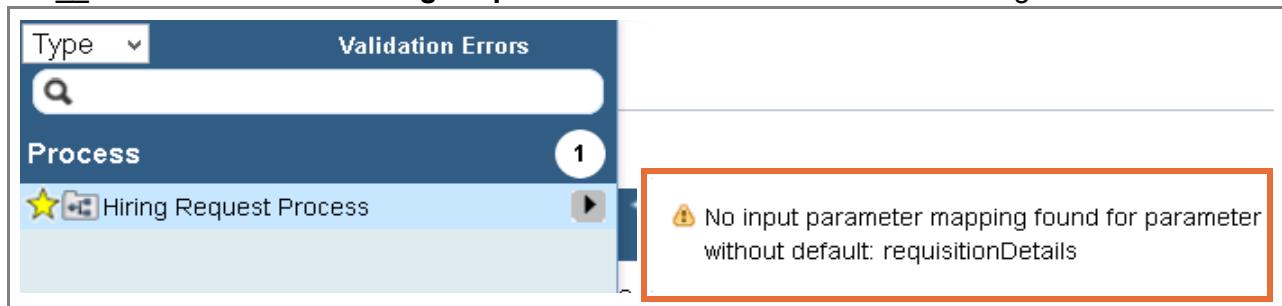
- a. Open the **HR Recruitment Processes** process application.
- b. Validation errors must be resolved before you deploy a process application to a different environment. The instructions thus far introduced an error. You correct this error before the deployment your process application.



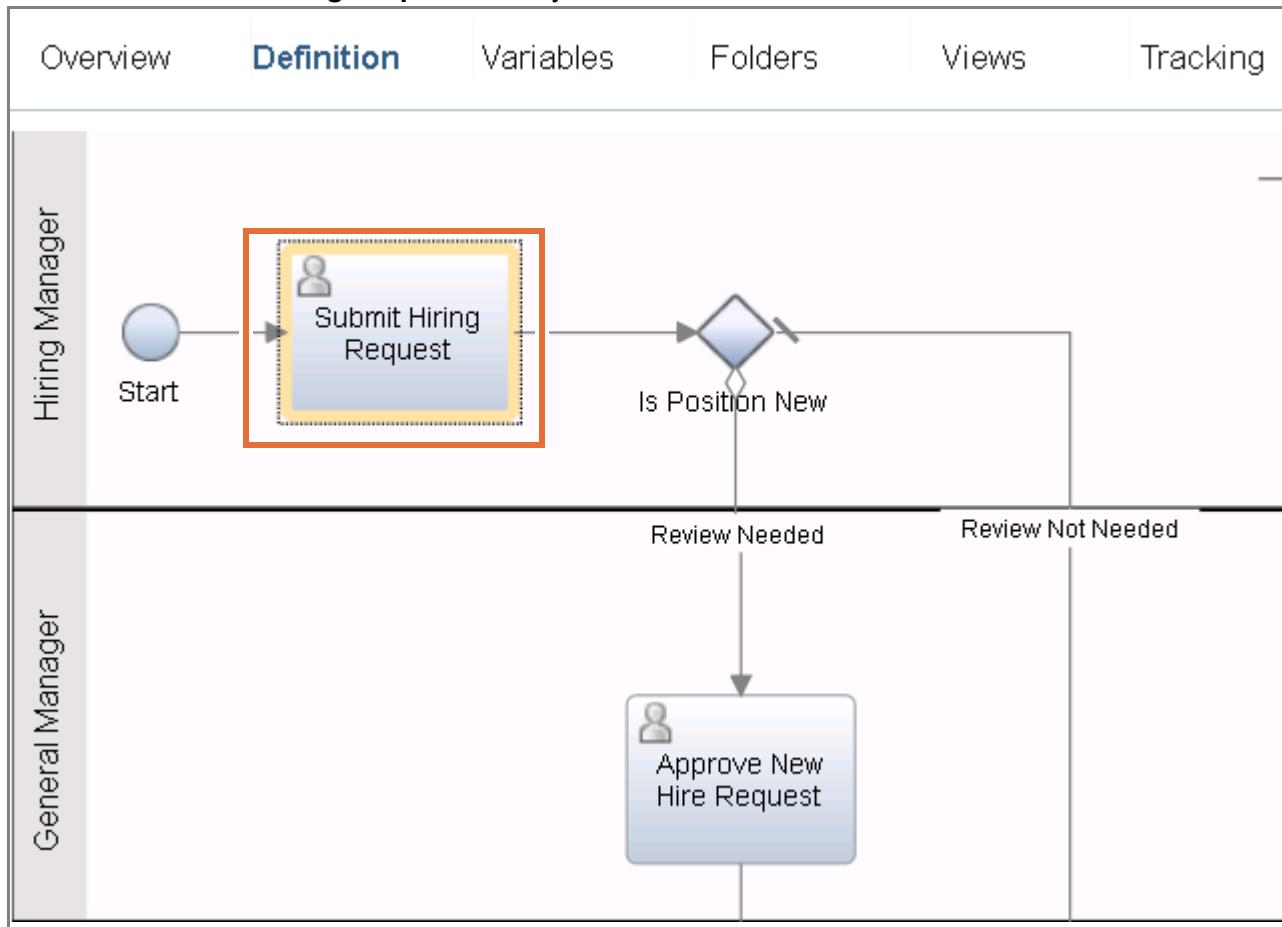
- c. In this example, the Hiring Request Process contains an error. To resolve this error, click the **Validation Errors** category to check the name of the process that contains the error.



