Course Exercises Guide

Process Implementing with IBM Business Process Manager Standard V8.5.7 - I

Course code WB823 / ZB823 ERC 2.0

IBM Training

October 2017 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.

Contents

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™	CICS®	DB™
DB2®	developerWorks®	Express®
FileNet®	IBM Connections™	IBM MobileFirst™
ILOG®	PartnerWorld®	PureApplication®
Rational®	Redbooks®	WebSphere®

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

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

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

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

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

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

Exercises description

This course includes the following exercises:

- Exercise 1. 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.

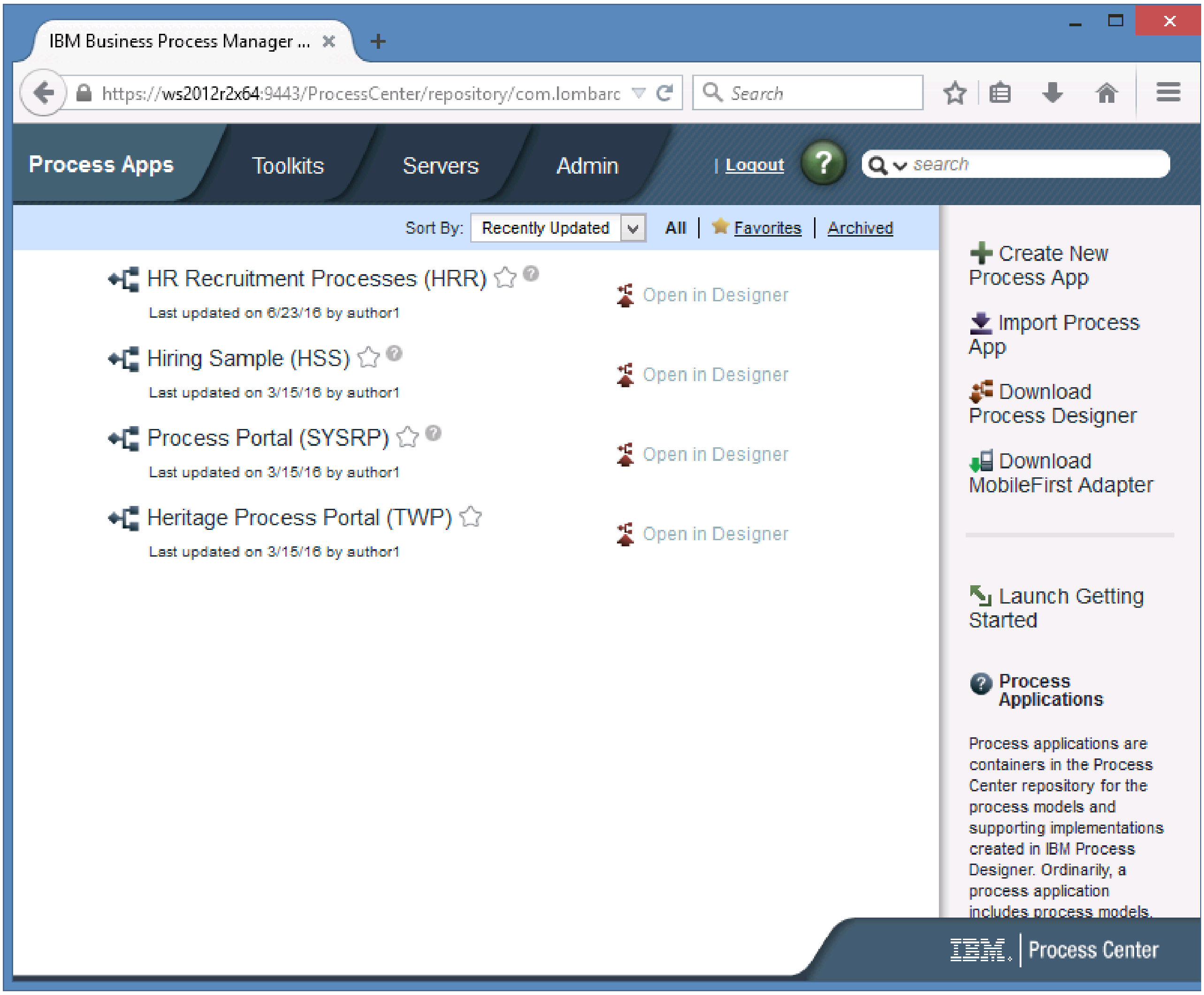
Most exercises include required sections, which should always be completed. It might be necessary to complete these sections before you can start later exercises. Some exercises also include optional sections that you might want to complete when you have sufficient time and want an extra challenge.

Important

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

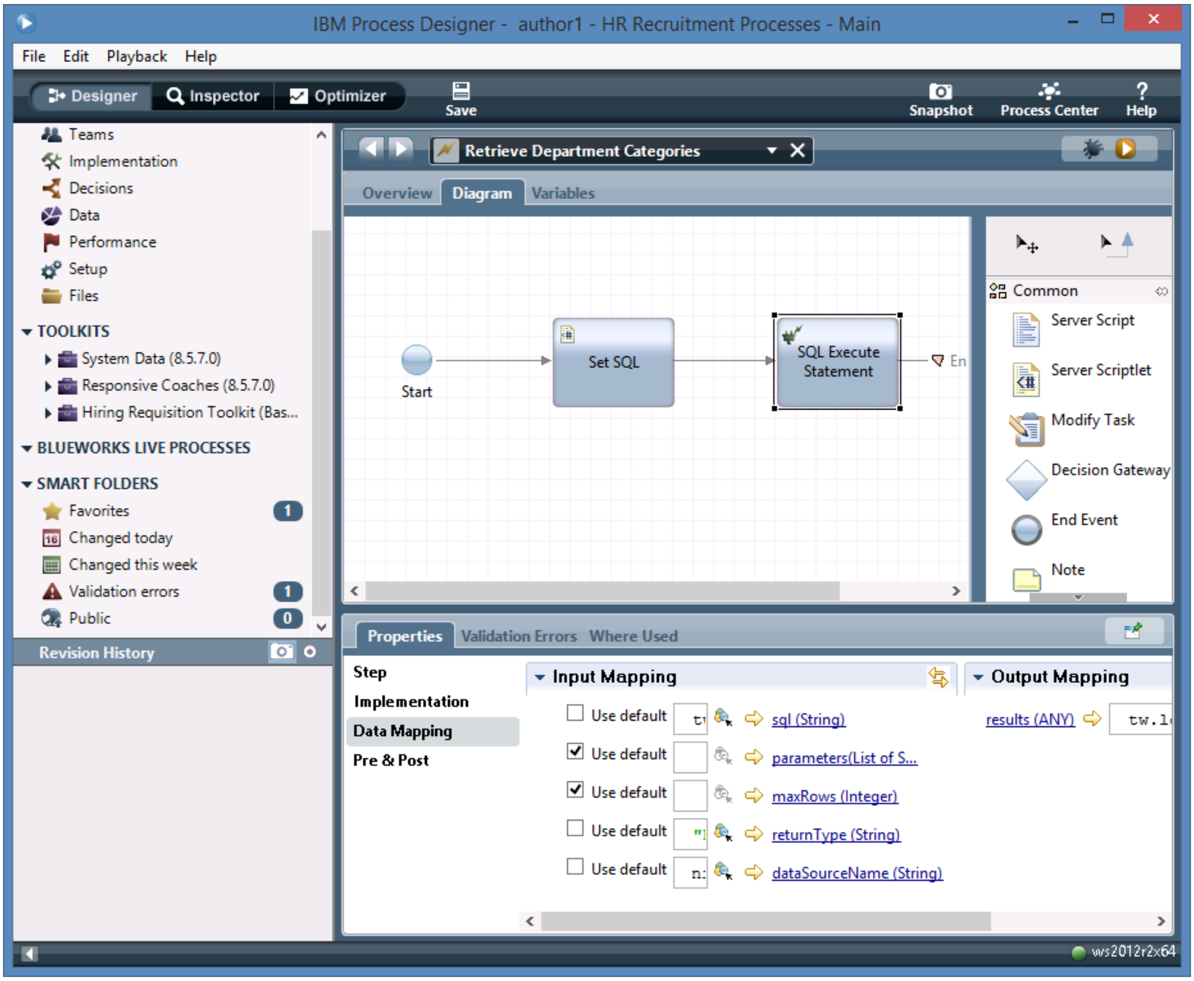
Important

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



This browser-based tool uses your desktop browser to create most of your process application. The web Process Designer is cross-browser compatible, meaning the Designer runs on multiple operating systems. For more information on browser and system compatibility, go to the **Detailed System Requirements Supported Software** tab for IBM Business Process Manager Standard 8.5.7.

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


IBM Process Designer - author1 - HR Recruitment Processes - Main
File Edit Playback Help
Designer Inspector Optimizer Save
Snapshot Process Center Help
Teams
Implementation
Decisions
Data
Performance
Setup
Files
TOOLKITS
System Data (8.5.7.0)
Responsive Coaches (8.5.7.0)
Hiring Requisition Toolkit (Bas...
BLUEWORKS LIVE PROCESSES
SMART FOLDERS
Favorites
Changed today
Changed this week
Validation errors
Public
Revision History
Retrieve Department Categories
Overview Diagram Variables
Start
Set SQL
SQL Execute Statement
Common
Server Script
Server Scriptlet
Modify Task
Decision Gateway
End Event
Note
Properties Validation Errors Where Used
Step
Implementation
Data Mapping
Pre & Post
Input Mapping
Output Mapping
Use default
sql (String)
parameters(List of S...
maxRows (Integer)
returnType (String)
dataSourceName (String)
results (ANY)
ws2012r2x64

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 Web Process Designer.

Objectives

After completing this exercise, you should be able to:

- Start IBM Business Process Manager
- Create a process application by using Web 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, PI files, and others. The course lab files can be found in the following directory:

`C:\labfiles` for the Windows platform

`/usr/labfiles` for the Linux platform

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

Note

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

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`	`web1sphere`
Windows 2012 R2	`administrator`	`web1sphere`
IBM Process Designer	`author1`	`author01`
IBM Business Process Manager WebSphere Application Server cell administrator	`administrator`	`web1sphere`

Note

For IBM Business Process Manager on Cloud users

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

The data in the data dictionary must also be added to the database in 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 Web 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

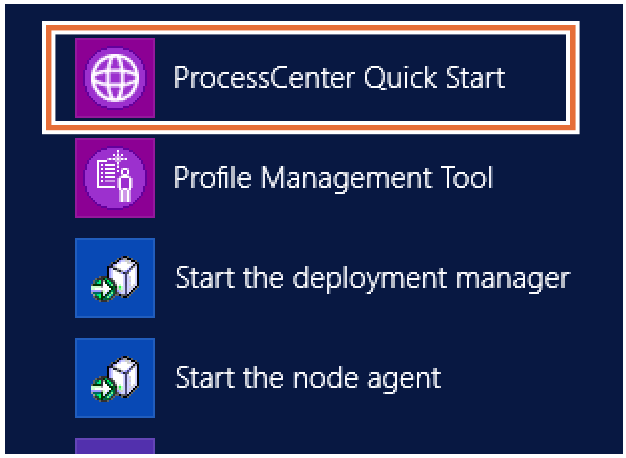
All three server configurations must be started in order, starting with the Deployment Manager profile, then the Node Agent profile, and followed last by the Deployment Environment. To accomplish this task quickly, IBM Business Process Manager provides a Quick Start routine to run the server start in order.

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

Note

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

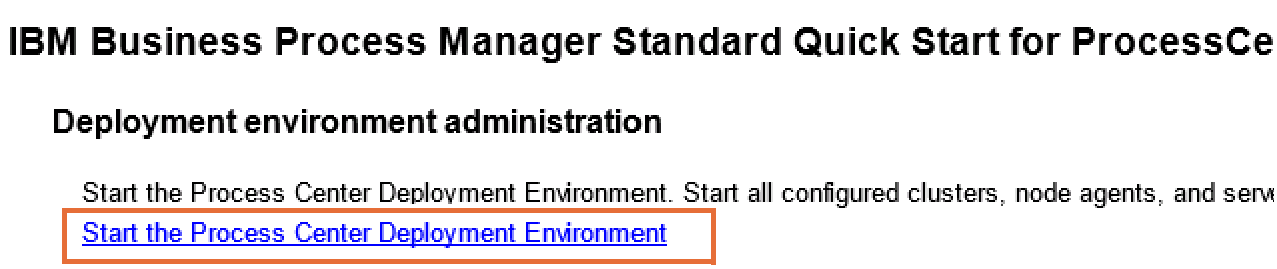
__ a. Click **Start > Down Arrow > IBM > ProcessCenter Quick Start**.



Important

Some of the artifacts like the start menu folder and the Process Designer are labeled 8.5 even though the product version is 8.5.7. You can verify the server version by the System Data Toolkit version that is installed on the Process Center later in the course.

__ b. In the **IBM Business Process Manager Quick Start** window, click the **Start the Process Center Deployment Environment** link.

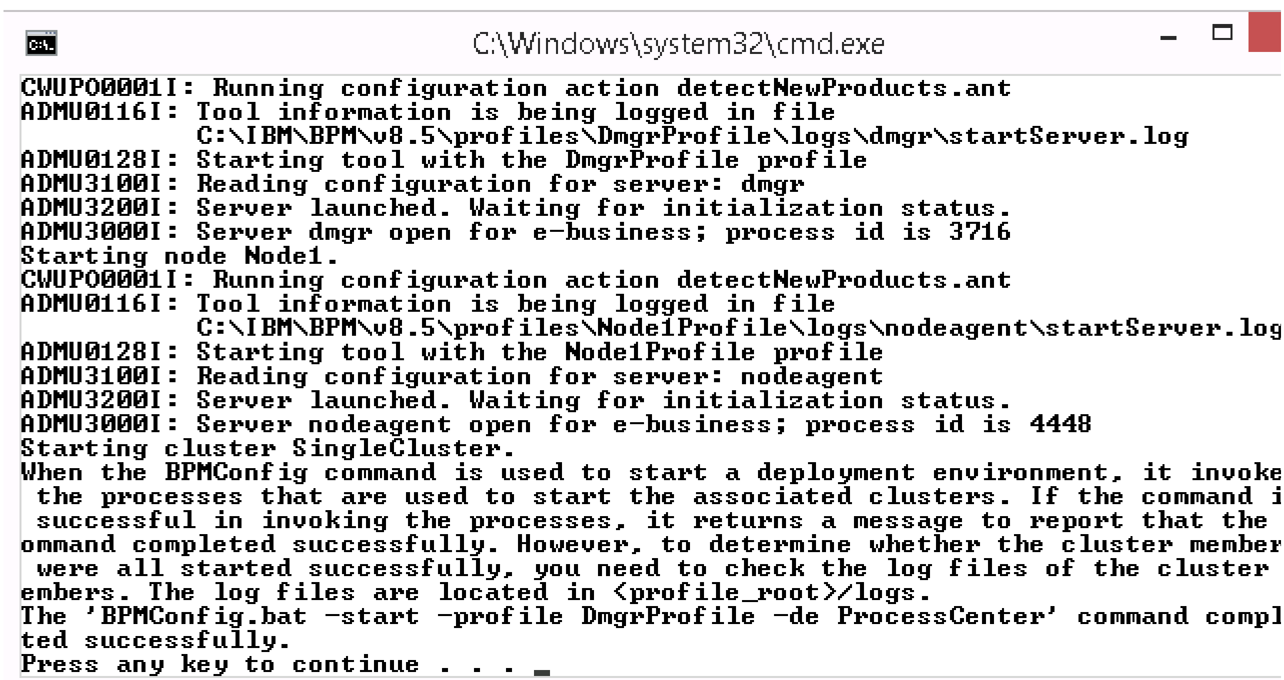


It takes a few moments to start the deployment environment.

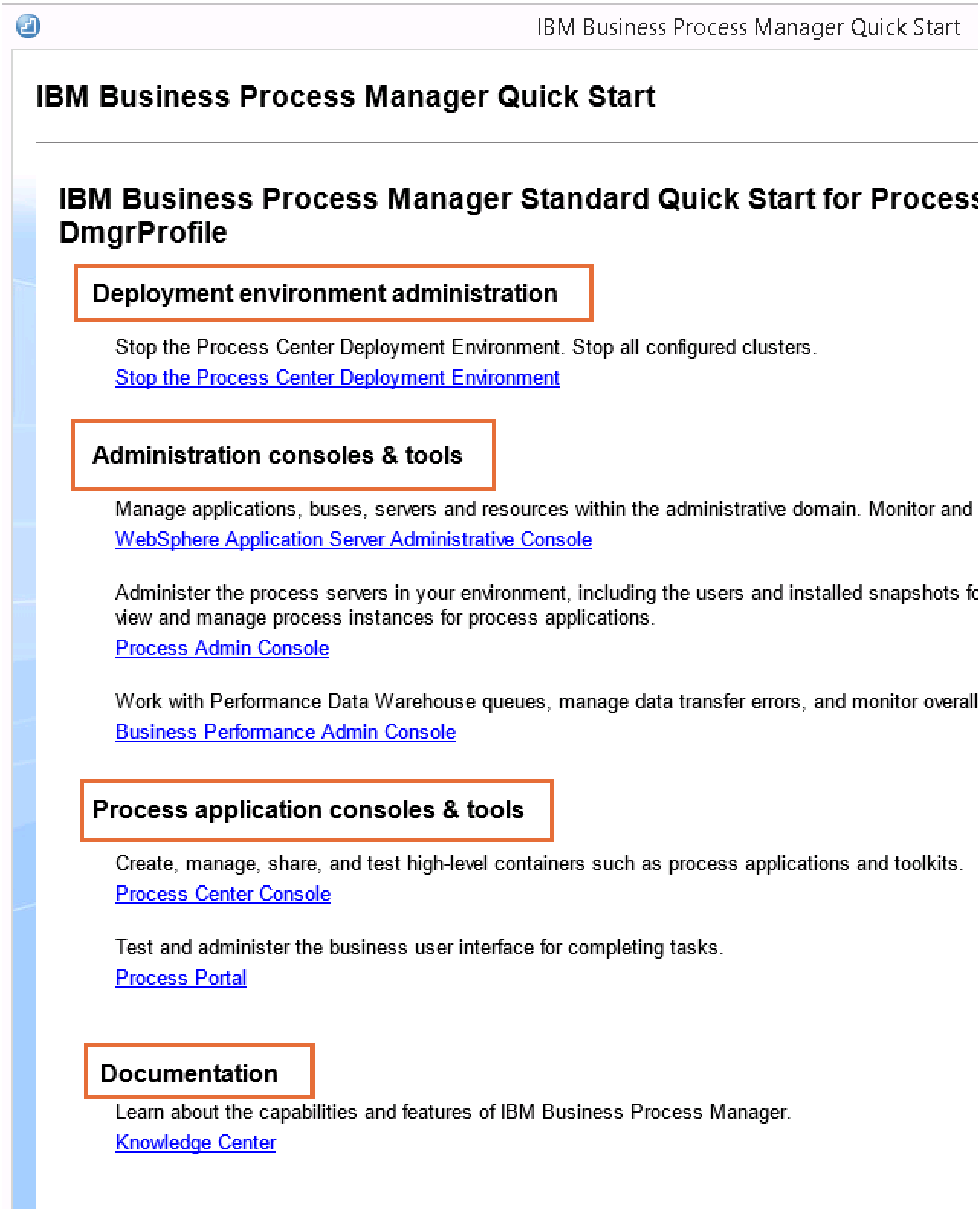
Important

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

__ c. A command window opens. Press any key when it prompts **Press any key to continue**.



__ d. 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 fc
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

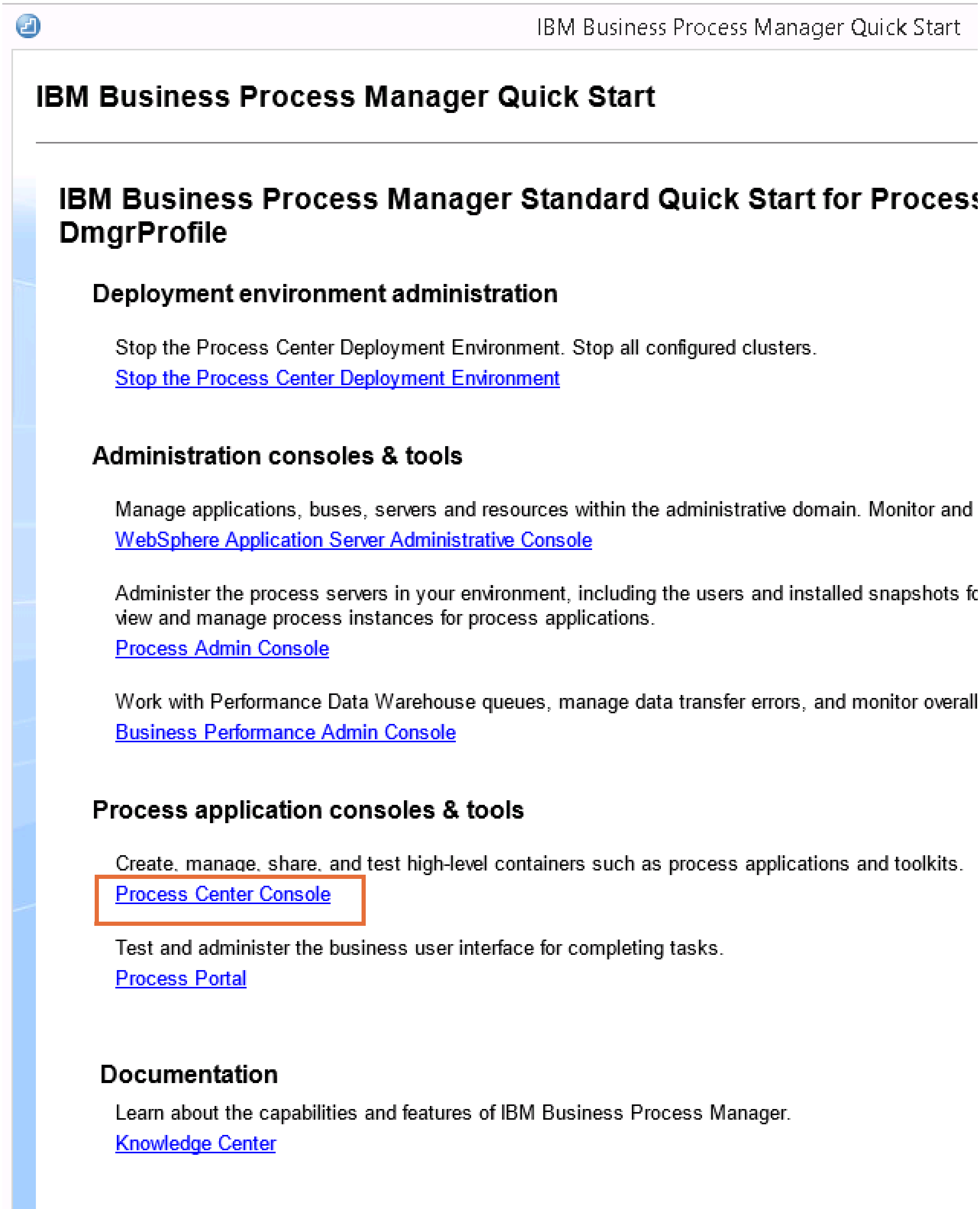
Note

For IBM Business Process Manager on Cloud users

Click the **Launch in the Process Portal** tile of your IBM Business Process Manager on Cloud home screen to access the links to the tools that are shown in the Quick Start menu.

__ 2. Start the **IBM Web Process Designer**.

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

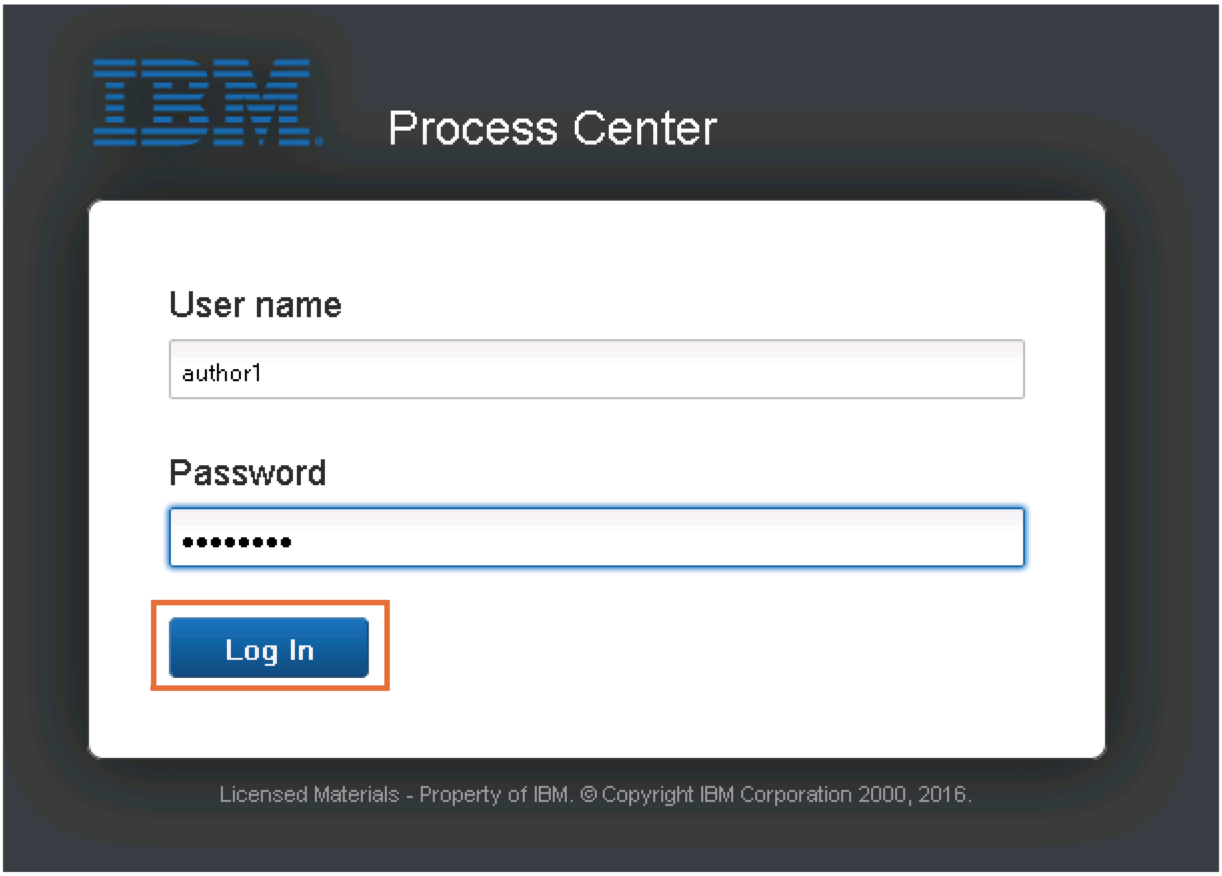


__ b. Minimize the **IBM Business Process Manager Quick Start** browser window and the **ProcessCenter Quick Start** window so that you can use the page to access the **Process Portal** later in the exercises.

__ 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**.

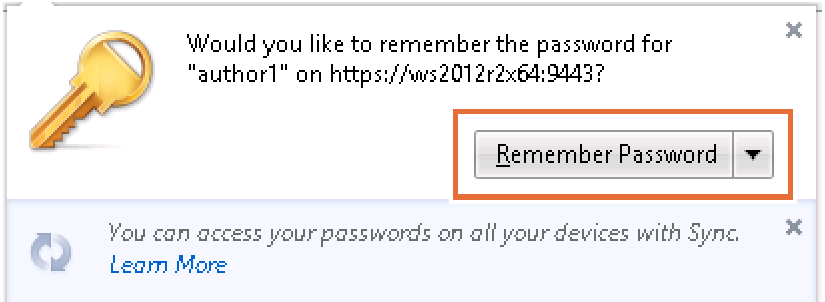
__ e. Click **Log In**.



Important

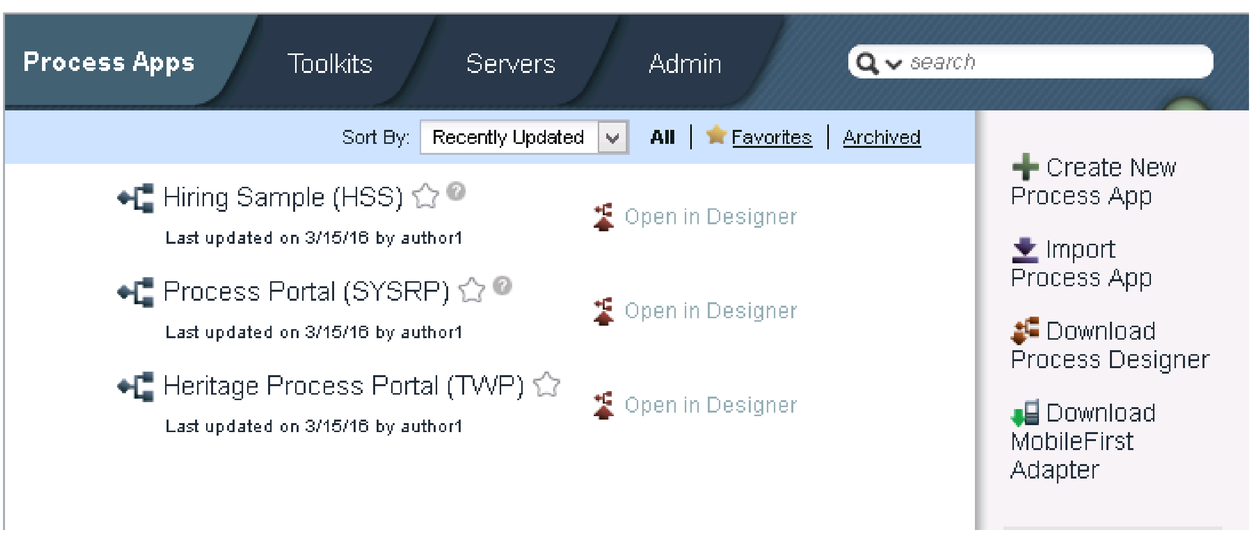
At certain times when working inside the web-based Process Designer, the session times out due to inactivity. If at any time during this course you see an Authentication Required prompt, insert the same credentials that are shown (`author1` and `author01`) and click **OK**.

__ 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.



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 Web 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, and numerous ways are available to accomplish the same thing.

Note

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

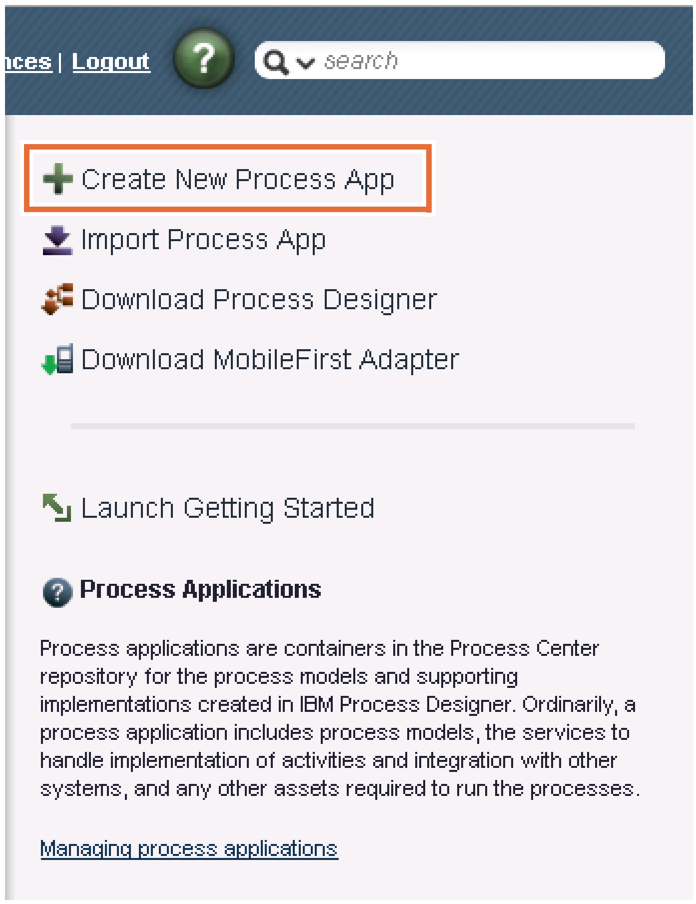
Part 2: Create a process application

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 Web 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.

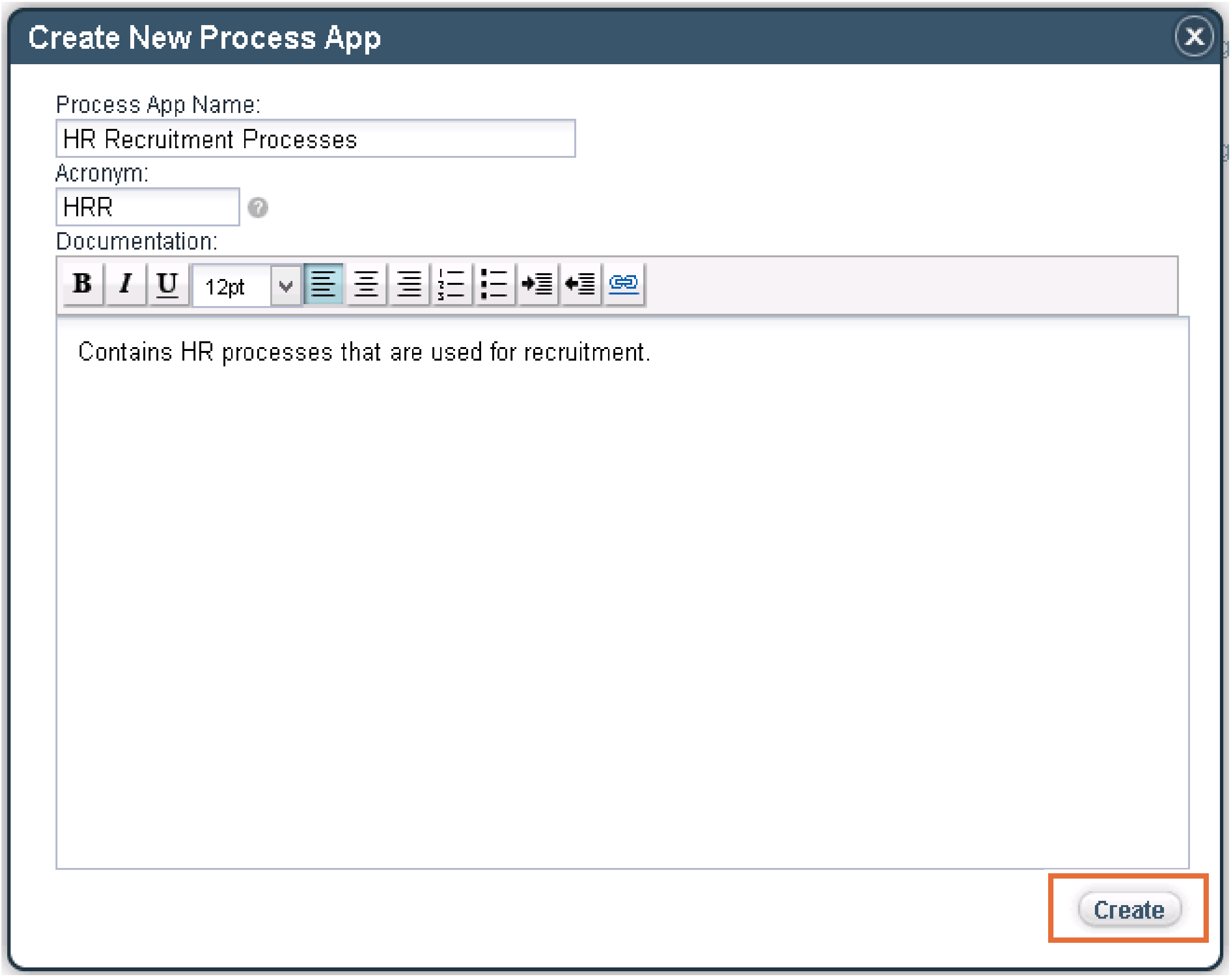
__ a. Click the **Create New Process App** link on right pane of IBM Web Process Designer.



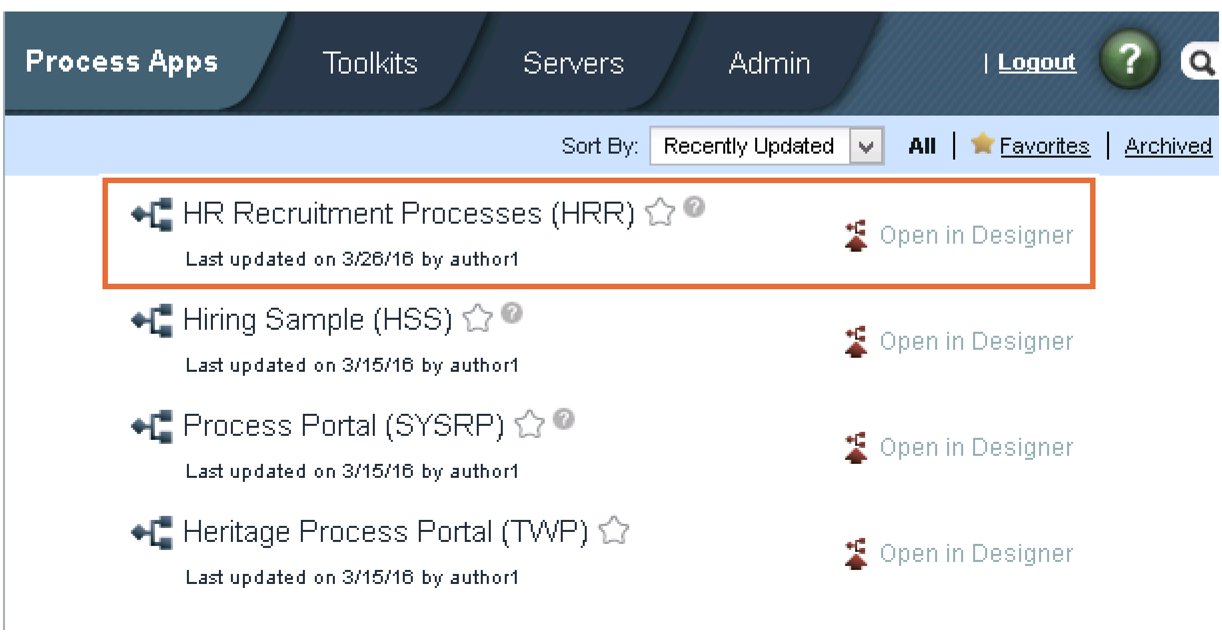
__ b. As soon as the **Create New Process App** dialog box is displayed, enter following values:

- 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**.

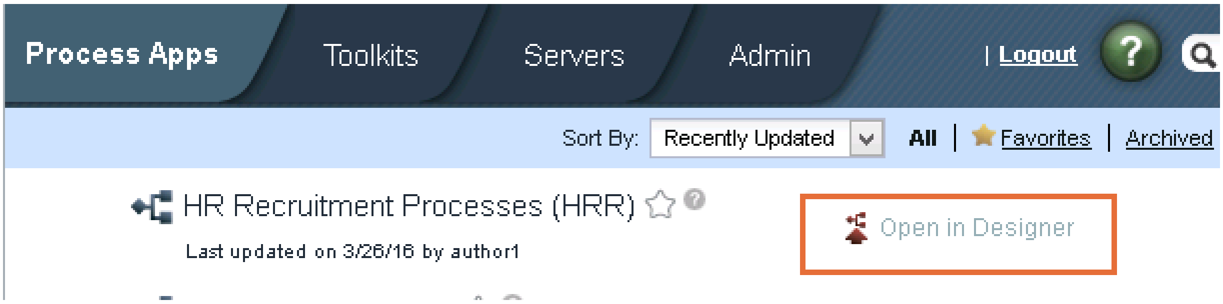


The process application is created and is now in your list of process apps.

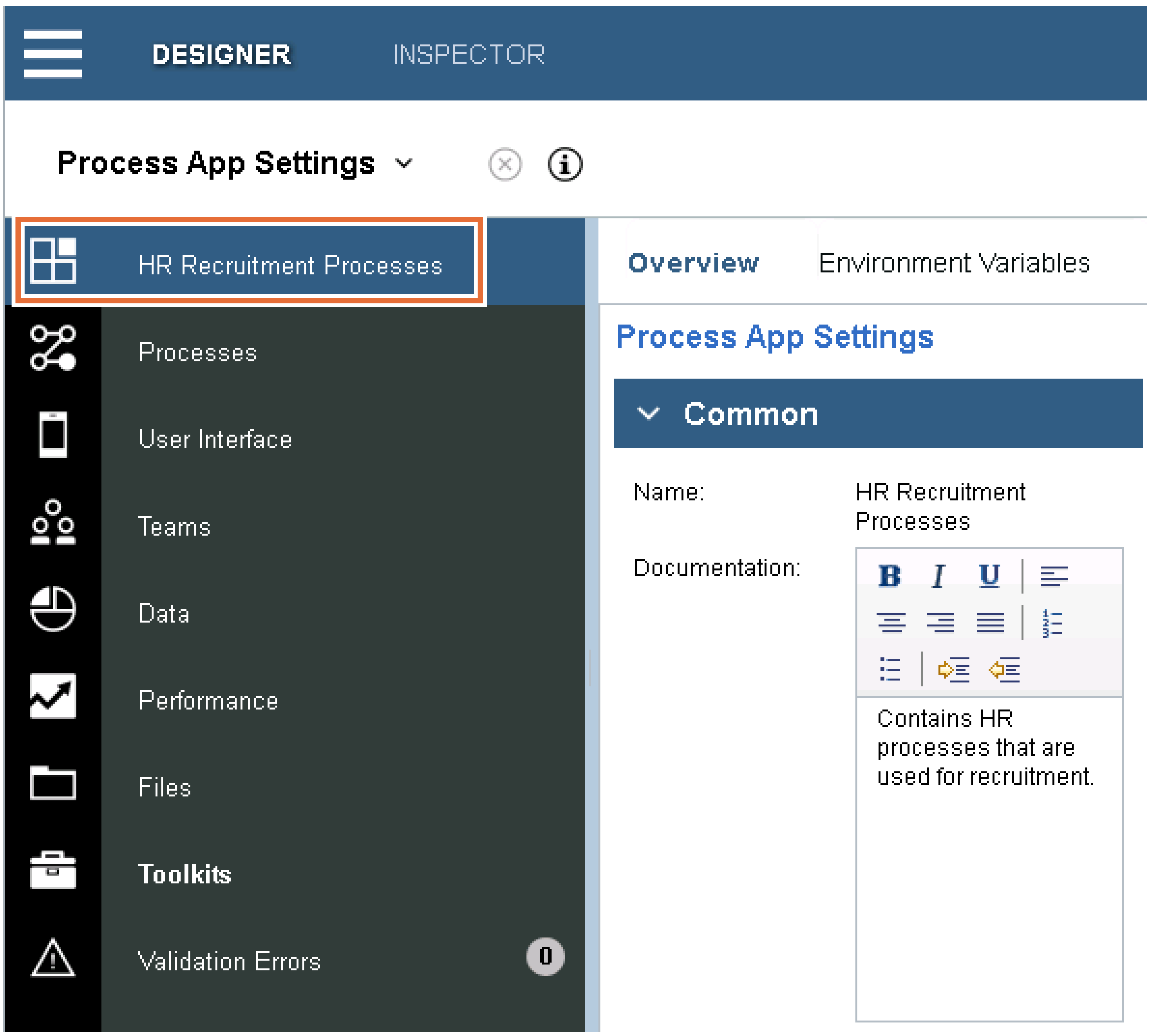


__ 2. Open the HR Recruitment Processes process application.

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



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



Important

The highlighted area shows the process application that you are editing. Notice that the initial screen is also an opportunity to edit the process application settings such as name, acronym, description, and authorization. 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 Web 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.

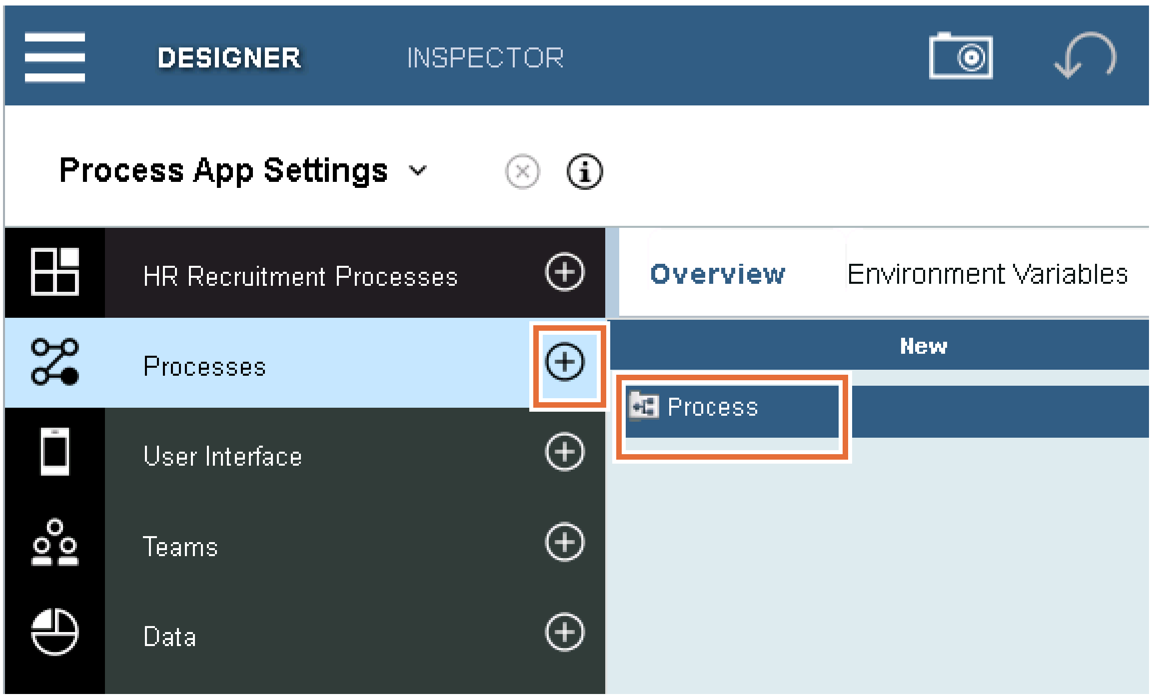
To create a process, you identify the user activities that are needed to complete the process and then group those activities into a process. You identify the documents that are used and the teams of users that work on the activities. You also identify the conditions that are required to start and complete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

Part 3: Create a process

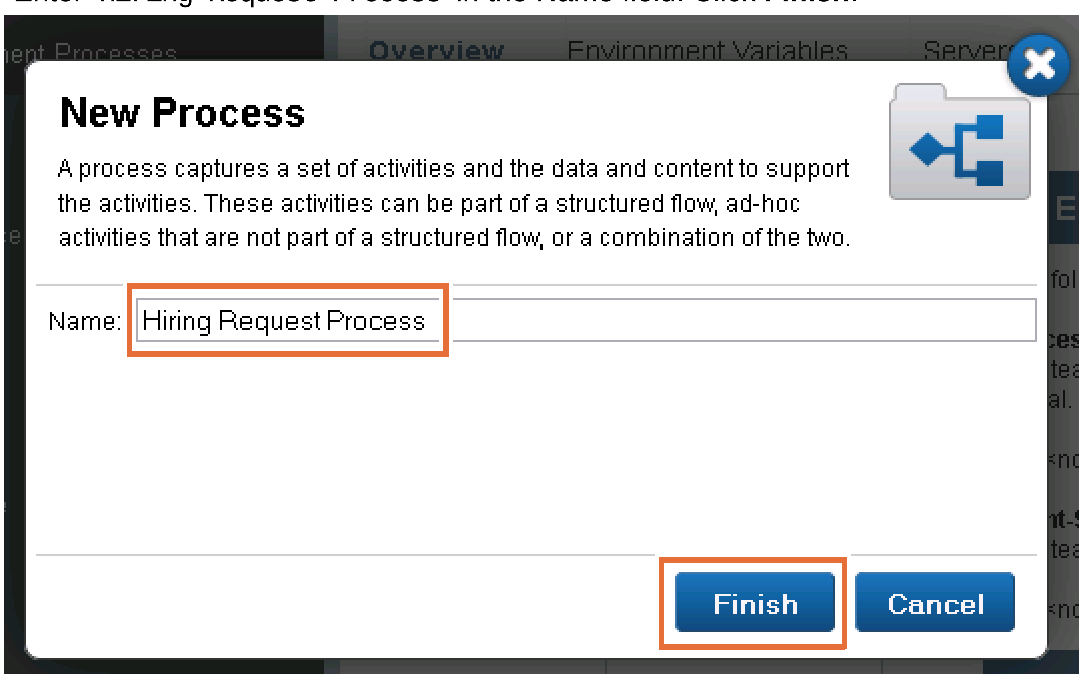
A process defines the activities that are needed to resolve a specific business problem. A process also defines who works on these activities and the steps they take to resolve the problem. At run time, a business user works with an instance of a process.

__ 1. Create a process.

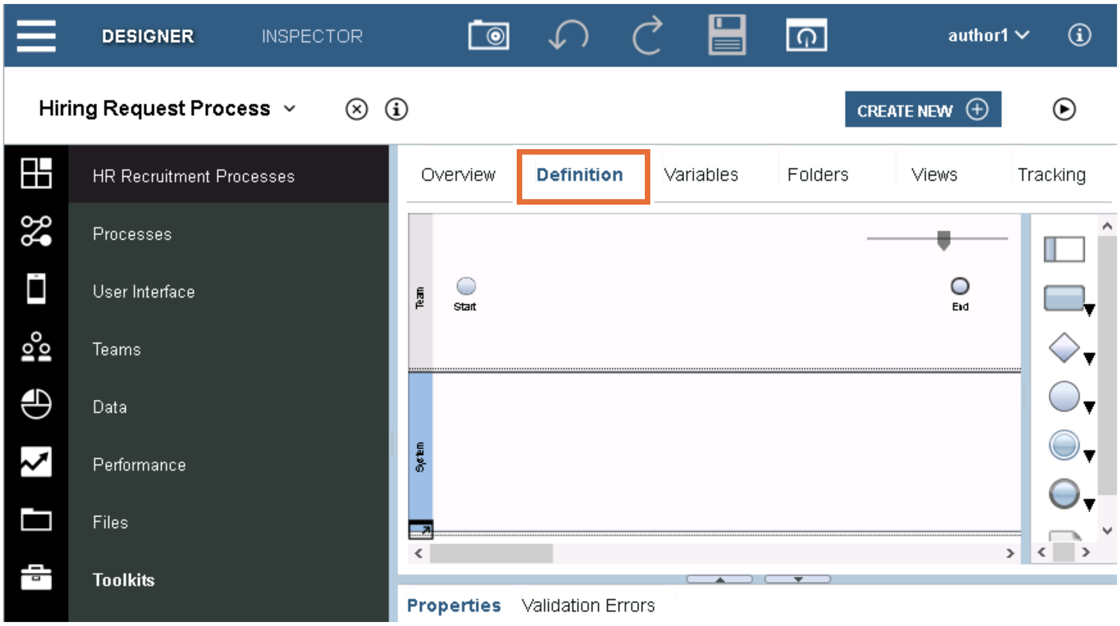
__ a. Return to the web Process Designer client application. From the process library, click the (**+**) plus sign next to Processes and select **Process**.



__ b. Enter `Hiring Request Process` in the Name field. Click **Finish**.

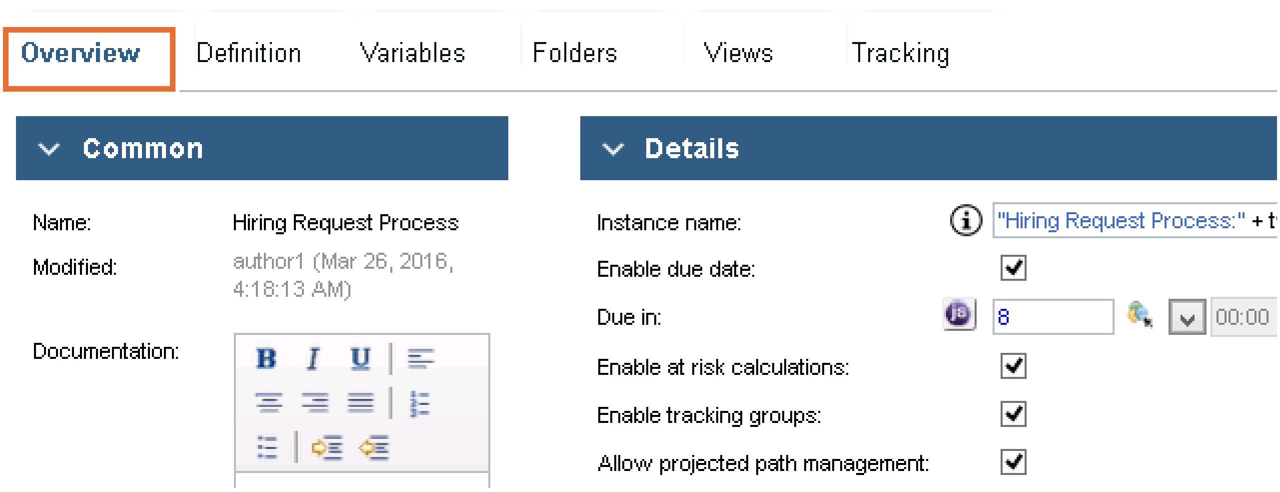


The process is created, and the initial view is shown under the **Definition** tab. By default you have a process with a team lane, a system lane, a Start event, and an End event. Nothing is special about these events. They are created by default because you must have a Start event and an End event when you create a process in IBM Business Process Manager Web Process Designer.

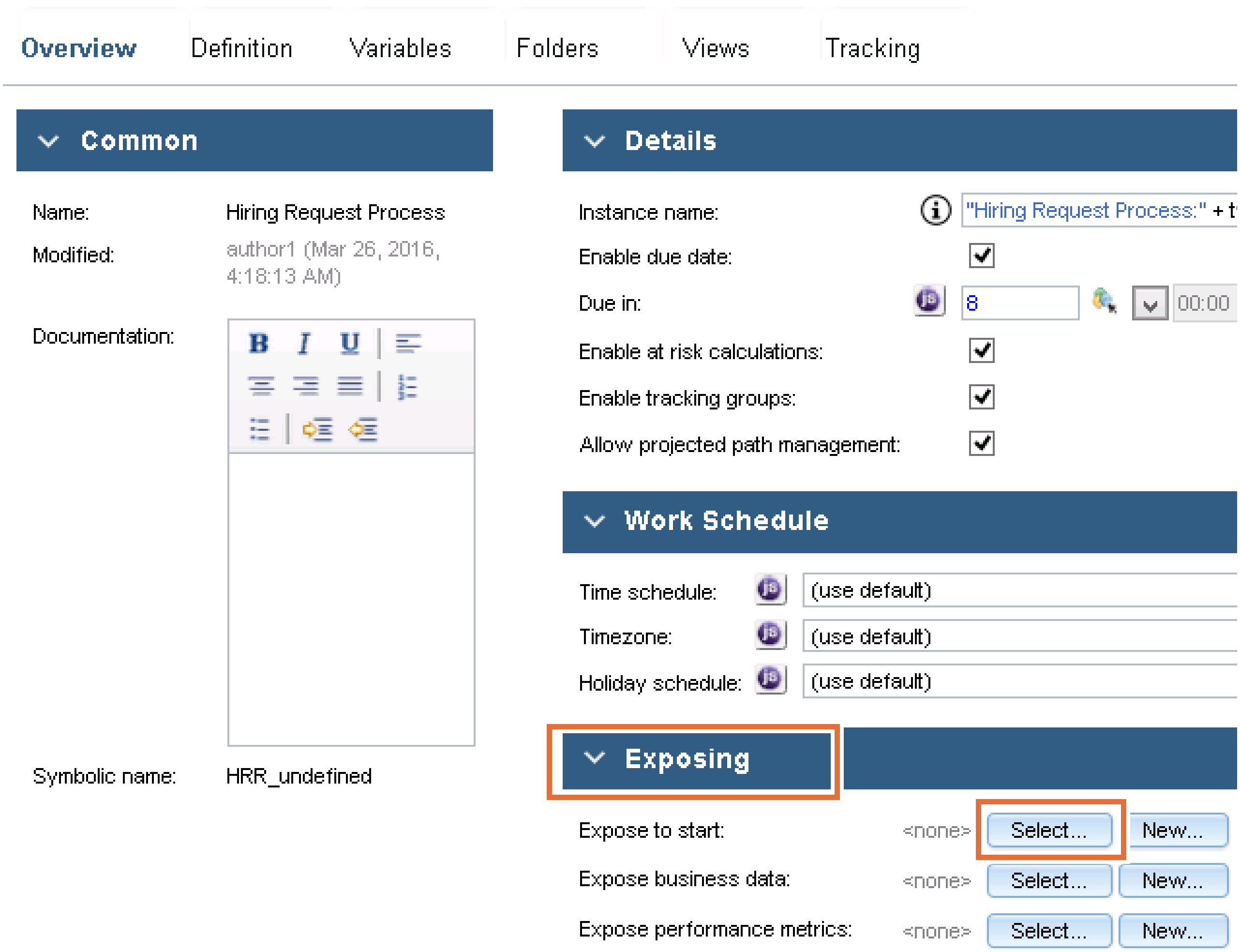


__ 2. Expose the process to **All Users** so that all the users can start the process.

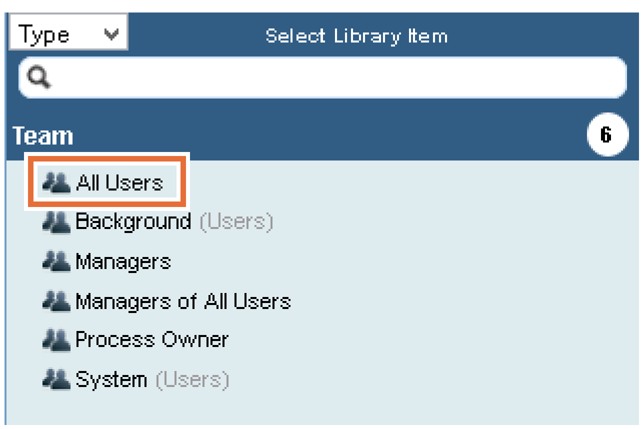
__ a. Click the **Overview** tab.



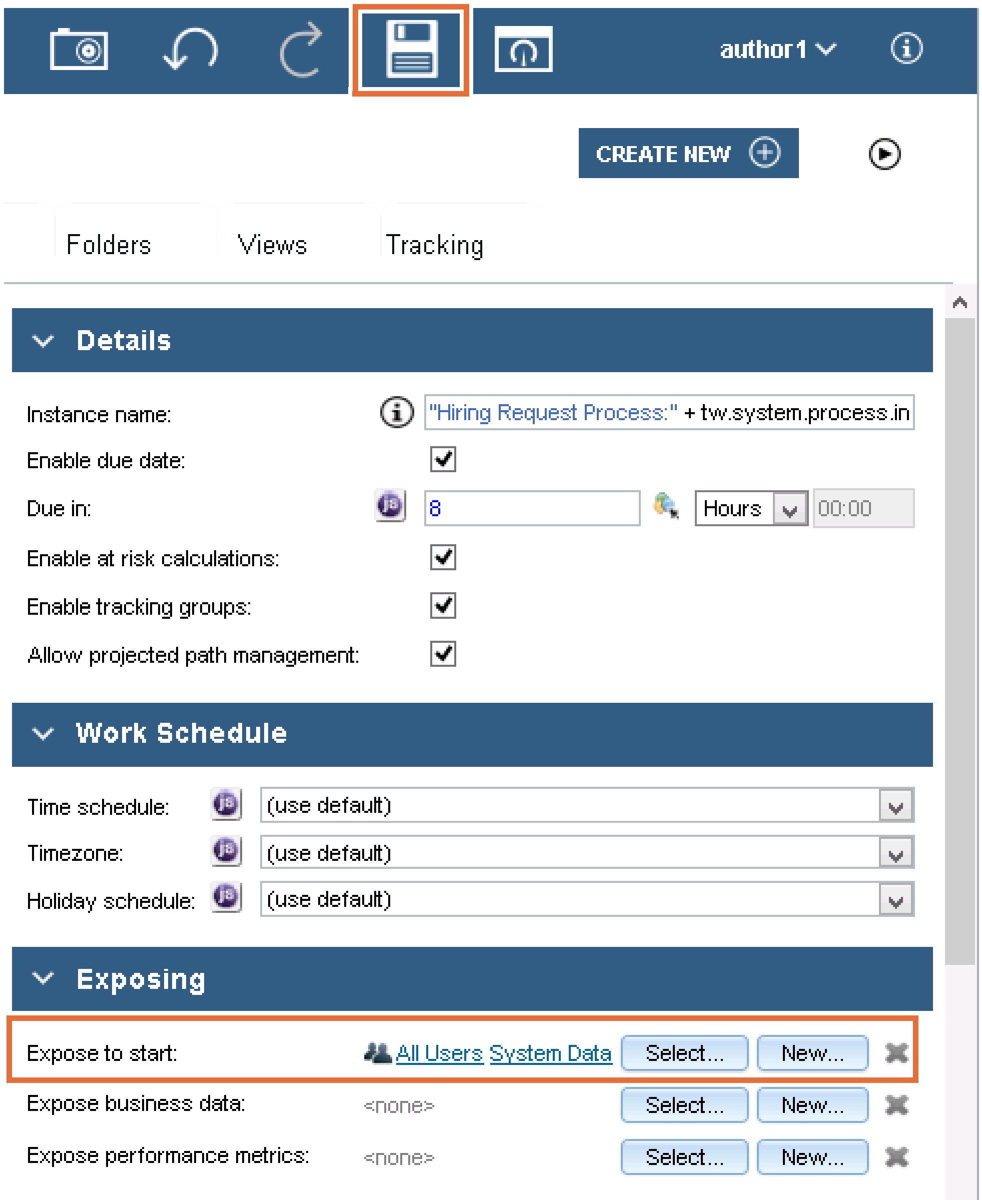
__ b. Under the Exposing section, click **Select** for the Expose to start option.



__ c. Select **All Users**.



__ 3. Click **Save** to save your changes.



Hint

You can also use the short cut **Ctrl+S** to save your work.

Note

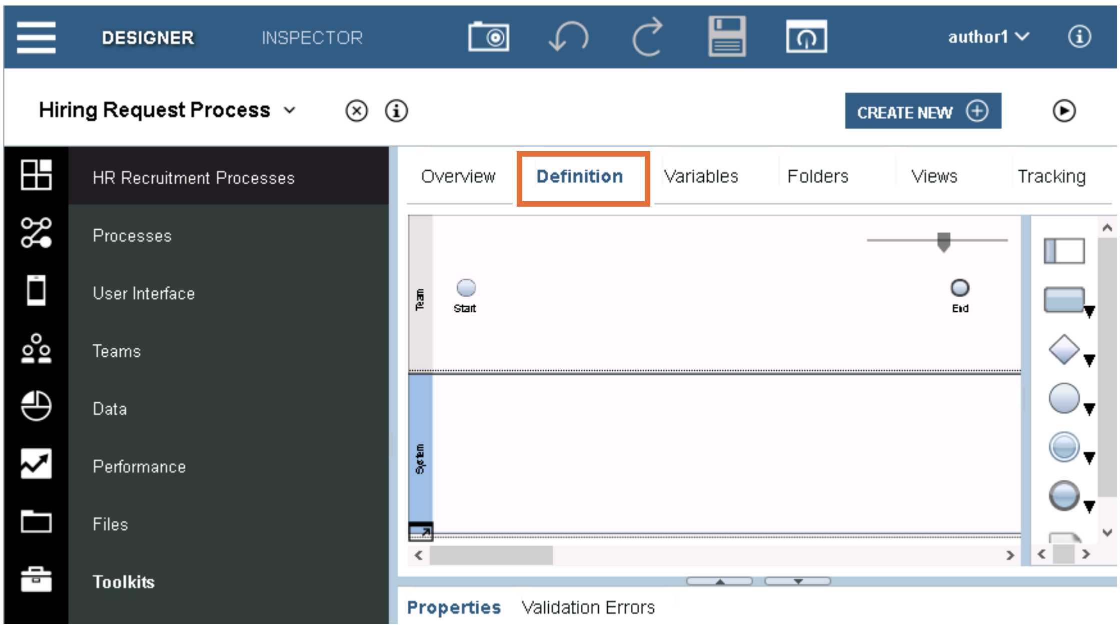
You can assign a team whose members can start a process, or an instance owner's team whose members can work with the process at run time in the Process Portal. Currently, you assign all users to the HR Recruitment Processes process, and in a later unit and lab exercise in this course, you learn how to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

Part 4: Create process activities

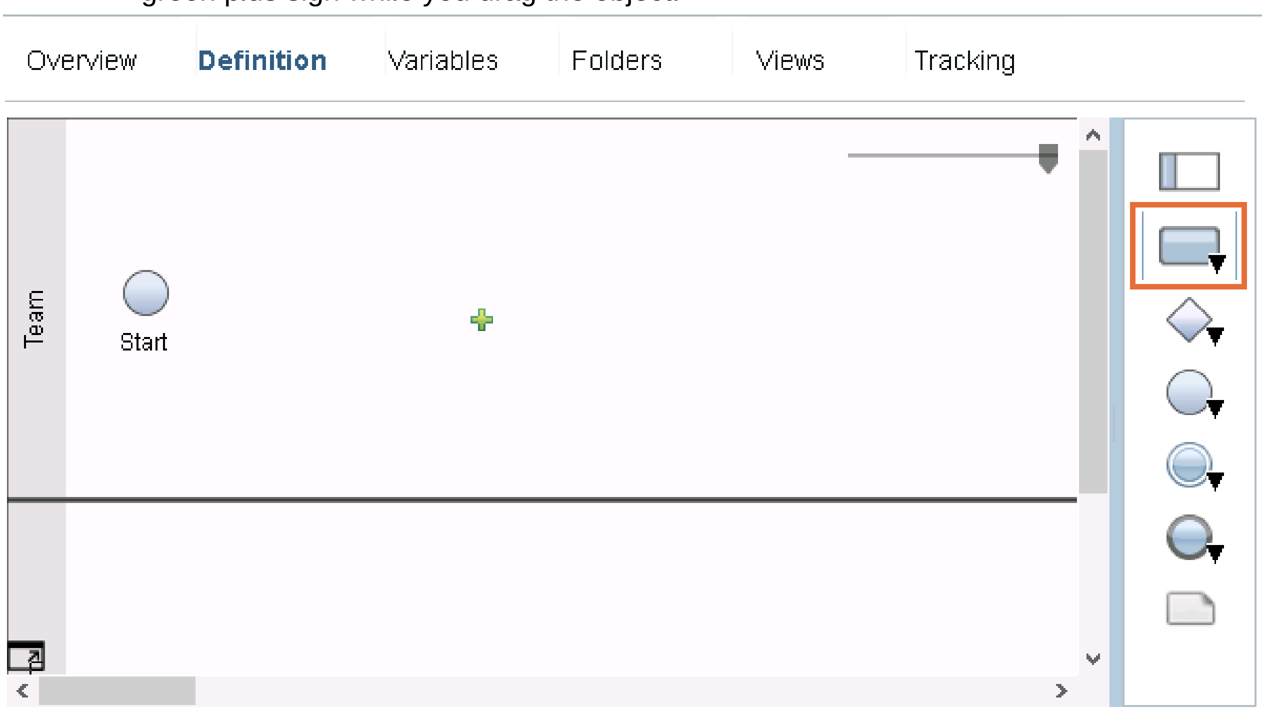
An activity in a process is a discrete task that a person or a system completes as part of that process. Typically a process has a number of activities. Initially you add an activity in a process and classify it as required or optional. Next, you repeat the process to create three required activities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

__ 1. Create a required process activity that is named **Submit Hiring Request**.

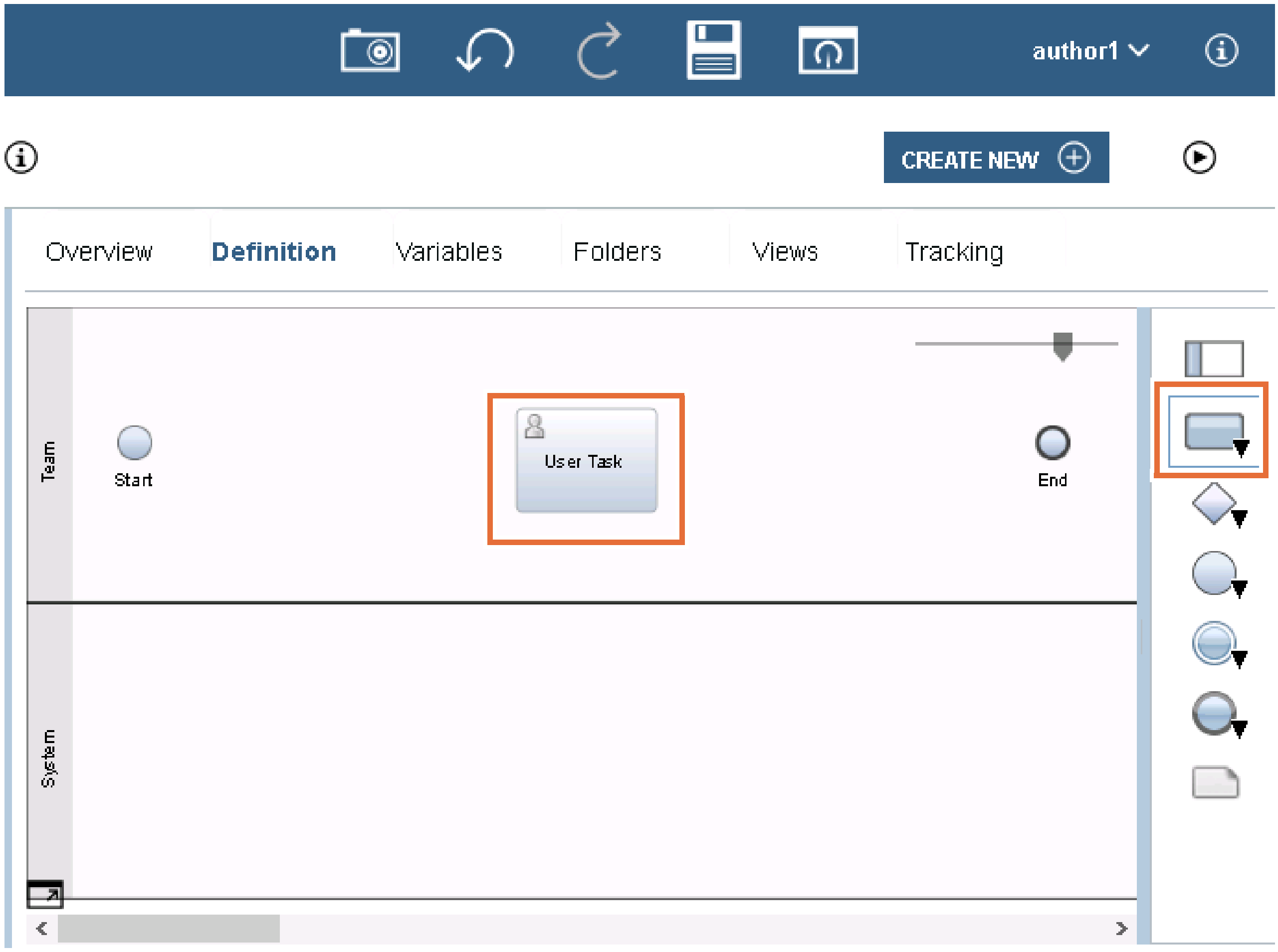
__ a. From the Hiring Request Process, click the **Definition** tab.



__ 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.

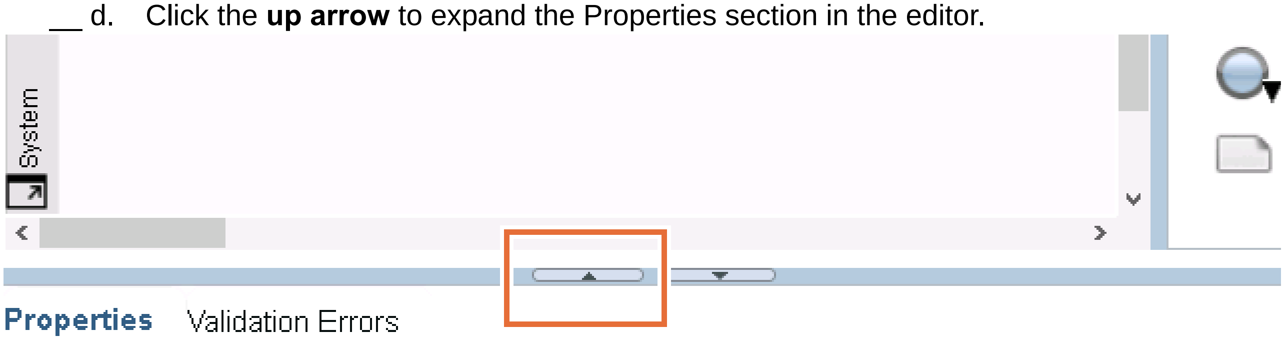


Hint

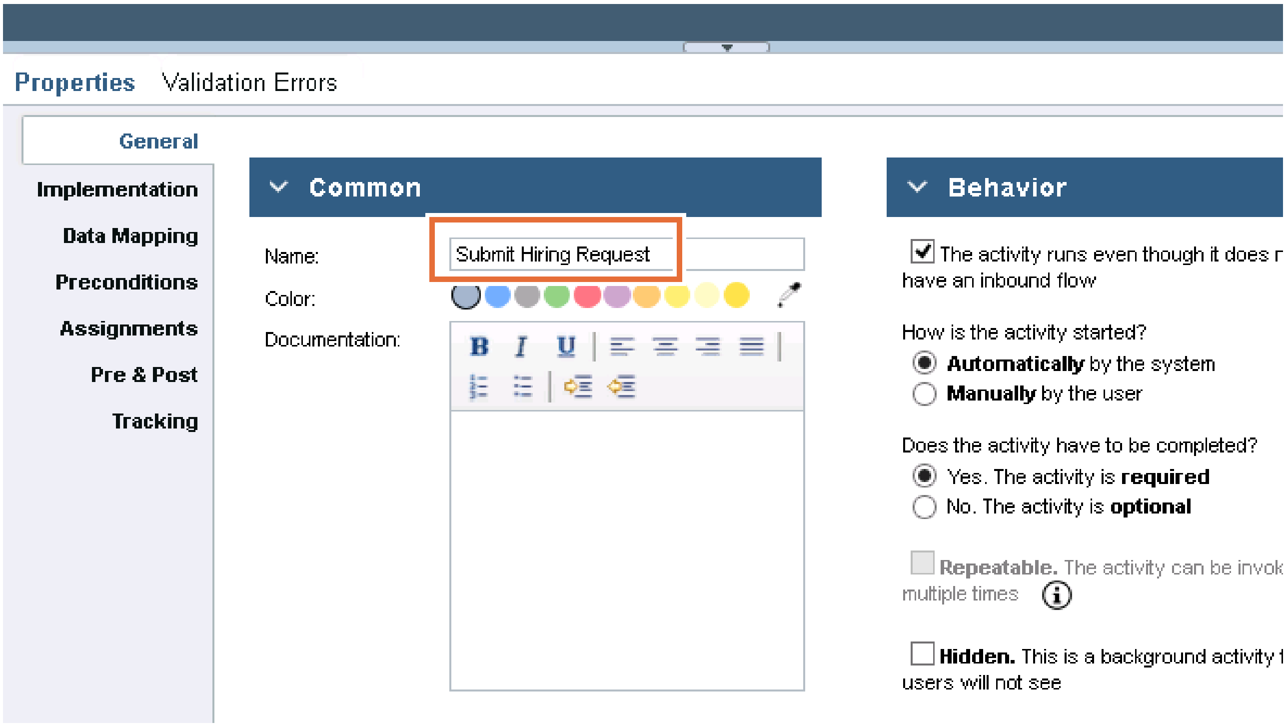
Click the element in the palette and hold for 1 second. When you drag to the canvas, a green plus sign verifies that you are dragging an element to the canvas. If you want to place an item above another item on the canvas, you can drag it to the item that must appear below it. Then, release the element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

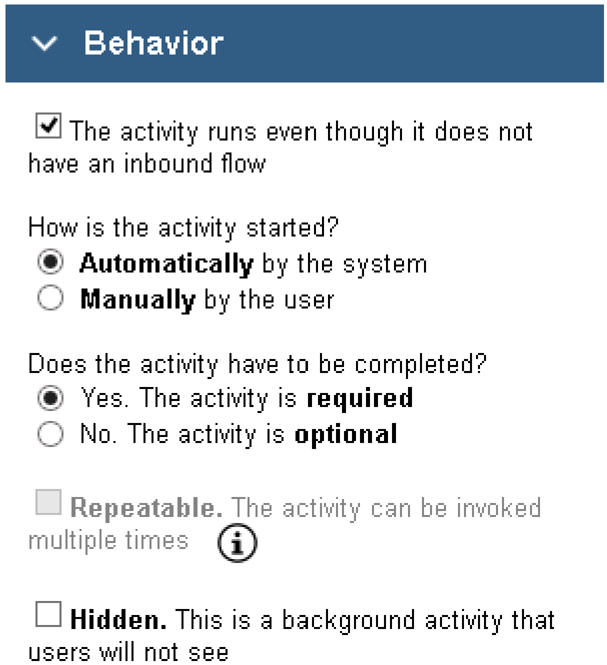


__ e. In the **Properties > General > Common** section, enter `Submit Hiring Request` as the name.

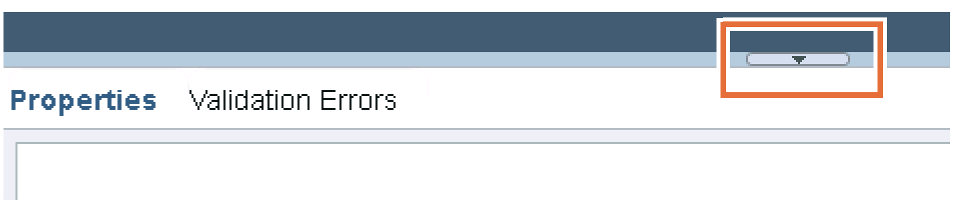


__ f. In the **Properties > General > Behavior** section, verify that the following options are selected by default:

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

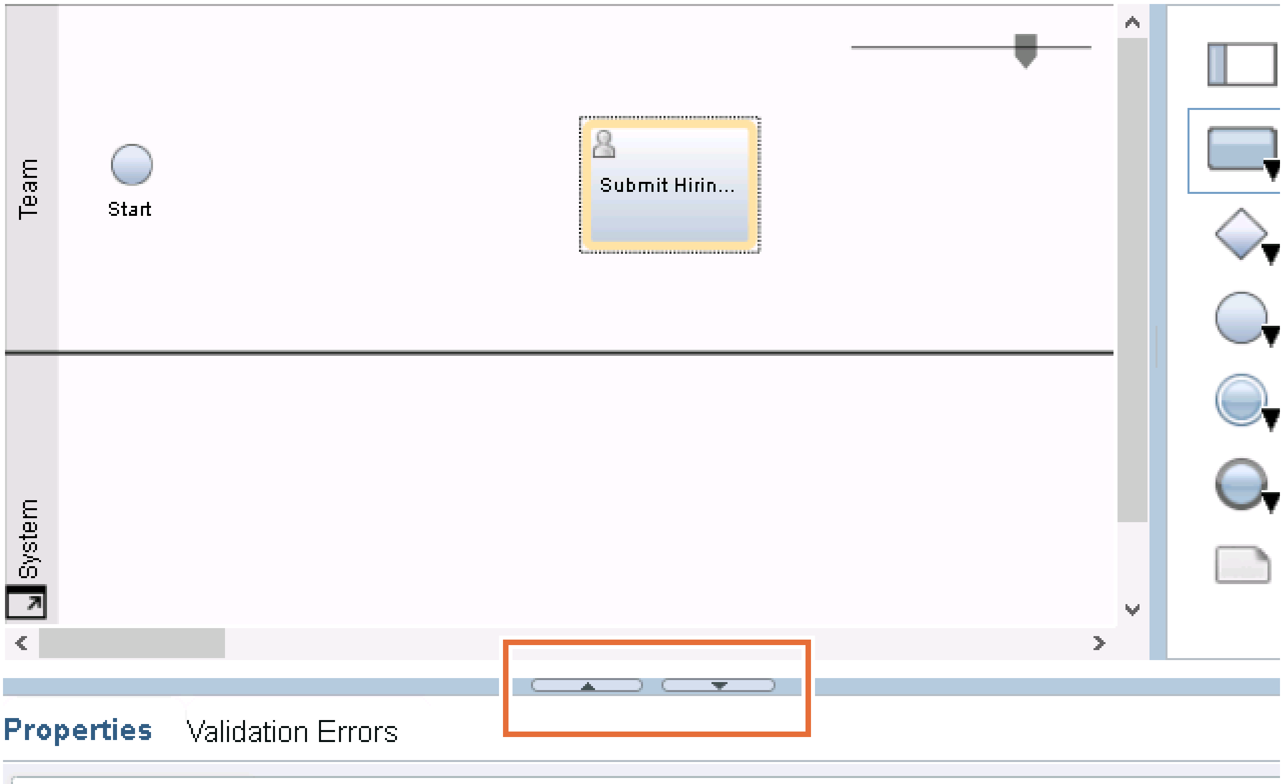


__ g. Click the **down arrow** to collapse the Properties section.



Reminder

You can click the down/up arrow to minimize or maximize the properties section for a process in the web Process Designer.



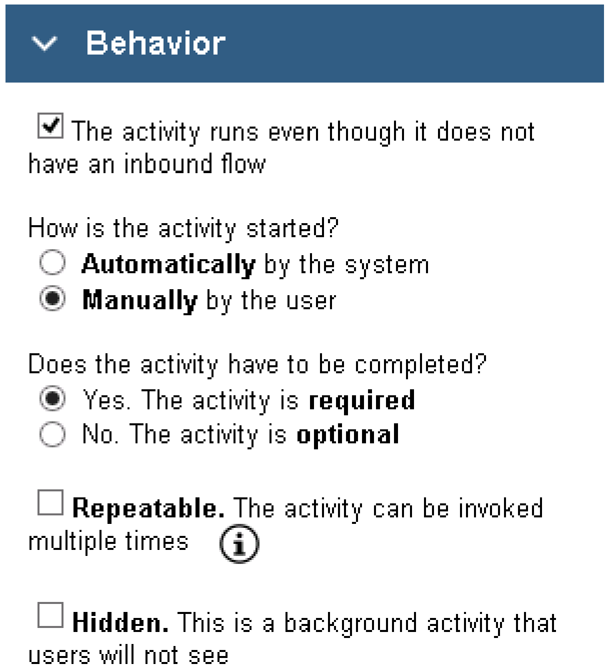
__ 2. Create a manually started, required activity that is named **Approve Hire Request**.

__ a. Drag an activity from the palette to the right of the Submit Hiring Request activity.

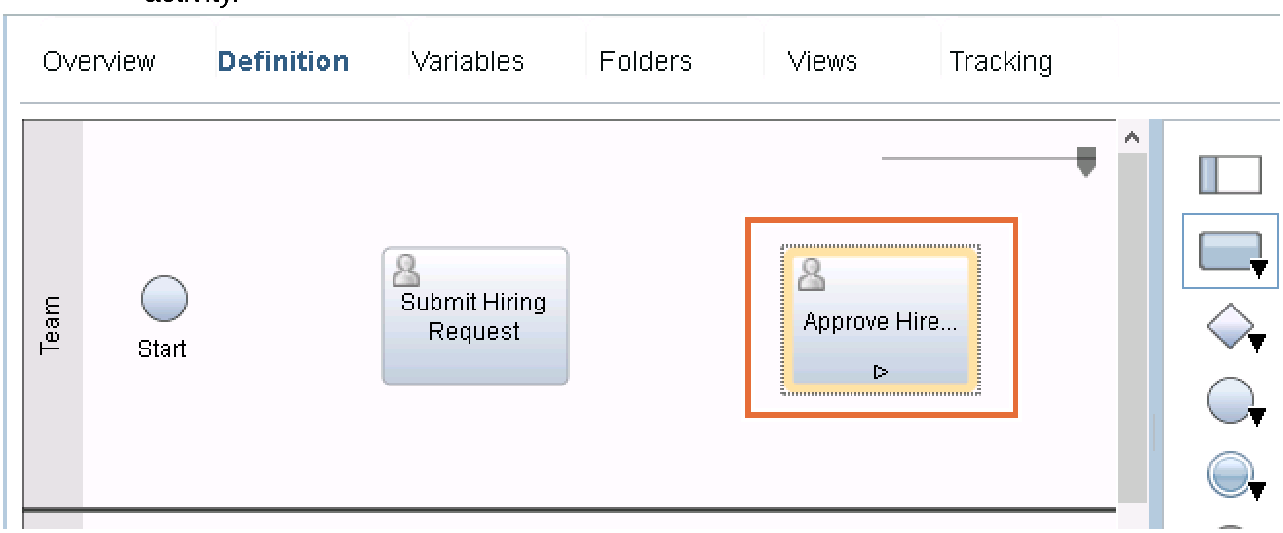
__ b. Double-click the activity to rename it as `Approve Hire Request` and press the Enter key to apply the name change.

__ c. In the **Properties > General > Behavior** section, select the following options:

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



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



__ 3. Create a manually started, required activity that is named **Complete Hire Request**.

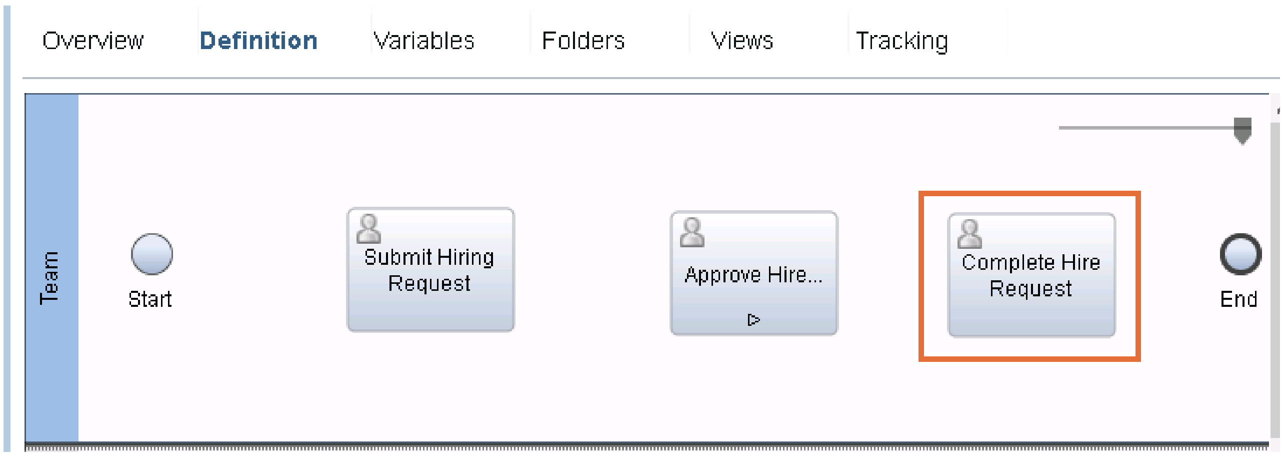
__ a. Drag an activity from the palette to the right of Approve Hire Request activity.

Note

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

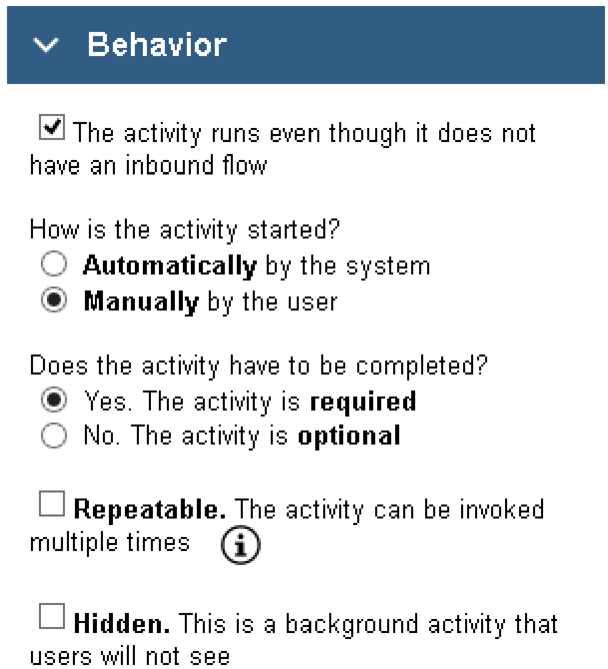
- Click the activity and then drag it in the position that you want.
- Click the activity and then use the arrow keys on the keyboard to move it in the position that you want.

__ b. Rename the activity as: `Complete Hire Request`



__ c. In the **Properties > General > Behavior** section, select the following options:

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

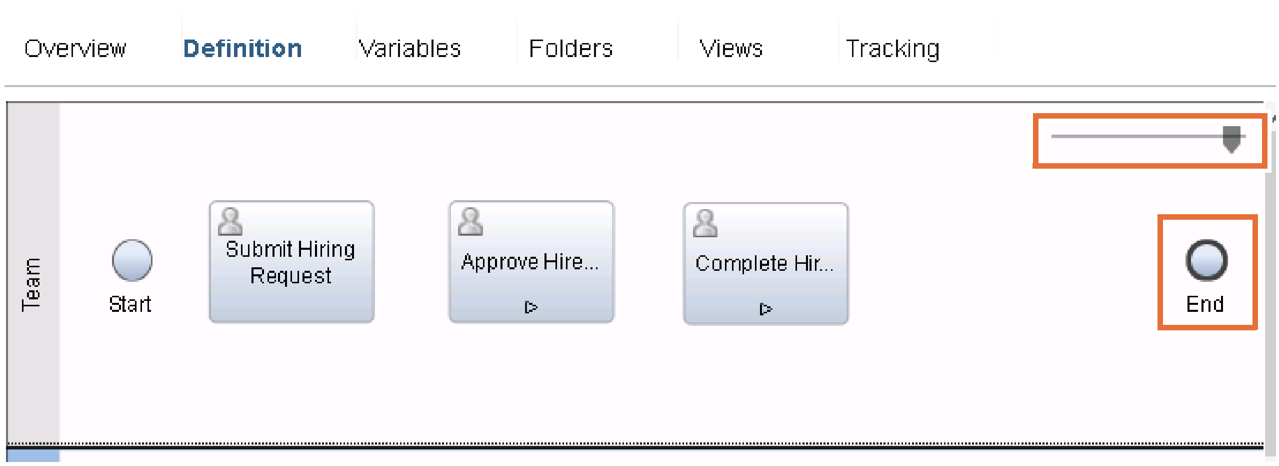


__ d. Save your changes.

Note

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

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



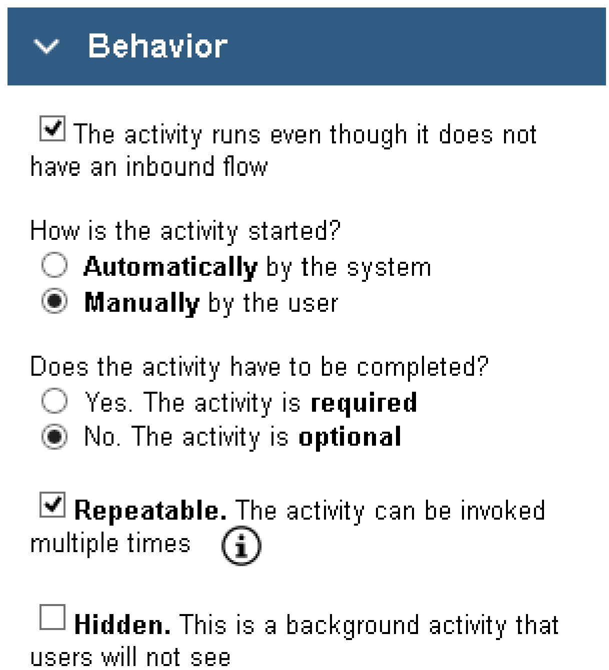
__ 4. Create an optional, manually started, repeatable activity named **Review Salary**.

__ a. Drag an activity from the palette to the right of Complete Hire Request activity.

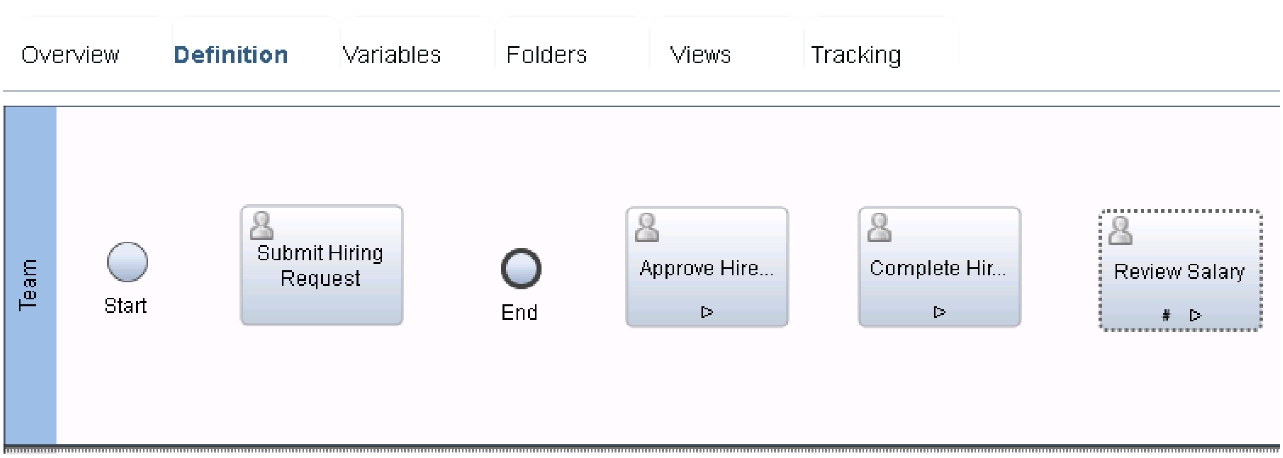
__ b. Rename the activity as: `Review Salary`

__ c. In the **Properties > General > Behavior** section, select the following options:

- This activity runs even though it does not have an inbound flow: Enabled.
- **How is the activity started?** Manually.
- **Does the activity have to be completed?** No. The activity is optional.
- **Repeatable**: Enabled.
- **Hidden**: Disabled.



__ 5. Create a flow in the process.

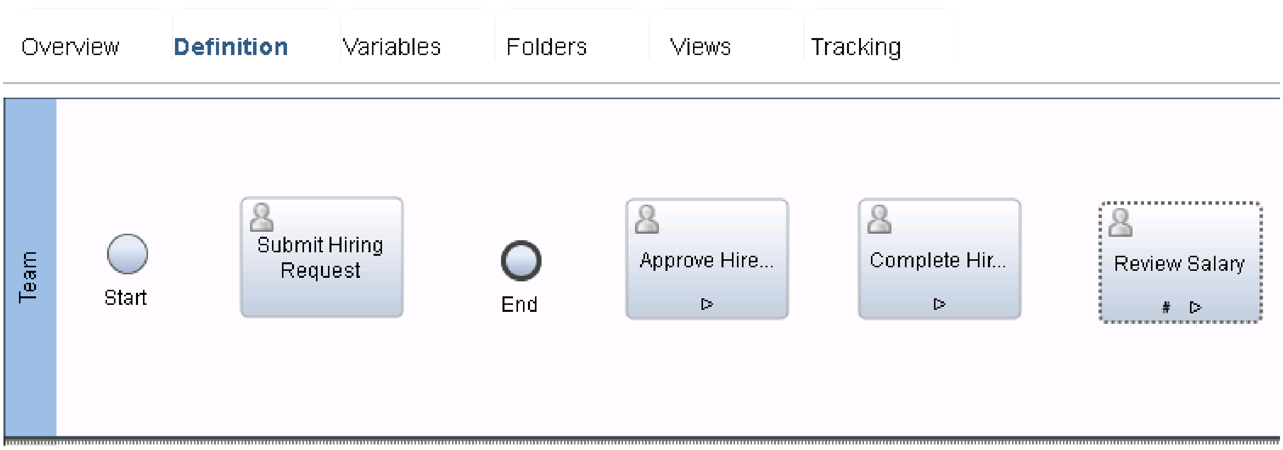


Note

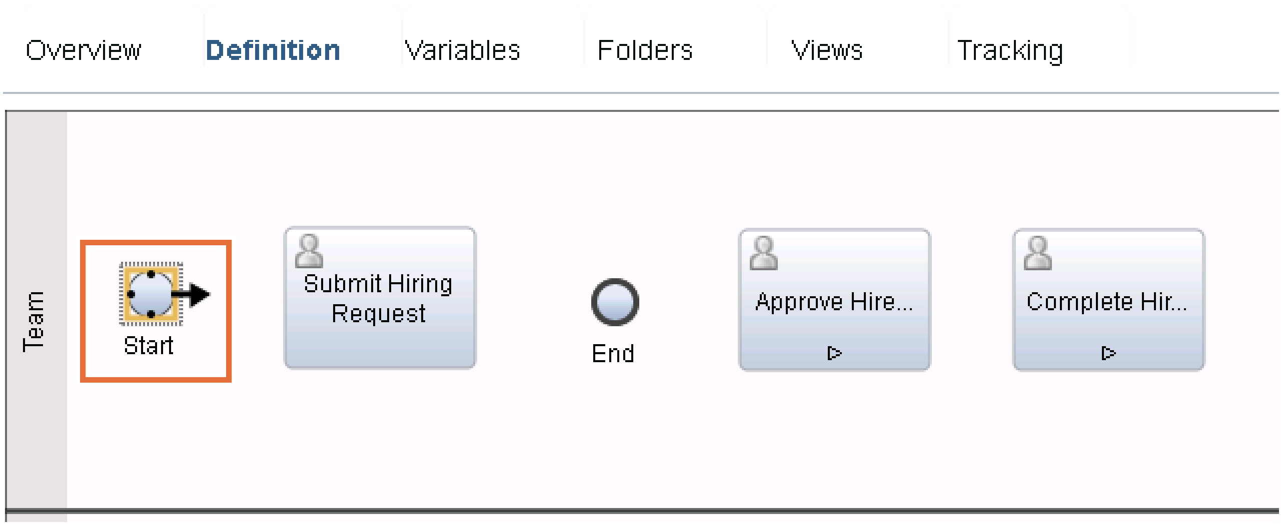
In this exercise, you modeled an as-is process with ad hoc activities, and in the next exercises you work on creating a to-be process model. Because the activities in the current process can occur at any time, you modeled the process by using ad hoc activities.

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.

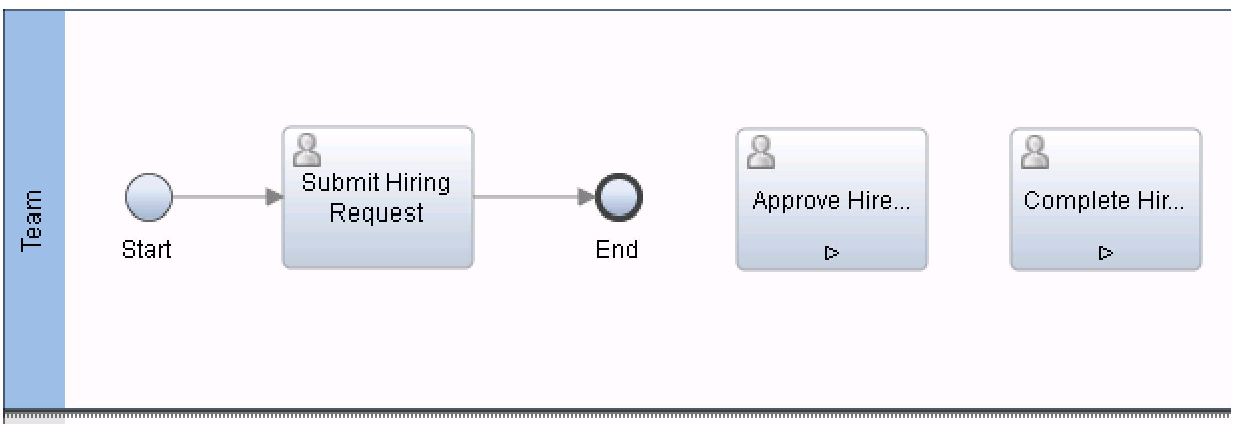


__ b. Hover over the **Start** event to see a flow that emerges out from the control point on the edge.



__ 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.



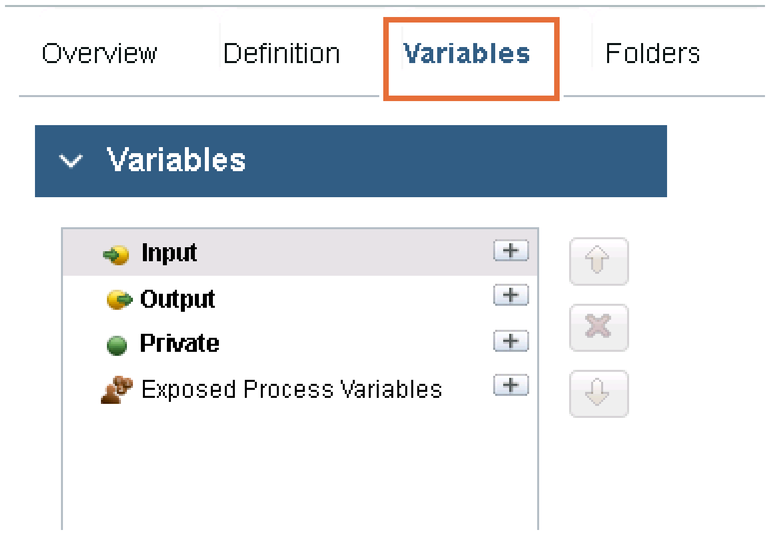
__ e. Save your changes.