__ d. Hover over the **Hiring Request Process** to check the error message.



__ e. To resolve the parameter mapping errors, open the **Hiring Request Process**. Click the **Submit Hiring Request** activity.



__ f. Open the **Properties > Data Mapping** menu.

- __ g. In the **Input Mapping** section, map the input to the `requisitionDetails` (`HiringRequisition`) variable by typing `tw.local.requisitionDetails`.

The screenshot shows a 'Input Mapping' section. On the left, there is a text input field containing the value 'tw.local.requisitionDetails'. This field is highlighted with an orange border. To the right of the field is a small icon of a person with a gear and a yellow arrow pointing right, followed by the text 'requisitionDetails (Hiring...)'.

- __ h. In the **Output Mapping** section, map the first output to the `requisitionDetails` (`HiringRequisition`) variable by typing `tw.local.requisitionDetails`.
- __ i. In the **Output Mapping** section, map the second output to the `isNewPosition` (`String`) variable by typing `tw.local.isNewPosition`.

The screenshot shows an 'Output Mapping' section. On the left, there are two output variables listed: '[requisitionDetails \(HiringRequisition\)](#)' and '[isNewPosition \(String\)](#)'. To the right of each variable is a yellow arrow pointing right. To the far right, there are two corresponding target variables: '`tw.local.requisitionDetails`' and '`tw.local.isNewPosition`'. Both of these target variables are highlighted with orange borders.

- __ j. Select the **Approve New Hire Request** activity.
- __ k. Open the **Properties > Data Mapping** menu.
- __ l. In the **Input Mapping** section, map the input to the `requisitionDetails` (`HiringRequisition`) variable by typing `tw.local.requisitionDetails`.

The screenshot shows an 'Input Mapping' section. On the left, there is a text input field containing the value 'tw.local.requisitionDetails'. This field is highlighted with an orange border. To the right of the field is a small icon of a person with a gear and a yellow arrow pointing right, followed by the text 'requisitionDetails (Hiring...)'.

- __ m. In the **Output Mapping** section, map the first output to the `requisitionDetails` (`HiringRequisition`) variable by typing `tw.local.requisitionDetails`.

- n. In the **Output Mapping** section, map the second output to the `isNewPosition` (`String`) variable by typing `tw.local.isNewPosition`.

Output Mapping

requisitionDetails (HiringRequisition)	→	tw.local.requisitionDetails
isNewPosition (String)	→	tw.local.isNewPosition

- o. Save your changes.

Part 2: Check the history of the process application

- 1. Before taking the final snapshot of the process application for deployment, you can check the history of the actions that you performed on the process application.
- a. Return to the Process Center.



- b. Click the **HR Recruitment Processes** process application.

Process Apps	Toolkits	Servers	Admin	Preference
Sort By: Recently Updated All HR Recruitment Processes (HRR) <small>Last updated on [redacted] by author1</small>				
Procurement Sample (STPPS1) <small>Last updated on [redacted] by author1</small>				
Hiring Sample Advanced (HSAV1) <small>Last updated on [redacted] by author1</small>				

c. Click History.

The screenshot shows the Process Apps interface with the 'History' tab highlighted by a red box. The application name 'HR Recruitment Processes (HRR)' is visible, along with tabs for 'Schemas', 'Snapshots', 'History' (which is active), and 'Manage'. A dropdown menu 'Track' is set to 'Baseline'. Below the tabs, there's a status message 'Current' with a camera icon, and a note 'Last changed on [redacted] by author1'.

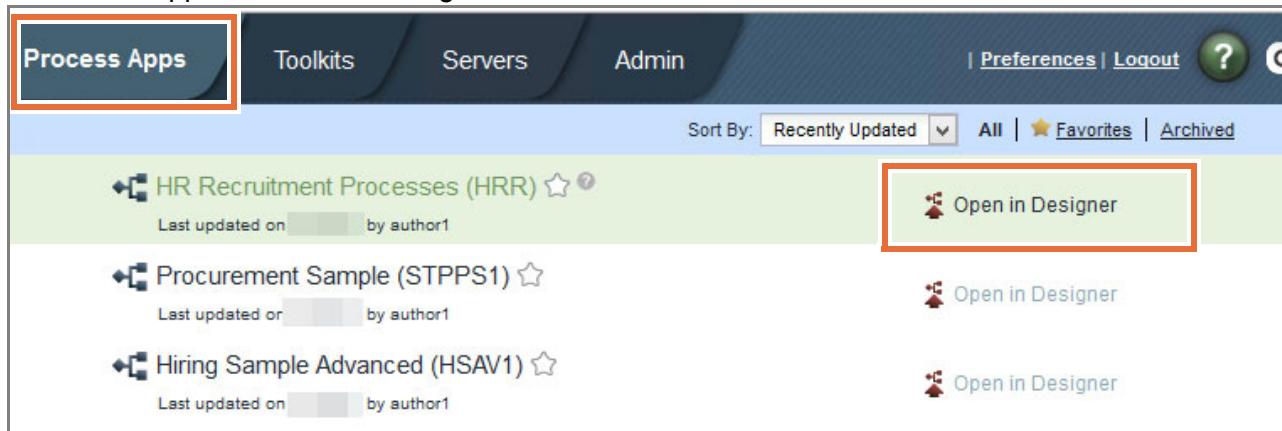
d. Verify the set of actions that you performed on the process application from the beginning.

The screenshot shows the Process Apps interface with the 'History' tab selected. The application name 'HR Recruitment Processes (HRR)' is visible, along with tabs for 'Schemas', 'Snapshots', 'History' (which is active), and 'Manage'. The history log displays several audit entries:

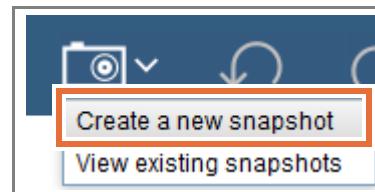
- Playback_1 snapshot was modified in Main by author1 on [redacted] at 10:40:44 AM
- Playback_1 snapshot was created in Main by author1 on [redacted] at 10:40:44 AM
- Playback_3 snapshot was modified in Main by author1 on [redacted] at 2:40:21 AM
- Playback_3 snapshot was created in Main by author1 on [redacted] at 2:40:20 AM
- HR Recruitment Processes was updated as a dependency from null to Playback 3 for null by author1 on [redacted]
- HR Recruitment Processes was updated as a dependency from null to Playback 2 for null by author1 on [redacted]
- HR Recruitment Processes was updated as a dependency from null to Playback 2 for null by author1 on [redacted]
- HR Recruitment Processes was updated as a dependency from null to Baseline for null by author1 on [redacted]
- HR Recruitment Processes process app modified by author1 on [redacted] at 3:50:44 AM
- Main track was created by author1 on [redacted] at 3:50:36 AM
- HR Recruitment Processes process app created by author1 on [redacted] at 3:50:34 AM