Part 5: Create process variables

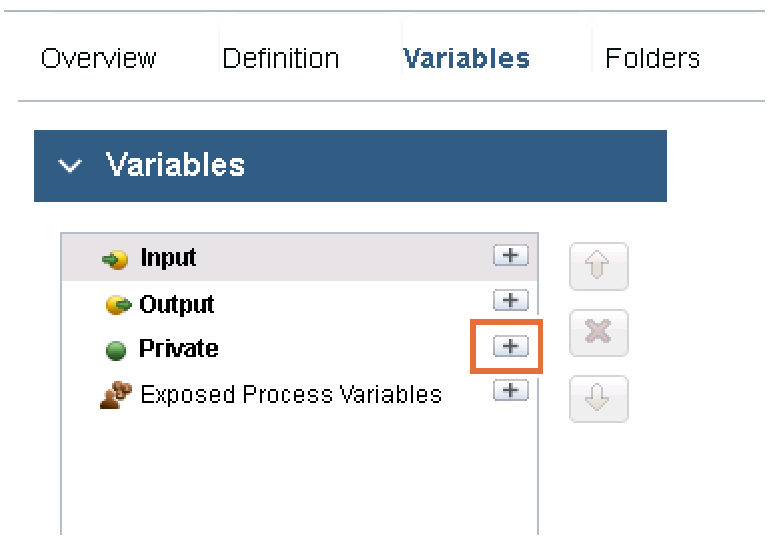
A variable contains information that can be shared between activities. A variable can have a primitive type such as String. A variable can also be a business object with either a complex or a simple type.

__ 1. Create the variables.

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

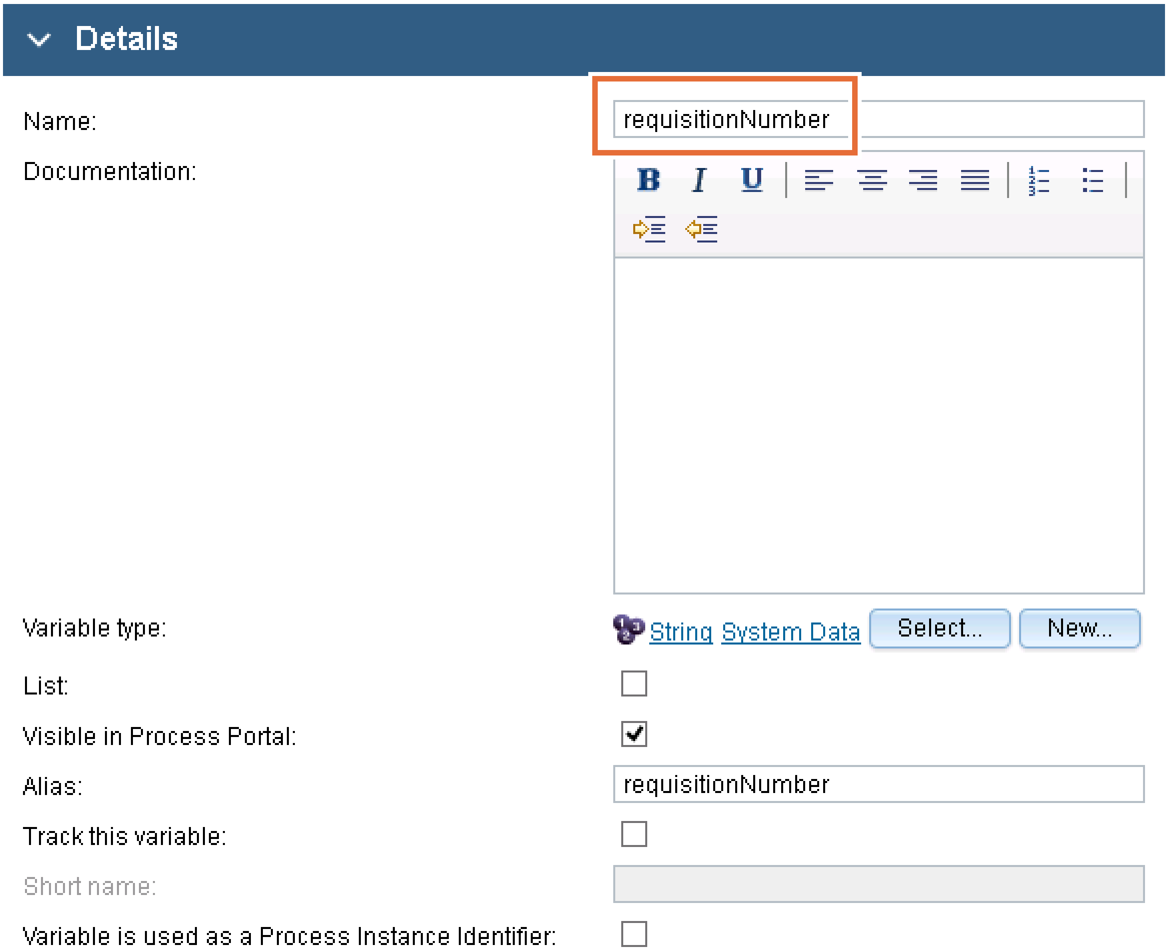


__ b. Click (**+**) next to **Private** to add a variable.



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

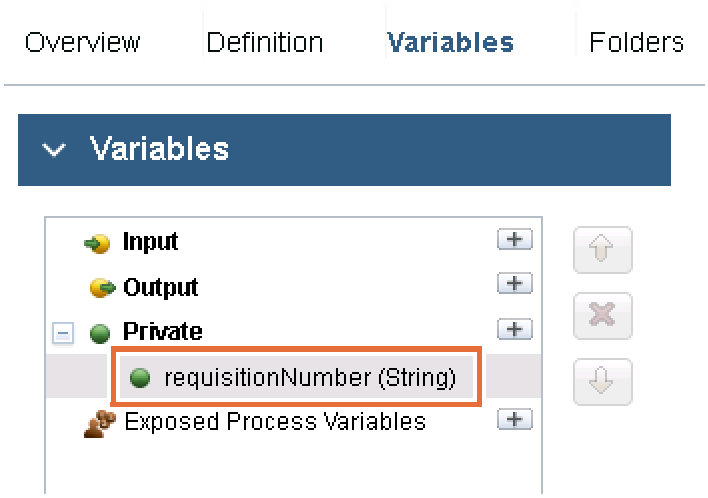
- **Name**: `requisitionNumber`
- **Variable Type**: String
- **List**: Disabled
- **Visible in Process Portal**: Enabled
- **Alias**: `requisitionNumber`



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.

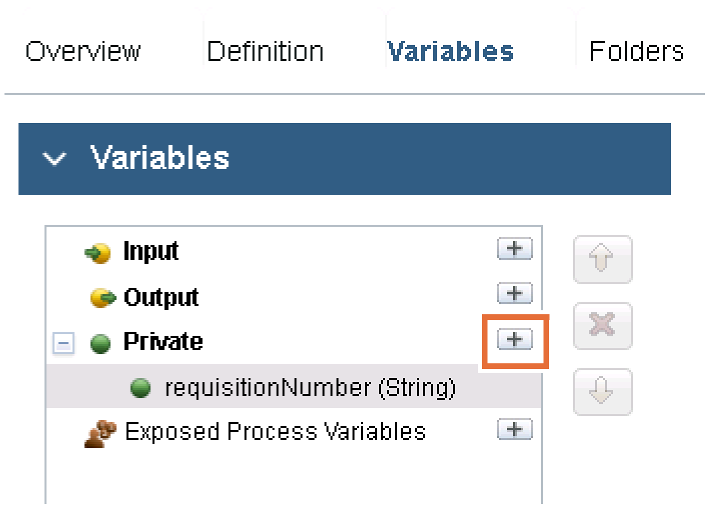
__ d. The variable is added in the Private list under the Variables section.



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

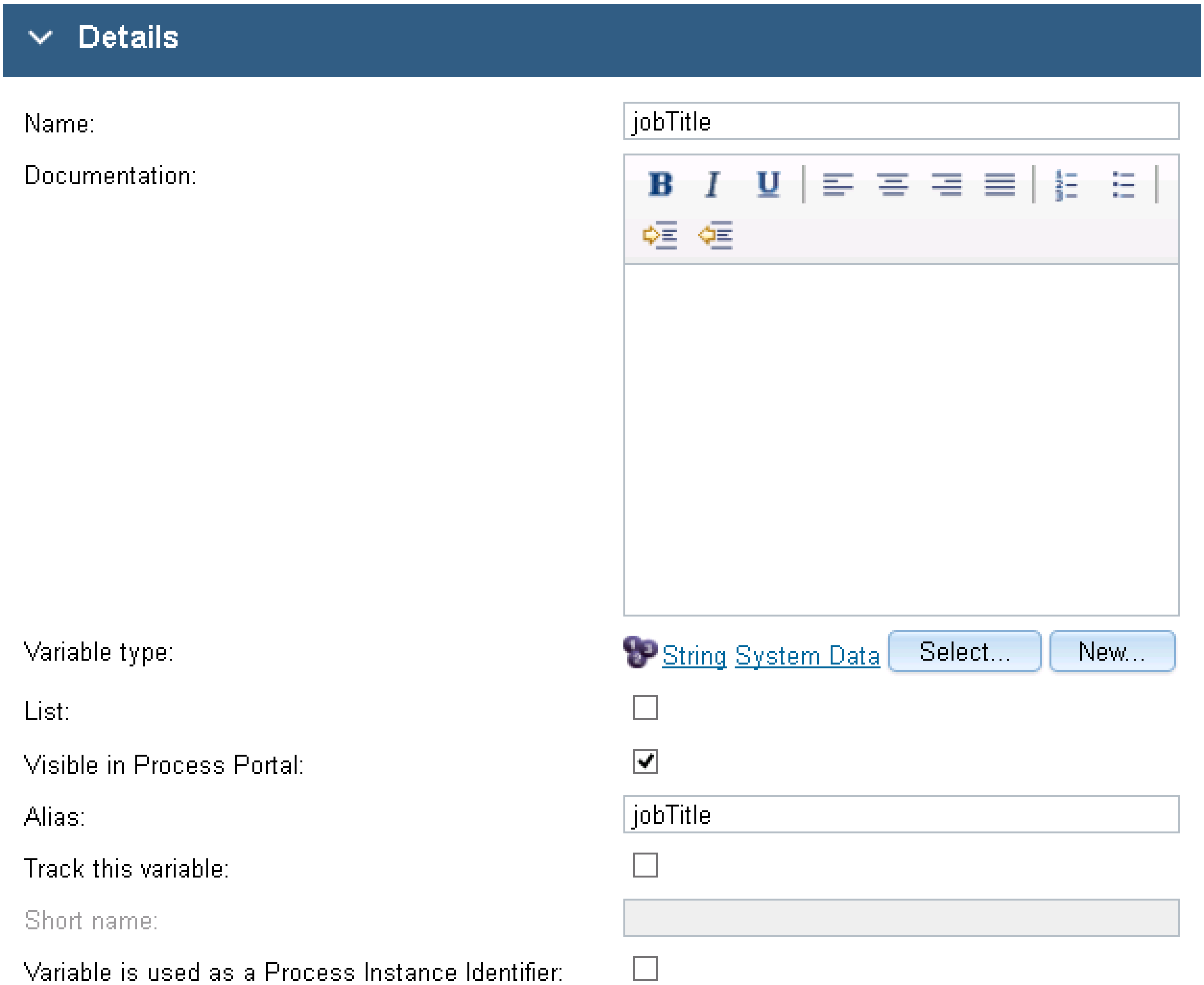
Name	Variable Type	List	Visible	Alias
`jobTitle`	String	No	Yes	`jobTitle`
`salary`	Integer	No	Yes	`salary`
`department`	String	No	Yes	`department`

__ a. Click (**+**) next to Private to add another private variable.



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

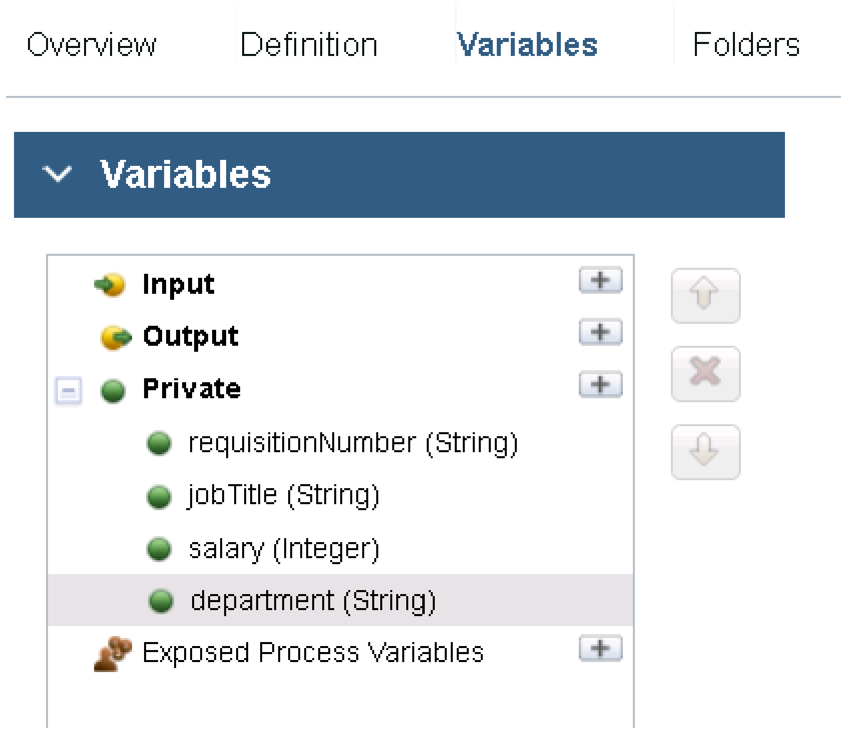
- **Name**: `jobTitle`
- **Variable Type**: String
- **List**: Disabled
- **Visible in Process Portal**: Enabled
- **Alias**: `jobTitle`



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

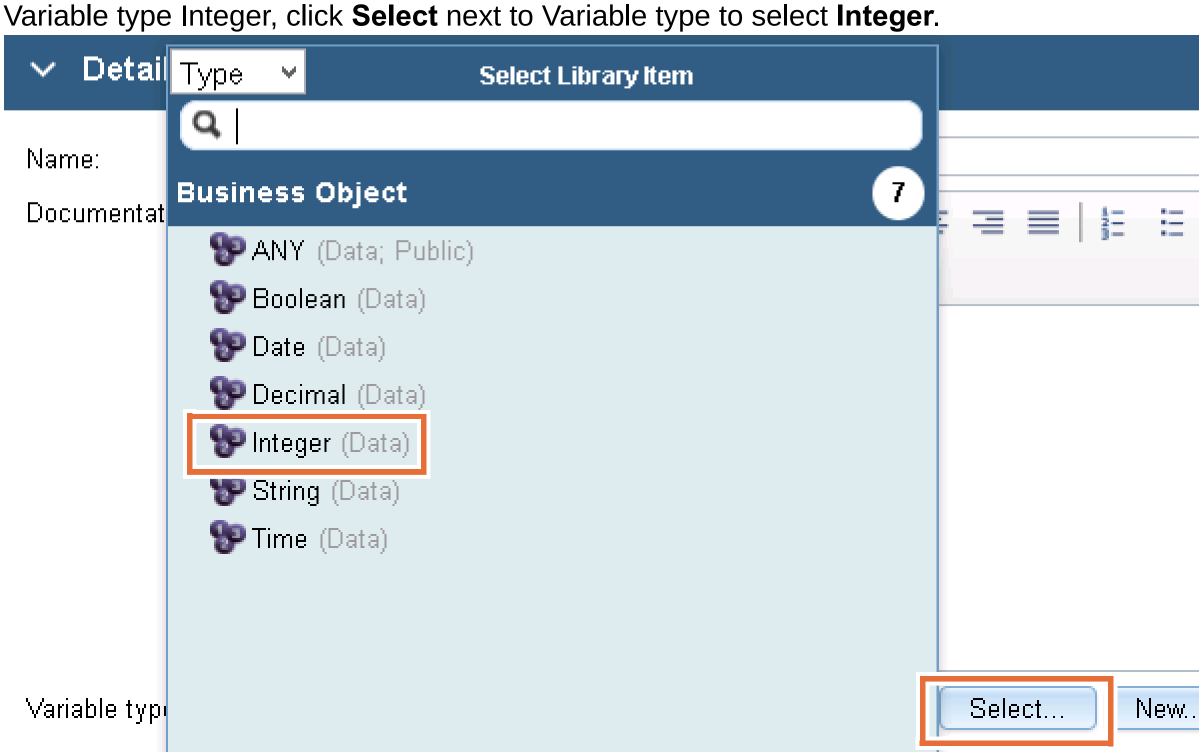
Name	Variable Type	List	Visible	Alias
`salary`	Integer	No	Yes	`salary`
`department`	String	No	Yes	`department`

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



Hint

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

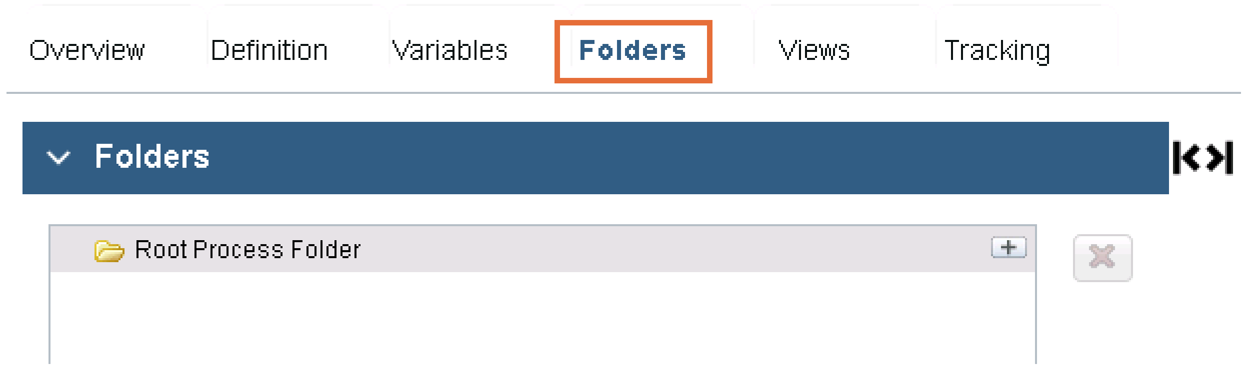


Part 6: Create process folders

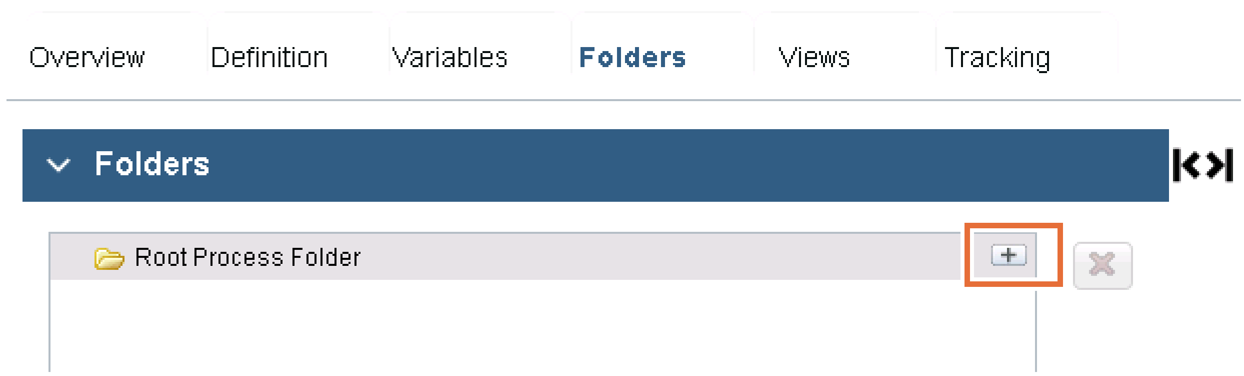
Folders provide a way of grouping documents that are related to a process. You can create a folder structure that knowledge workers or users in Process Portal can use to add documents that are required to complete the process. You create process folders in the process editor.

__ 1. Create process folders.

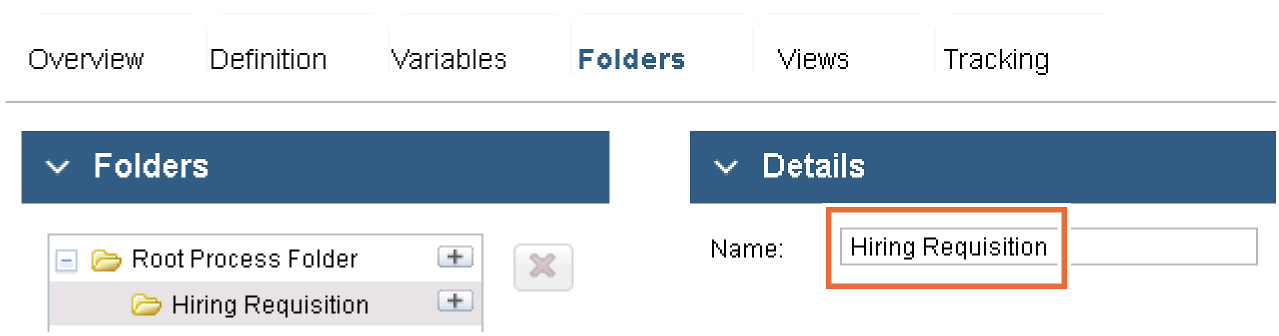
__ a. From the Hiring Request Process, click the **Folders** tab.



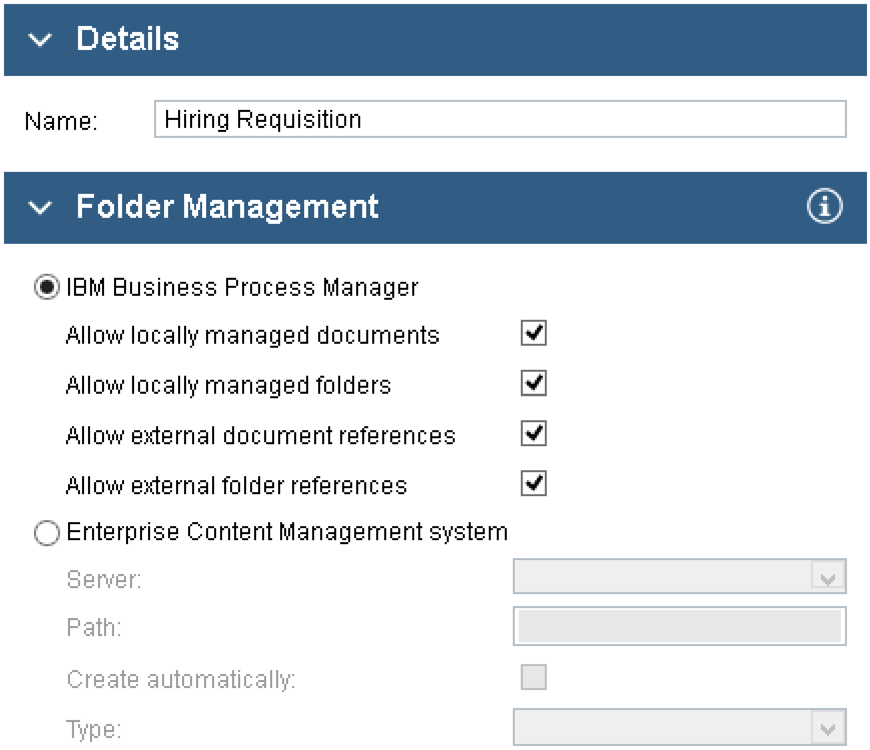
__ 2. Click (**+**) next to **Root Process Folder** to add a folder.



__ 3. Enter `Hiring Requisition` as the name.



__ 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.



__ 5. Click **Save** to save your changes. You created a new folder.

Information

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 web 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.

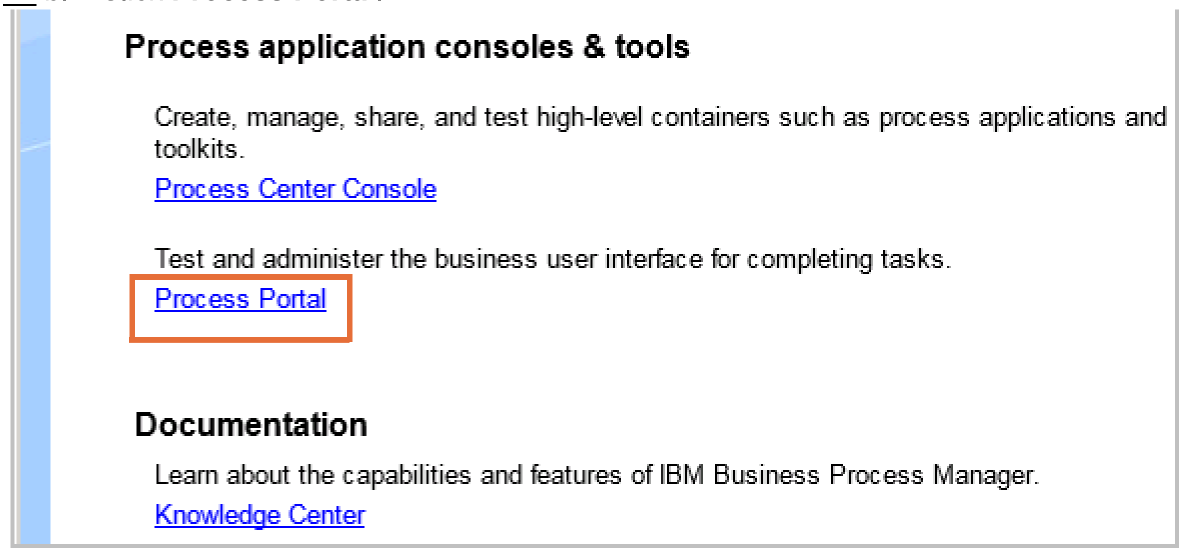
Note

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.

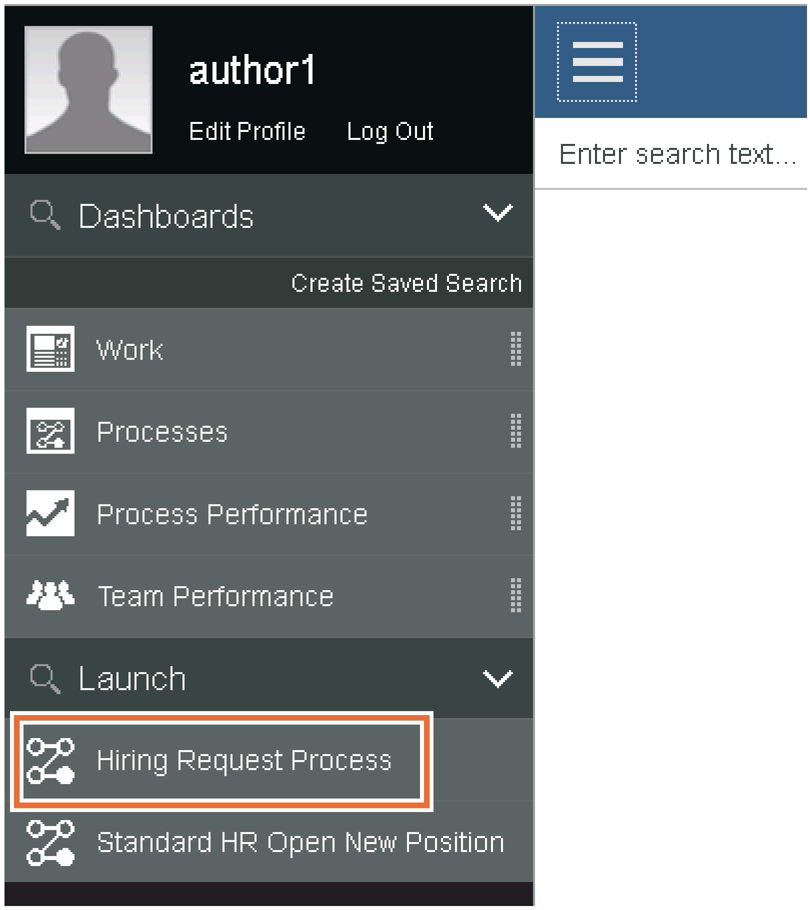
__ b. Click **Process Portal**.



Note

If prompted for credentials, enter `author1` for the **Username** 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**.

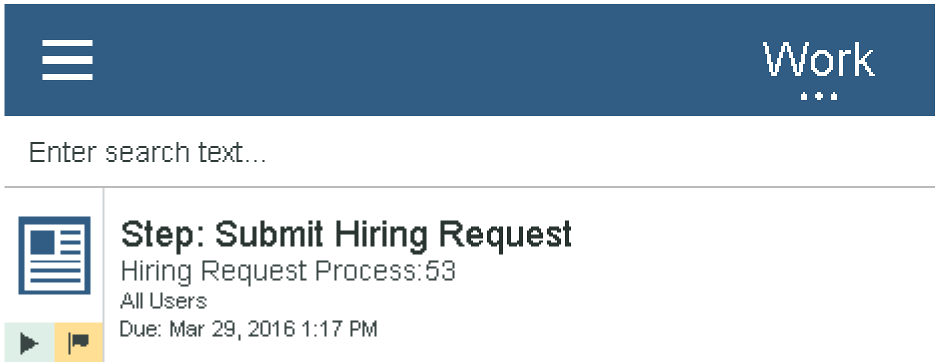


A process instance is started. You see the message at the bottom of the window.

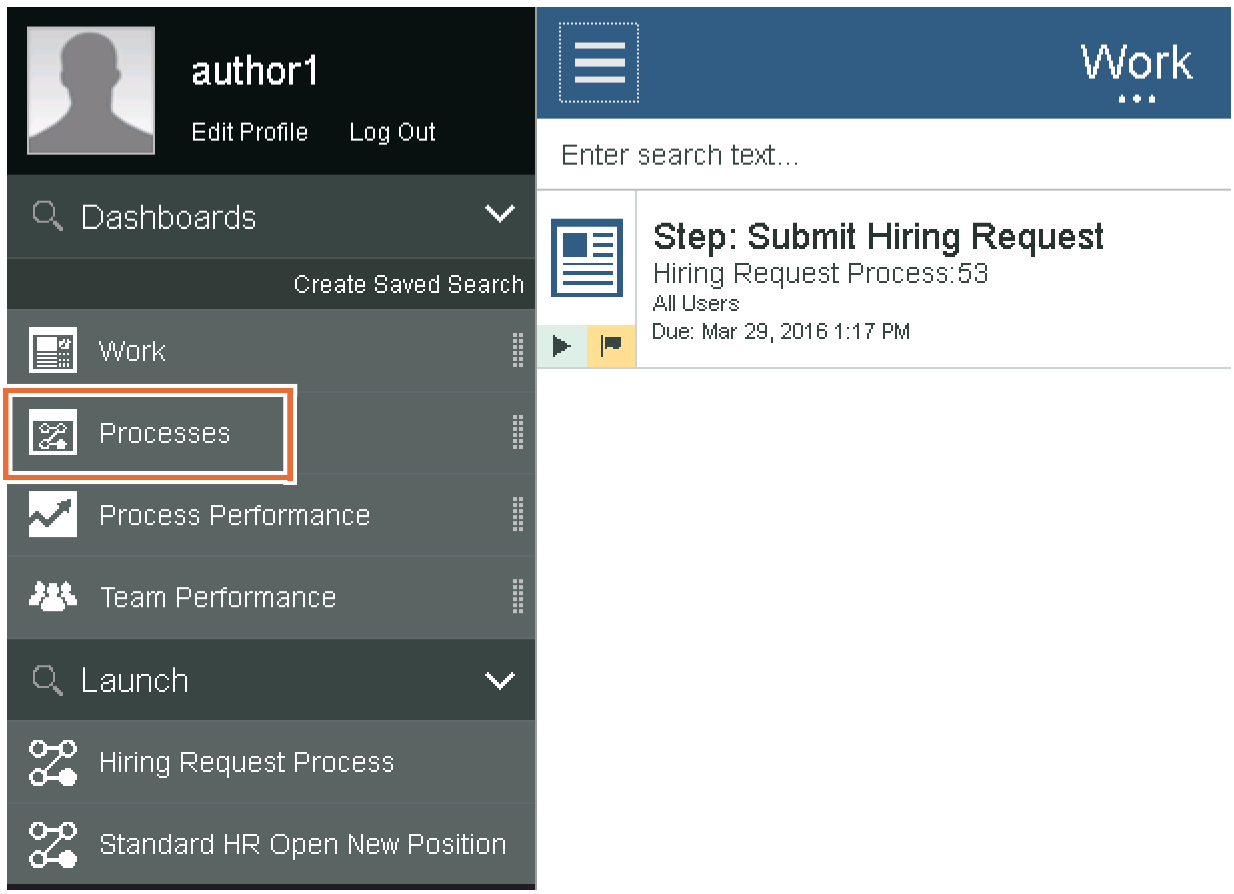
Note

Due to 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.

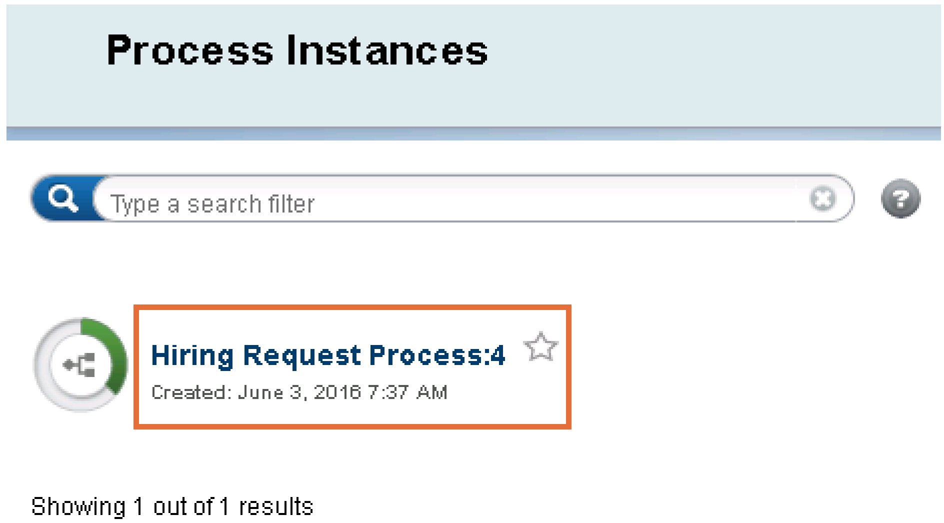
__ 3. Under **Work**, the **Submit Hiring Request** activity shows as an open task.



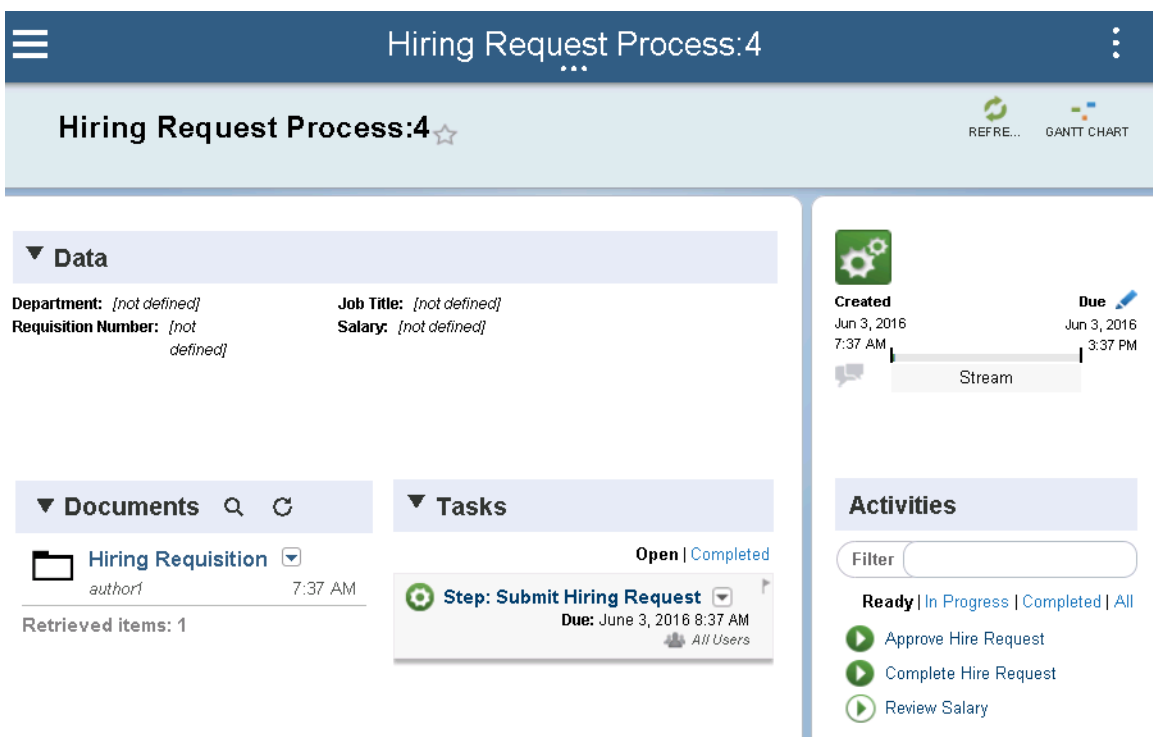
__ 4. In the Library menu, click **Processes**.



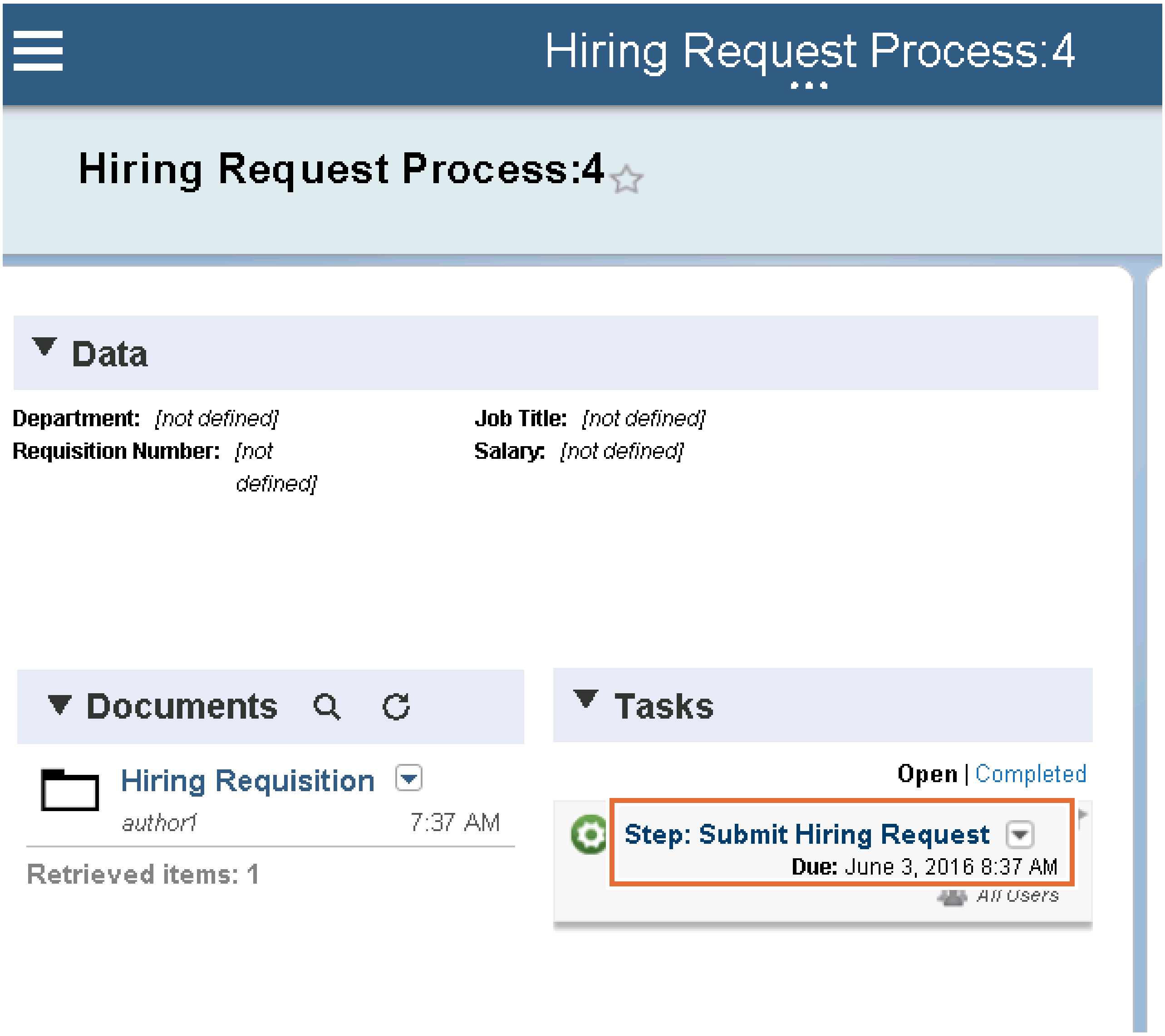
__ 5. The **Hiring Request Process** instance is listed. Click the **Hiring Request Process** instance.



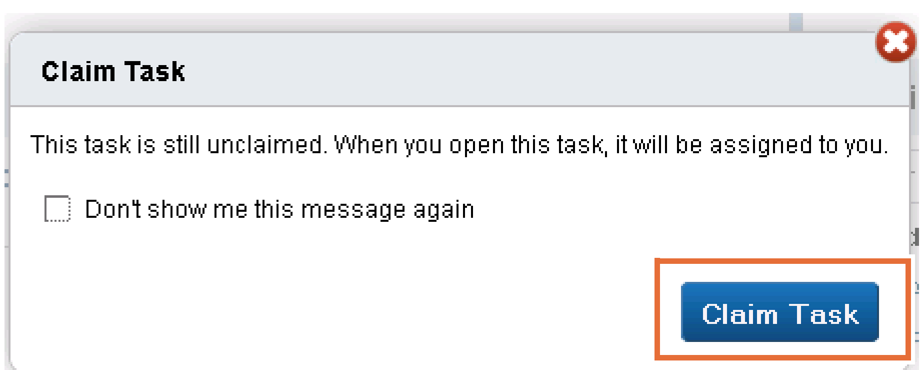
__ 6. The process instance window lists the open and ad hoc tasks.



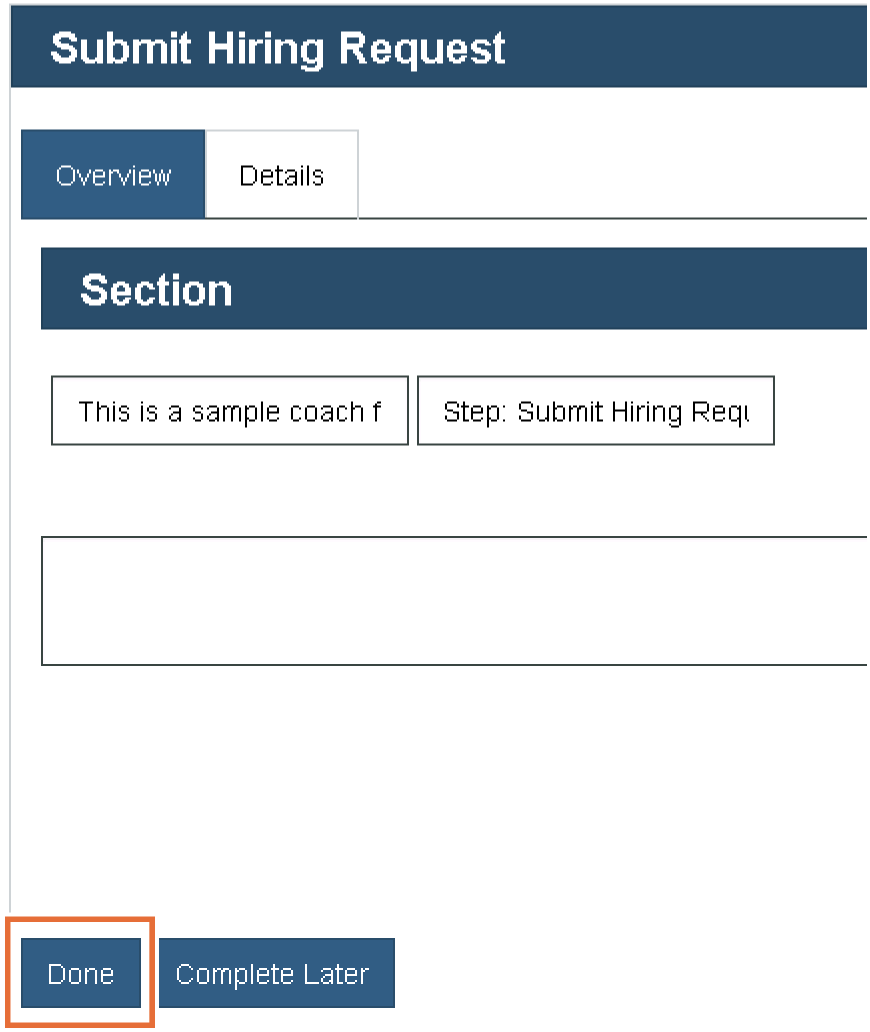
__ 7. Click **Step: Submit Hiring Request** under the Tasks section.



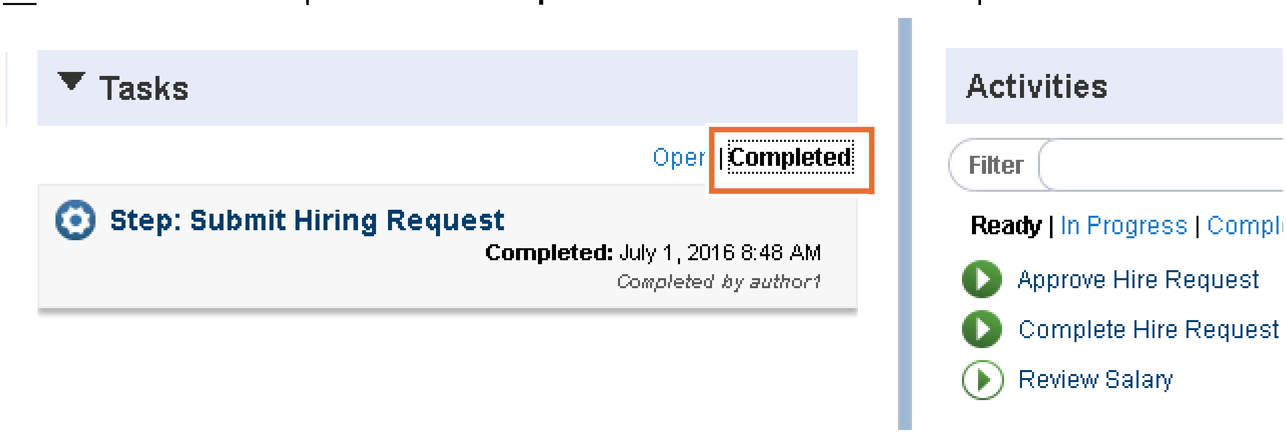
__ 8. Click **Claim Task**.



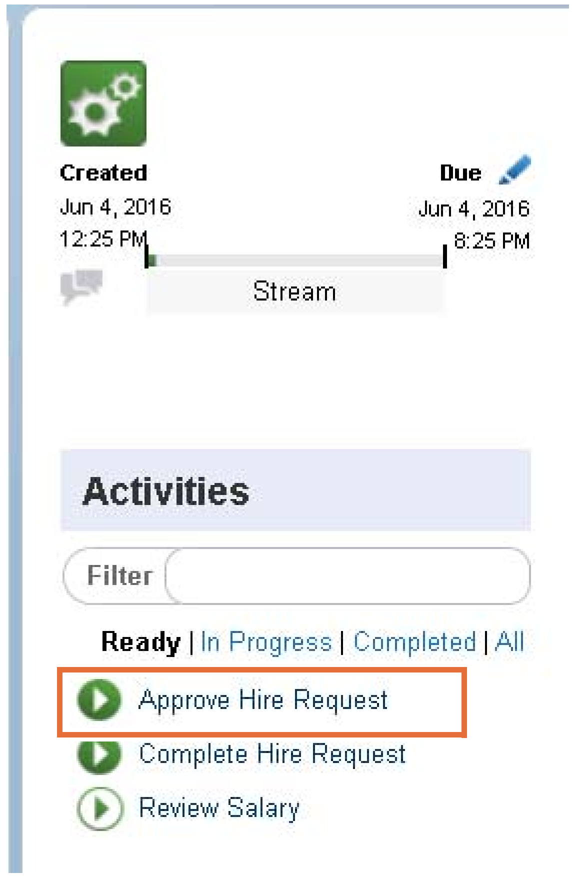
__ 9. The task is assigned to you and the default human service coach is shown. Click **Done** to complete the task.



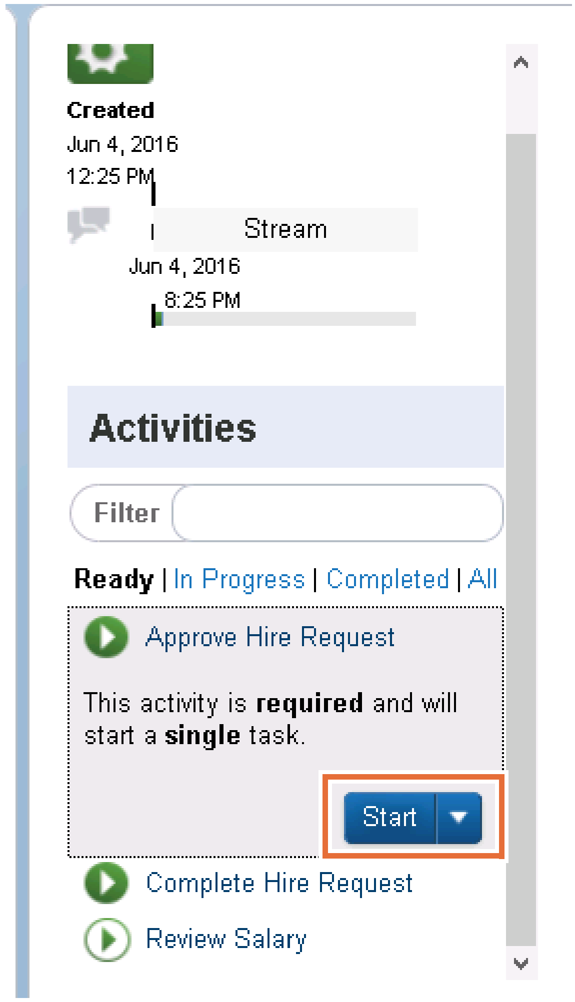
__ 10. The task is completed. Click **Completed** to view the task in the completed tasks list.



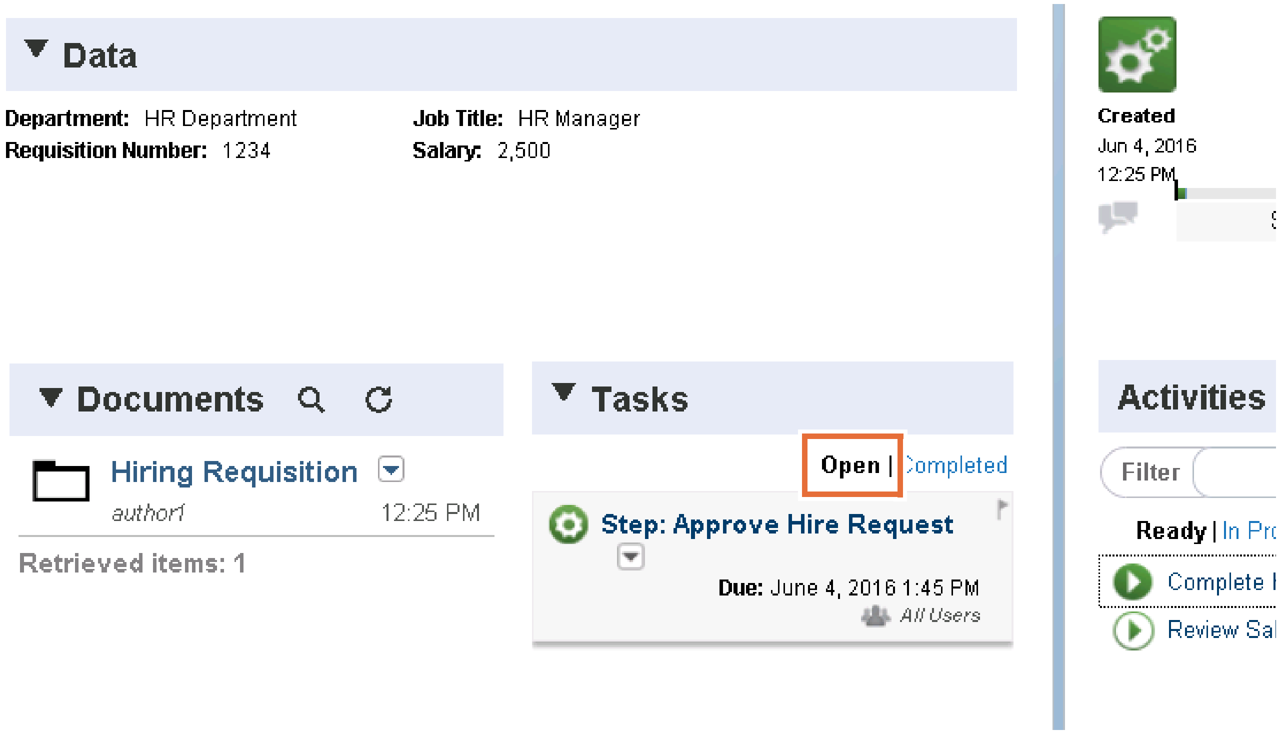
__ 11. Click the ad hoc activity **Approve Hire Request**, which is in a Ready state on the right pane.



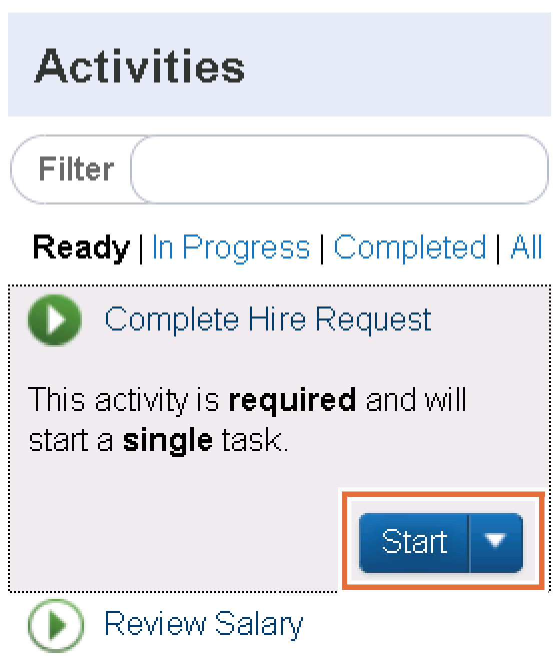
__ 12. Click **Start** to start this task.



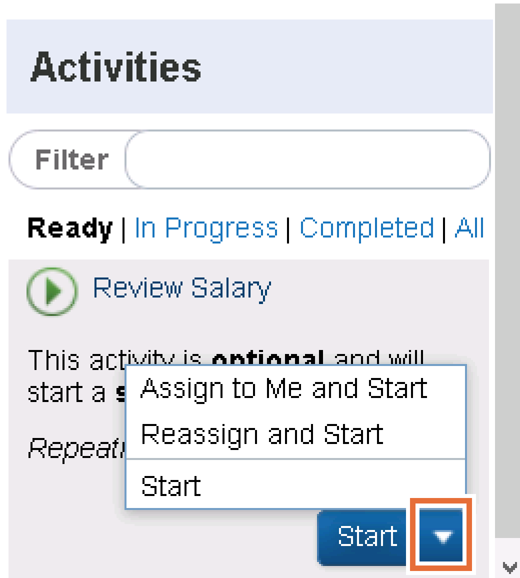
__ 13. The **Approve Hire Request** task is created. Click **Open** to view this task in the open tasks list.



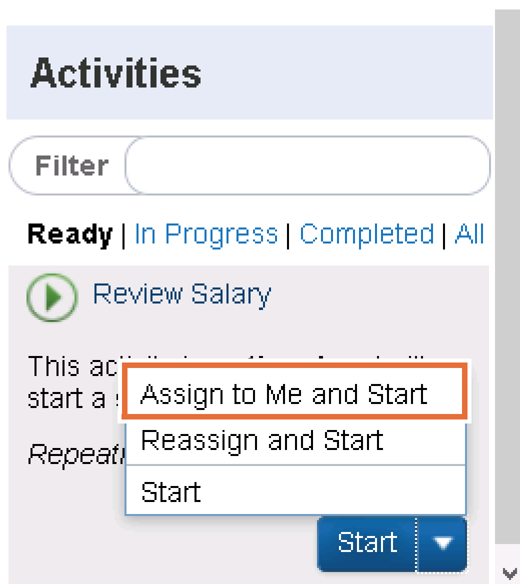
__ 14. Select the ad hoc activity **Complete Hire Request** in the right pane and click **Start** to start this task. It is also added in the open tasks list.



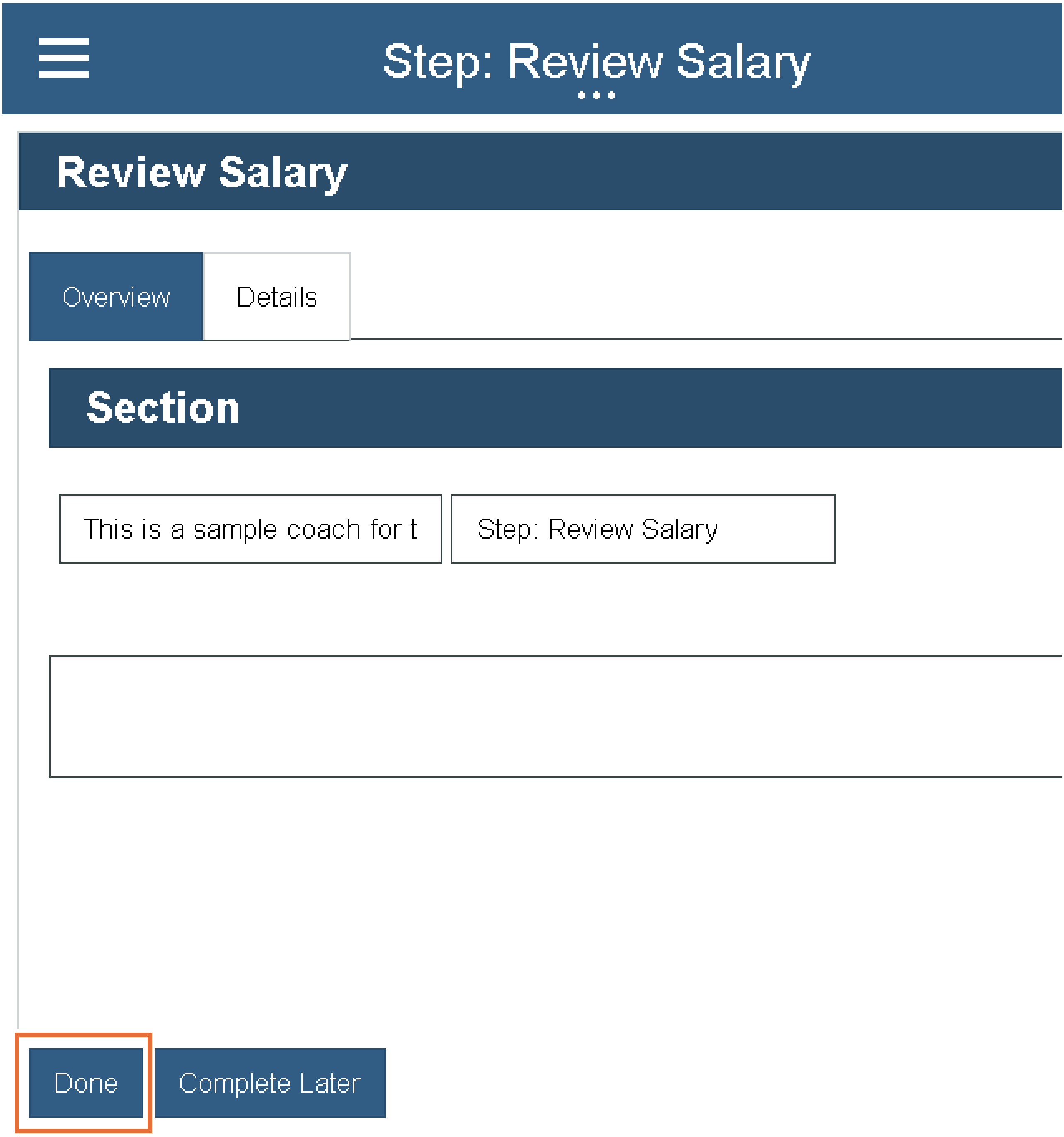
__ 15. Select the optional activity **Review Salary** in the right pane and open the menu next to the Start button.



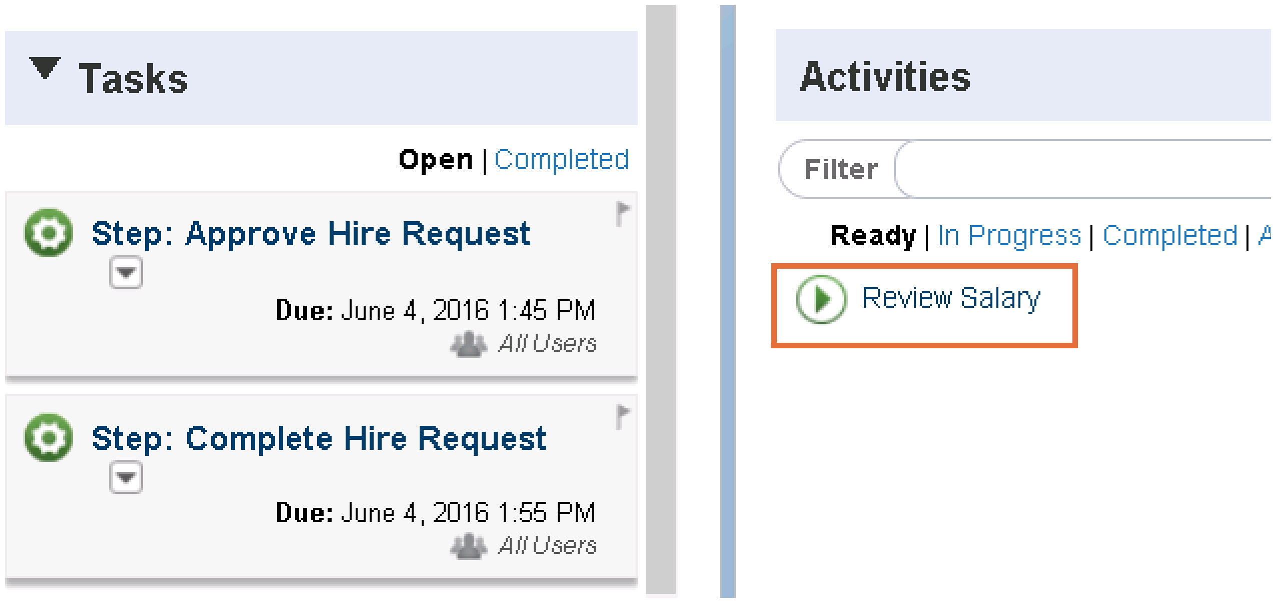
__ 16. Select **Assign to Me and Start**.



__ 17. The task is assigned to you and the default human service coach is shown. Click **Done** to complete the task.



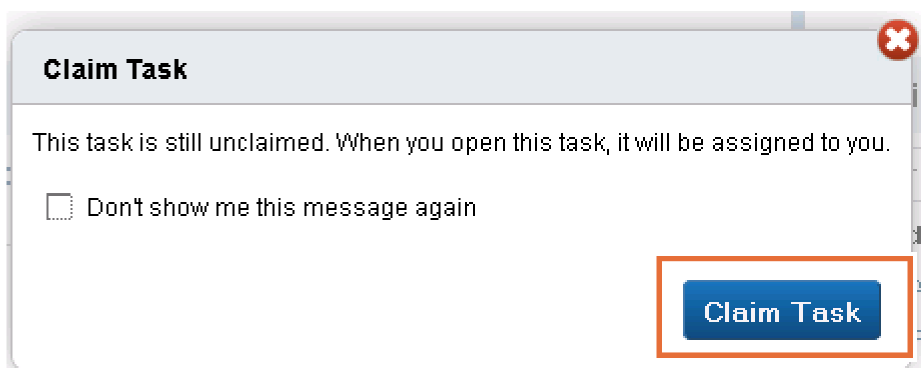
__ 18. Because the **Review Salary** activity is repeatable, it remains in the Ready (optional) state.



__ 19. Complete the **Approve Hire Request** task.

__ a. Click the **Step: Approve Hire Request** link.

__ b. Click **Claim Task**.



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

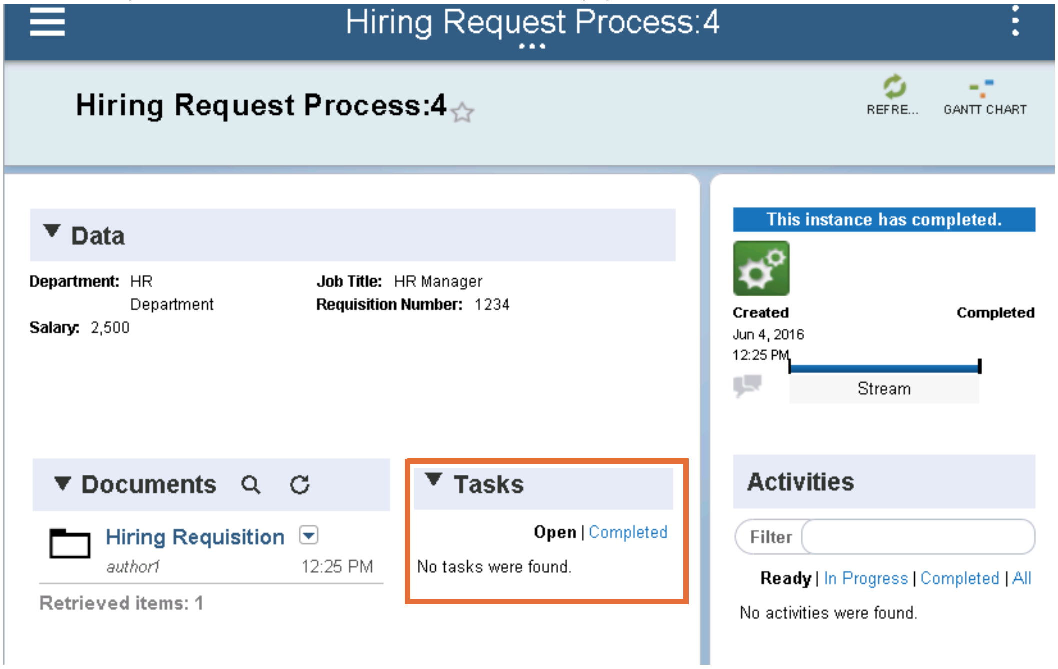
__ 20. Complete the **Complete Hire Request** task.

__ a. Click the **Step: Complete Hire Request** link.

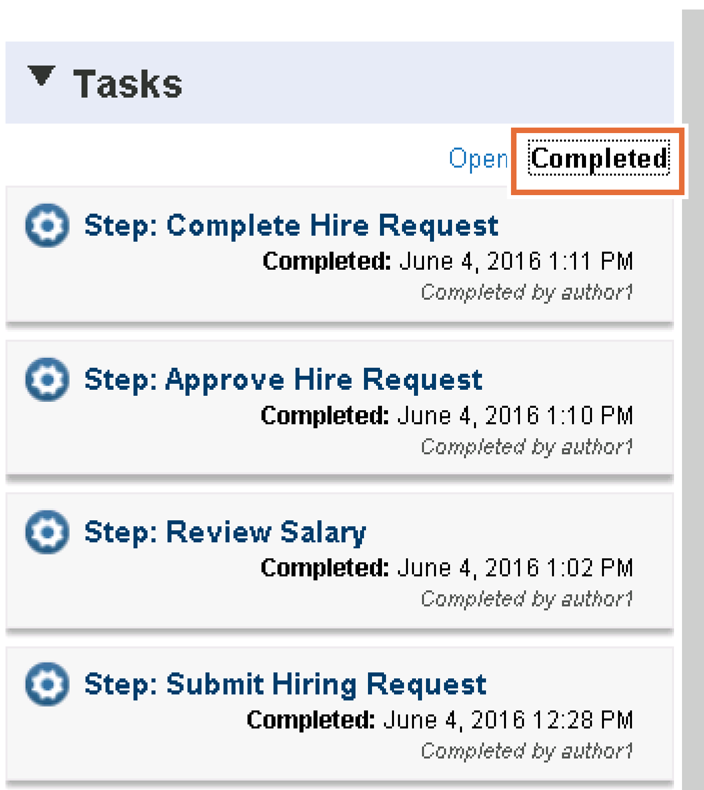
__ b. Click **Claim Task**.

__ c. Click **Done** to complete the task.

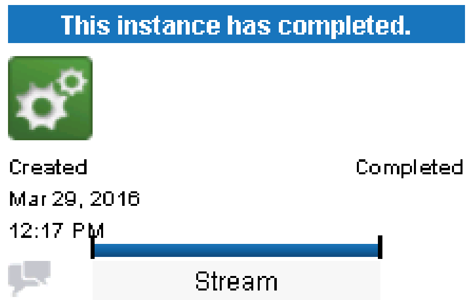
All the tasks are completed and the **Tasks** section is empty.



__ 21. After you complete all the activities, click **Completed** under the Tasks section to show the list of completed tasks.

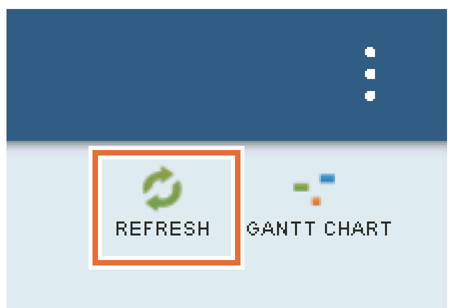


After the tasks status updates, the instance is completed. It might take up to 1 minute for the system to process the update.



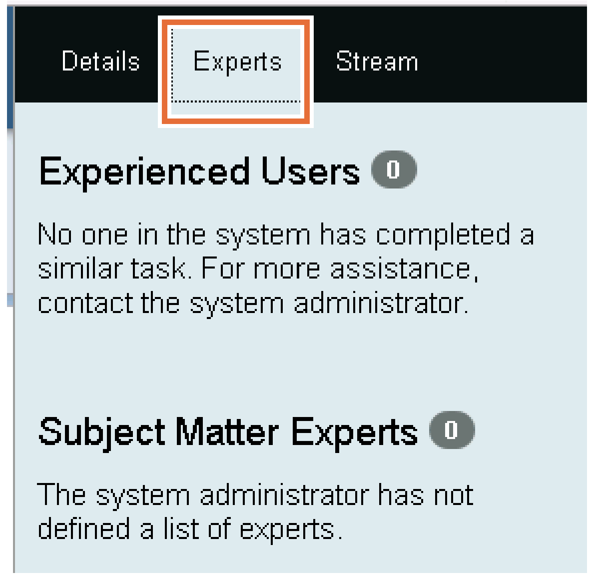
Note

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.

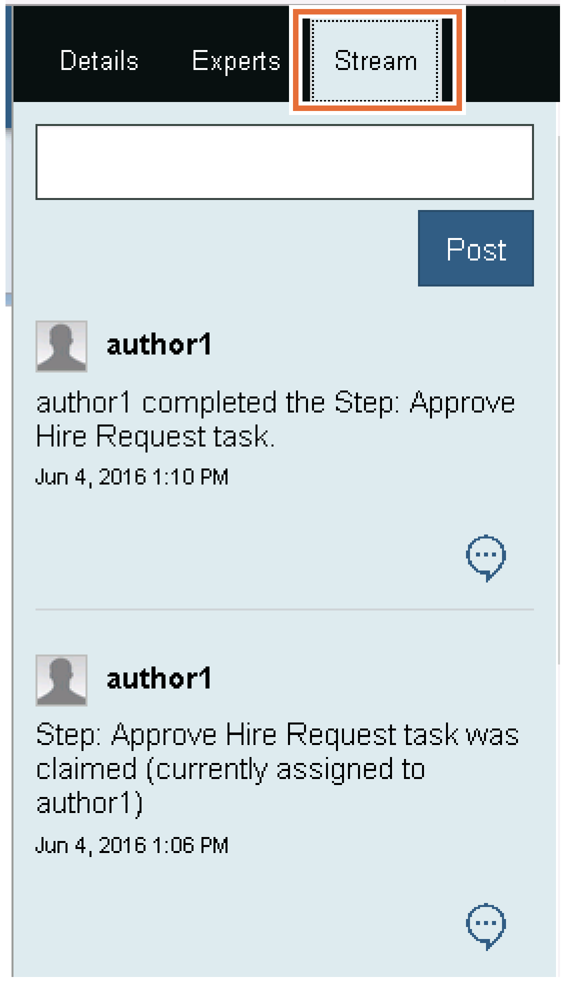


__ 22. Click the **Context menu** icon at the upper-right corner of Process Portal and note that the details of the activities completed.

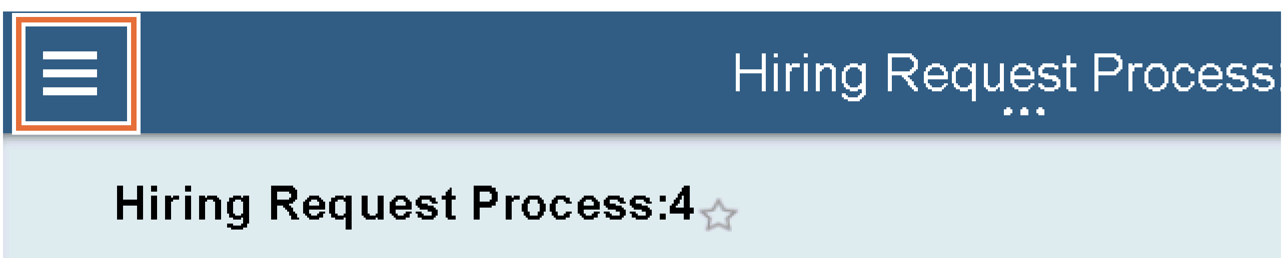
__ 23. Click the **Experts** tab. This tab lists the experienced users and subject matter experts in the system. Because you completed an instance of the task, you are added to the list as soon as another user creates an instance of the process. As a result, users can ask for assistance to complete tasks inside the portal.



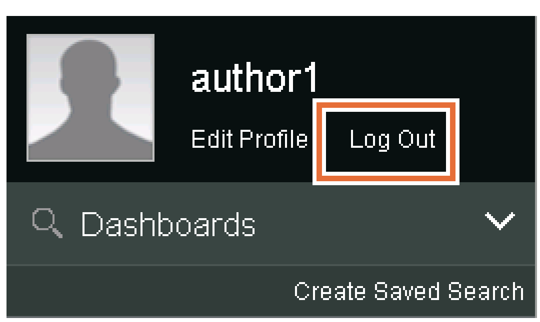
__ 24. Click the **Stream** tab. This tab shows the actions that occur for this instance. You can also post a comment to the stream that appears immediately in the stream. You add a comment to the stream in the later exercise.



__ 25. Click the **Library** icon on the left side of the window.



__ 26. Click **Log Out**.



__ 27. Close the Process Portal tab in the browser.

__ 28. 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 also 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 Web 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 their 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.

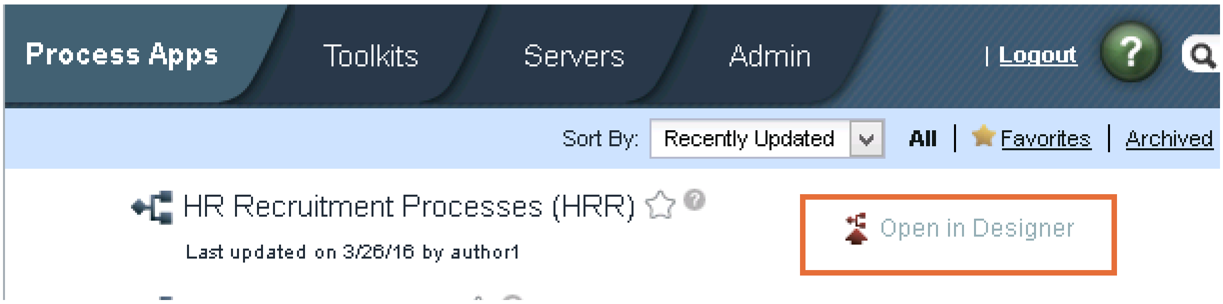
- 2.5: When a request violates the established salary guidelines of the company, the HR Administrator can approve or reject the requested salary override.
- 2.6: If the salary override is approved, the request is posted to the HR Positions database and made available for dissemination.
- 2.7: If the 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 to-be process model.

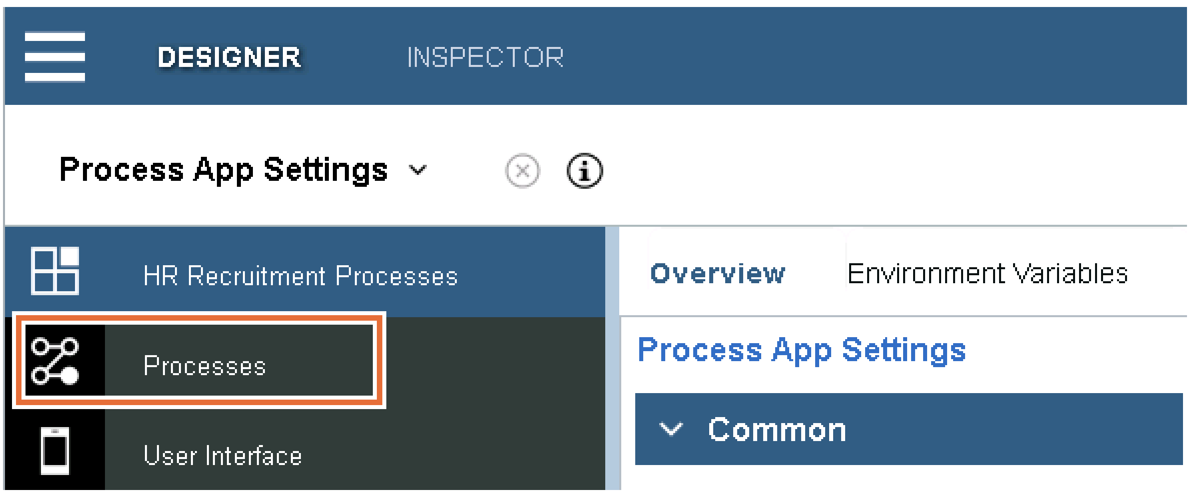
Part 1: Open the business process in Process Center

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

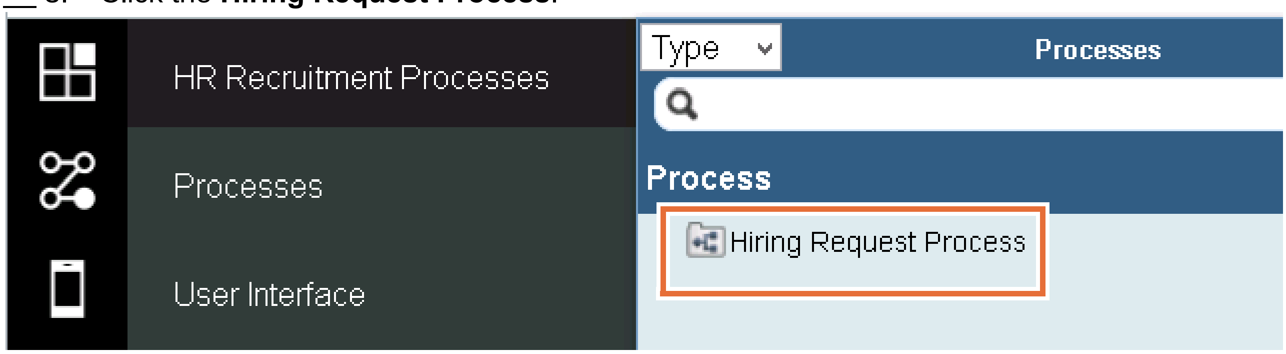
__ 1. Open the **HR Recruitment Processes (HRR)** process application unless it is already open.



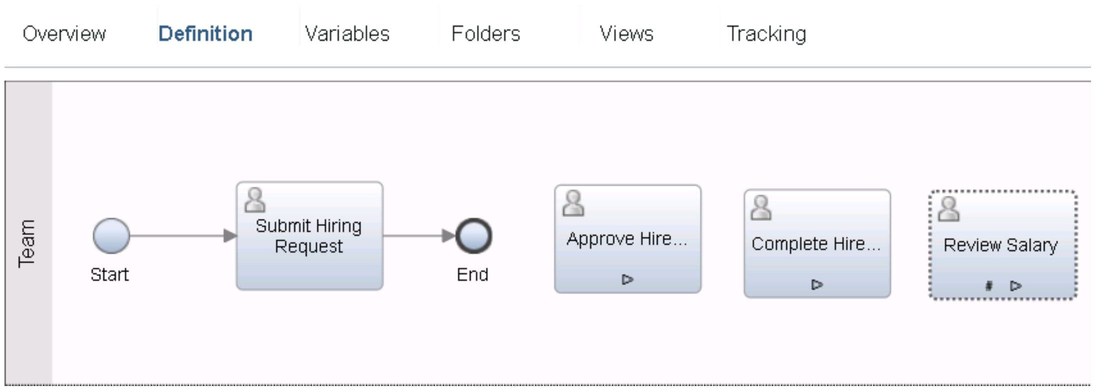
__ 2. Click **Processes**.



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



__ 4. The process opens in the web Process Designer editor.



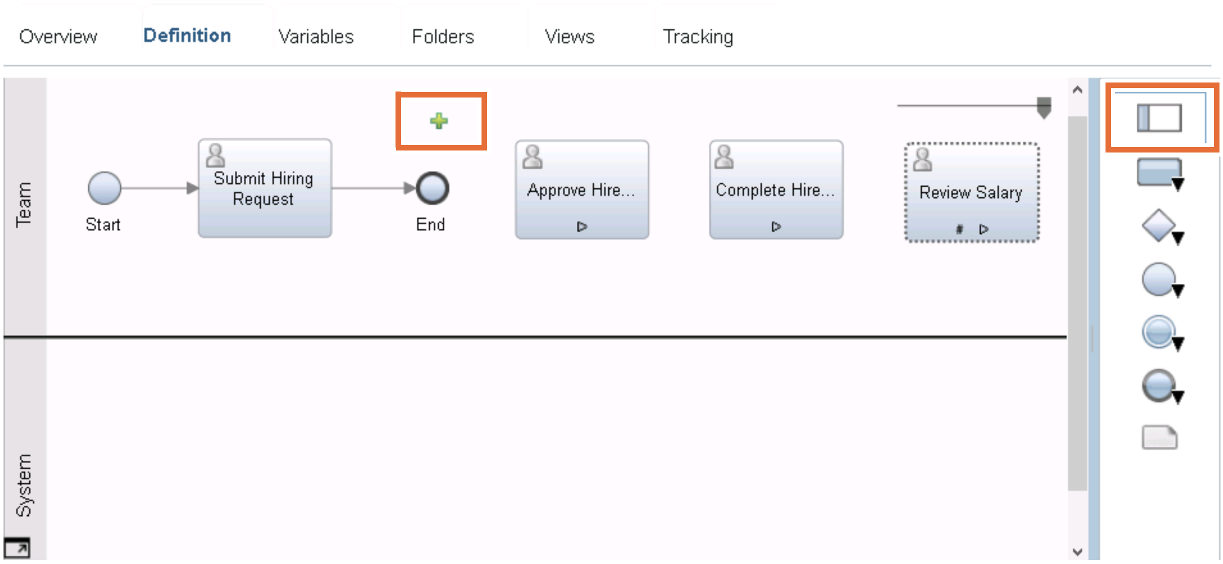
Part 2: Add lanes to the business process

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.

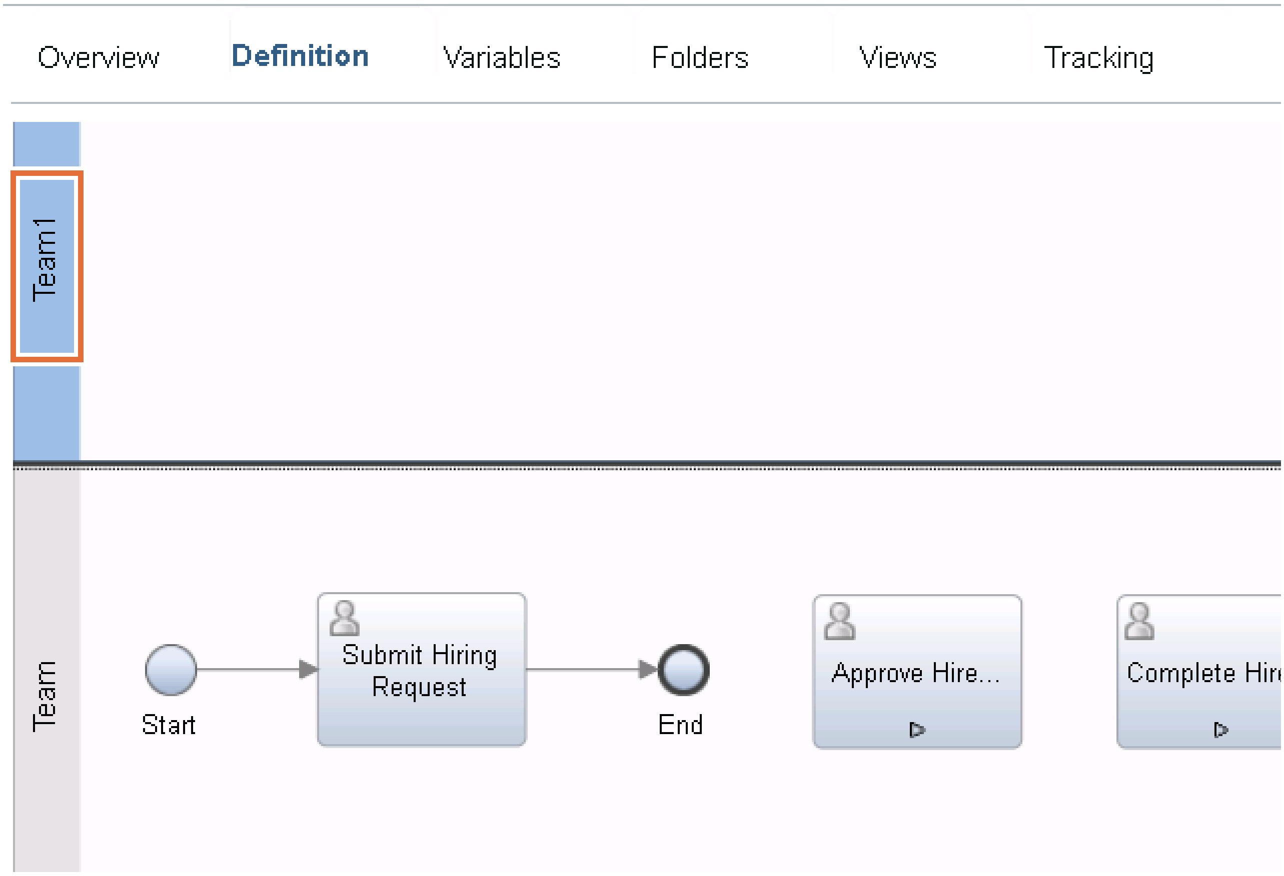
__ 1. Model the process teams.

__ 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. There is also one System team 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.

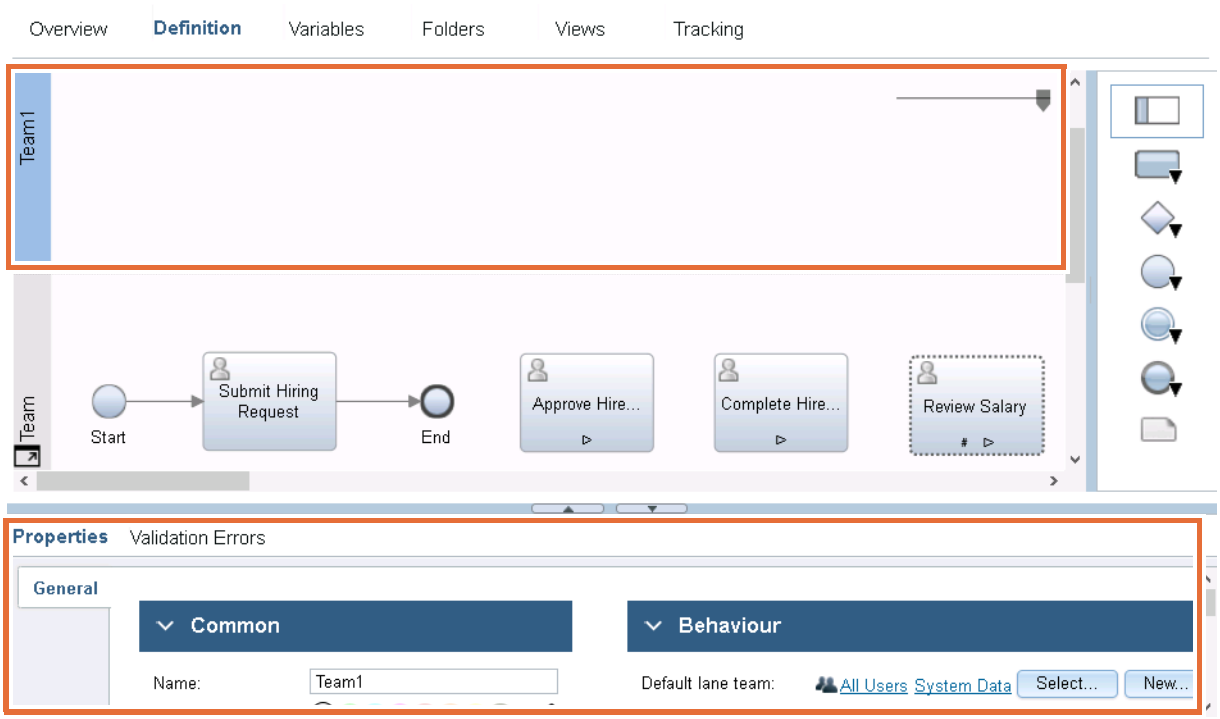


__ c. A lane with name Team1 is added to the process.

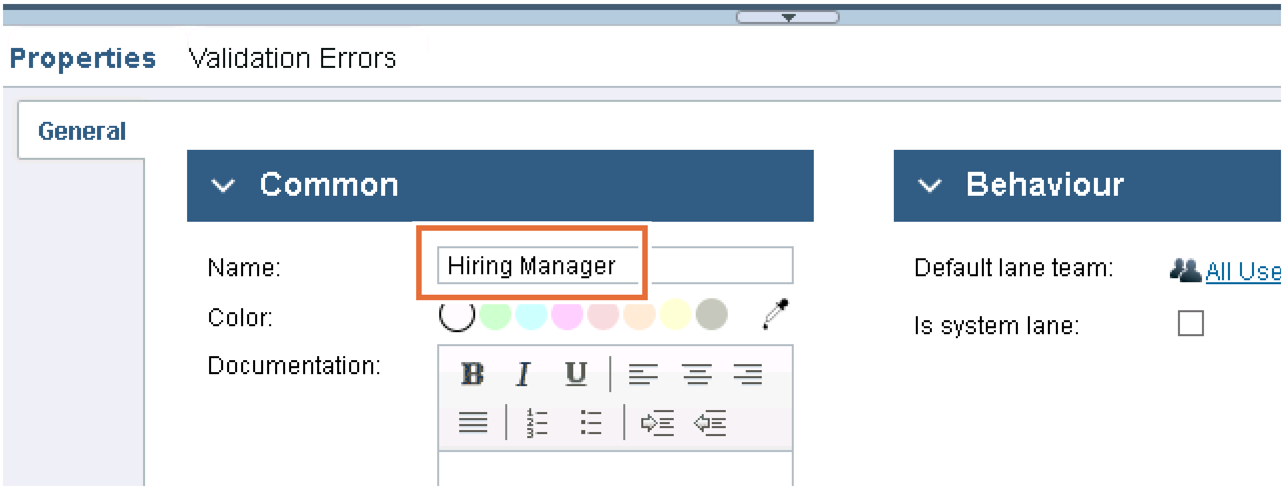


__ 2. Set the properties of the team lanes.

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



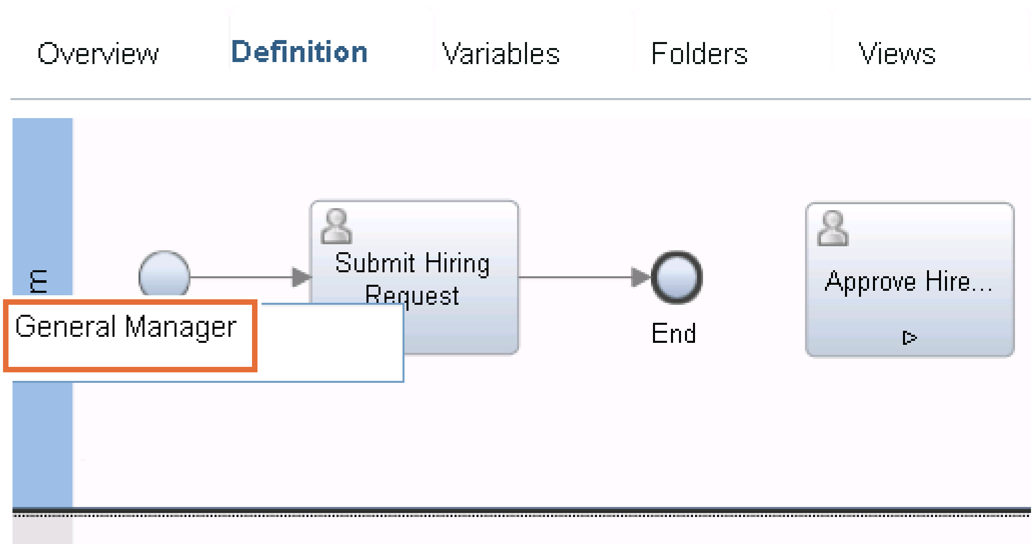
__ b. In the **Properties > General > Common** section, change the name of the top lane from the current name to: `Hiring Manager`



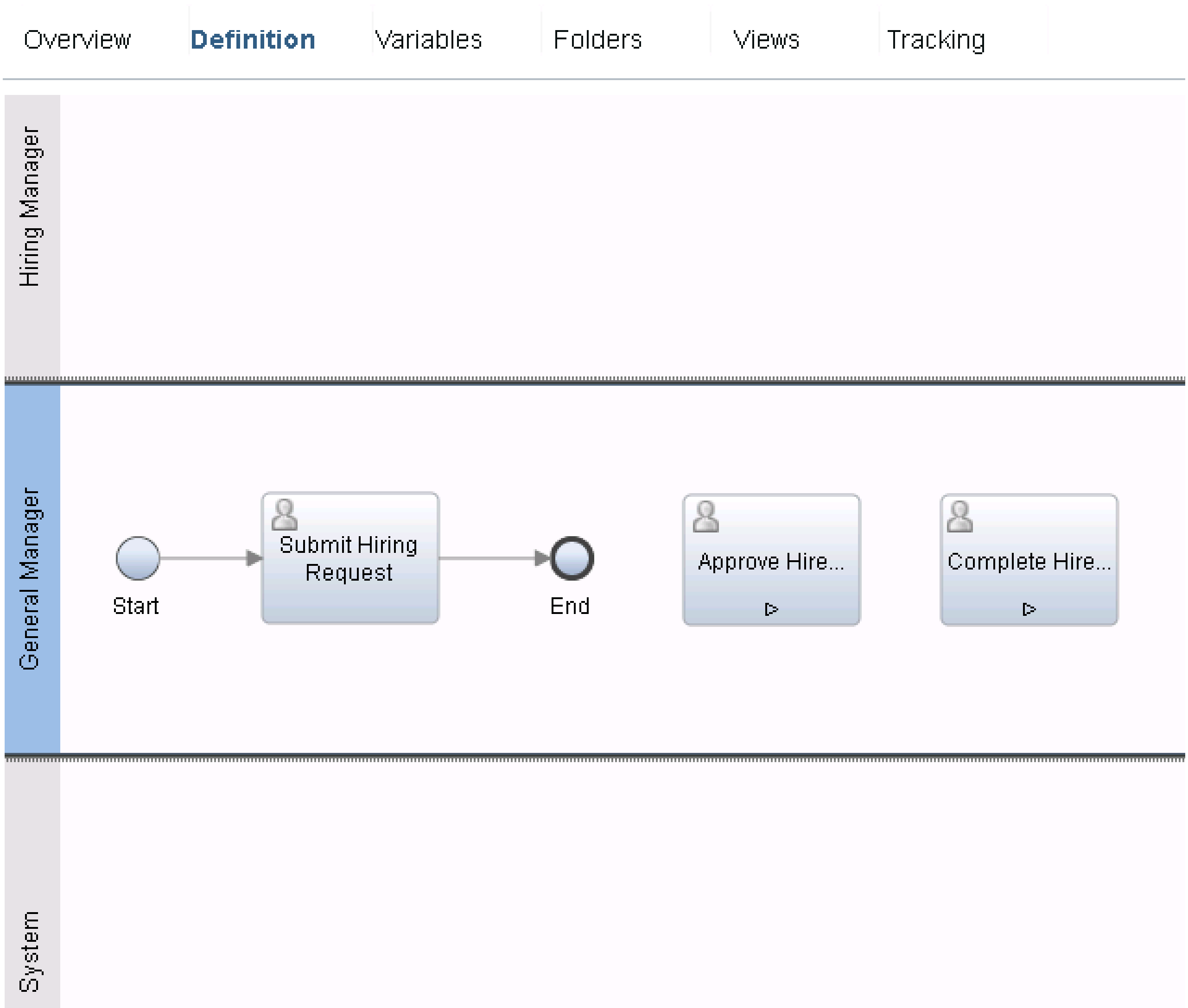
__ 3. Rename the second lane labeled Team.

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

__ b. Change the name of the second lane from `Team` to: `General Manager`



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

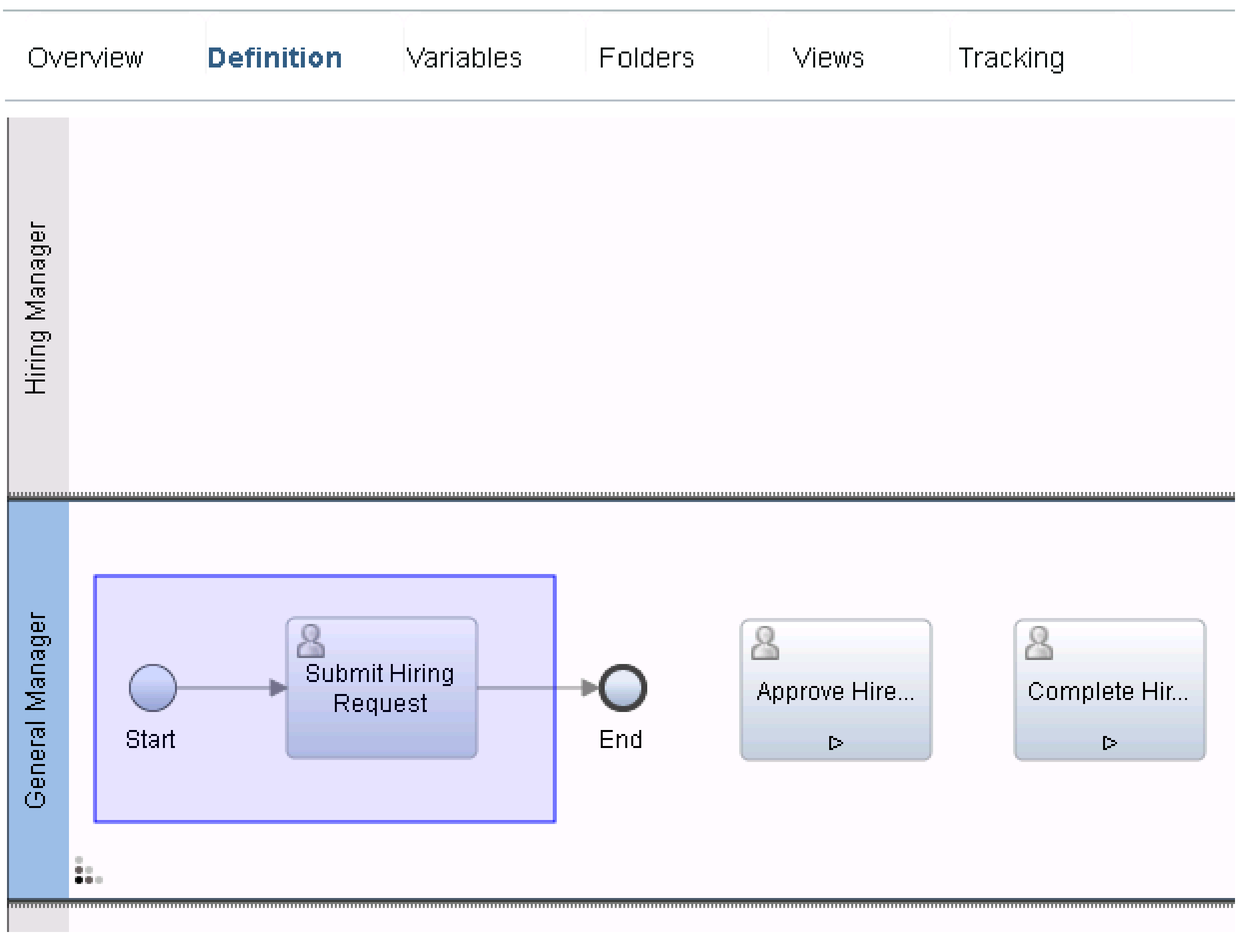
To accomplish the task of adding activities and more events to the process model, use the process requirements that are provided at the beginning of Exercise 2. Read the main process requirements and determine the activities from the requirements. Because the requirements were written down, it might 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:

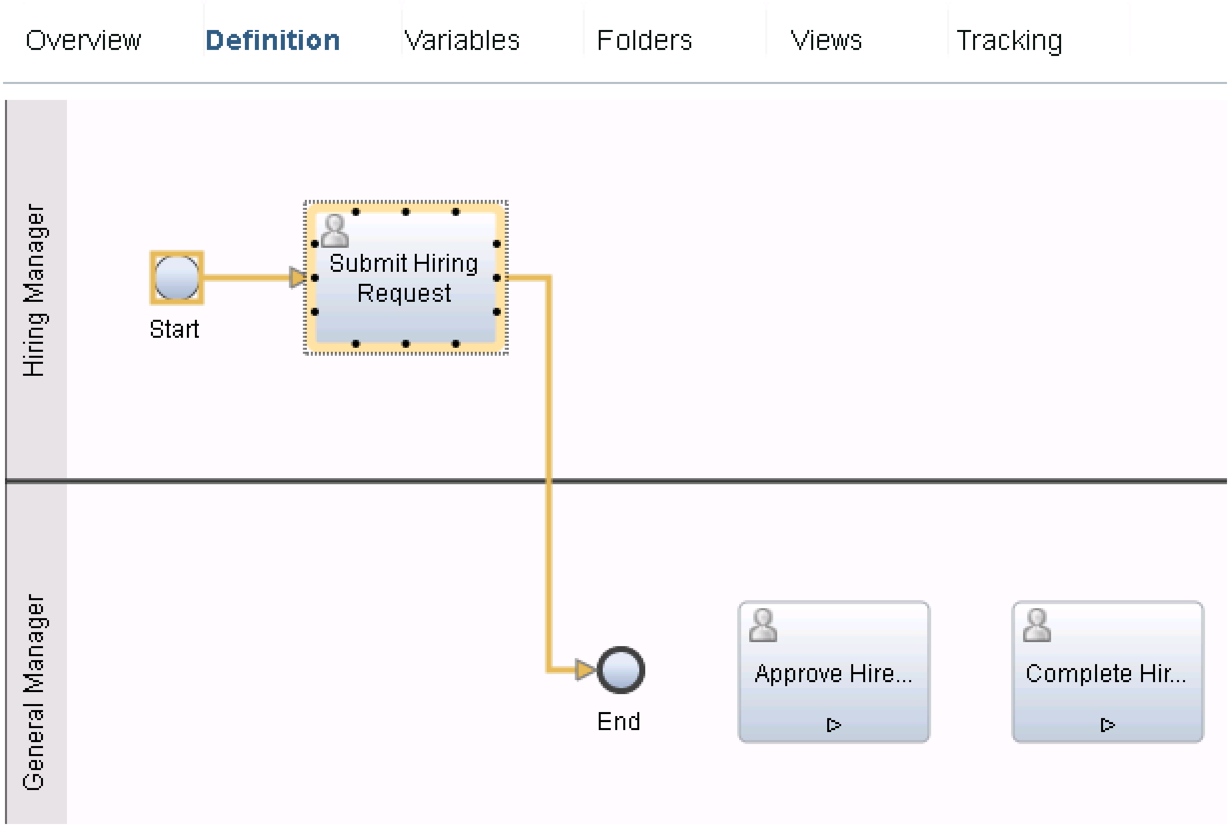
- Hiring Manager (team): Submit Hiring Request
- General Manager (team): Approve New Hire Request
- System (team): Approve Hire Request and Complete Hire Request

__ 1. Arrange the activities to the team lanes to establish the correct process flow. Model the activities and set their properties.