Part 3: Create a snapshot of the process application

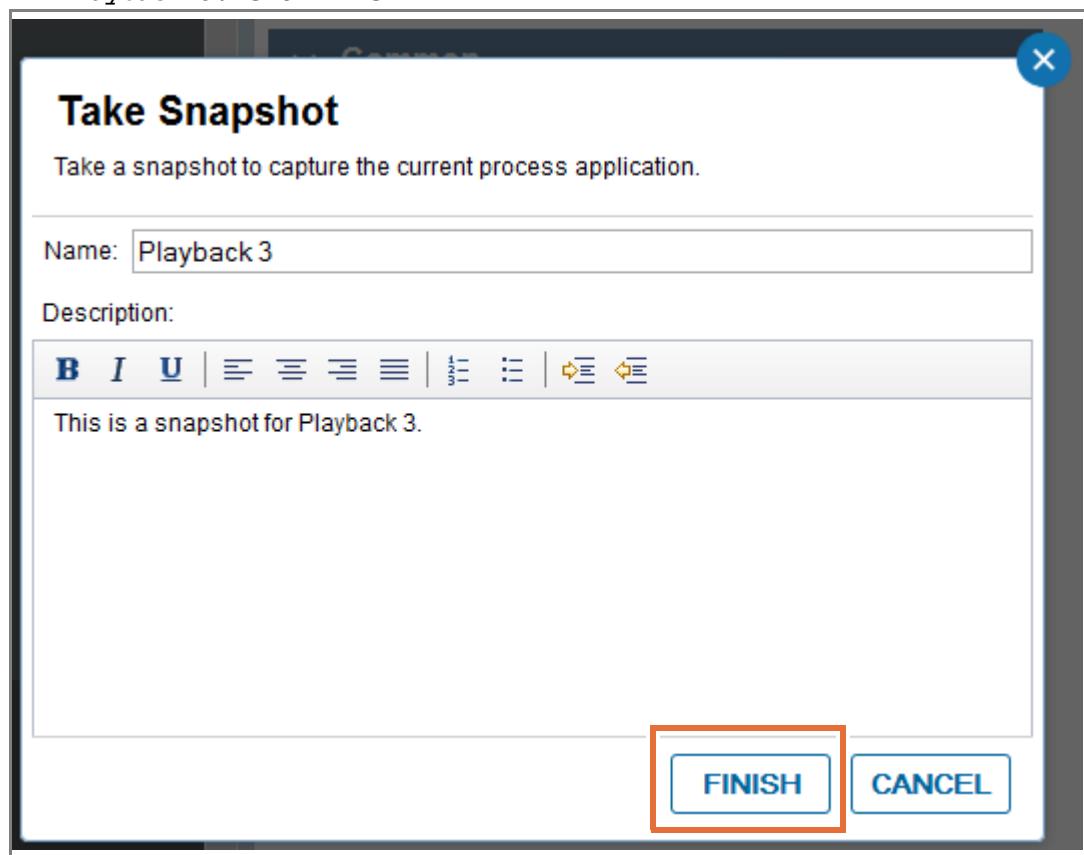
- 1. You created snapshots throughout development from the previous exercises. This exercise is for the final deployment to the next environment. Create a snapshot of the **Hiring Request Process** in IBM Process Designer.
- a. Click the **Process Apps** tab. Open the **HR Recruitment Processes** process application in the Designer.



- b. Expand the **Snapshot** menu on the top of the Process Designer window. Click **Create a new snapshot**.



- ___ c. Give your snapshot the name **Playback 3** and description: This is a snapshot for Playback 3. Click **Finish**.



- ___ d. Click **OK** when the pop-up window is shown.
 ___ e. Return to the **Process Center**.
 ___ f. Click the **HR Recruitment Processes** process application.

The screenshot shows the Process Center interface. The top navigation bar includes 'Process Apps', 'Toolkits', 'Servers', 'Admin', and 'Preference'. Below the navigation bar, there's a search bar with 'Sort By: Recently Updated' and a dropdown menu. The main area displays a list of process applications:

- HR Recruitment Processes (HRR)** (highlighted with a red box)
- Procurement Sample (STPPS1)
- Hiring Sample Advanced (HSAV1)

 Each item has a small icon, a last update timestamp, and an author name (author1). To the right of each item is an 'Open in Design' button with a red star icon.

- g. Your snapshot is displayed in the list of snapshots.

The screenshot shows the 'Process Apps' interface. At the top, there are tabs for 'Process Apps', 'Toolkits', 'Servers', and 'Admin'. Below that, a breadcrumb navigation shows 'HR Recruitment Processes (HRR)'. There are three main sections: 'Current', 'Recent', and 'Schemas'. The 'Current' section shows a schema named 'Current' last changed by 'author1'. The 'Recent' section shows a schema named 'Test Schema' last changed by 'author1'. The 'Schemas' section shows a schema named 'Playback 3 (P3)' with a '(New)' status indicator. This 'Playback 3 (P3)' entry is highlighted with an orange border.



Information

Your process application is ready to deploy to the test environment. If your account is authorized to deploy this process application, you can use the IBM Process Designer to deploy the snapshot. You can also use the Process Admin console to do the deployment.

End of exercise

Exercise review and wrap-up

In this exercise, you prepared for the final snapshot. If any validation errors were present in the process application, you resolved any problems. Next, you looked at the history of the actions that were performed on the process application and created the snapshot for deployment.

Appendix A. Conducting playbacks

Overview

After the end of each playback in the development cycle, test your work in the Process Portal. Conducting a playback is a good time to demonstrate to business stakeholders how data flows in the process application.

Objectives

At the end of this exercise, you should be able to:

- Log on to the Process Portal
- Create an instance of a process
- Demonstrate the business data flow through coaches

Introduction

A playback between business and IT is a focused demonstration of a partially implemented process model at the designated development phase. The goal of a playback is to encourage discussion, build consensus, improve collaboration, and ultimately approve the process model. Playbacks thus enable the iterative agile development of the process application.

Playbacks provide early visibility and input from the business group on the process application functions. Often times, the perspective of the business group benefits developers because they quickly identify adjustments to requirements well before the final product is implemented.