__ a. To drag the **Start** event and the **Submit Hiring Request** activity to the **Hiring Manager** lane, drag a box to select these two items in the process.

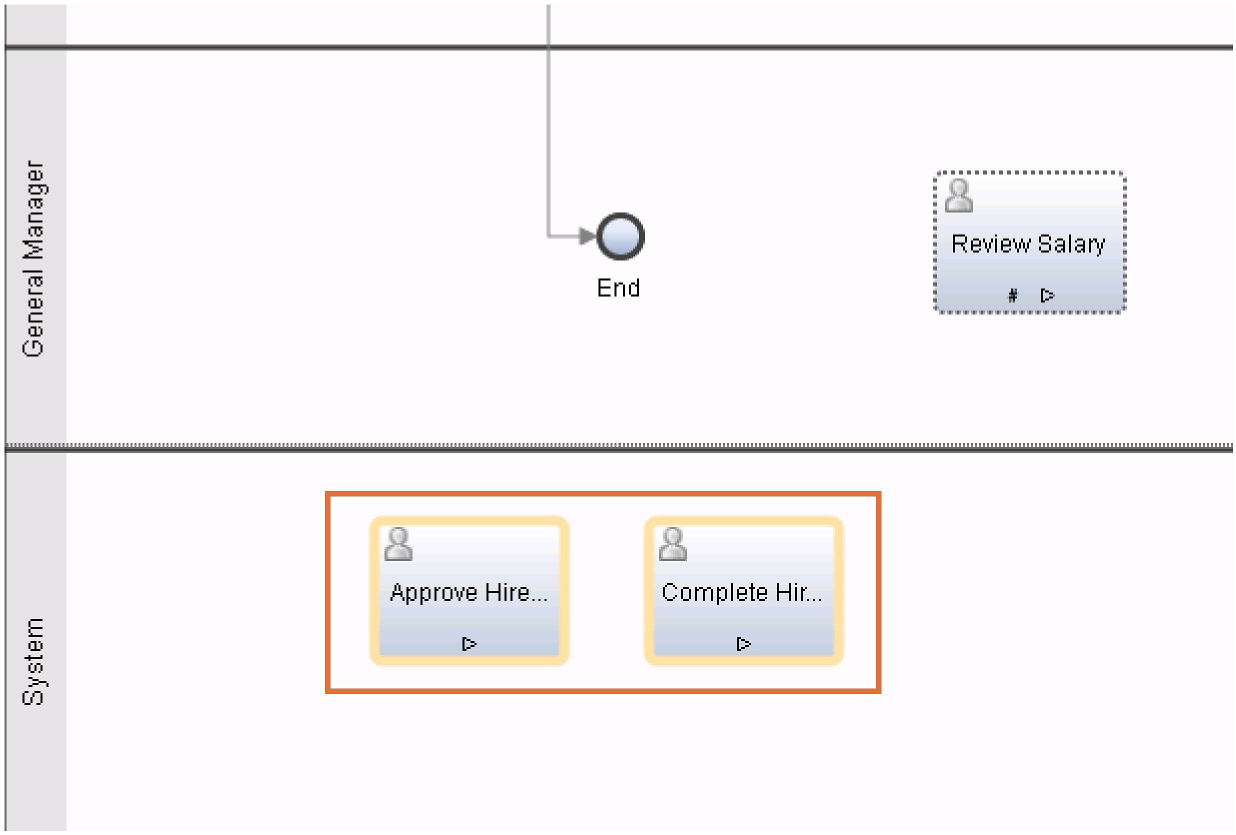


__ b. Drag the selected activities in the **Hiring Manager** lane.

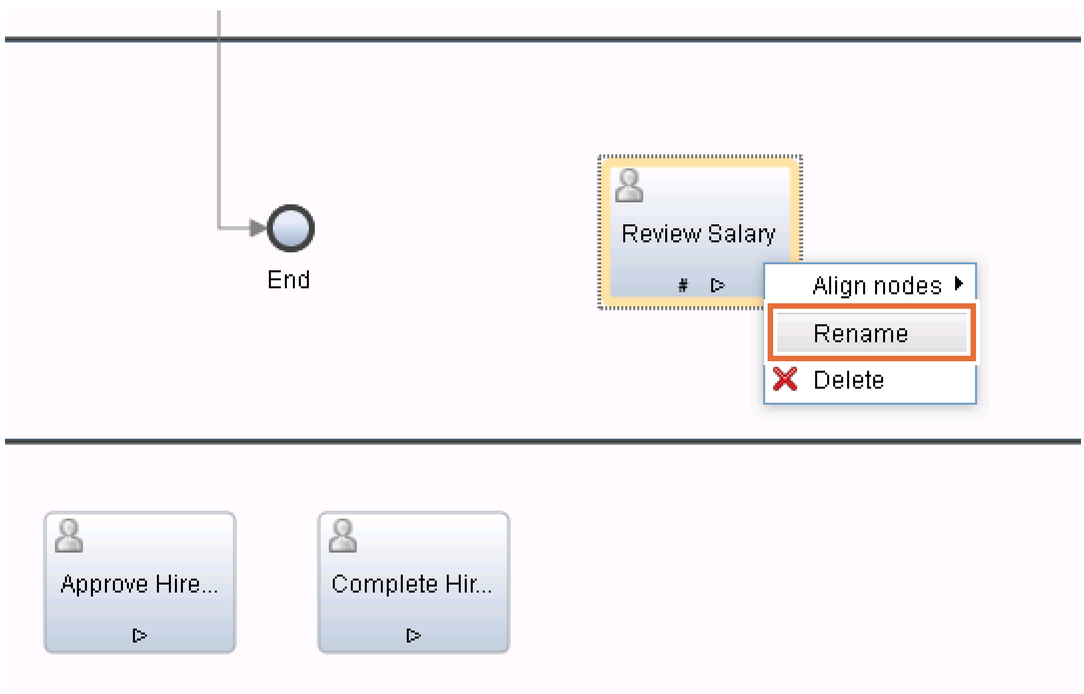


__ c. Drag a box to select the **Approve Hire Request** activity and the **Complete Hire Request** activity.

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



__ e. Right-click the **Review Salary** activity and select **Rename**.



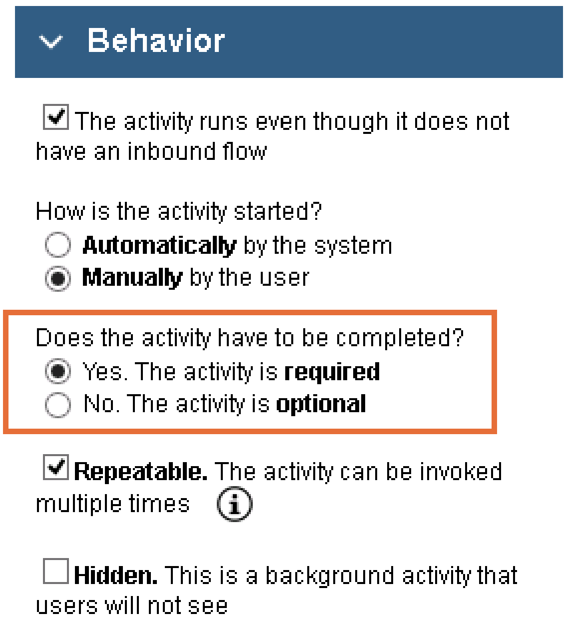
__ f. Enter the name: `Approve New Hire Request`

Reminder

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. Select the **Approve New Hire Request** activity.

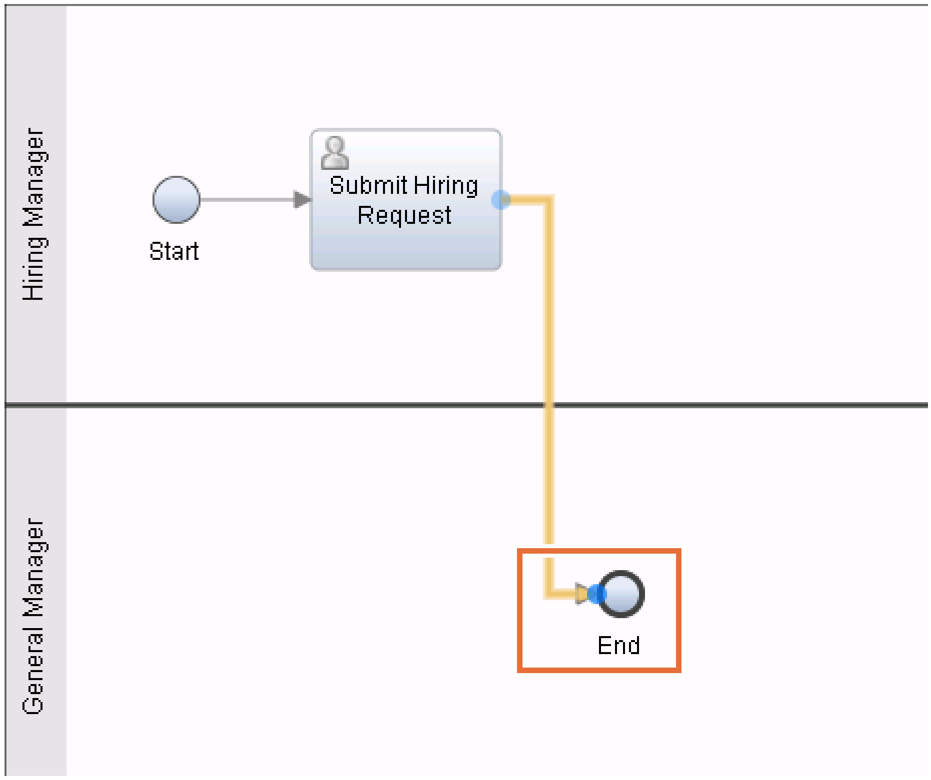
__ h. In the **Properties > General > Behavior** section, to change this activity from optional to required, select the option: **Yes. The activity is required**.



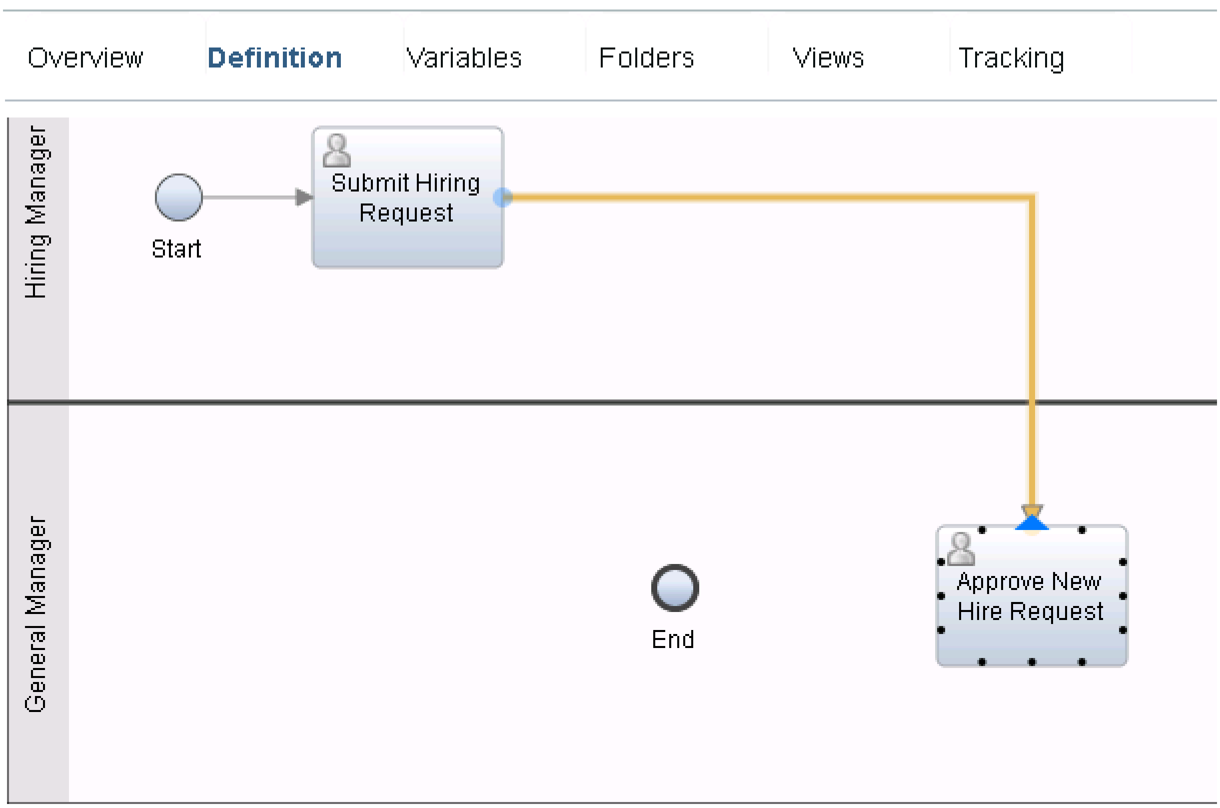
__ 2. Add sequence flow to the components in the diagram.

__ a. Select the sequence flow between **Submit Hiring Request** activity and the **End** event.

__ b. Hover over the edge of **End** event to see a blue endpoint of the sequence flow.

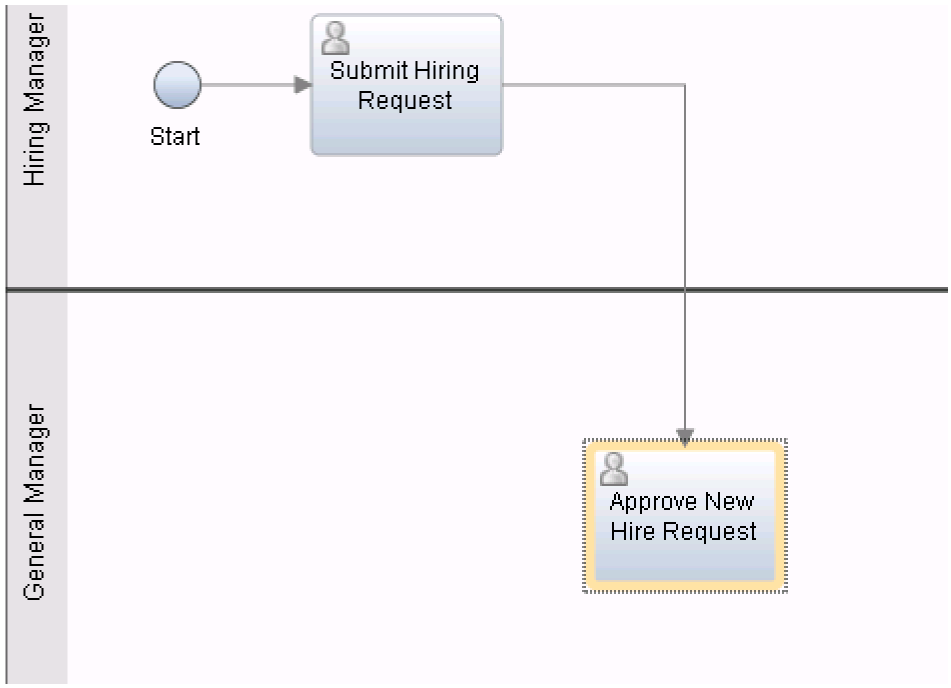


__ c. Hold this blue point and drag it to an anchor point on the **Approve New Hire Request** activity in the General Manager lane.



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

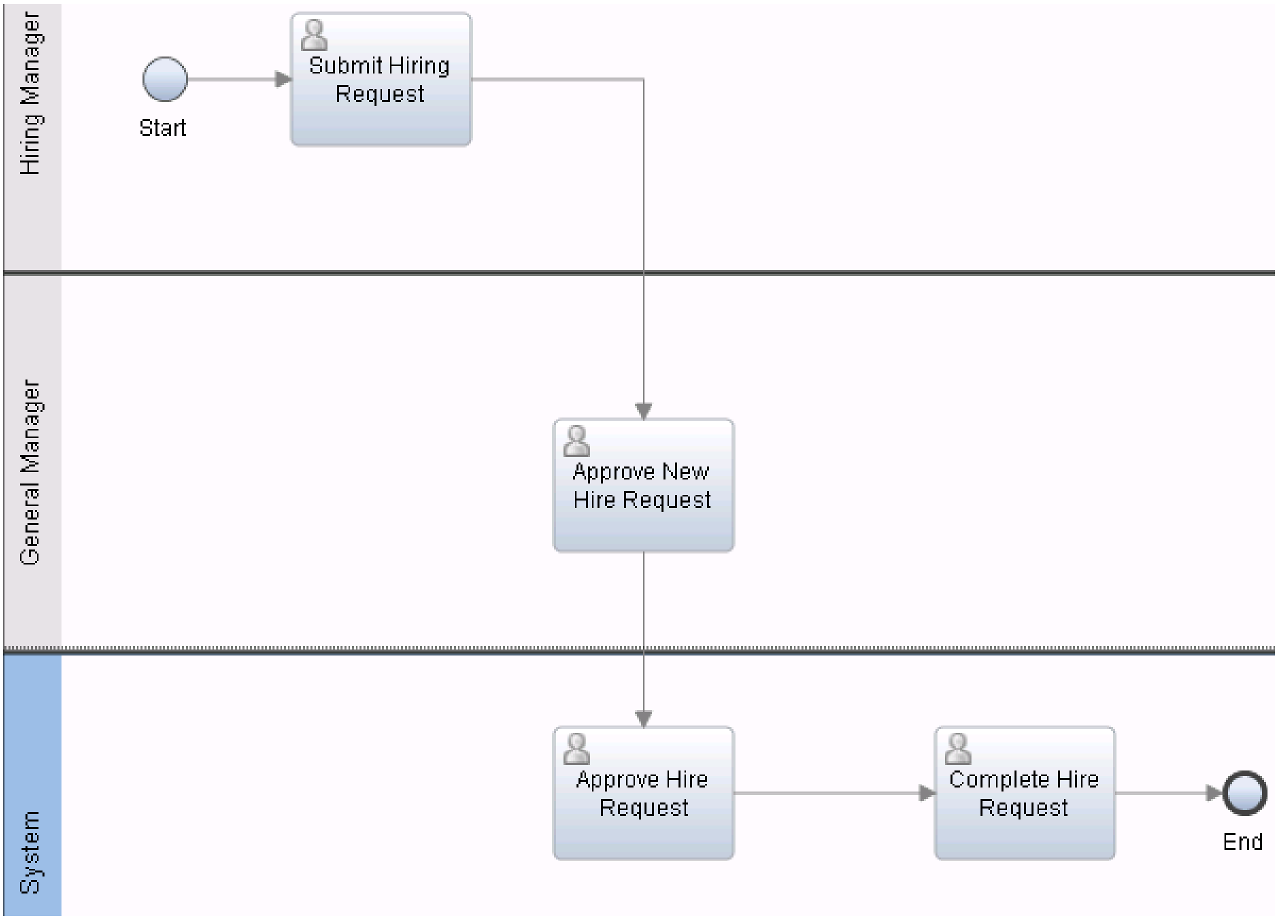
__ e. Select the **Approve New Hire Request** activity and drag it below the Submit Hiring Request activity to align the activities with straight flow lines in the process.



__ 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.

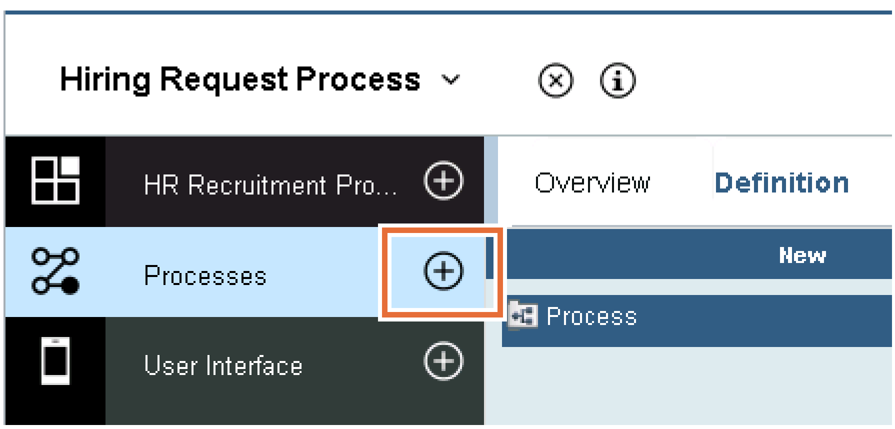
Because more than one team is involved in the **Approve Hire Request** activity, it is no longer a "task" but a nested process. The nested process has lanes, such as System, HR Administrator, and Hiring Manager. Activities include Check Hire Request, Override Hire Request, and Negotiate Hire Request.

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.

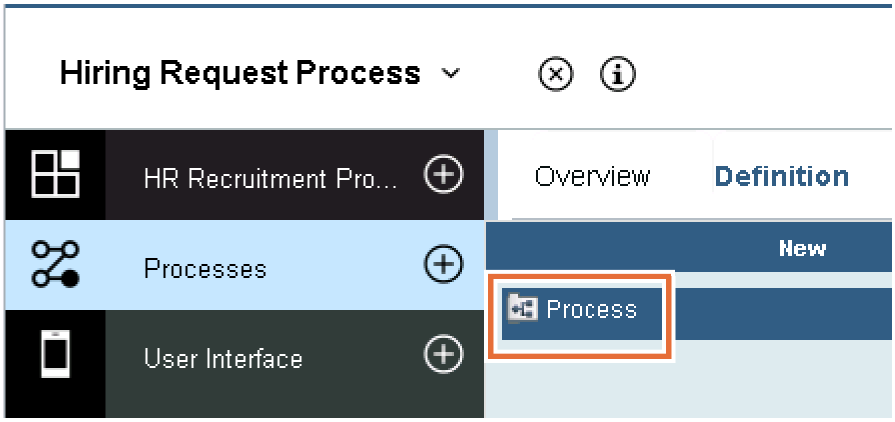
Approve Hire Request is a nested process and might be reused in other HR processes, so you create a linked process for this scenario.

__ 1. Create a process that is called **Approve Hire Request**.

__ a. From the process library, use your mouse to hover over **Processes** and then click the (**+**) plus sign.

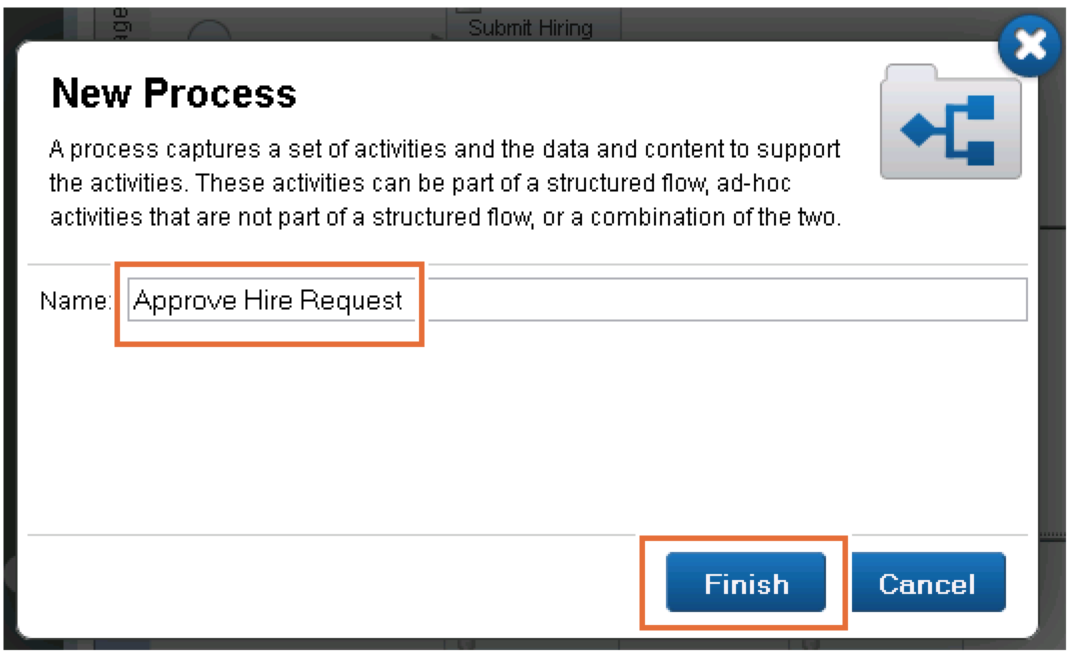


__ b. From the **New** menu, click **Process**.



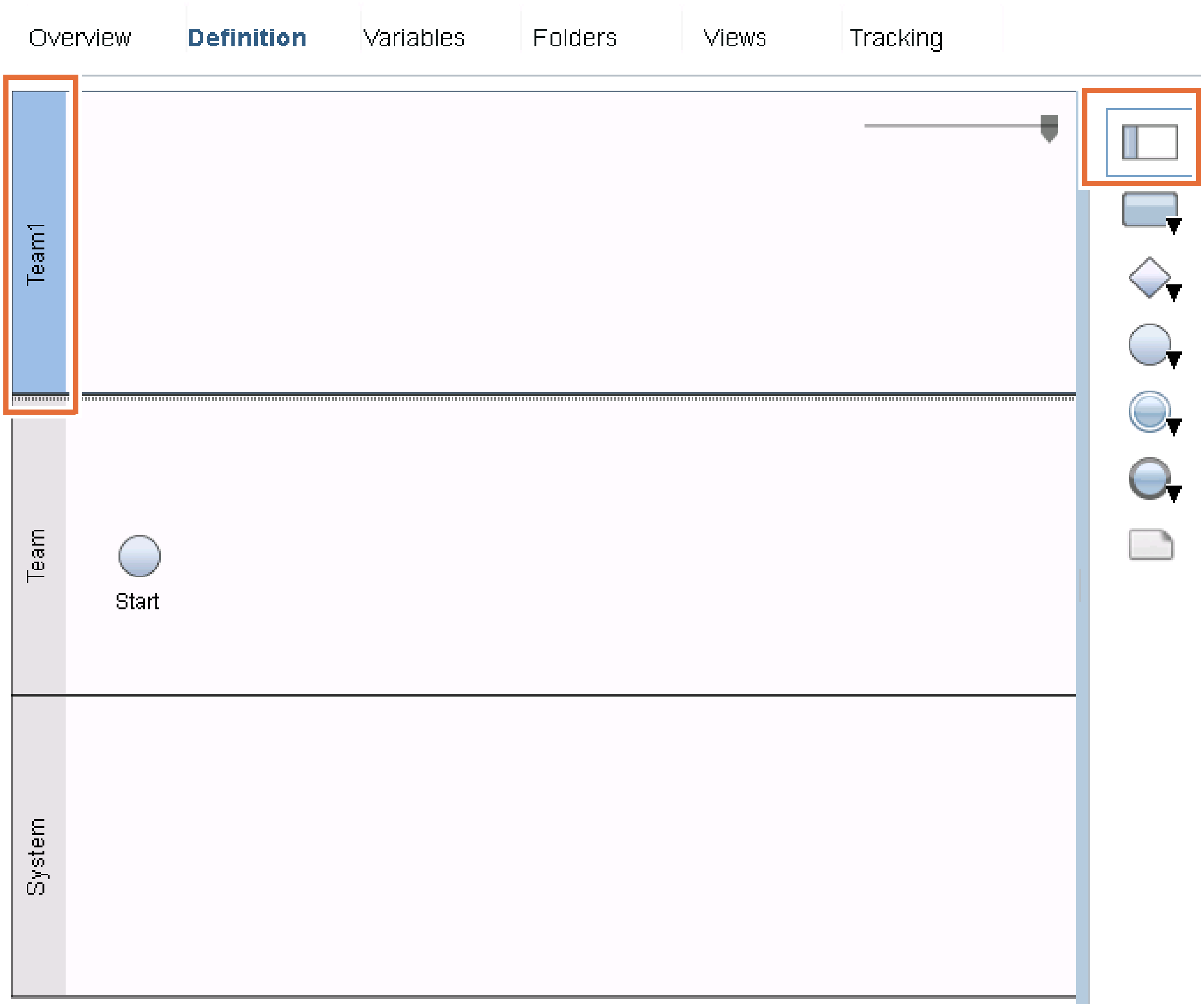
__ c. Enter `Approve Hire Request` for the **Name**.

__ d. Click **Finish**.



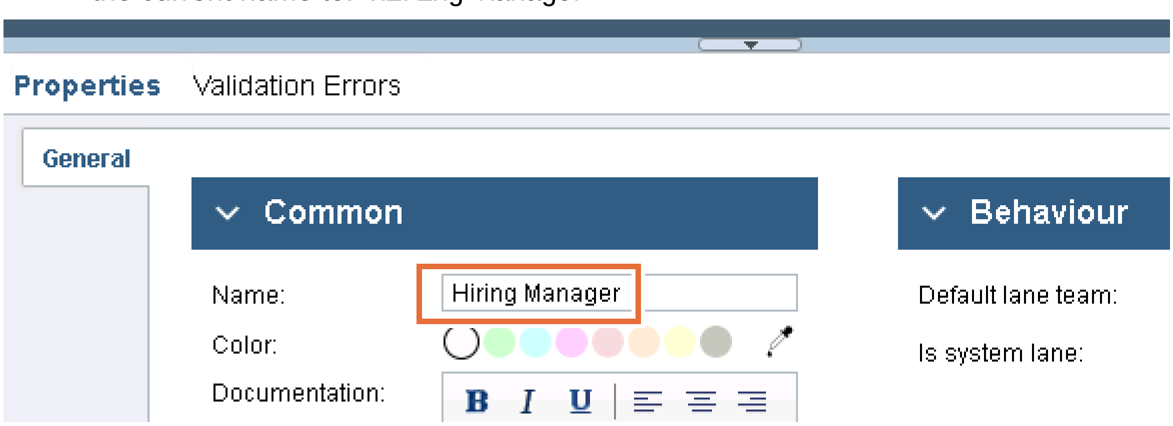
__ 2. Create the System, HR Administrator, and Hiring Manager lanes.

__ a. Click the **Lane** icon from the palette and drag one lane from the palette to the canvas above the default Team lane.

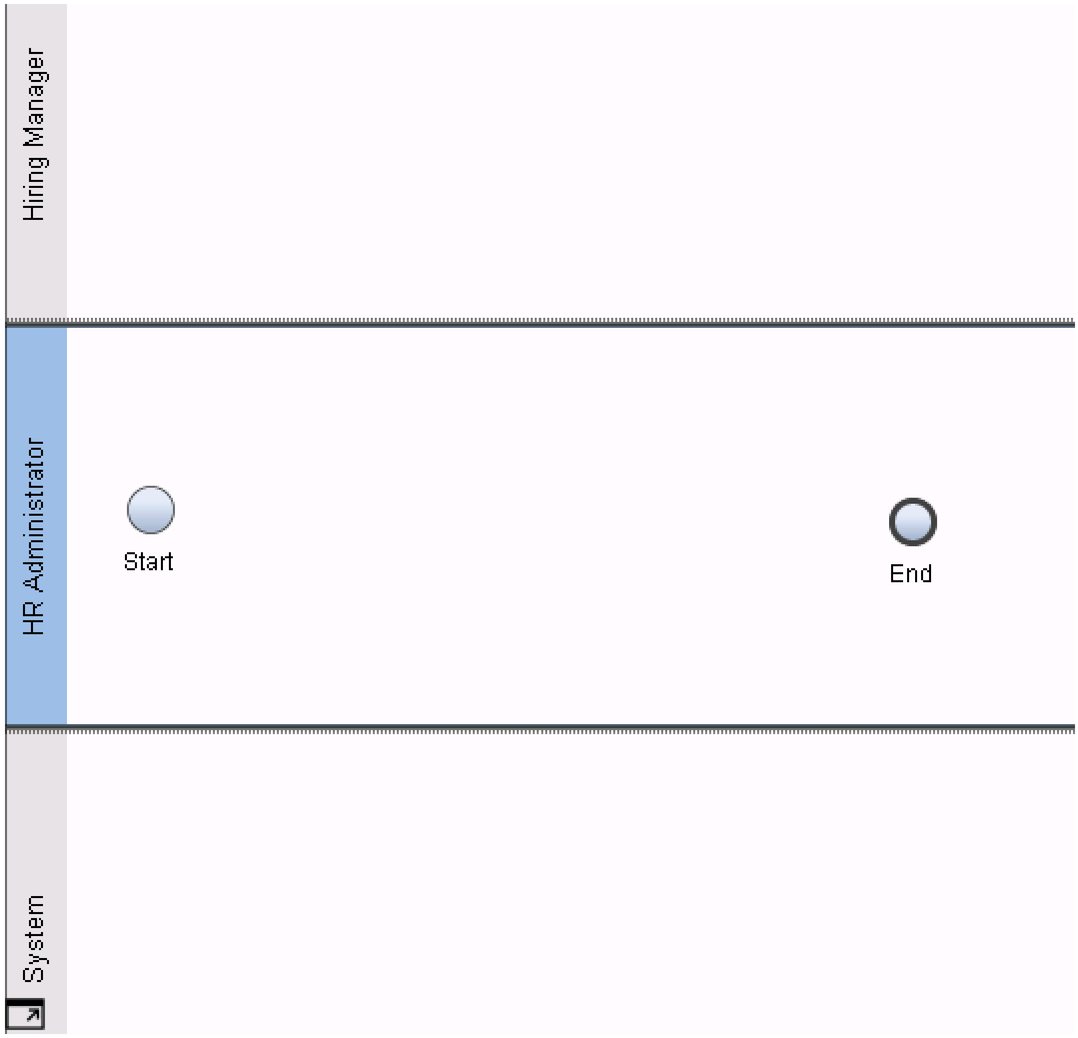


__ b. Click the **top lane** to select the entire lane. The Properties tab populates at the bottom of the screen.

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



__ d. Select the second lane, **Team**, and rename the lane to: `HR Administrator`

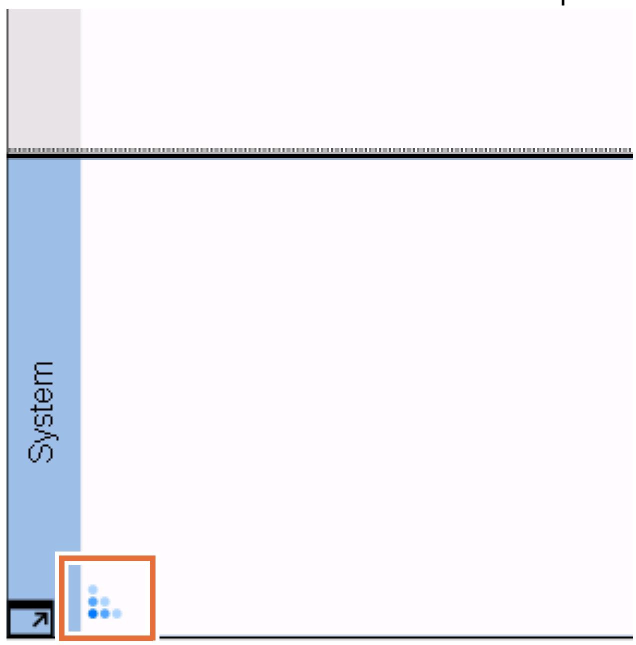


Reminder

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

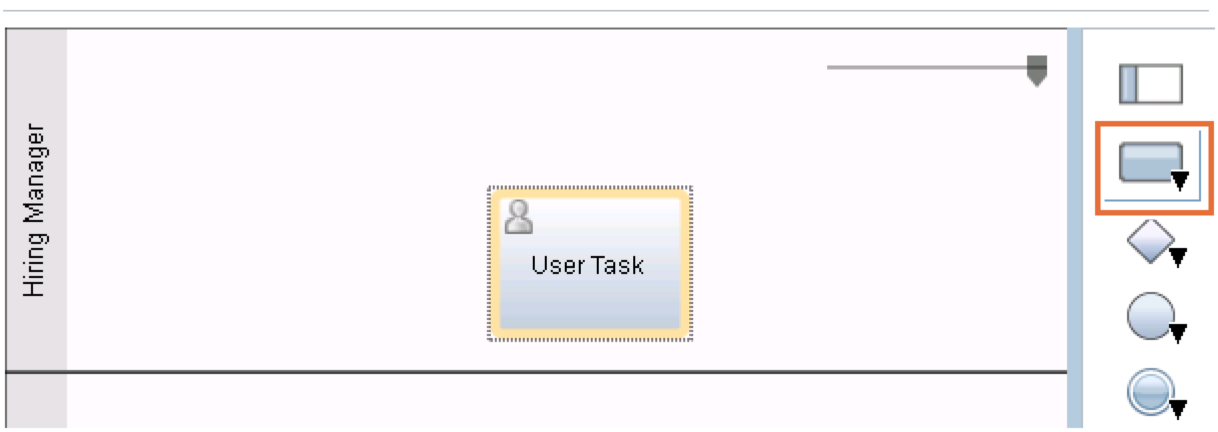
Note

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.

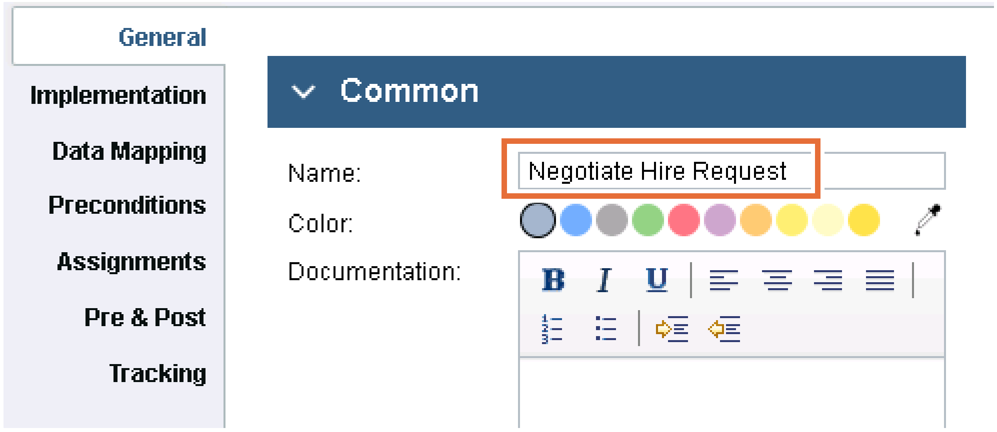


__ 3. Create the Check Hire Request, Override Hire Request, and Negotiate Hire Request activities.

__ a. Drag an activity from the palette to the **Hiring Manager** lane.



__ b. Rename the activity to: `Negotiate Hire Request`



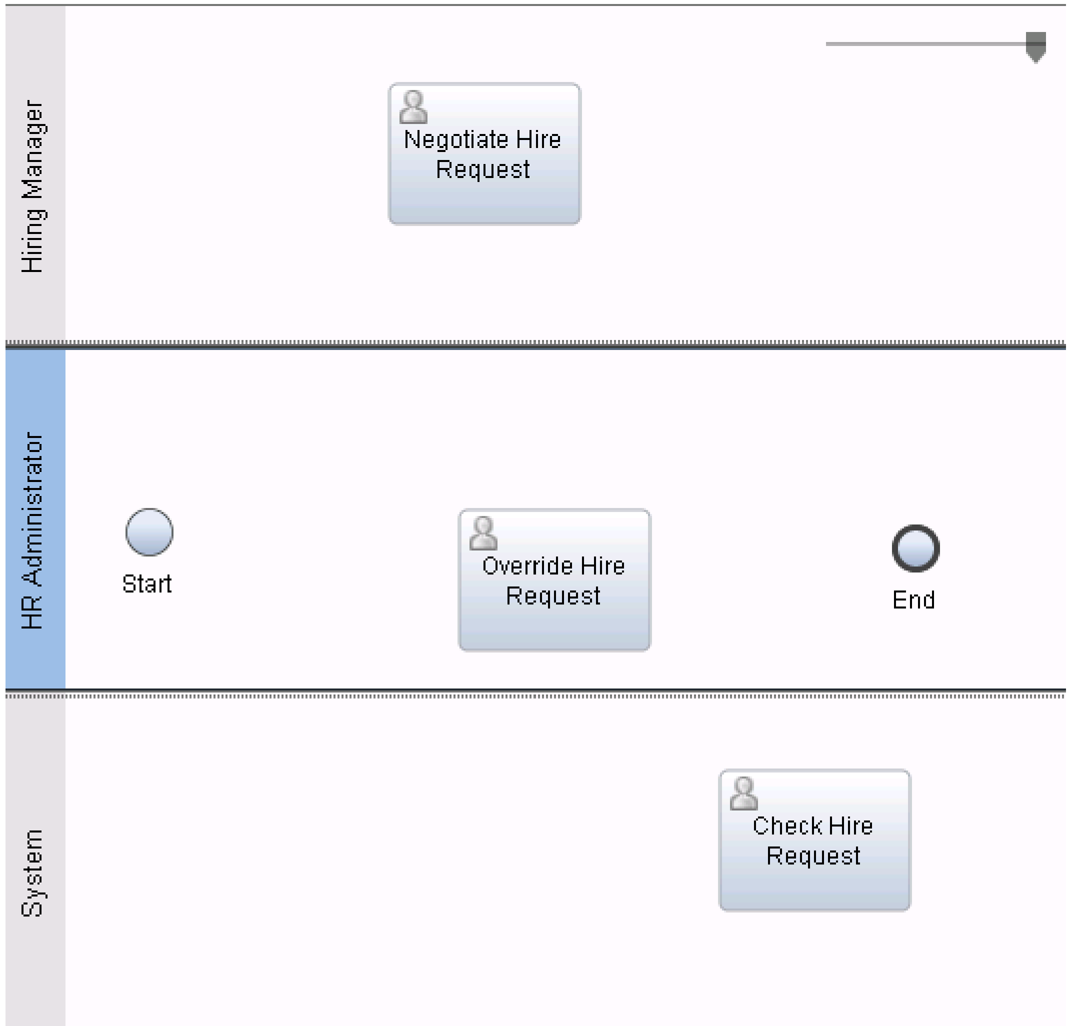
Reminder

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

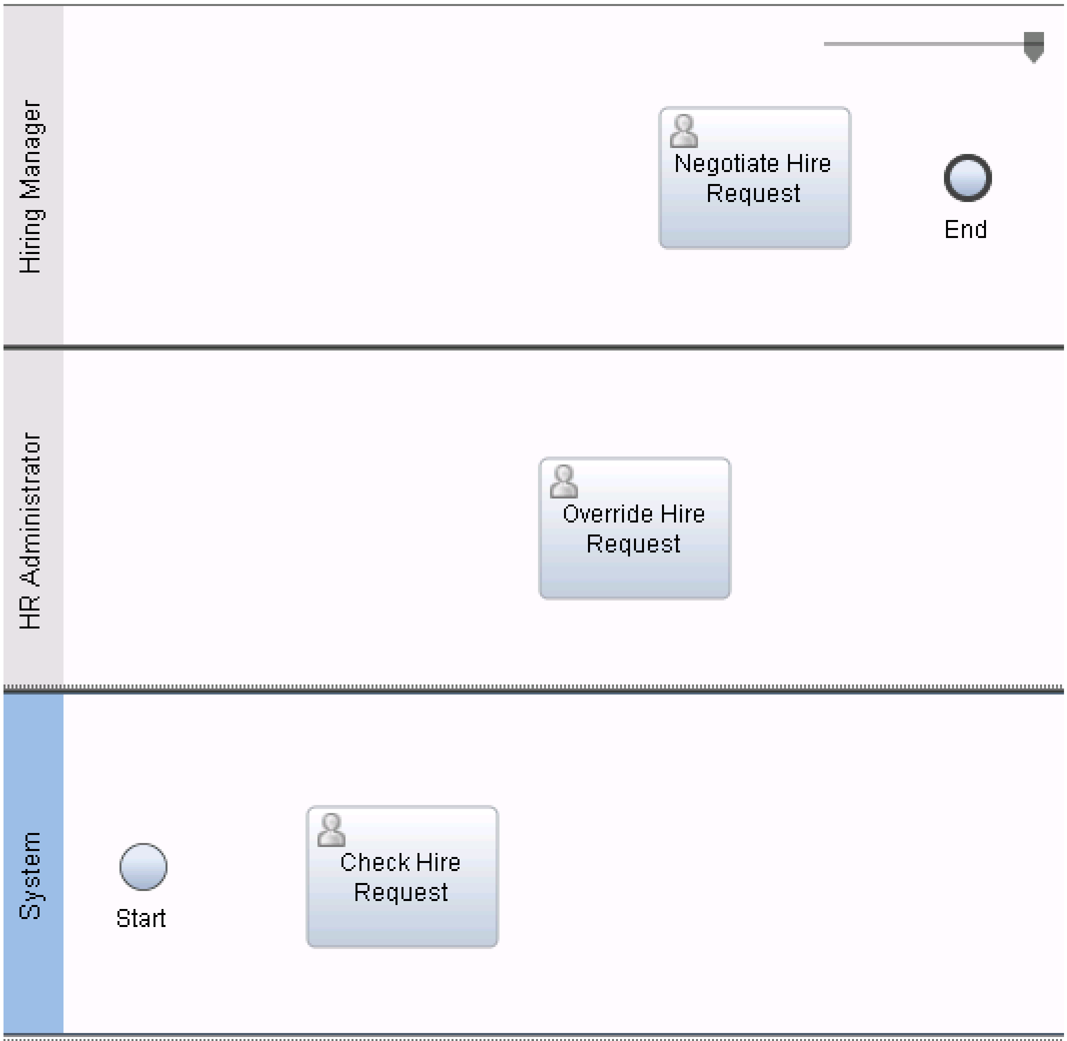
__ c. Repeat dragging two more activities to their corresponding lanes and changing the names until your process is complete and the locations of your activities match the nested diagram table.

Table 1. Nested diagram

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


Hiring Manager
Negotiate Hire Request
HR Administrator
Start
Override Hire Request
End
System
Check Hire Request

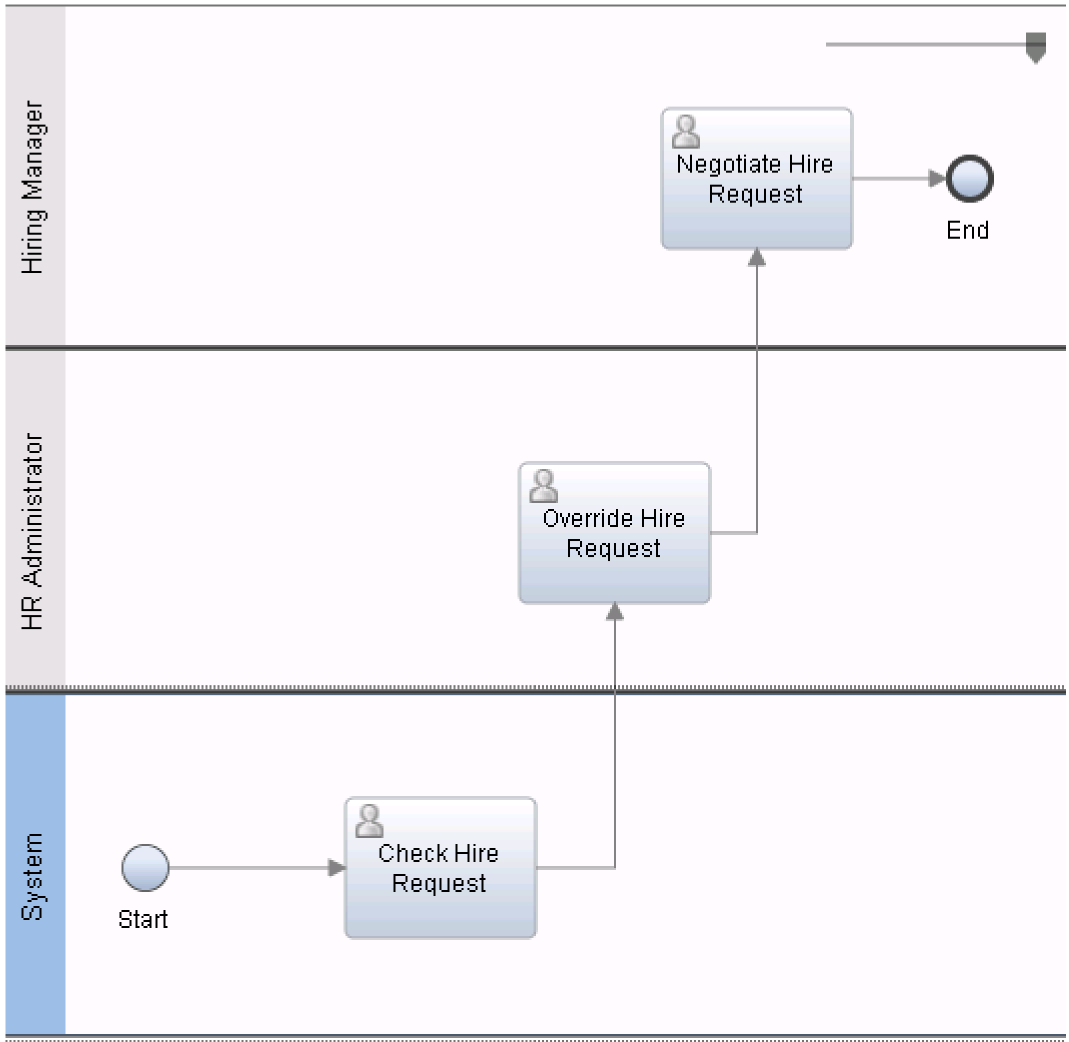
__ d. Move your **Start** event before the first activity, **Check Hire Request** in the System lane, and the **End** event after the **Negotiate Hire Request** activity in the Hiring Manager lane. Arrange the activities in an upward diagonal line.