1. Log on to the Process Portal. The URL is:

`http://<server URL:port>/portal`

2. Demonstrate process flow. Complete a coach and submit the data.
3. Show the effects of business data as it changes the process flow.

Requirements

Completing the exercises for this course requires a lab environment that includes the exercise support files, the IBM Process Designer, and the IBM Business Process Manager Process Center development environment.

This exercise relies upon the `HR Recruitment Processes` process application.

Exercise instructions

The ability to shift direction during each playback phase is key to reaching the ultimate BPM project target. The appreciably faster time to value that the BPM team and business realize with this approach sets the direction for future BPM project development.

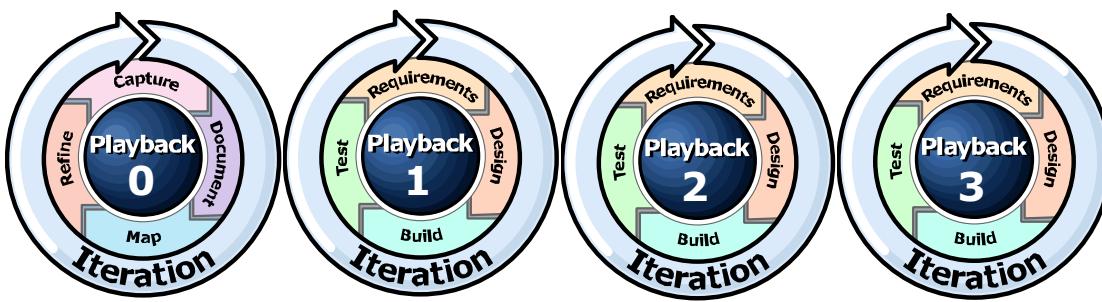
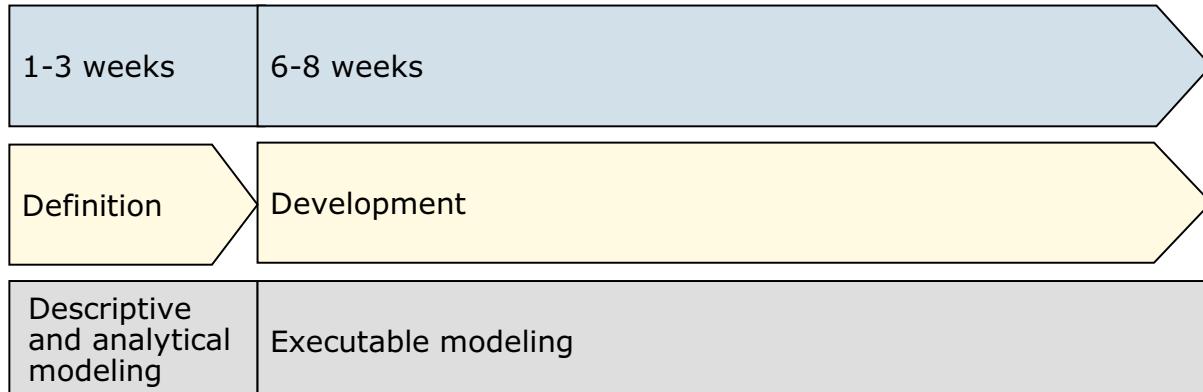
Themed playbacks

Often playbacks are conducted as themed phases of development. The number of playbacks depends on the number of themes that the organization wants to use and the complexity of the business process needs. This guide can be used as a primer for just about any BPM project development effort.

IBM Training

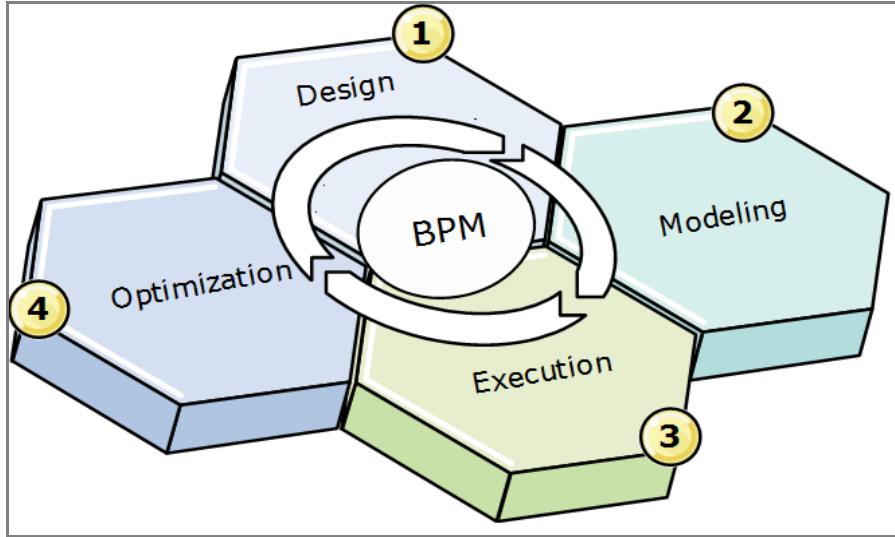
IBM

Playback cycles



Project management and the BPM lifecycle

Completion of the project management means that a large portion of the BPM lifecycle is complete. After the project development is complete, which is 9 – 14 weeks, the process application is monitored during production. The process performance data is then evaluated and analyzed for more efficiency. Business goals can also be altered based on process performance at this stage of the lifecycle. After completion, the next iteration of the project is initiated, and the project development with playbacks begins again.



1. Design goals:

- Capture executive vision
- Process nomination
- Process prioritization
- Process discovery
- Process analysis

2. Modeling goals:

- Create a process model
- Process adjustments
- Process simulation

3. Execution goals:

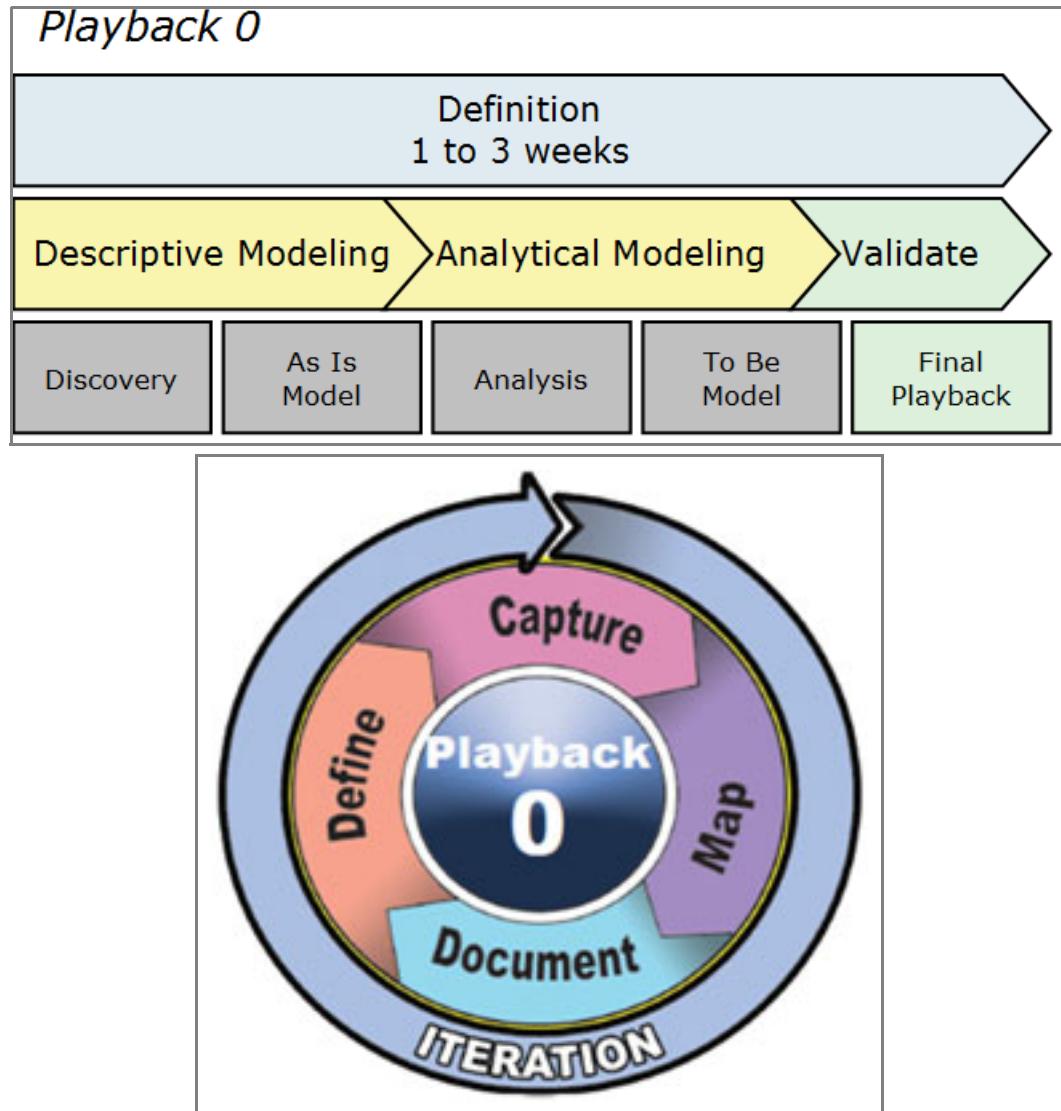
- Implement the process model as a process application
- Adjust business process requirements as needed
- Deploy and monitor the process application