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.

__ e. Finally, create a sequence flow from the **Start** event to **Check Hire Request**, from **Check Hire Request** to **Override Hire Request**, from **Override Hire Request** to **Negotiate Hire Request**, and then from **Negotiate Hire Request** to the **End** event.



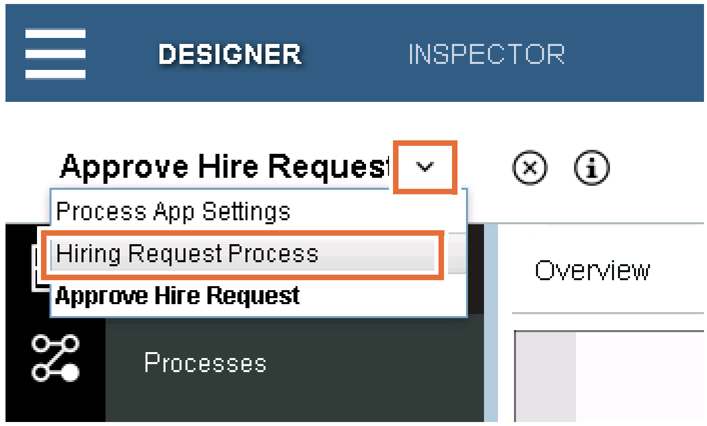
Reminder

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.

__ f. Save your process application.

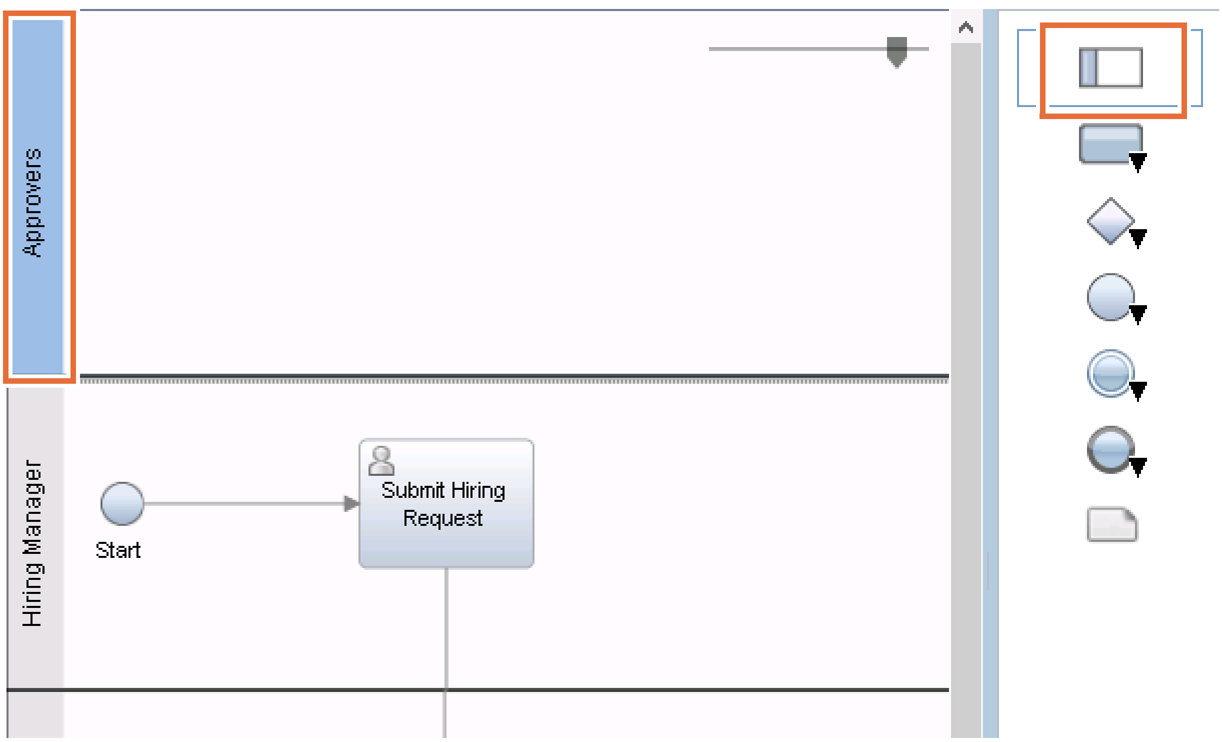
Part 5: Attach the linked process

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

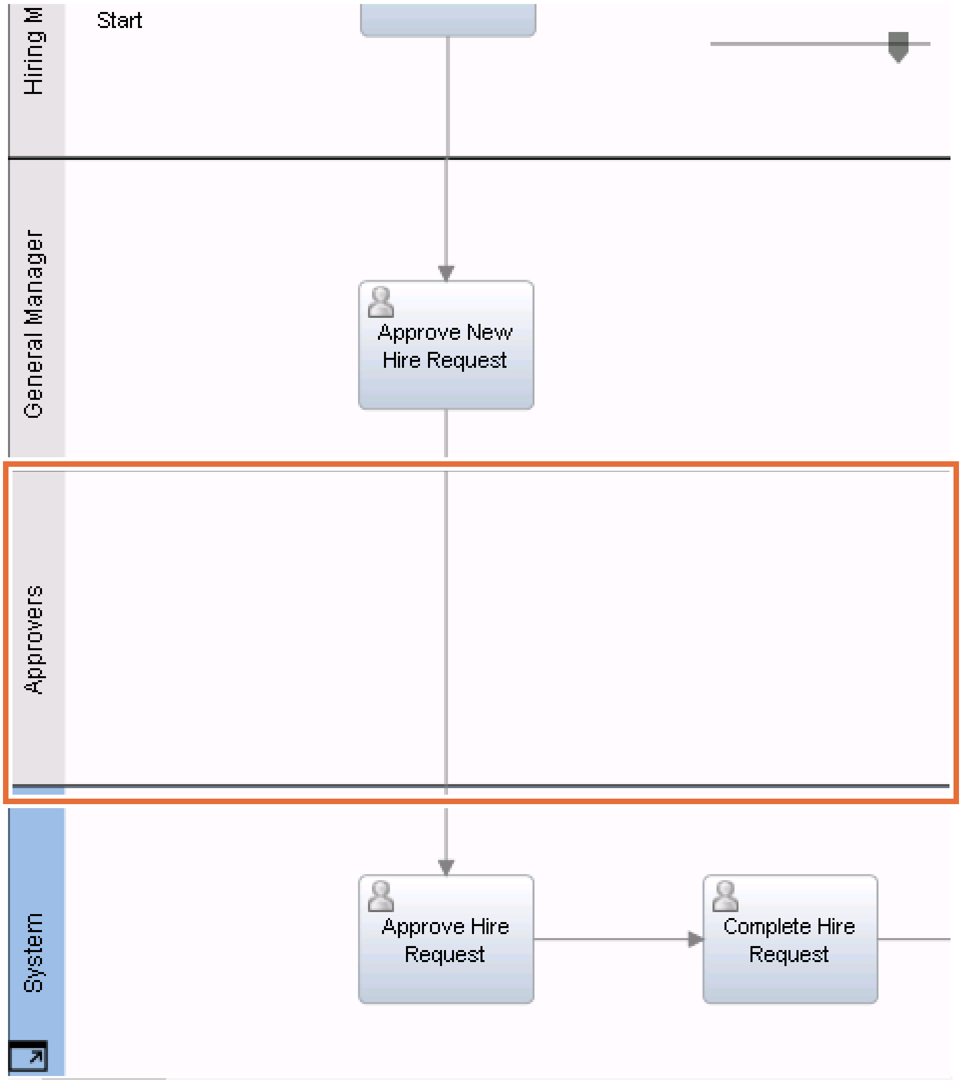


__ 2. Change the implementation of the existing **Approve Hire Request** activity to a nested process.

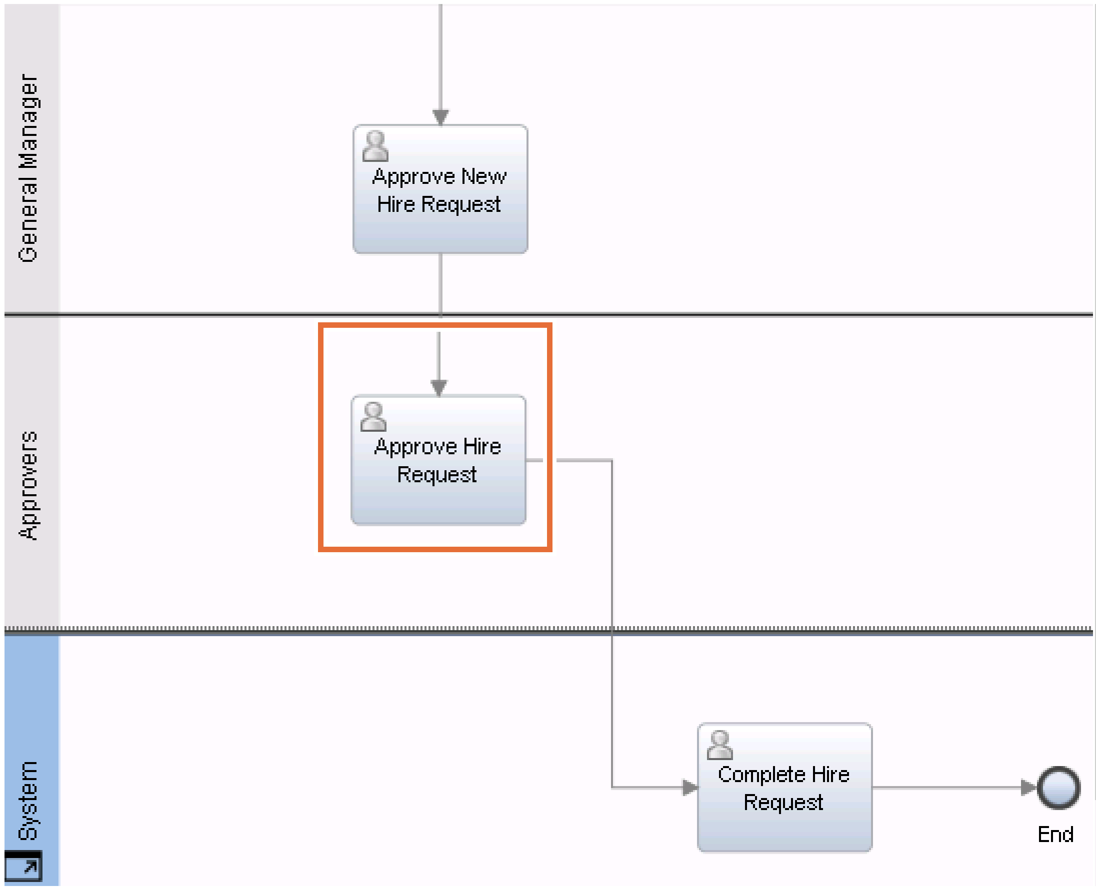
__ a. Drag a new lane above the Hiring Manager lane in the diagram and name it: `Approvers`



__ b. Select the **Approvers** lane and drag the lane down so that it is just above the **System** lane.



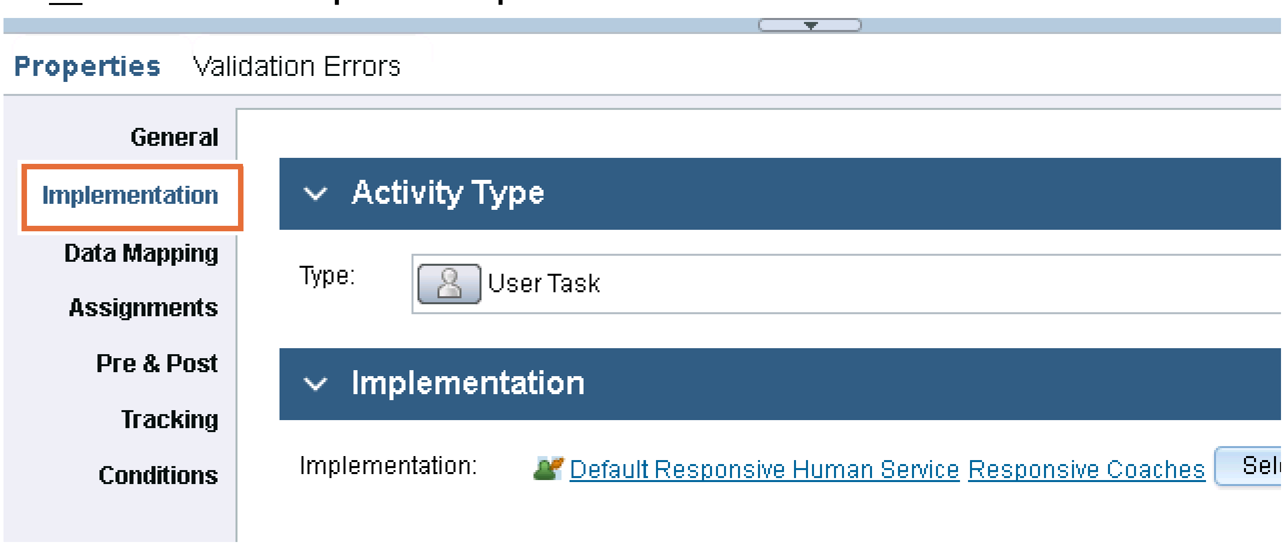
__ c. Drag the existing **Approve Hire Request** activity into the **Approvers** lane.



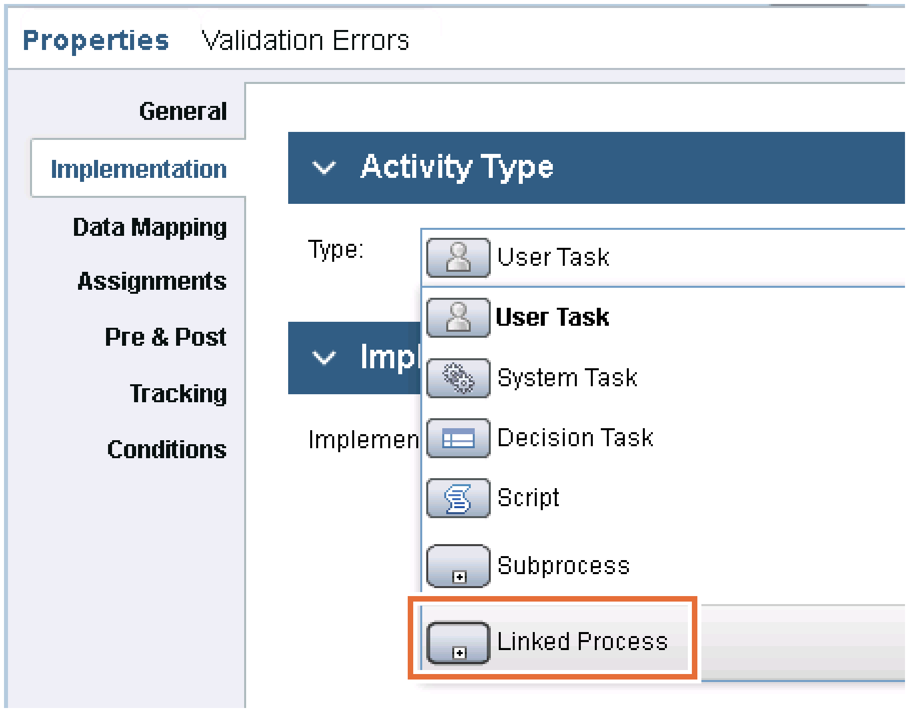
Reminder

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

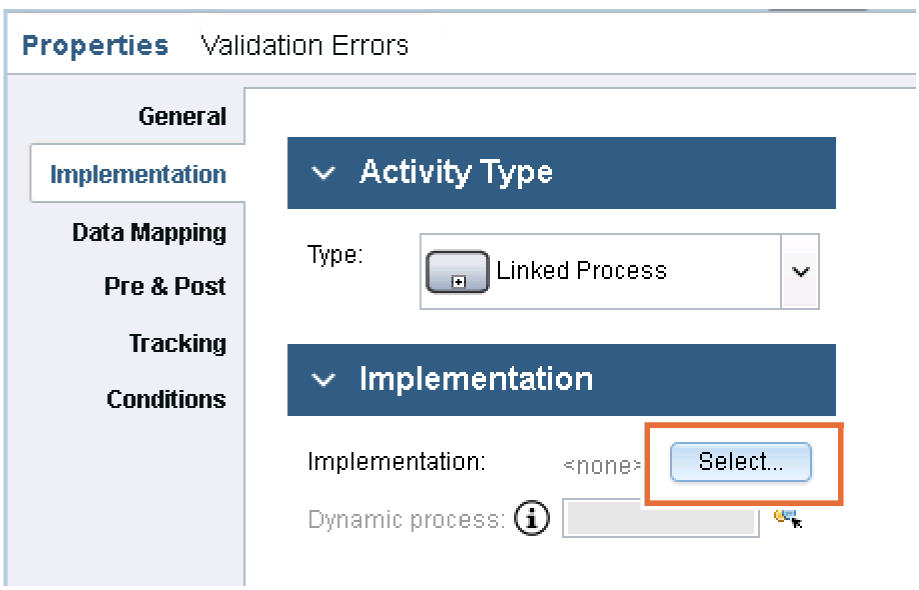
__ d. Select the **Properties > Implementation** menu.



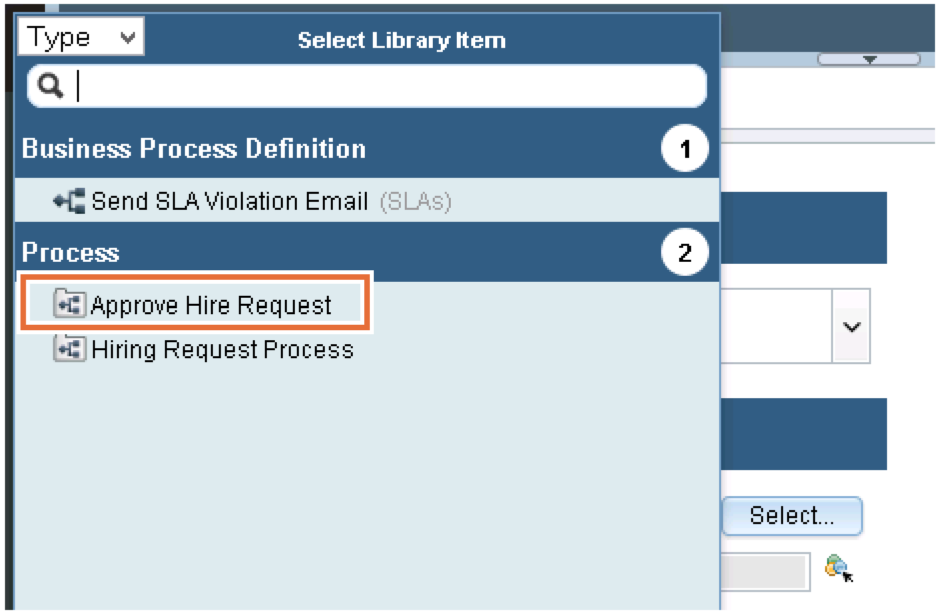
__ e. In the Activity Type section, select the Type as **Linked Process**.



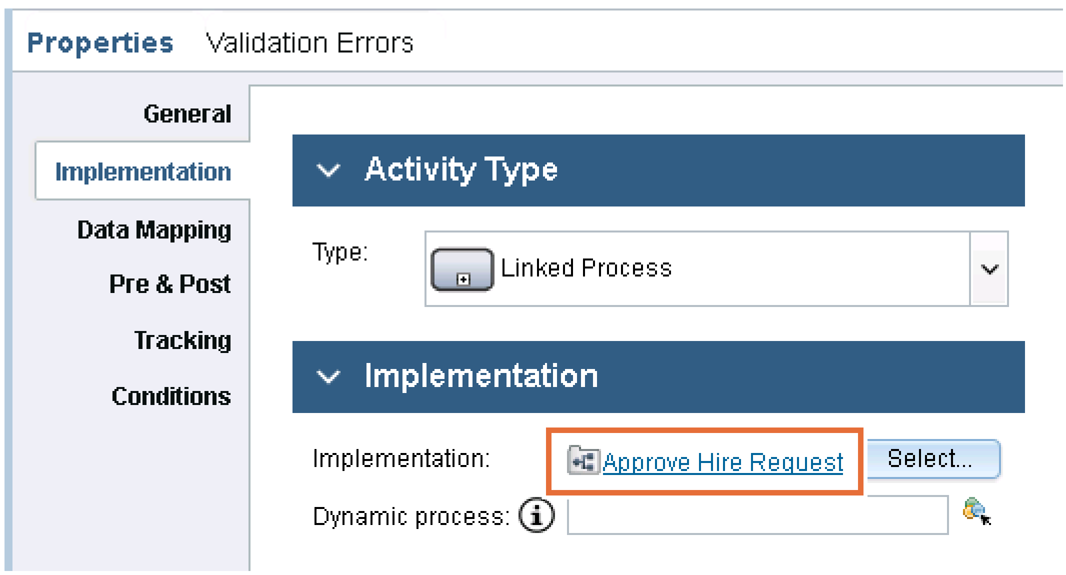
__ f. Under the Implementation section, click **Select** next to Implementation.



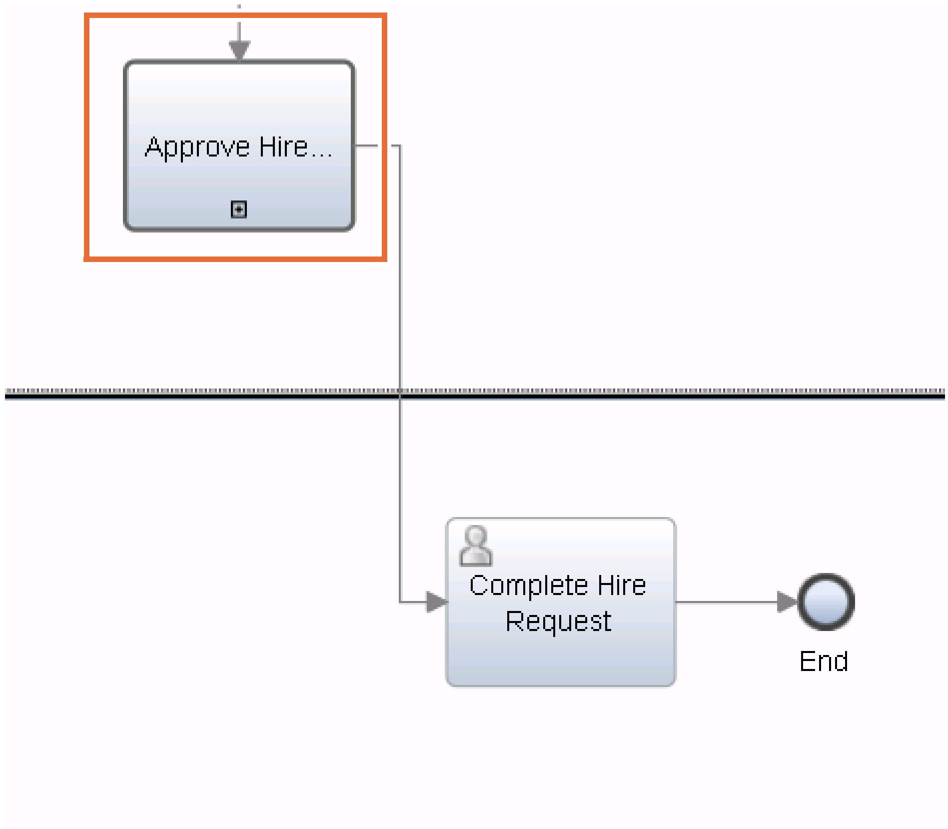
__ g. Select **Approve Hire Request** from the Process menu.



__ 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 Web 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

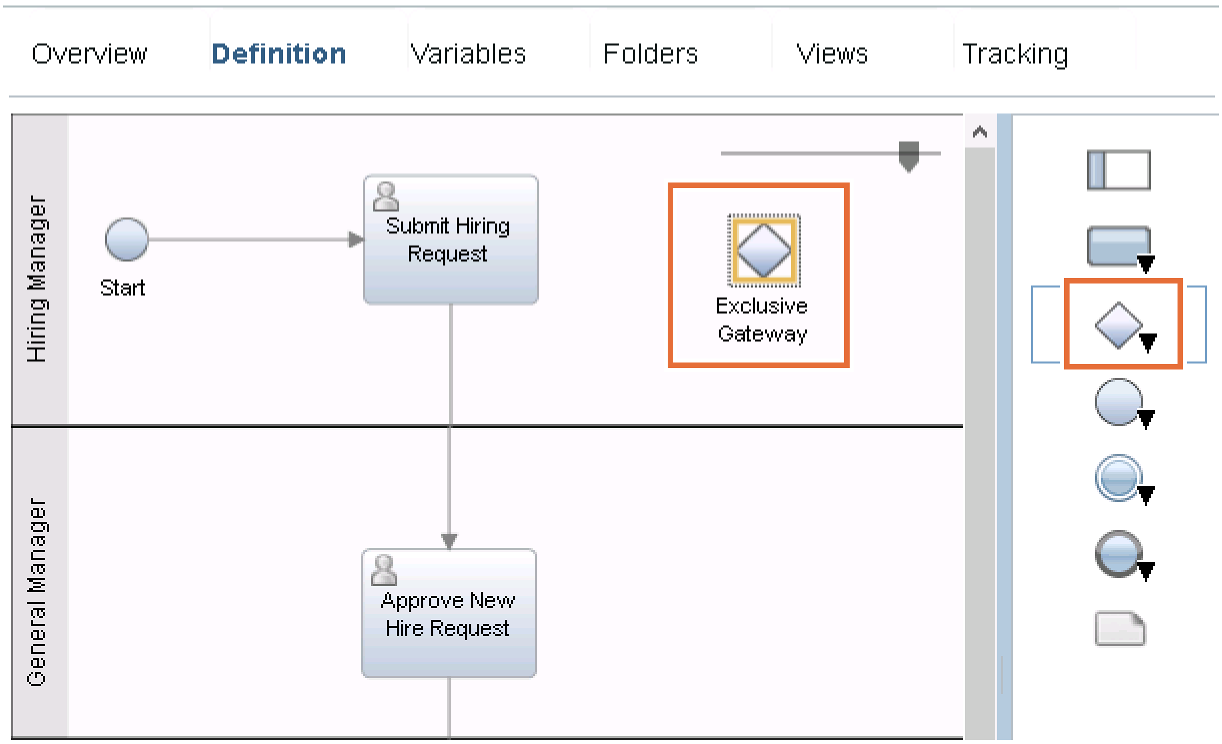
Part 1: Create gateways for parent process

Process flow controls are known as 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 a **Gateway** from the palette onto the canvas to the right of the **Submit Hiring Request** activity.

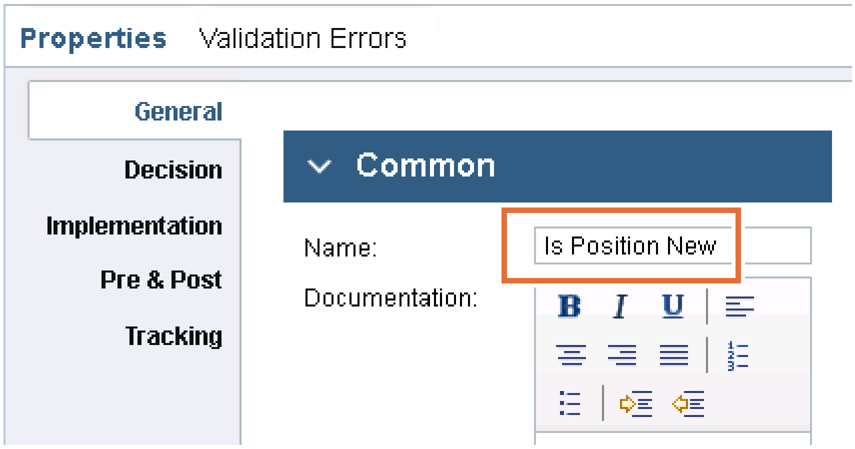


Note

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**.

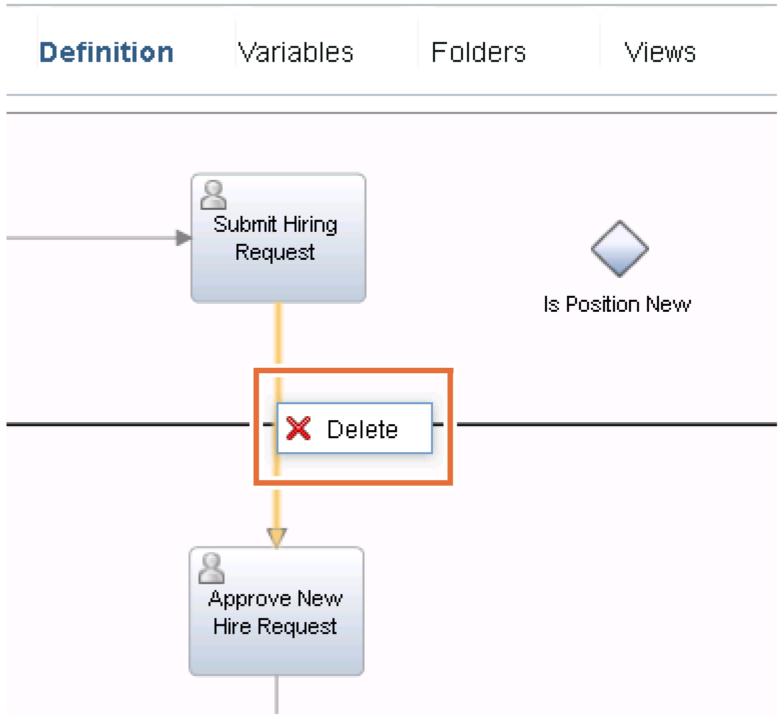


__ b. In the **Properties > General > Behavior** section, verify that `Exclusive Gateway` is the **Gateway type**.

__ 3. Connect the gateway.

__ a. Select the sequence flow between **Submit Hiring Request** and **Approve Hew Hire Request**.

__ b. While selected, right-click the sequence flow and click **Delete**.



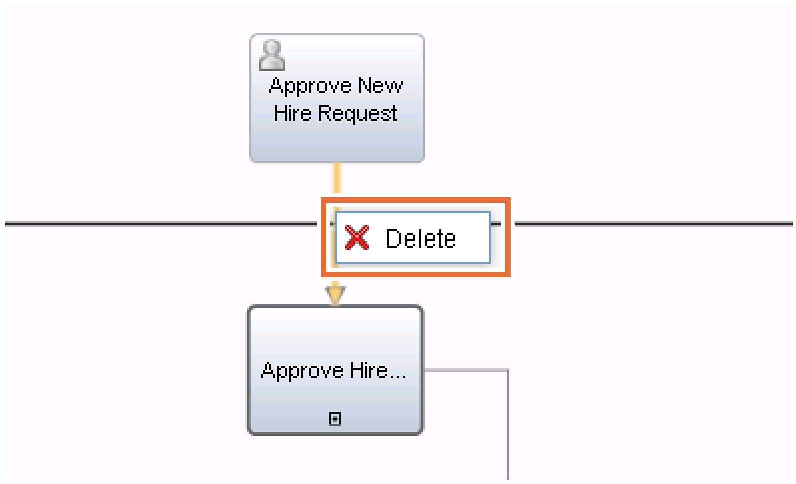
Note

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

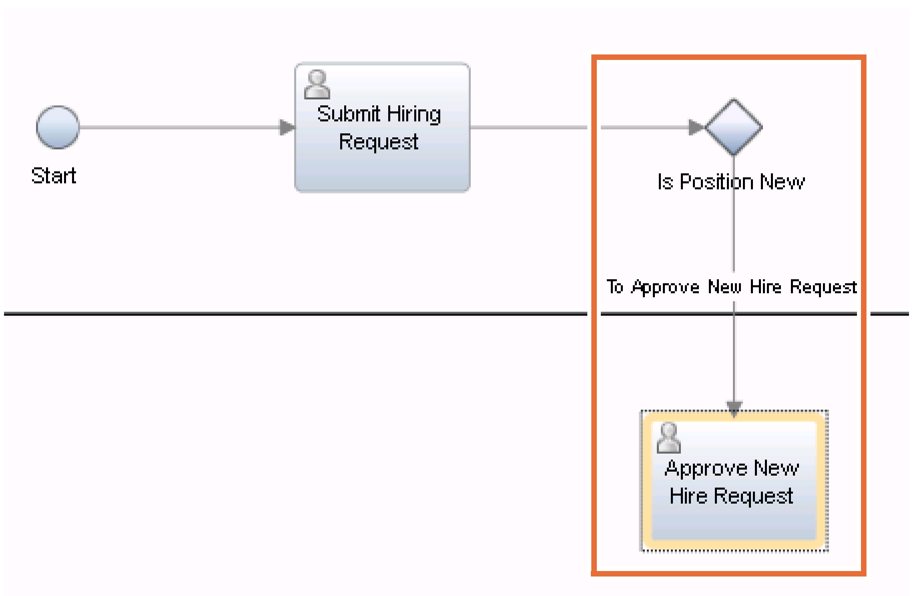
__ c. Connect the sequence flow between **Submit Hiring Request** and the **Is Position New** gateway.

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

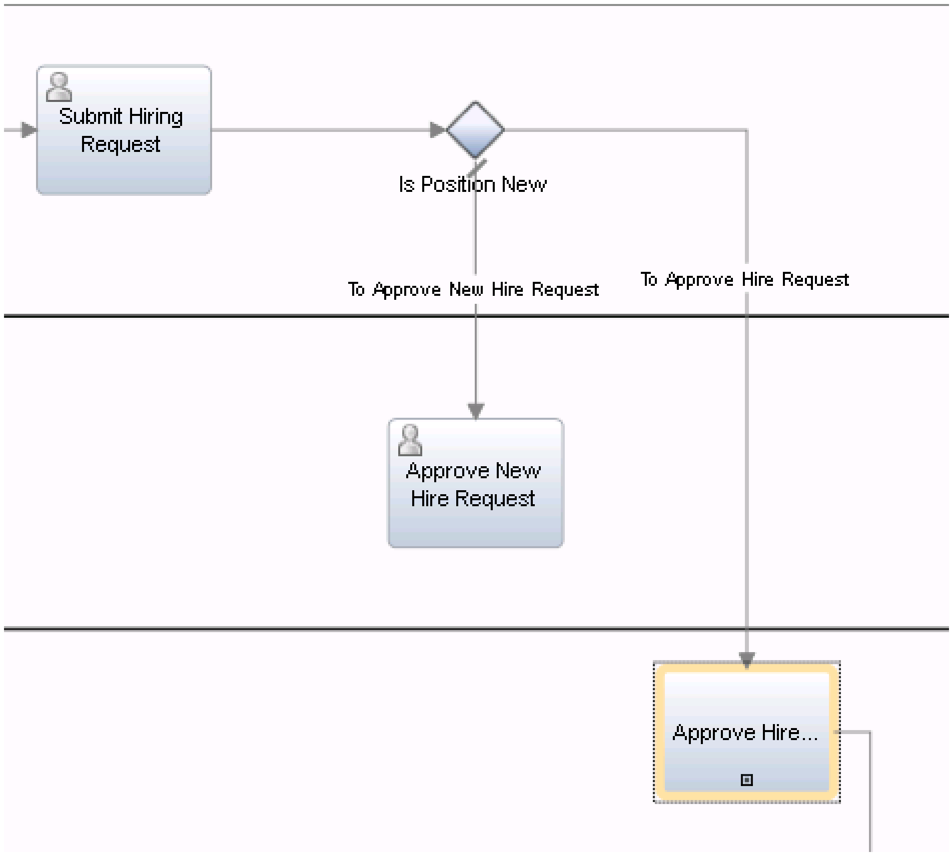
__ e. Right-click the flow and click **Delete**.



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



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

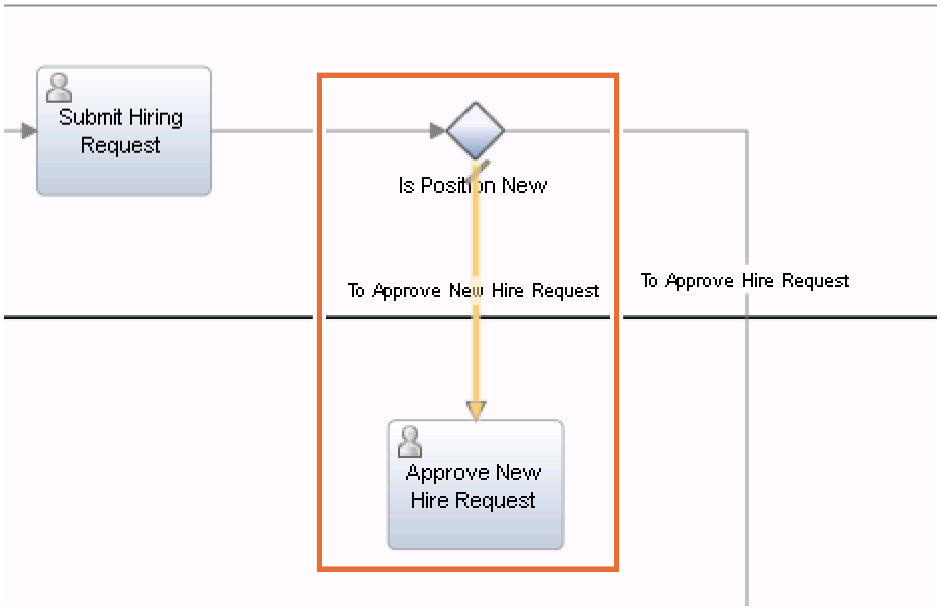


Important

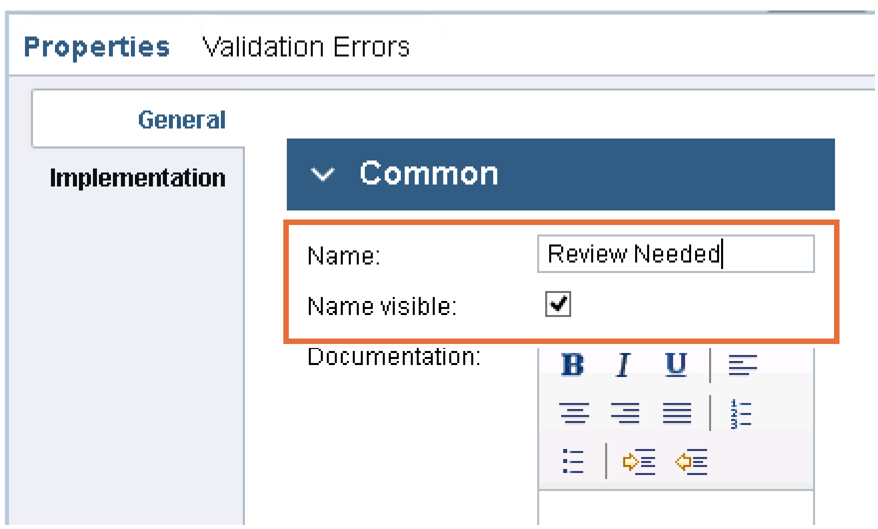
Your sequence flows do not have to exactly match the exercise diagrams in this lab. The sequence flows change after implementation of the gateways. The flows become default or conditional (do or do not have a diagonal hash marker) according to the order you draw them, so your flows might vary from the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

__ 4. Label the flows.

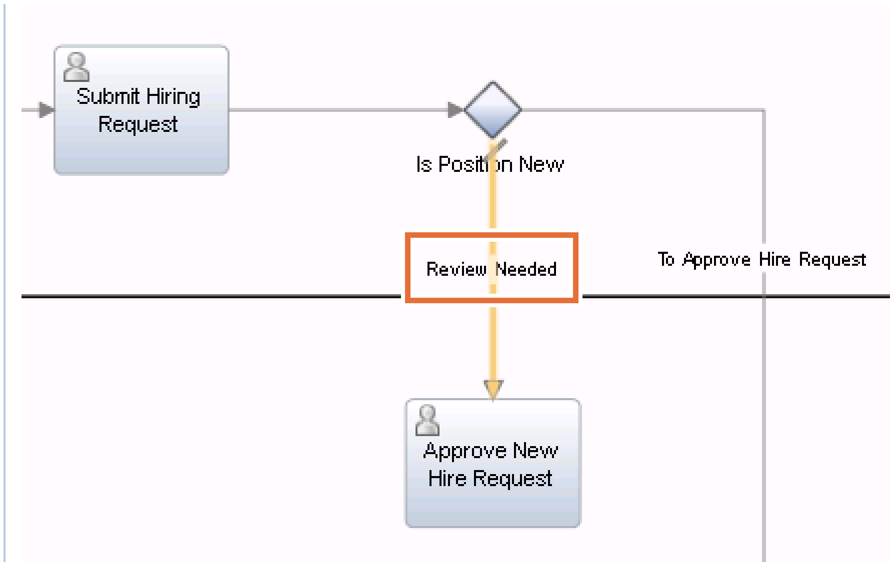
__ a. Select the flow between the **Is Position New** gateway and **Approve New Hire Request**.



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

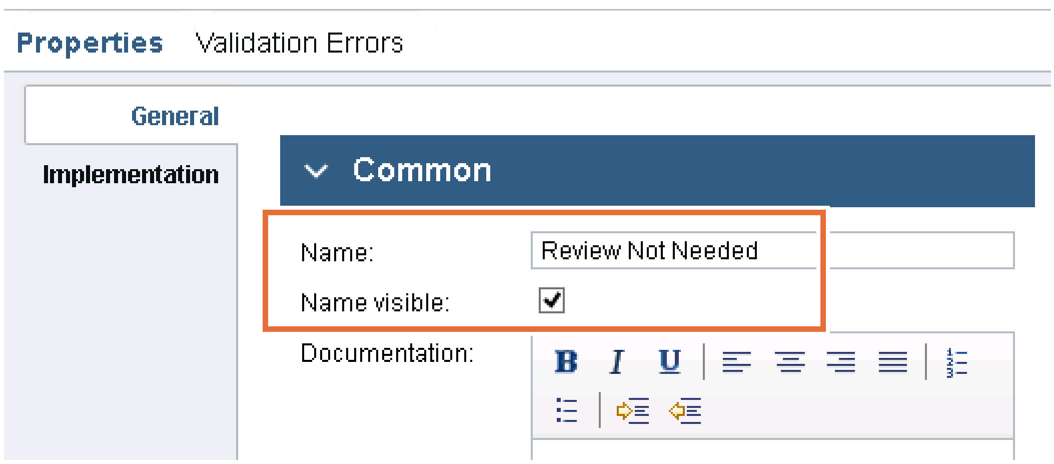


The flow displays the label.

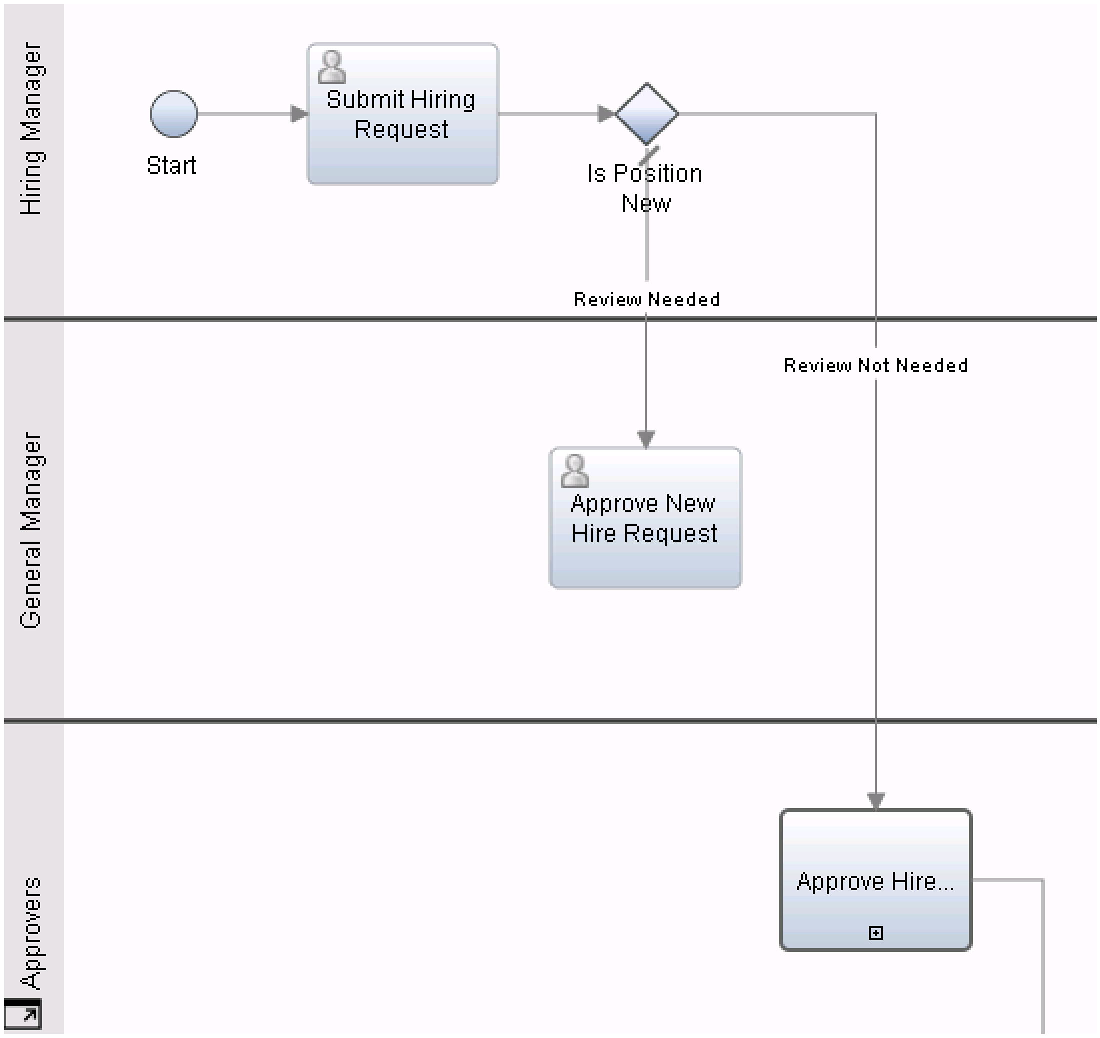


__ c. Select the flow between the **Is Position New** gateway and **Approve Hire Request**.

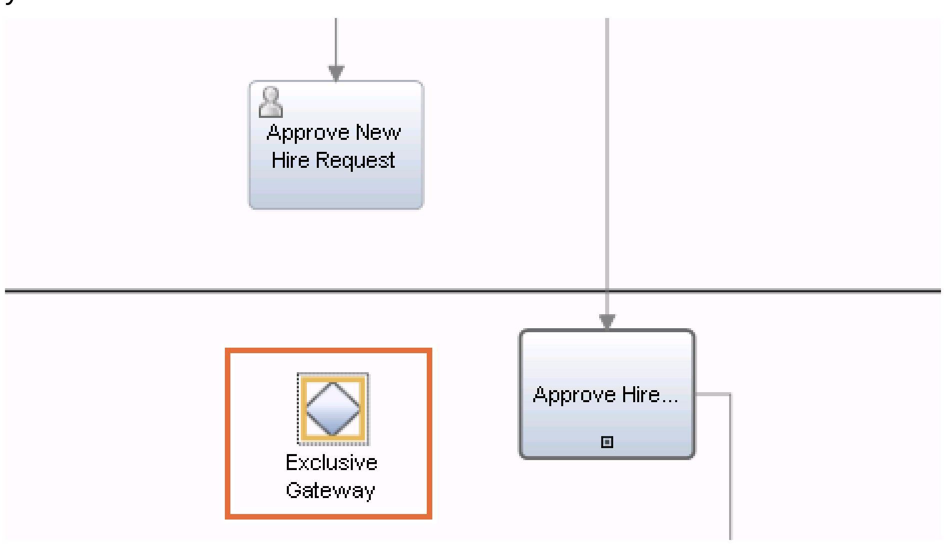
__ d. In the **Properties > General > Common** section, enter `Review Not Needed` as the Name and select **Name visible**.



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

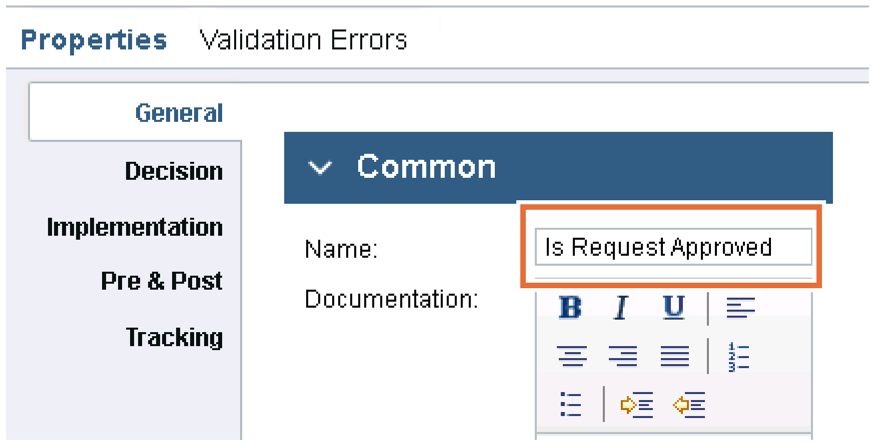


__ 5. Drag a **Gateway** from the palette onto the canvas to the left of the **Approve Hire Request** activity.

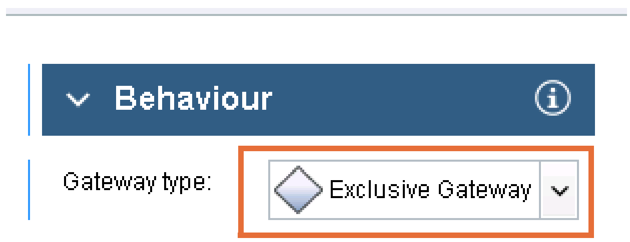


__ 6. Select the **Gateway** and set the properties.

__ a. Select the **Exclusive Gateway** and in the **Properties > General > Common** section, enter `Is Request Approved` as the Name.

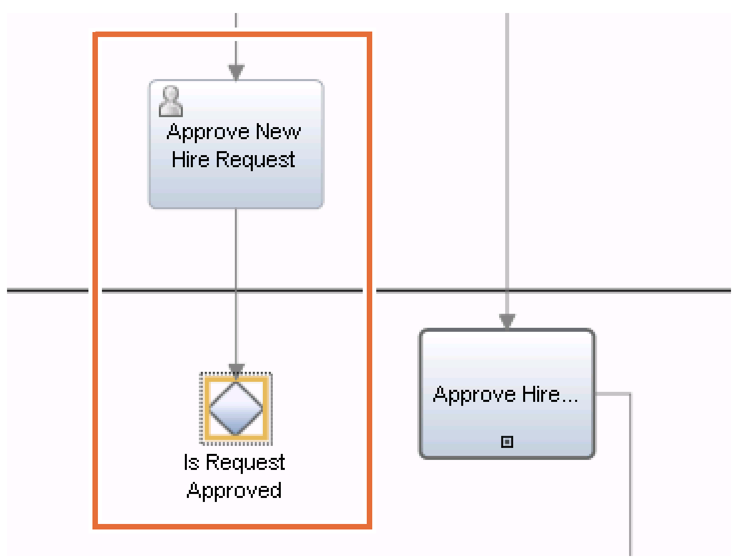


__ b. In the **Properties > General > Behaviour** section, verify **Exclusive Gateway** as the Gateway type.

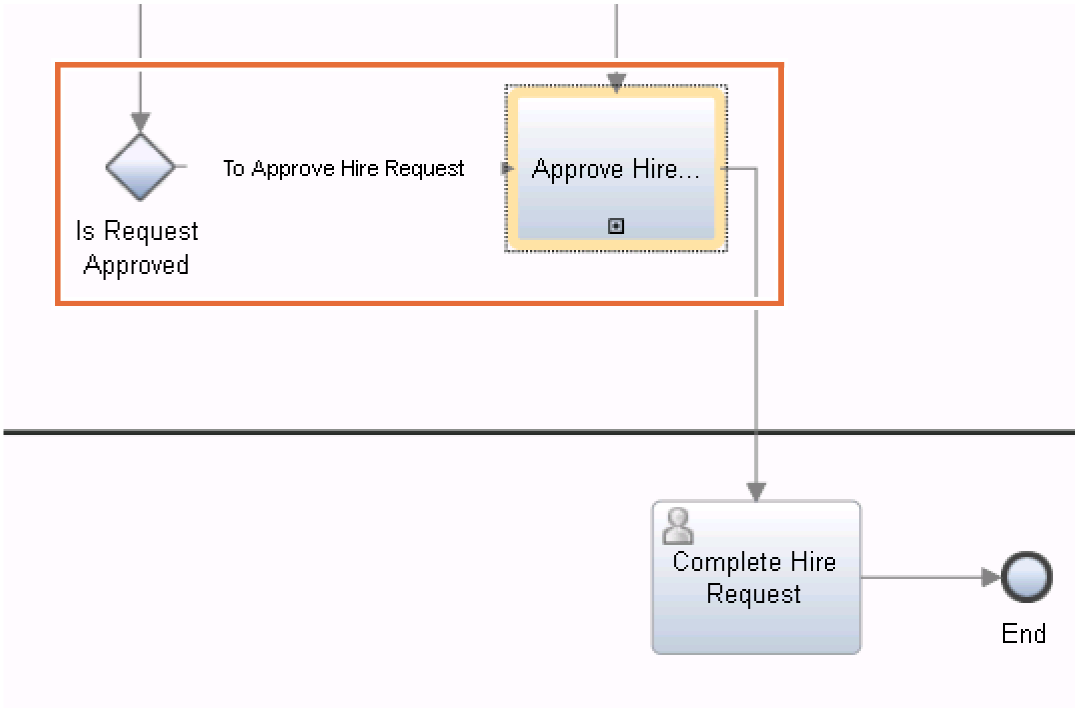


__ 7. Connect the gateway.

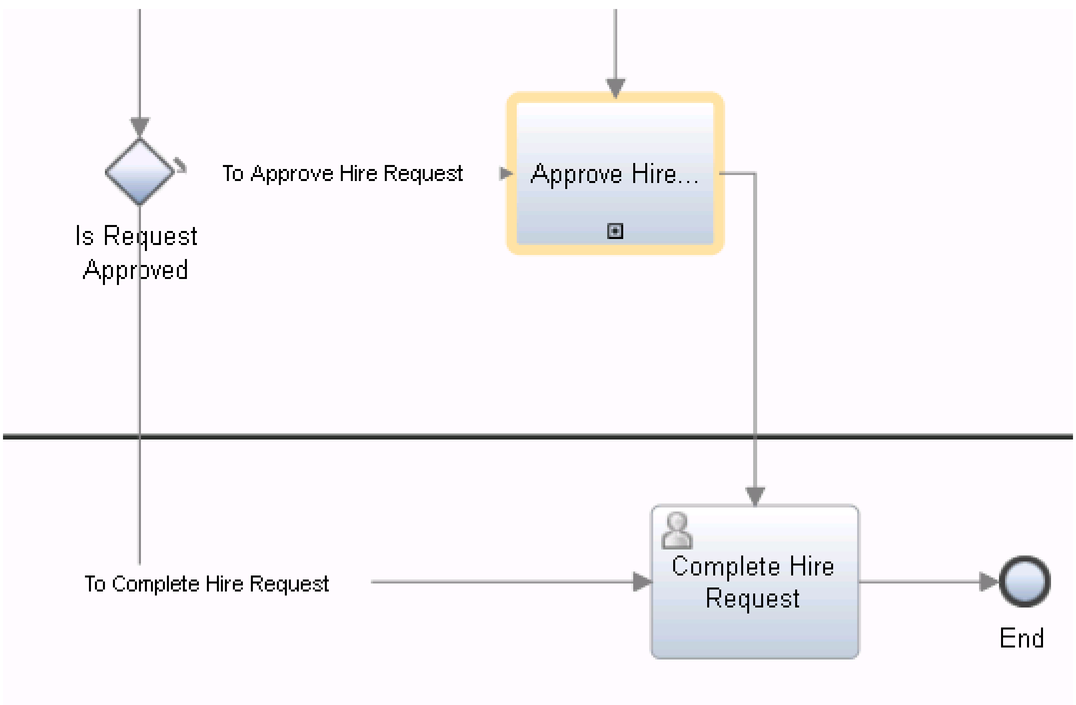
__ a. Connect the **Approve New Hire Request** to the top of the **Is Request Approved** gateway.



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



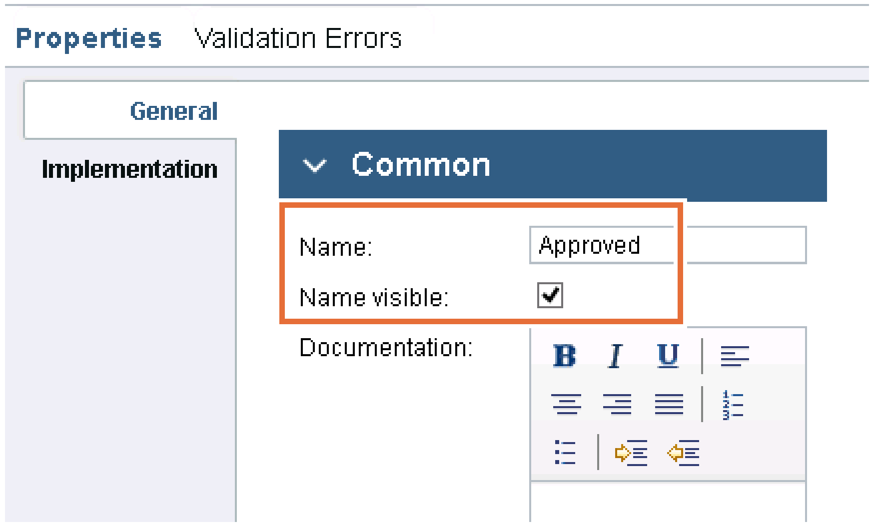
__ c. Connect the **Is Request Approved** gateway to the **Complete Hire Request** activity.



__ 8. Label the flows.

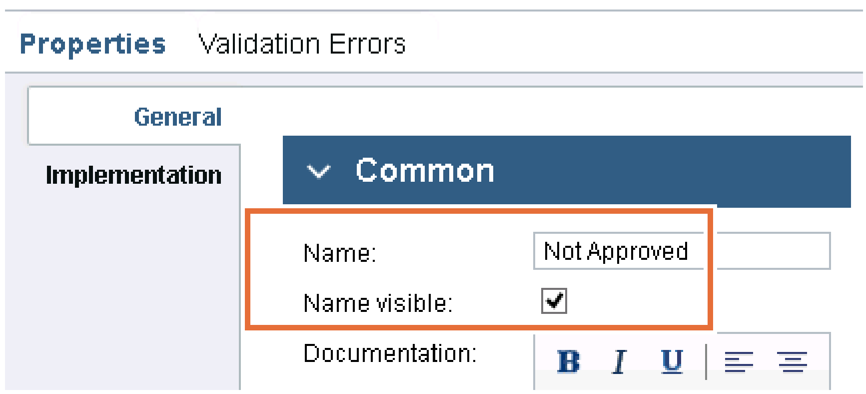
__ a. Select the flow between the **Is Request Approved** gateway and the **Approve Hire Request** linked process.

__ b. Name the flow `Approved` and select **Name visible**.

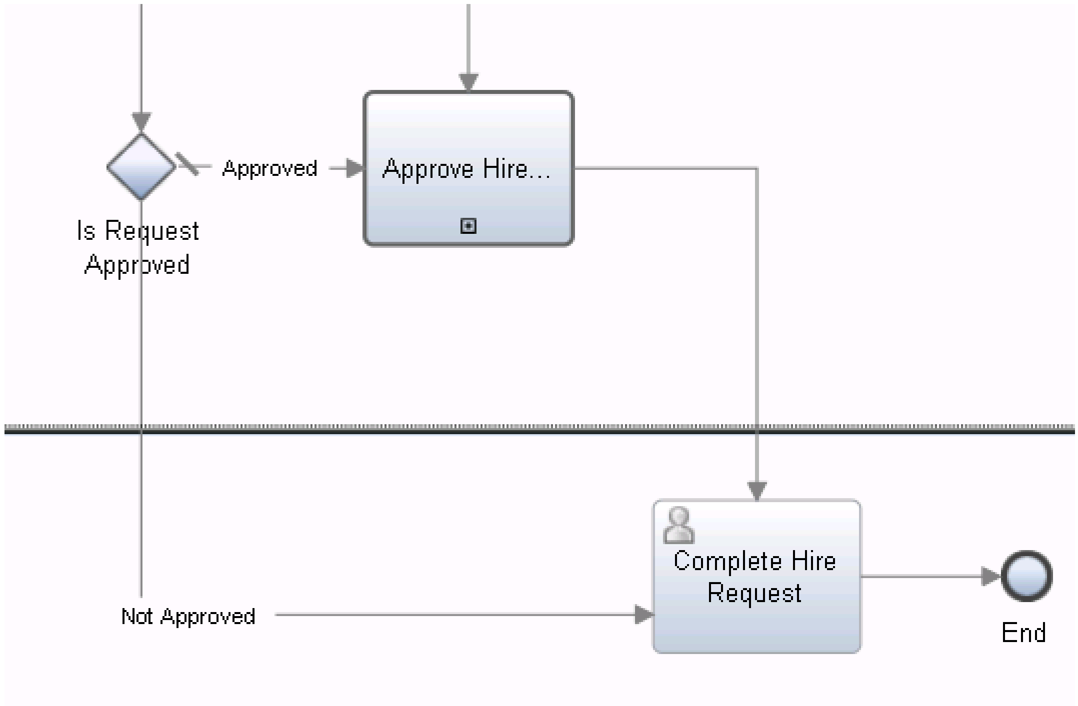


__ c. Select the flow between the **Is Request Approved** gateway and **Complete Hire Request**.

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



The flow displays the label:

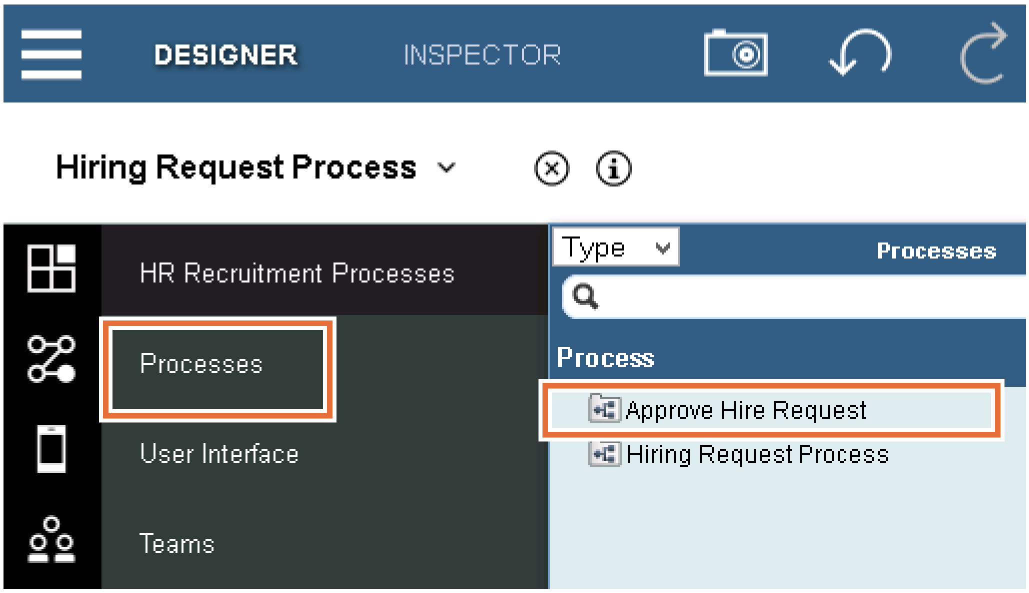


__ e. Click **Save** to save your process.

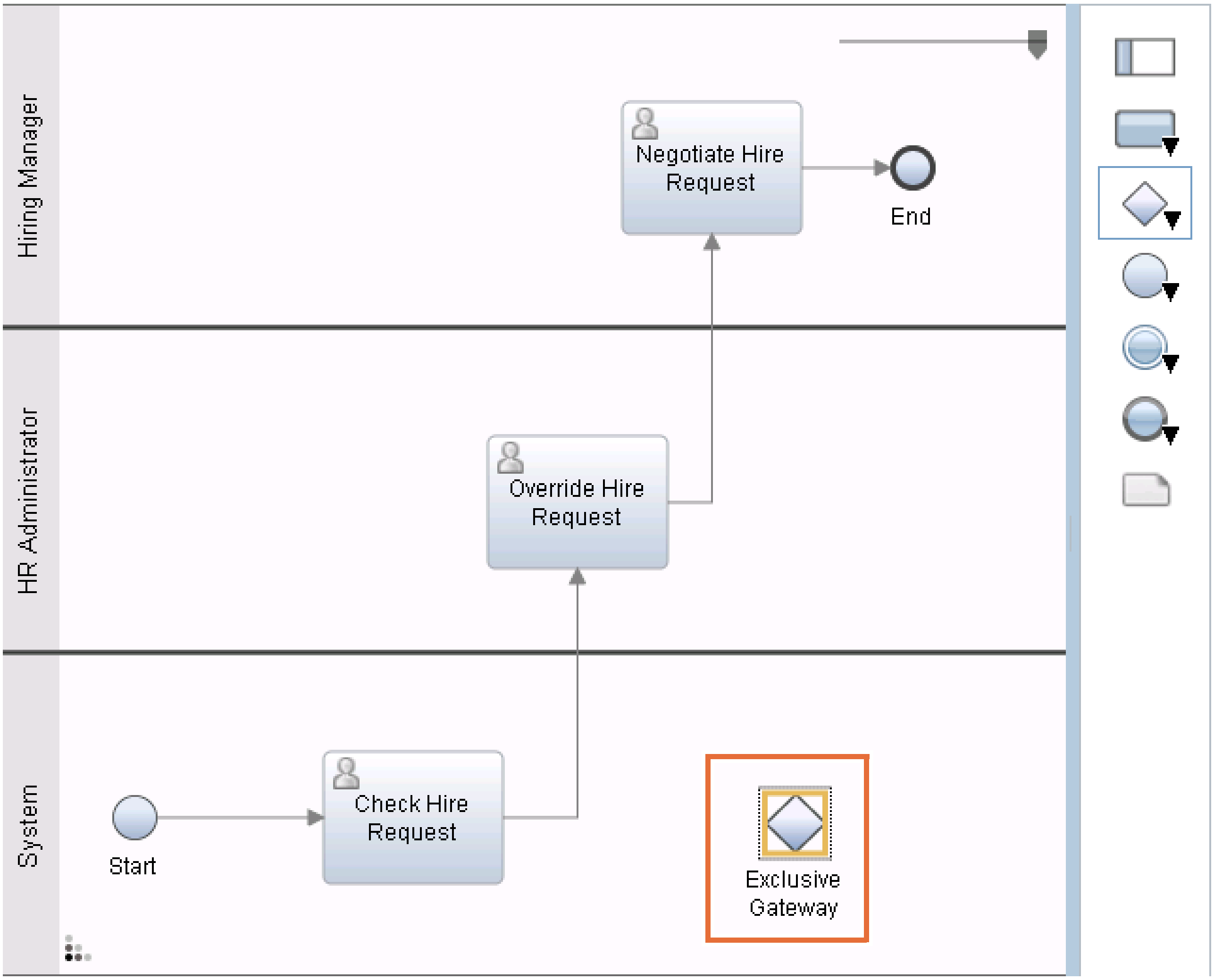
Part 2: Create gateways for the nested process

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

__ 1. 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.



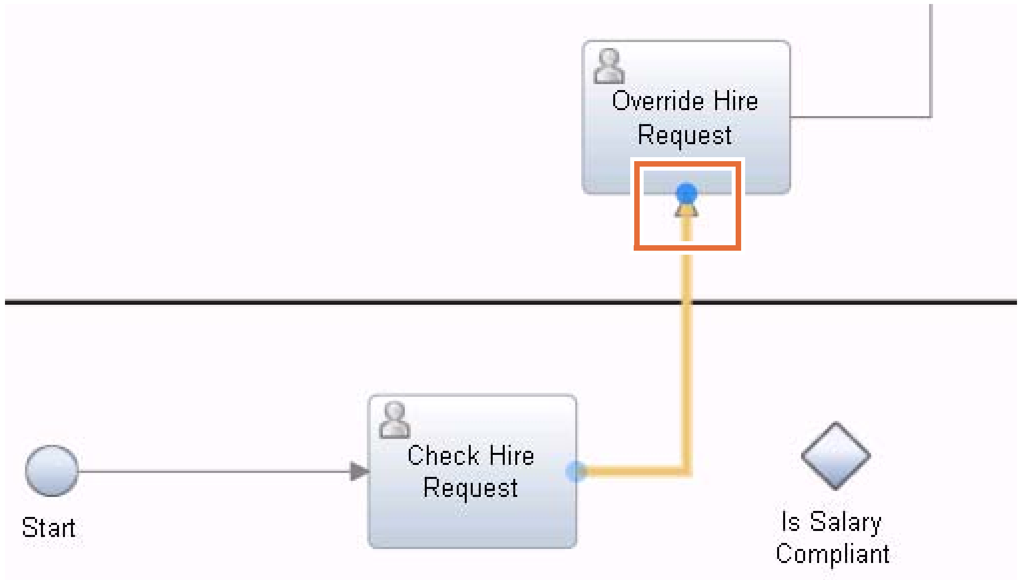
__ 3. Rename the gateway to: `Is Salary Compliant`

__ 4. Verify that the Gateway type is **Exclusive Gateway**.

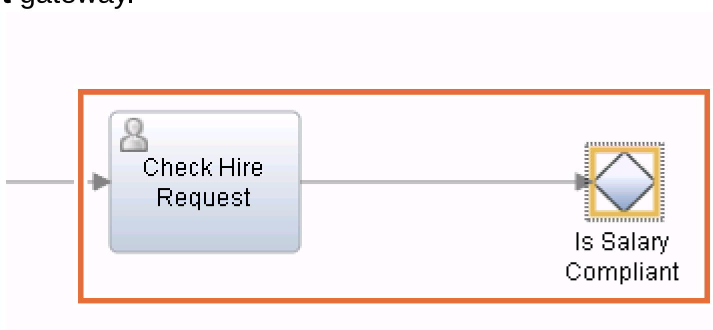
__ 5. Connect the gateway.

 __ a. Select the sequence flow between **Check Hire Request** and **Override Hire Request**.

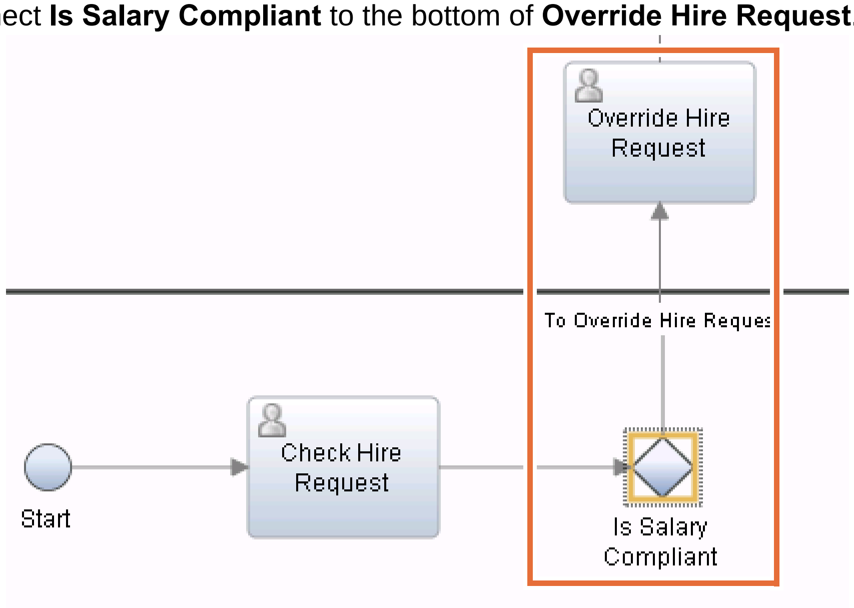
__ b. Hover over the tip of the sequence flow to see a blue point.



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

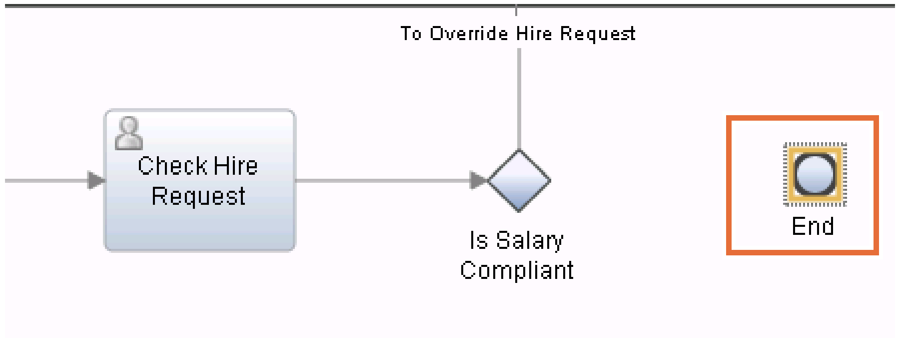


__ d. Connect **Is Salary Compliant** to the bottom of **Override Hire Request**.

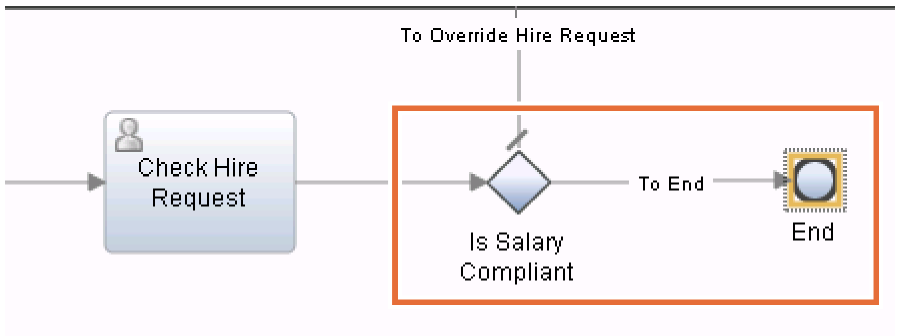


__ e. Remove the existing sequence flow between **Negotiate Hire Request** and the **End** event.

__ 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.

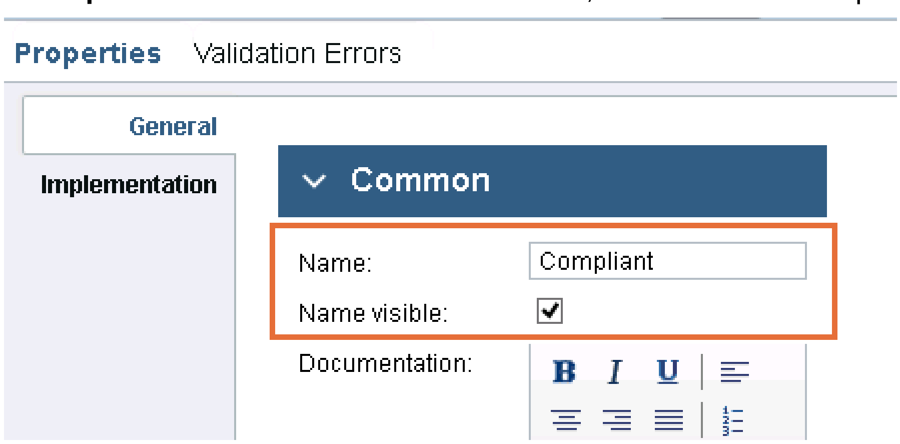
__ 6. Label and straighten the flows.

__ a. Select the flow between the **Is Salary Compliant** gateway and **Override Hire Request**.

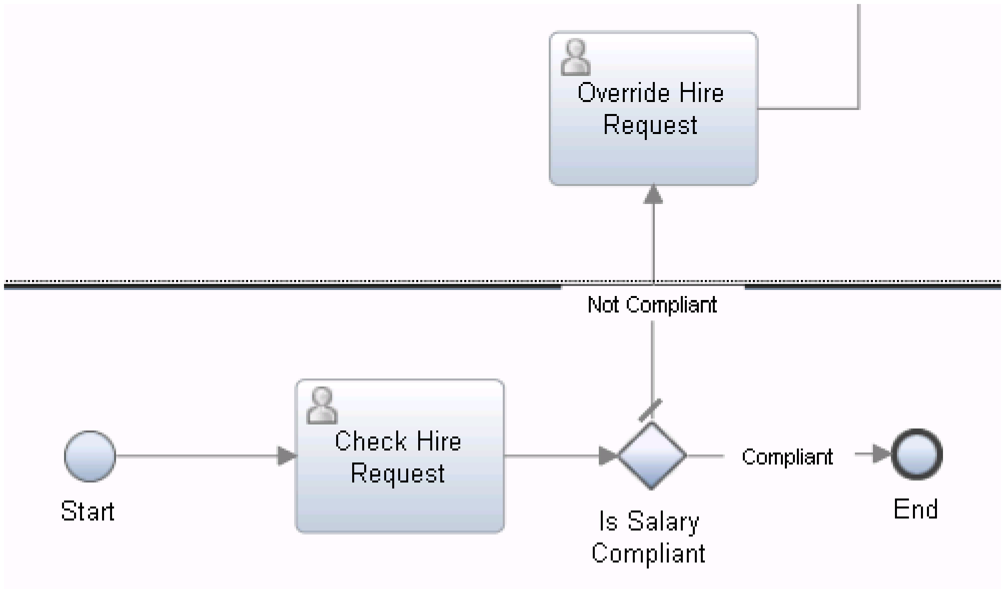
__ b. Name the flow: `Not Compliant`

__ c. Select the flow between the **Is Salary Compliant** gateway and the **End** event.

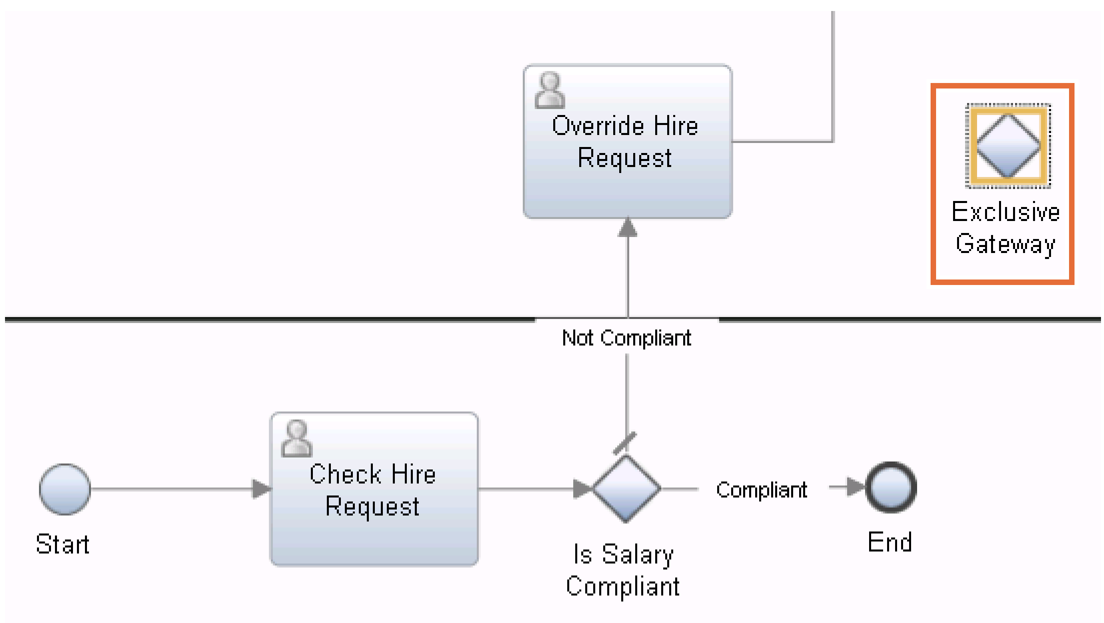
__ d. In the **Properties > General > Common** section, name the flow: `Compliant`



The flows display as follows:



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



__ 8. Click the **Gateway** and set the properties.

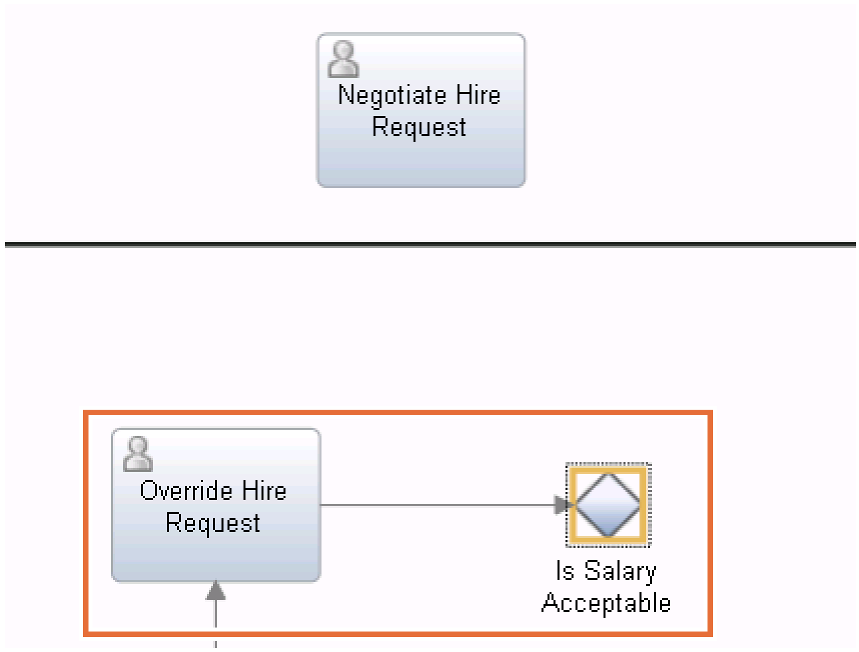
  __ a. In the Properties section, set the gateway name to: `Is Salary Acceptable`

  __ b. Save your changes.

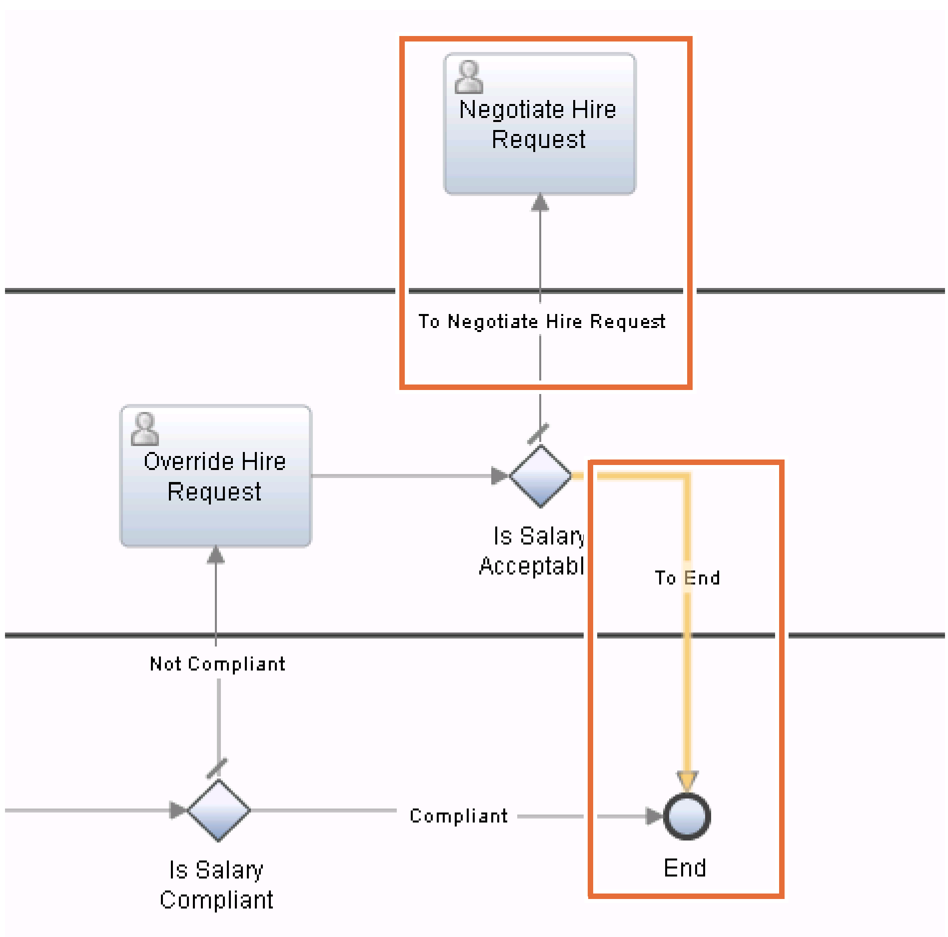
__ 9. Connect the gateway.

  __ a. Delete the flow between **Override Hire Request** and **Negotiate Hire Request**.

__ b. Connect the **Override Hire Request** to the right 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.



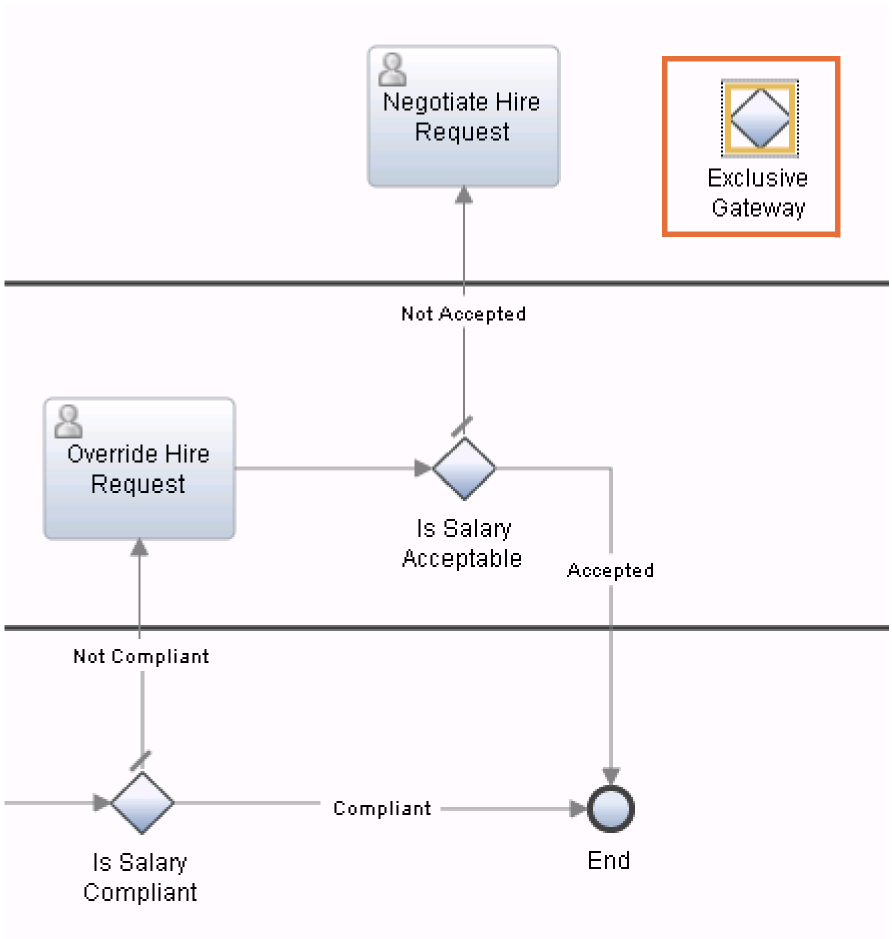
__ 10. Label the flows.

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`

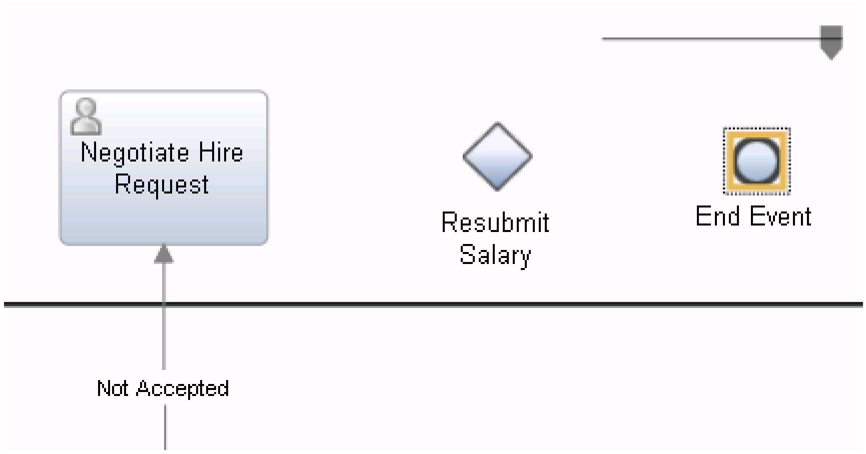
__ b. Name the flow between the **Is Salary Acceptable** gateway and the end event as: `Accepted`

__ 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.



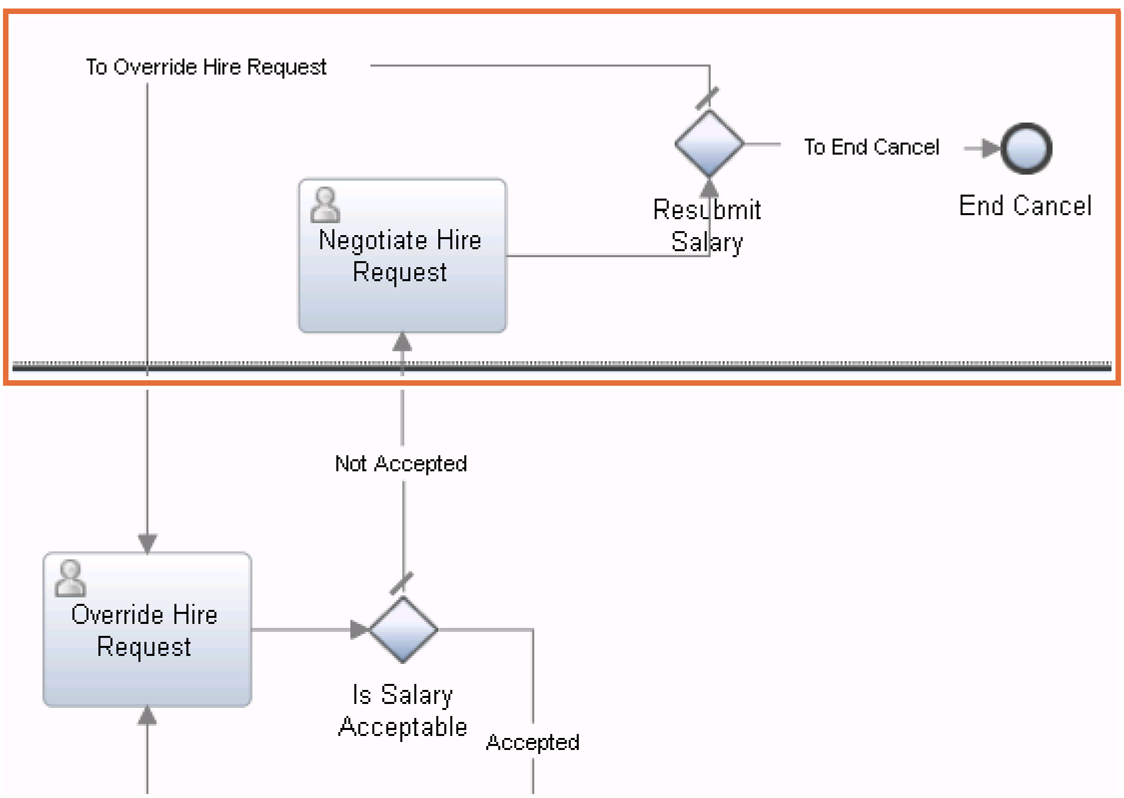
__ 14. Rename the **End** event as: `End Cancel`

__ 15. Connect the gateway.

__ a. Connect **Negotiate Hire Request** to the **Resubmit Salary** gateway.

__ b. Connect the **Resubmit Salary** gateway to the top of **Override Hire Request**.

__ c. Connect the **Resubmit Salary** gateway to the **End Cancel** event.



Important

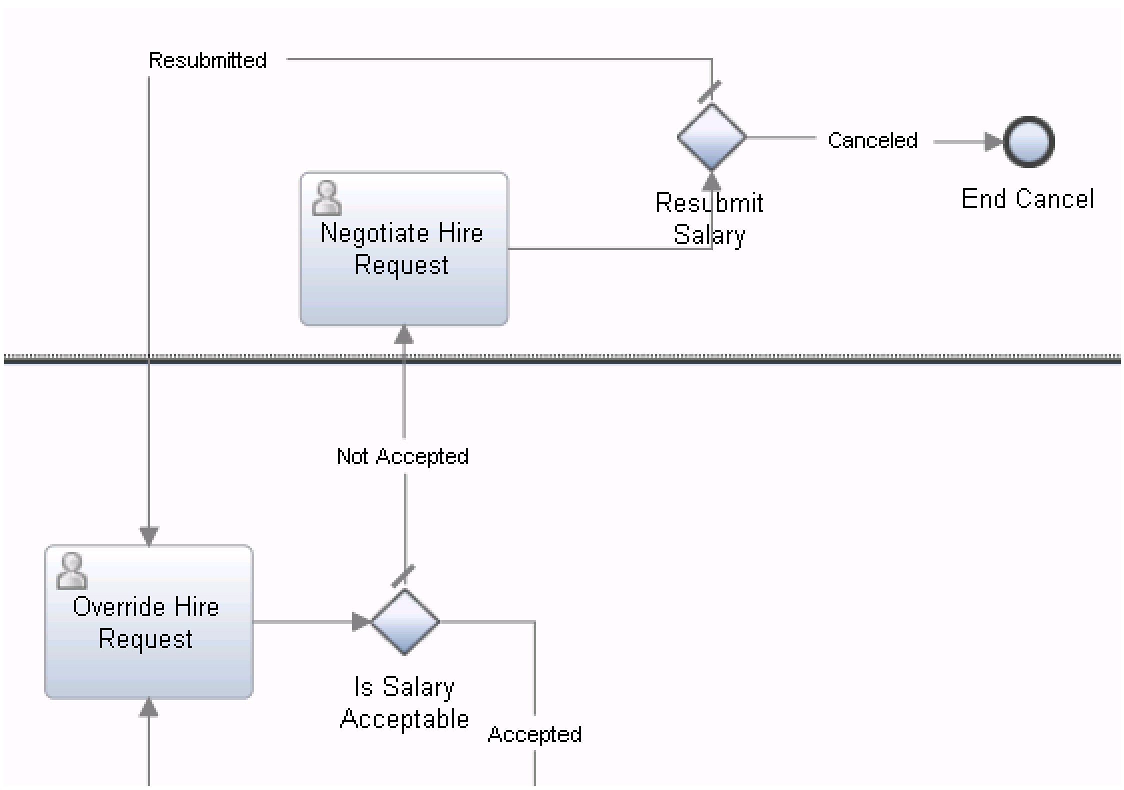
Make sure that you create the flows to these elements in the correct directions. You can align the activities by selecting and dragging them to a location where you can see them better.

__ 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**.

__ c. The process has all the flows labeled.



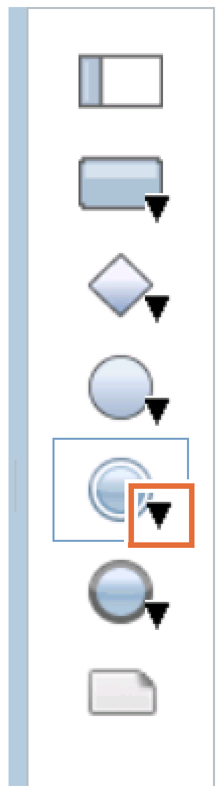
Part 3: Modeling timer intermediate events

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

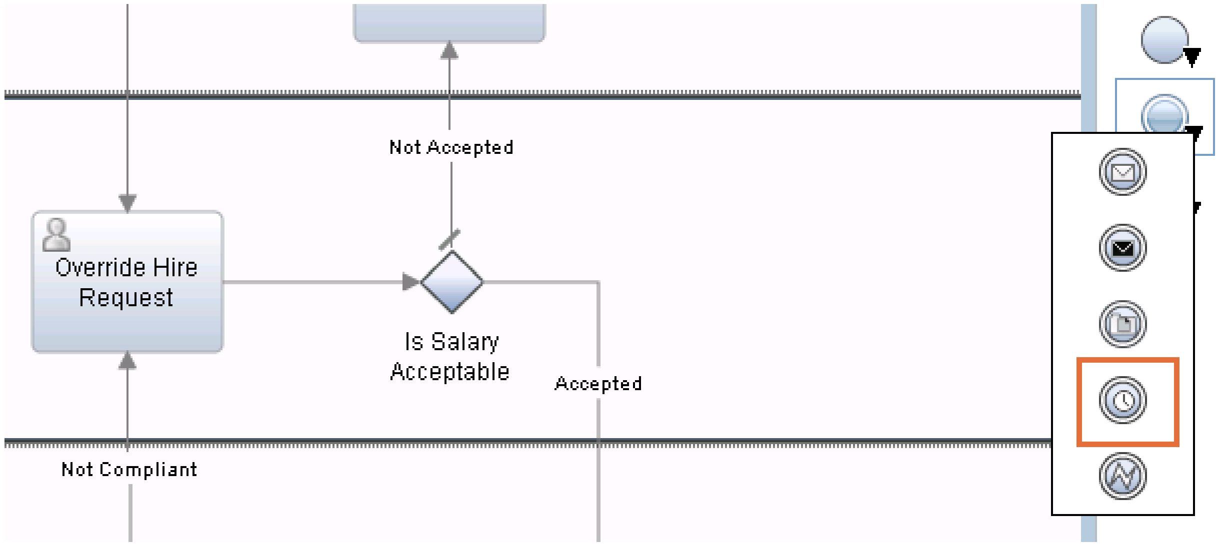
Refer to the core requirements for an escalation in 2.10 listed in Exercise 2. The HR Administrator has 4 hours to complete the review. If the review is not completed within 4 hours, an email is sent to the HR Administrator notifying the HR Administrator of the missed deadline.

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.

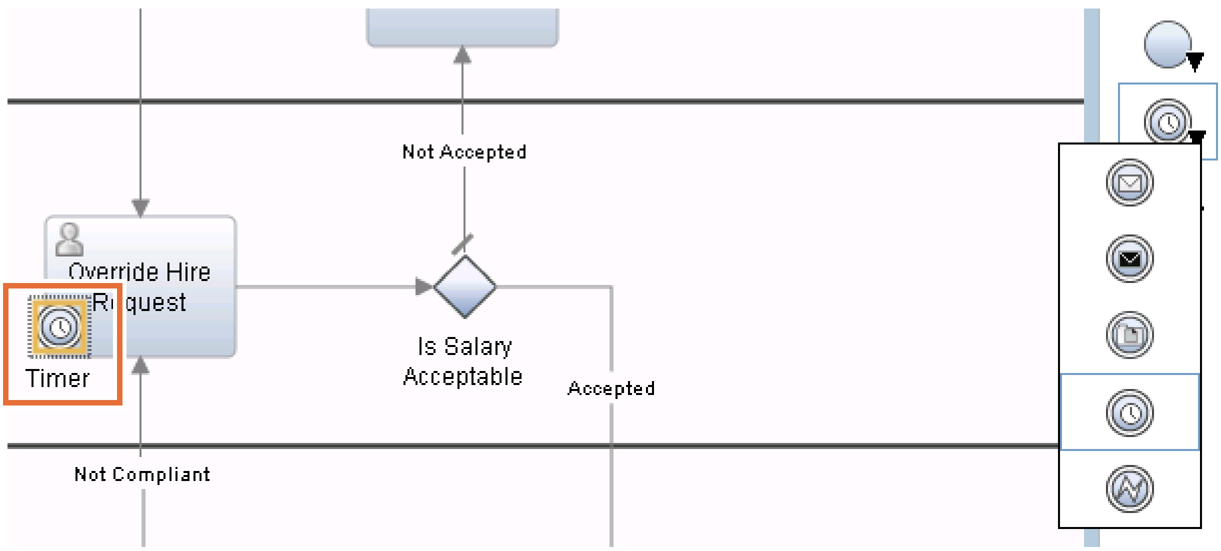
__ 1. In the **Approve Hire Request** process, select the **drop-down icon** beside the **Intermediate** event.



__ 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.



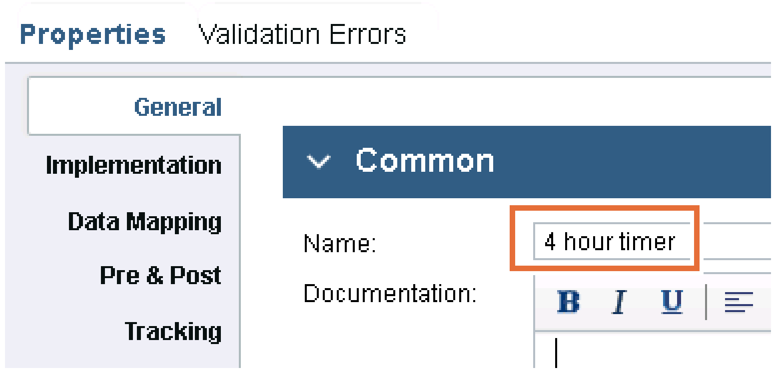
Important

For this exercise, make sure that you drag the **Timer** event onto the activity instead of to the left of the activity. In this case, you want a boundary or attached intermediate event instead of a sequence flow intermediate event.