4. Optimization goals:

- Analyze and evaluate process performance data
- Evaluate the business process ability to meet new business goals

Part 1: Playback 0: Capture, describe, and model the process

In Playback 0, documentation and process analysis set the correct framework for the process model creation, process automation, and process activity added value to gain efficiencies, visibility, and effectiveness for the business process. The entire Playback 0 stage typically takes 1 – 3 weeks to complete.



Part 2: Playback 1: Process flow data implementation, User interface design and implementation

In Playback 1, the process data model is implemented along with the appropriate process flow for the data. Flow data is different from business data in that flow data moves the process along from flow object to flow object. The most obvious examples of flow data are the data elements that the decision points use in the process or service diagrams. When a token comes to a decision point, the values of data elements are used to determine the next paths to take. Flow data also includes the following data:

- Data that is used to determine which activities to run
- Data that is used to determine who runs each activity
- Data that is used to determine when an activity is due or when an activity must be escalated

The process flow data ensures that the business process gets the right activities to the right people at the right time.

The enhancements that the business wants for process application user interfaces happen at this stage as well. Often, the initial development efforts are marred with requests to begin with full fidelity user interfaces. It is not uncommon to have this request because it is driven from a desire to impress executive level stakeholders with prototypical user interfaces that can be manipulated. It is better to reserve this type of development for the end of this playback stage. You ensure that the data model is in place, the functions of the user interface are approved, and all the remaining process application interactions and integrations are complete. Now the enhancements can be done without fear that rework might be needed later because of a change in requirements to the items in the prior playbacks. Changes can happen, but the likelihood is that those functional requirement changes are handled in Optimization and not within the development cycle because of consensus to move to this stage of development.

The new integrations and added features and functions are in the Hiring Requisition Process coaches, so it is now necessary to get a sign-off for the process application. Conduct a playback of your process to review the added enhancements.

Part 3: Conduct Playback 2: Integration

In Playback 2: Integration, the process interactions, and integrations are implemented so that the process can have all the functions that are needed to complete any process activity. This playback ensures that all business data is flowing, regardless of the source of that data. One example might be retrieving data from the different systems within an organization. So it is important to tap into that data, and that is why integrations play a vital role in having a full and robust process application. Other process interactions involve events within the process model. These intermediate events or even start events might need unique event handlers, such as listeners for messages that trigger an event. It would be the appropriate time to implement, test, and finalize all remaining process application interactions.

Validate that the decision service and business rules produce the expected results, which depend on the business data input into a coach.

Part 4: Conduct Playback 3: Hardening processes and services

In this playback, peer testing of all functions, all interface design, and all use cases can be done. The process application error handling allows the development team to fix any process application problems that were not detected earlier. After testing and approval, the move is made to deploy the process application to a test environment that is a user production environment so the process application can be monitored and tested under business conditions.

The last playback consists of validation of the error handling that you implemented in the process application. This milestone represents the last chance to change your Process Application before deployment to QA, Test, or Production environments. Hardening requires you to thoroughly examine all aspects to determine how your processes and services might break, and to design ways to catch these errors so your users do not encounter these errors. Time that is spent in this playback is critical to the success of the project, so do not treat Playback 3 as an afterthought.

Exercise review and wrap-up

This exercise provided an overview of all the playbacks from Playback 0 to Playback 3.

Appendix B. Modeling and implementation artifacts

The following pages contain the service types and what tools and components are available to build each service type.

Service type	Description
 Client-Side Human Service	Use a client-side human service when you want to create an interactive service. A human service is the only type of service that can contain coaches and postpones. Human services generate tasks in IBM Process Portal. Note: A human service is the only type of service that can call other nested human services.
 Service Flow	Use a service flow when it is necessary to coordinate with other nested services or to manipulate variable data. For example, if you want to implement data transformations or generate HTML for a coach, you can use a service flow. Service flows do not include Java or web service integrations directly. You can call a service flow from any other type of service, and a service flow can call other nested services.
Decision service 	Use a decision service when you want a condition to determine the implementation that is started. For example, when a certain condition evaluates to true, IBM Business Process Manager implements the JavaScript expression that you provide. Decision services cannot include Java or web service integrations directly. You can call a decision service from any other type of service, and a decision service can call other nested services.
 External Service	Use an external service when you want to integrate with an external system. An external service is the only type of service that can contain a Java or web service integration. You can call an integration service from any other type of service, and an integration service can call other nested services. Importing a Swagger file streamlines the creation of external services. You can also use an external service to call an Advanced Integration service. Use an Advanced Integration service when you want to integrate with a service that is created in IBM Business Process Manager Advanced.

Component icon	Available with	Description	
Coach		Use to create the interfaces for your human services. You can use coaches to easily add the fields, buttons, and other controls to enable users to participate in a business process. For more information, see Building coaches .	
Server Script		Use when you want to write JavaScript to run on the Process Server in the service context. The Server Script component is useful for parsing through variables and running programmatic commands. You can also use it to bind blocks of formatted text (for example, HTML, XML, or XSLT) directly to a service variable. This type eliminates the necessity to store large blocks of text in default values for variables.	
Intermediate Event		Only human service	Runs before the client-side human service completes. Can be used as a stay on page event or a postpone event. When attached to a service step, the intermediate event can be used to catch errors event.
Decision Gateway		All service types	Use to model a point in the process execution where only one of several paths can be followed, depending on a condition.
End event		All service types	Use to end service execution. For services that contain multiple paths, each path requires its own end event. Note: An end event is automatically included each time that you create a service.
Note		All processes and service types	Use to add information or document information about the process or service. Adding notes helps other developers understand your design.
Error end event		All service types	Use to purposely produce an error and end processing. You might, for example, use a Throw Exception component if you return too many rows from a database (over a limit that is normal and would bog down the server).

Error intermediate event

All service types

Use to listen for exceptions from the service component to which it is attached.

Service step

All service types

Use to incorporate other services in your current service. Nested services are generally defined to run specific, repeatable functions such as exception handling routines, integration with outside systems, or data manipulation. Nested services are often used in multiple process applications and likely exist in a toolkit. Note: Human and Ajax services cannot be nested.

Content Event

All processes

Use to integrate with an Enterprise Content Management system.

An ad hoc activity has no input wires and is started as required by knowledge workers or according to predefined preconditions, rather than by a predefined process flow. Such activities can be required or optional, and they can be defined as repeatable or to run at most once.

Ad hoc component icon	Available with	Description
	All service types	An ad hoc activity has no input wires and is started as required by knowledge workers or according to predefined preconditions, rather than by a predefined process flow. It can start automatically, as required, run once, or no precondition.
	All service types	This ad hoc activity must start manually.
	All service types	This ad hoc activity is optional.
	All service types	This ad hoc activity is manual and repeatable.
	All service types	This ad hoc activity has a precondition.

This table shows the state and its icon for each activity in Process Portal.

Status icon	Bucket name	Description	Option type
	Ready	Manual, required, not yet triggered, ready to be triggered.	Required
	Ready-Optional	Manual, optional, not yet triggered, ready to be triggered, does not have to be triggered.	Optional
	Working	Anything that is triggered but not yet completed.	Required/Optional
	Completed	Anything that is completed normally.	Required/Optional
	Waiting	Required, blocked on precondition. If manual, then go to Ready bucket; if automatic, then go to Working bucket.	Required
	Waiting-Optional	Optional, blocked on precondition. If manual, then go to Ready bucket; if automatic, then go to Working bucket.	Optional
	Failed	Go to Completed bucket, regardless of activity settings. If restarted, then all of the above apply.	Required/Optional

Appendix C. Challenge exercises

The following appendix provides a consolidation exercise for the Process Implementation course. It is optional and can be completed at the end of the course (if you are done with the core exercises earlier than other members of the class), or after the training class is complete.

Consolidation exercise

The **Post Position** activity in the sample process is not implemented. While completing this exercise, you can use your existing knowledge to provide a service for this activity.

The business determined that on completion of the hiring request, the details of the new job should be posted in the Positions table. The details of this table are contained in Appendix D: Data dictionary.

To assist you in your testing of the insertion of details into the Positions table, you create a separate service to read the rows that are contained in that table.

To accomplish this task, you use the Records control type. Details of how to use this control can be found in the Help file. In IBM Process Designer, click **Help > Help** to activate the IBM Business Process Manager product documentation.

Key steps

1. Insert a new record into the **Positions** table by using the details that are captured in the process.
2. Ensure that the **ID** field of the **Positions** table contains a unique value that is related to the instance ID of the process.
3. Set the **Status** field of the **Positions** table to **1** signifying that the new position is active.

Appendix D. Data dictionary

Training database

The JNDI is `jdbc/TrainingDB`.

The training database that is used in the foundation courses 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



© Copyright International Business Machines Corporation 2018.