__ 4. Click the **Intermediate** boundary event and set its properties.

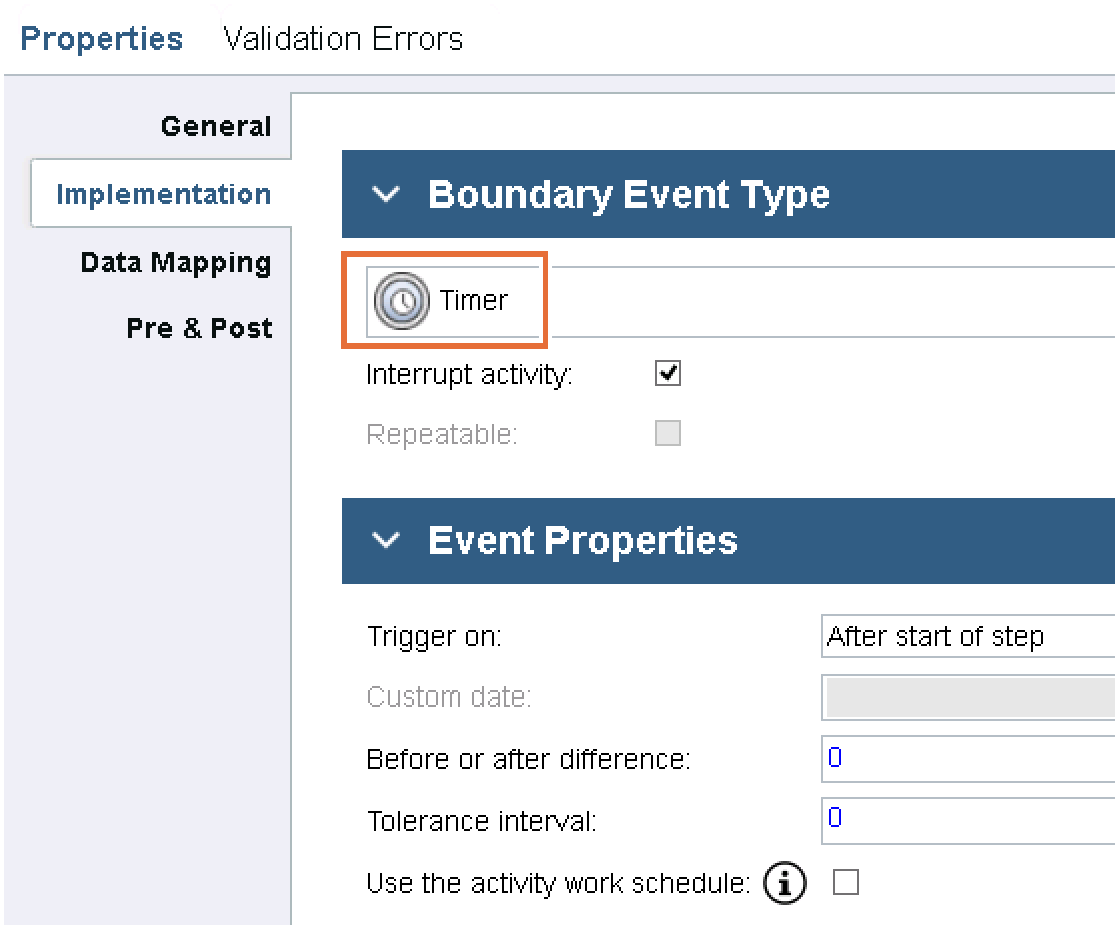
__ a. Select the **Timer** event.

__ b. In the **Properties > General > Common** section, change the name to: `4 hour timer`



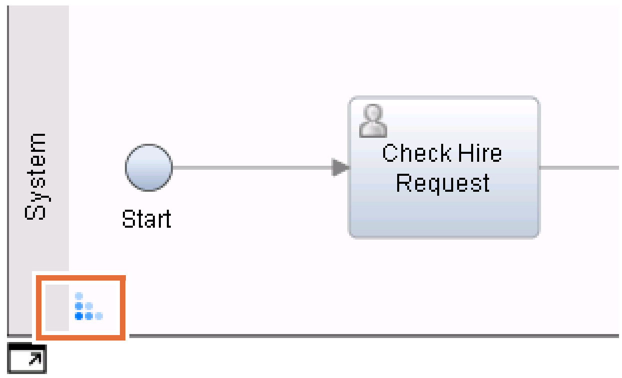
__ c. Select the **Properties > Implementation** menu.

__ d. 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.

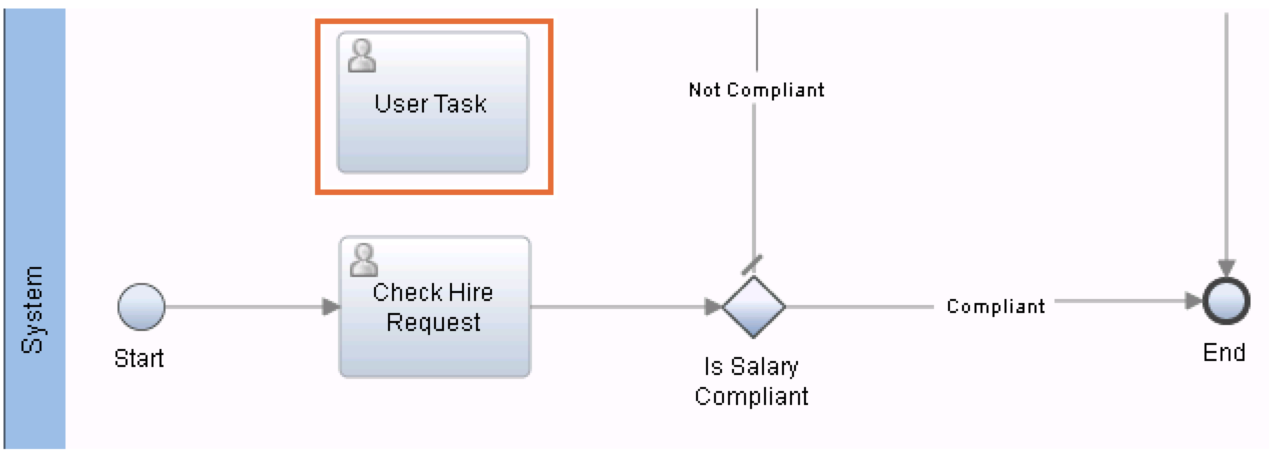


__ 5. Add the Notify HR Administrator activity.

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



__ b. Drag an activity from the palette to the **System** lane. Place it above the **Check Hire Request** activity.

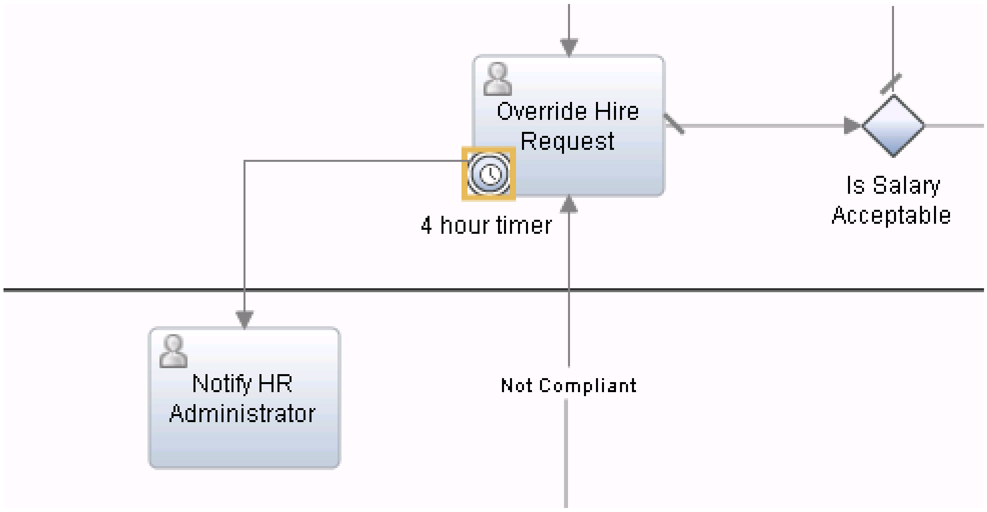


Hint

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

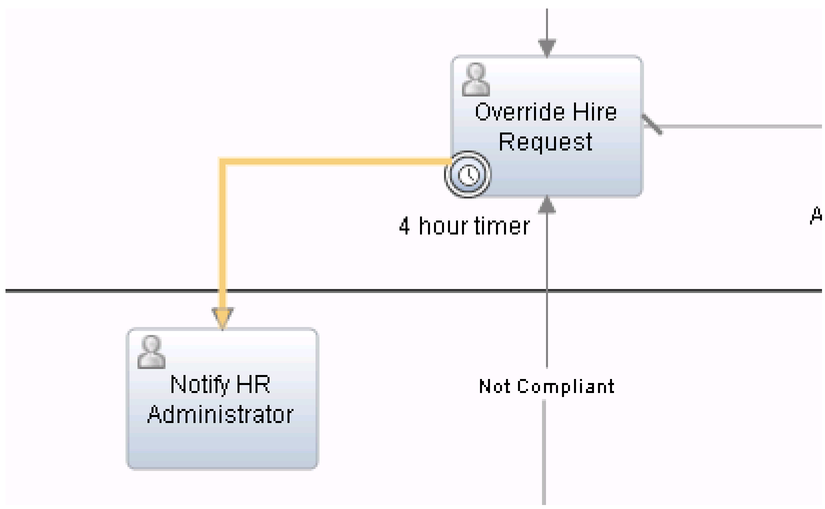
__ c. Rename the activity to: `Notify HR Administrator`

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

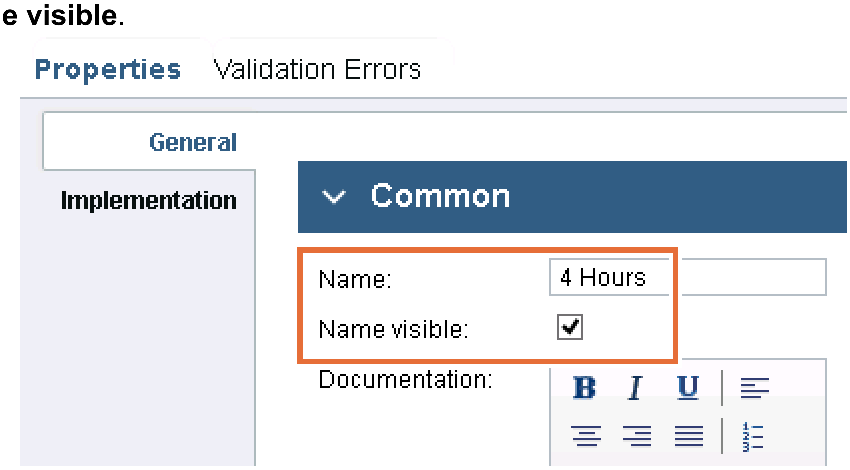


__ 7. Label the flow.

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

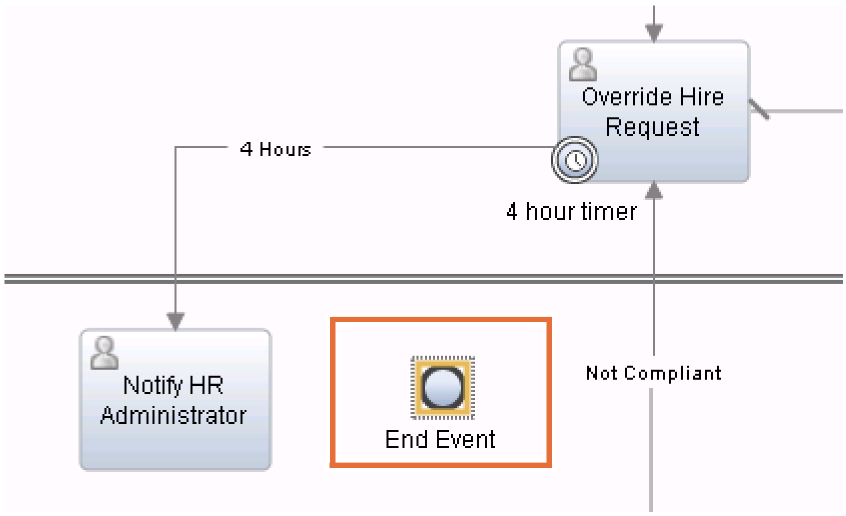


__ b. In **Properties > General > Common** section, name the flow as `4 Hours` and select **Name visible**.

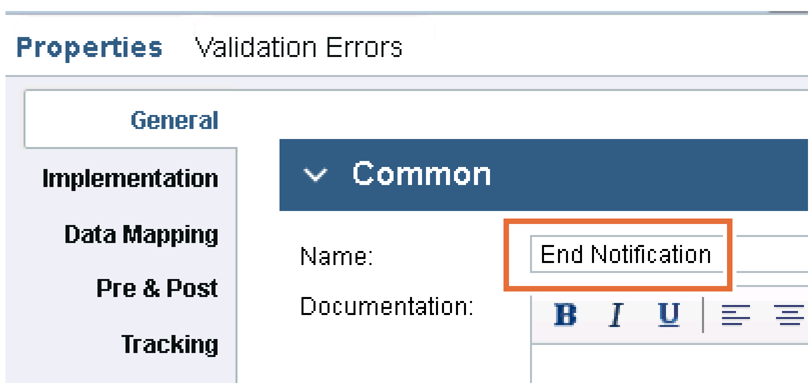


__ 8. Add an End Notification event.

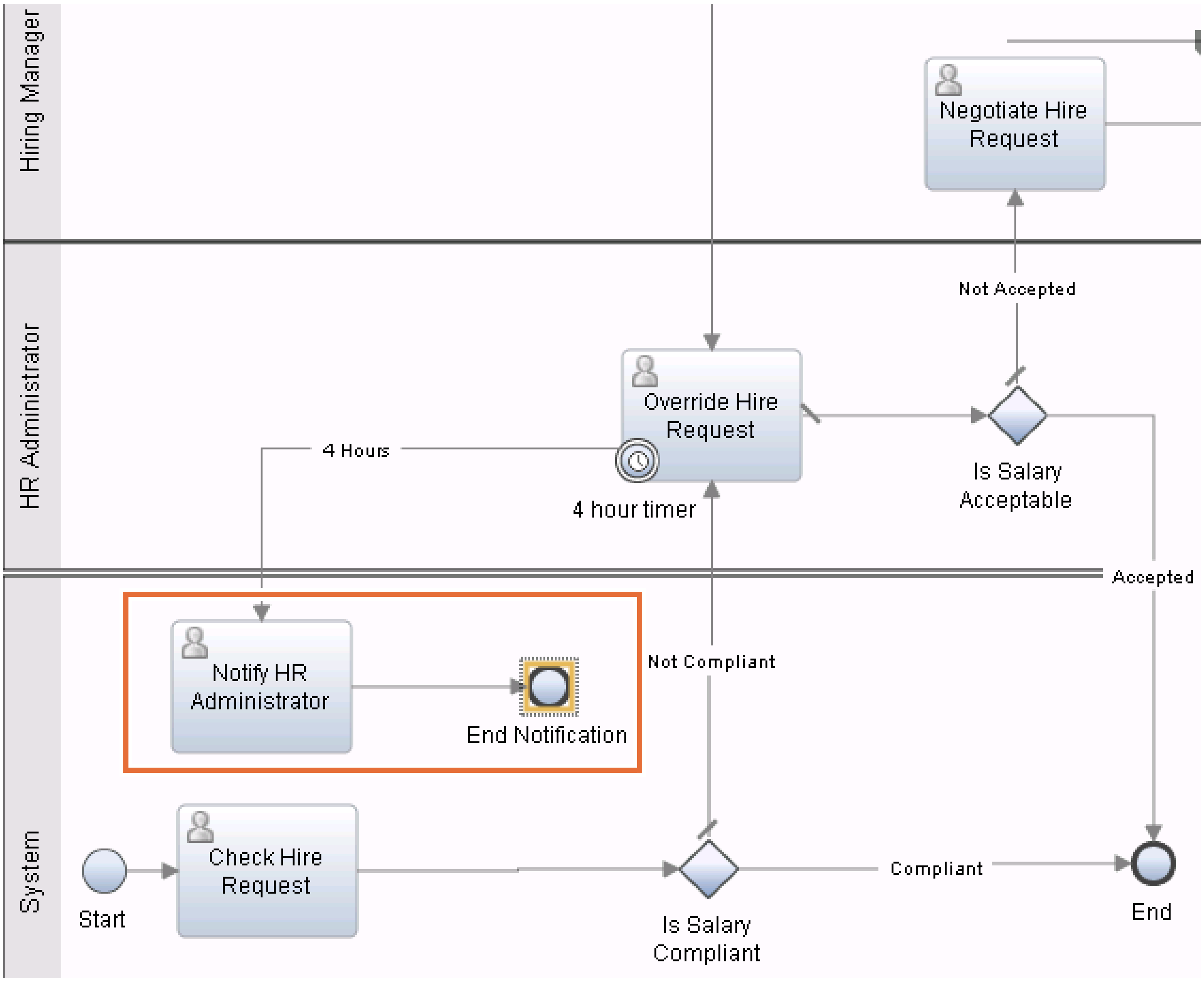
__ a. Drag an **End Event** icon from the palette onto the canvas to the right of **Notify HR Administrator**.



__ b. Rename the End event to: `End Notification`



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



__ a. Click **Save** to save your work.

Important

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 Web 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 Web 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.

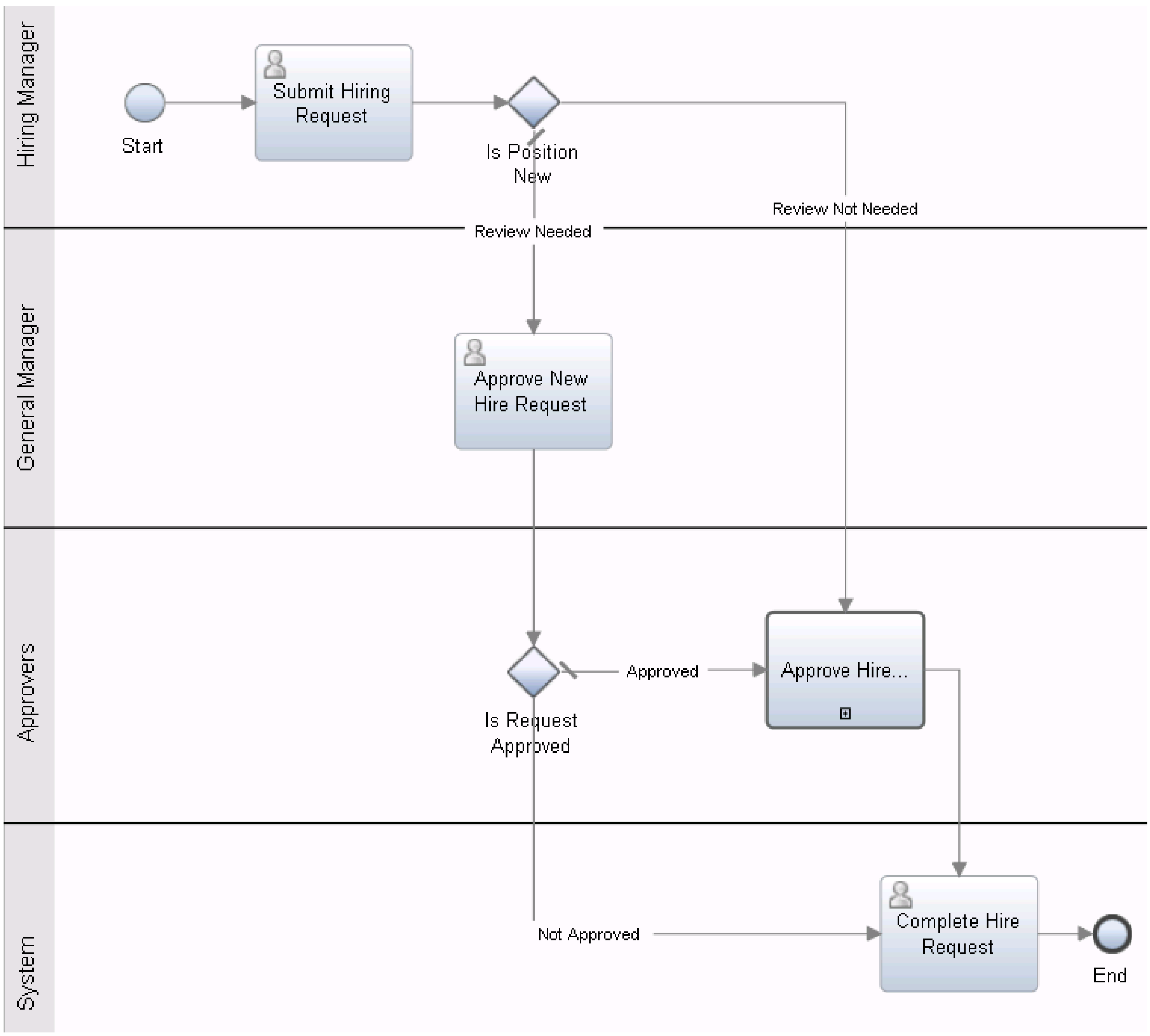
Exercise instructions

Part 1: Verify the process model

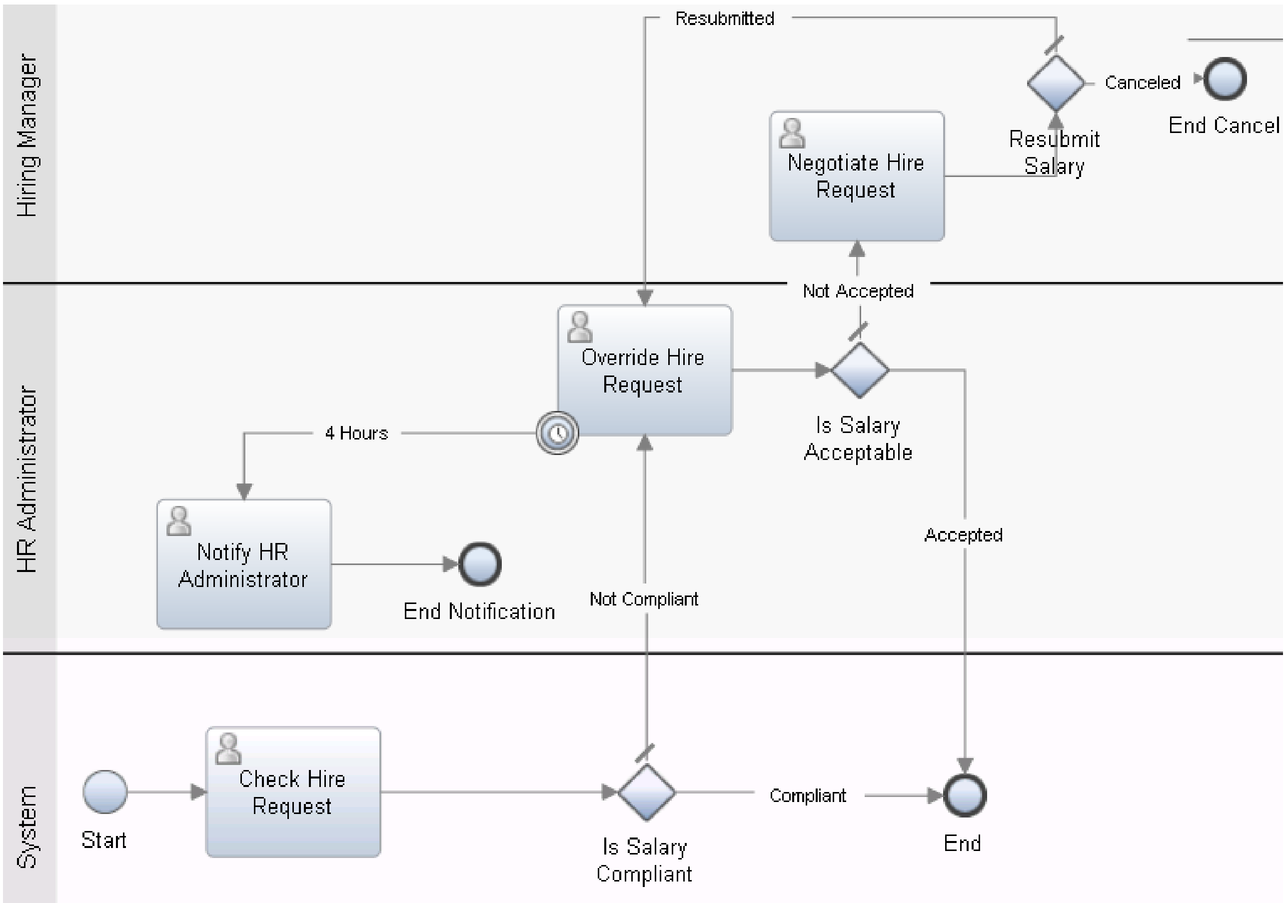
If you have a colleague or others nearby, see whether you can explain your new model to them to simulate a playback. If not, examine your process in IBM Web Process Designer and compare it to the diagrams.

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:

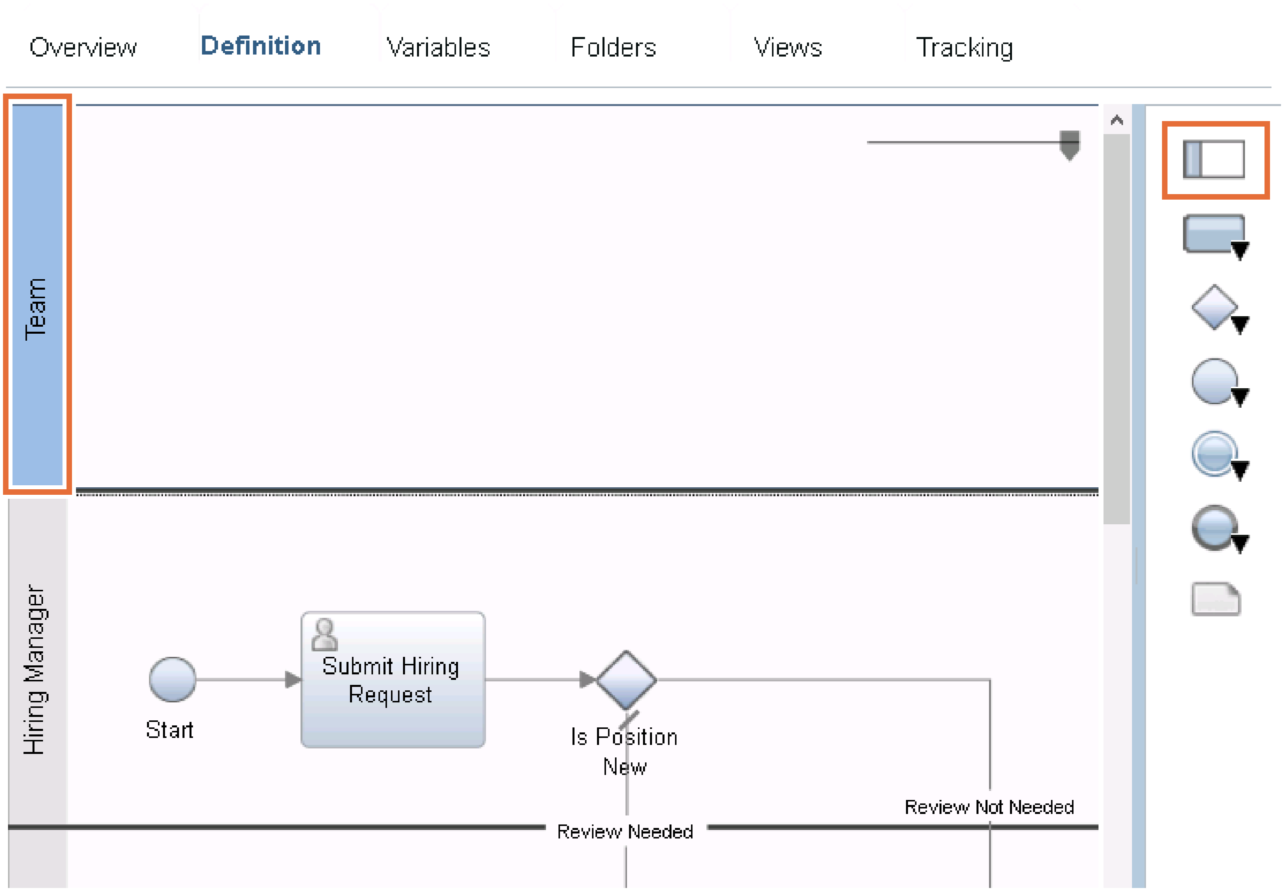
> Several job postings contain wording that violates a new set of hiring laws. To address this issue, the vice president suggests that a lawyer must review every job post at some point before the job opening is released and posted to the public. The lawyer checks for legal compliance and edits if necessary.

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.

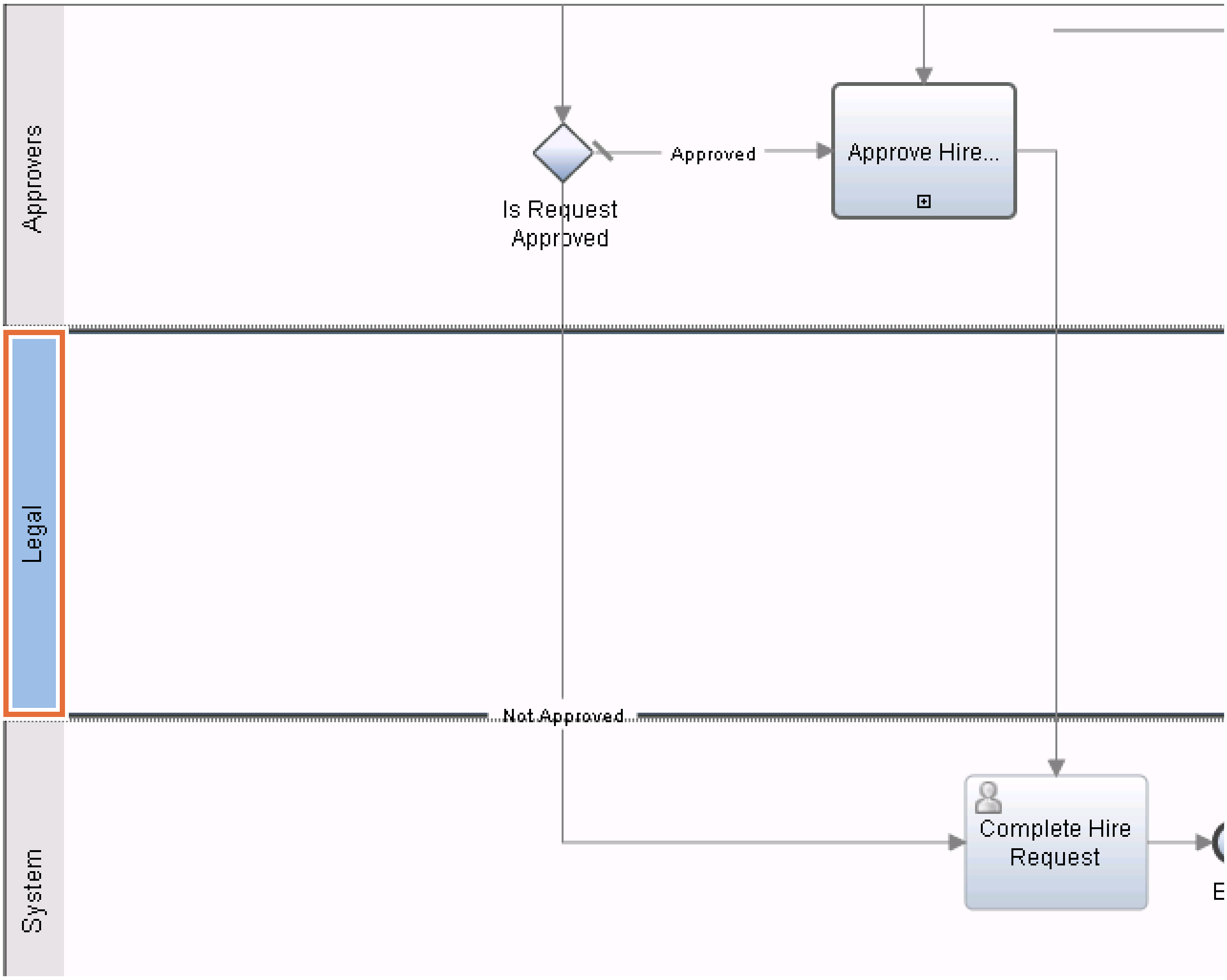
__ a. Open the **Hiring Request Process**.

__ b. Click the **Lane** icon and drag it from the palette to the canvas above the Hiring Manager lane.



__ c. Name the lane: `Legal`

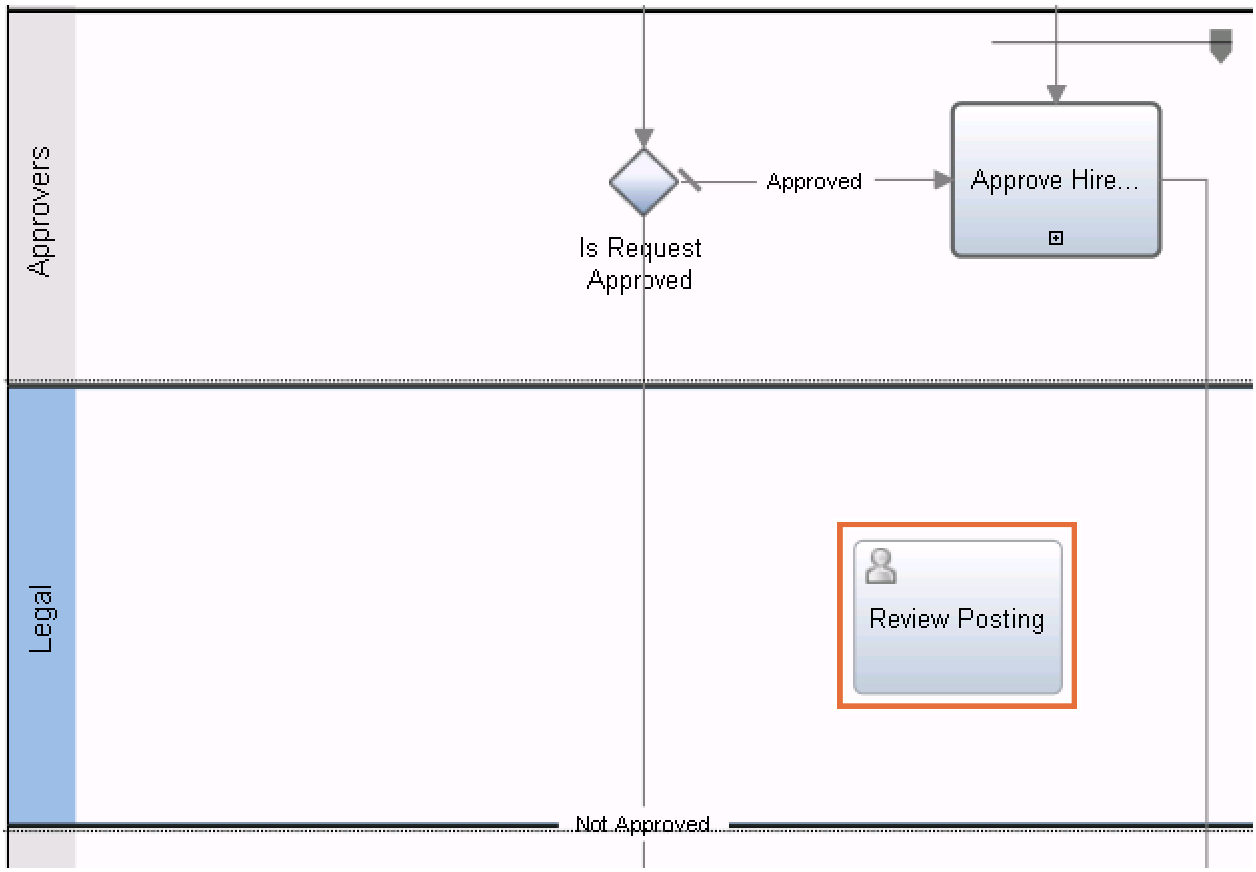
__ d. Drag the **Legal** lane between the **Approvers** and **System** lanes.



__ 2. Create an activity for a lawyer to review the hire request.

__ a. Drag an **Activity** from the palette to the **Legal** lane in the process.

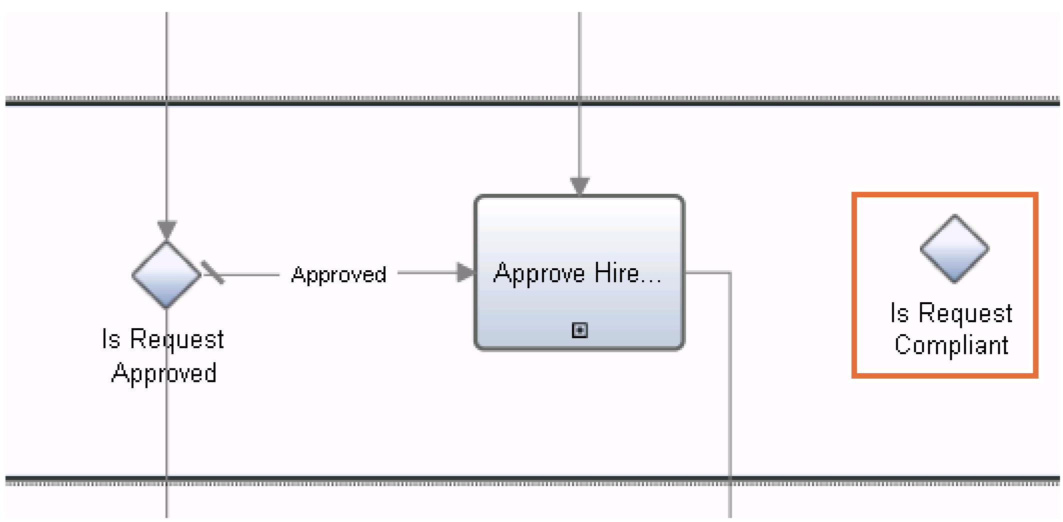
__ b. Name the activity: `Review Posting`



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

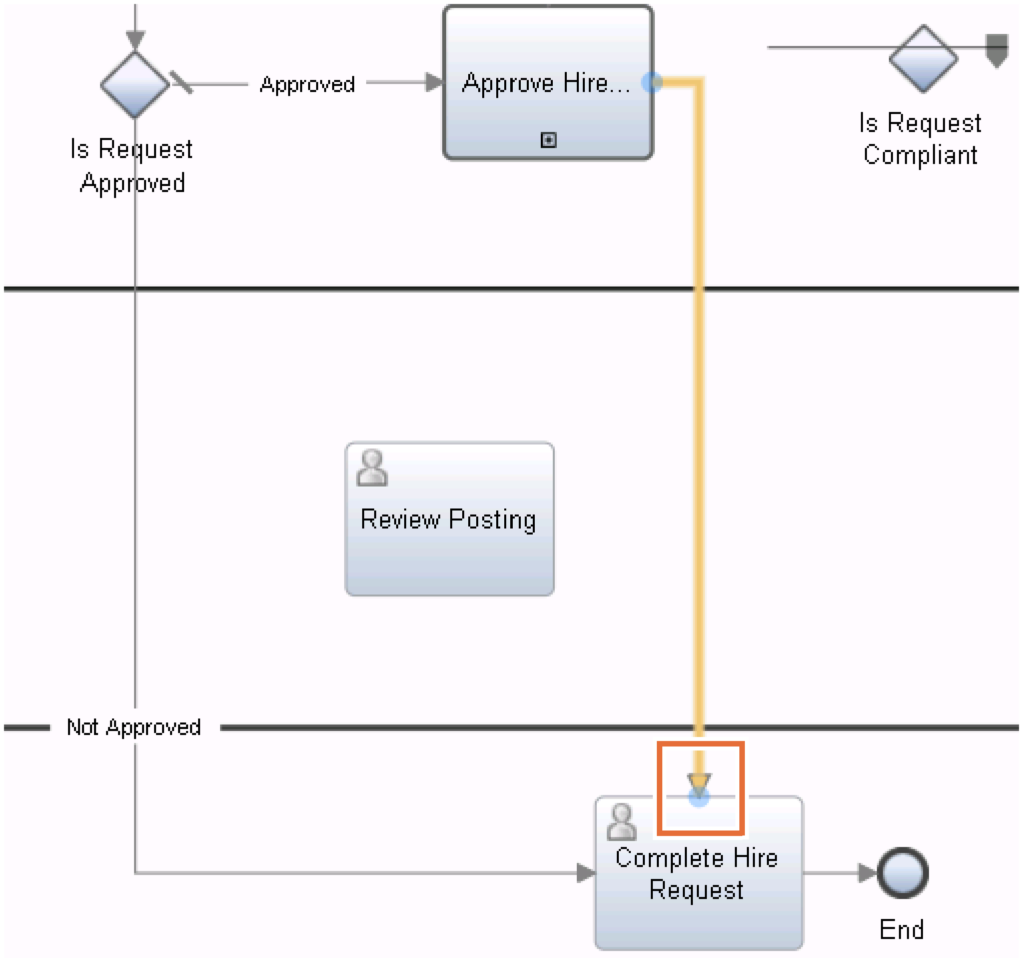
__ a. Drag a **Gateway** to the right of **Approve Hire Request**.

__ b. Rename the gateway to: `Is Request Compliant`

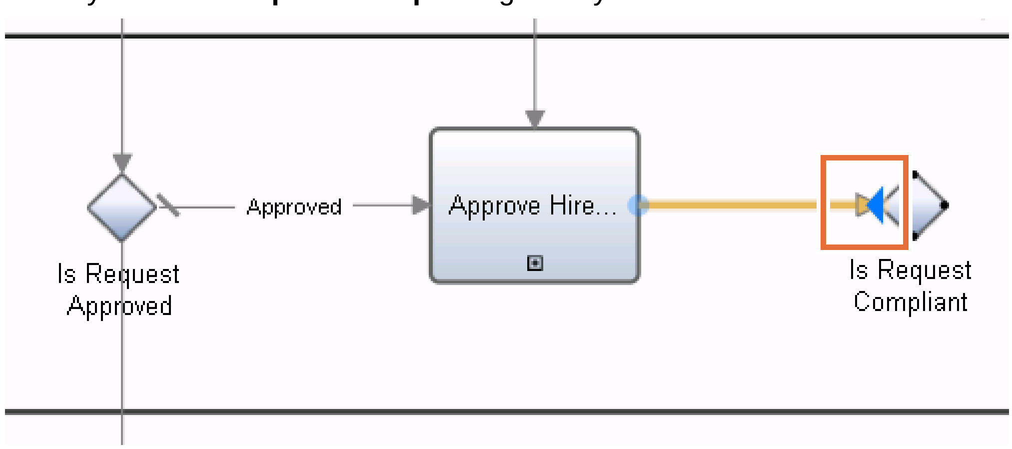


__ 4. Connect the **Review Posting** activity.

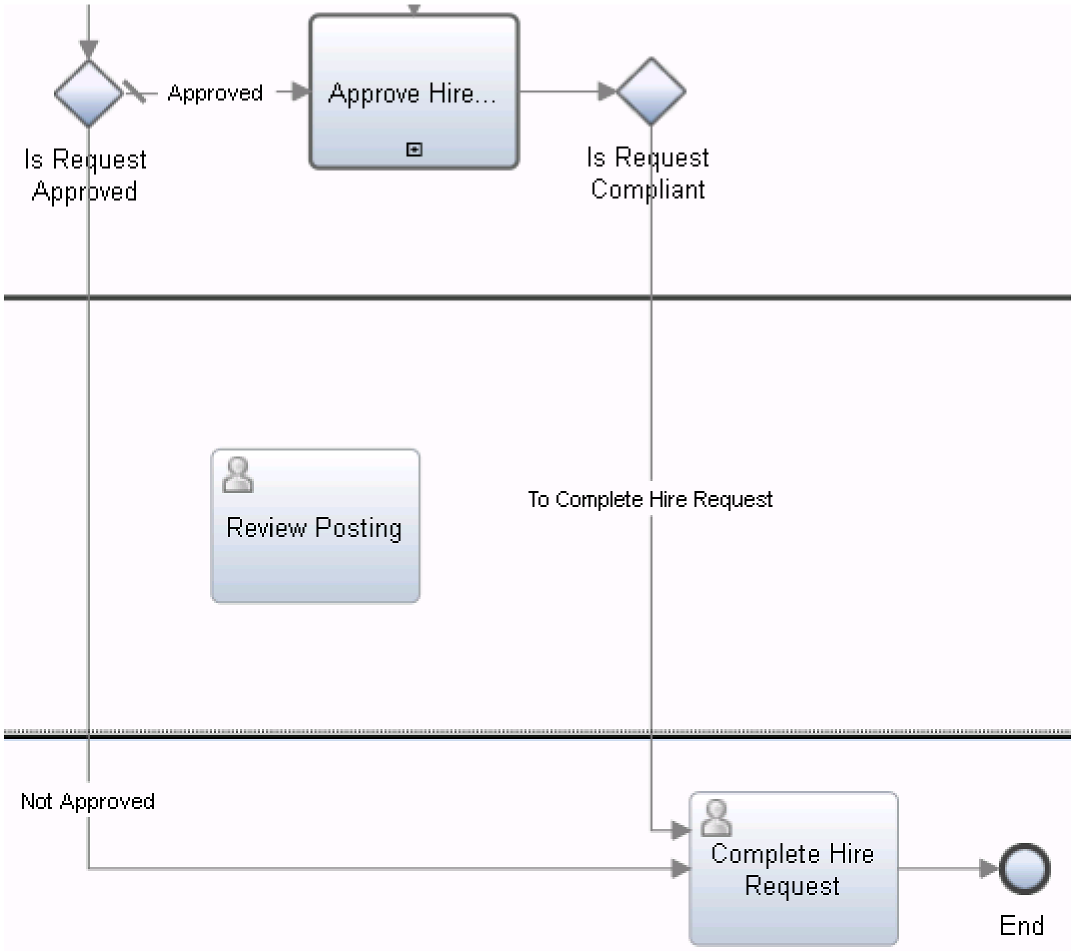
__ a. Select the sequence flow between **Approve Hire Request** and **Complete Hire Request** activities.



__ 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.

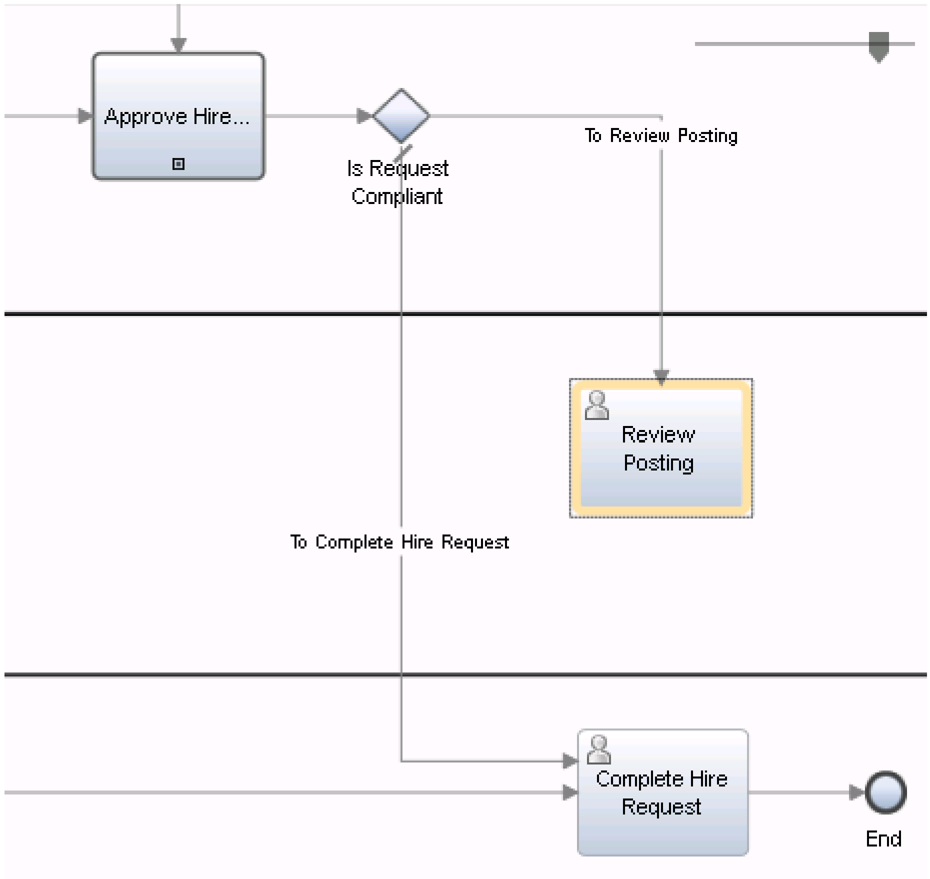


__ c. Connect the **Is Request Compliant** gateway to **Complete Hire Request**.

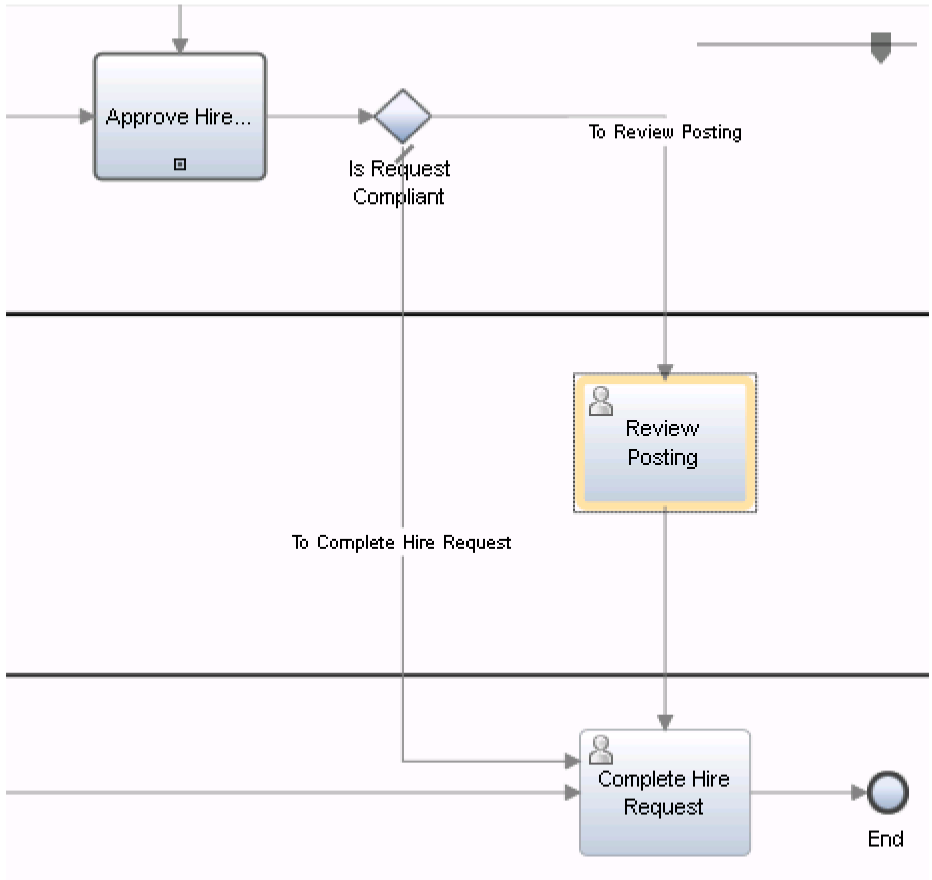


__ d. Drag the **Review Posting** activity on top of the **Complete Hire Request** activity.

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



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

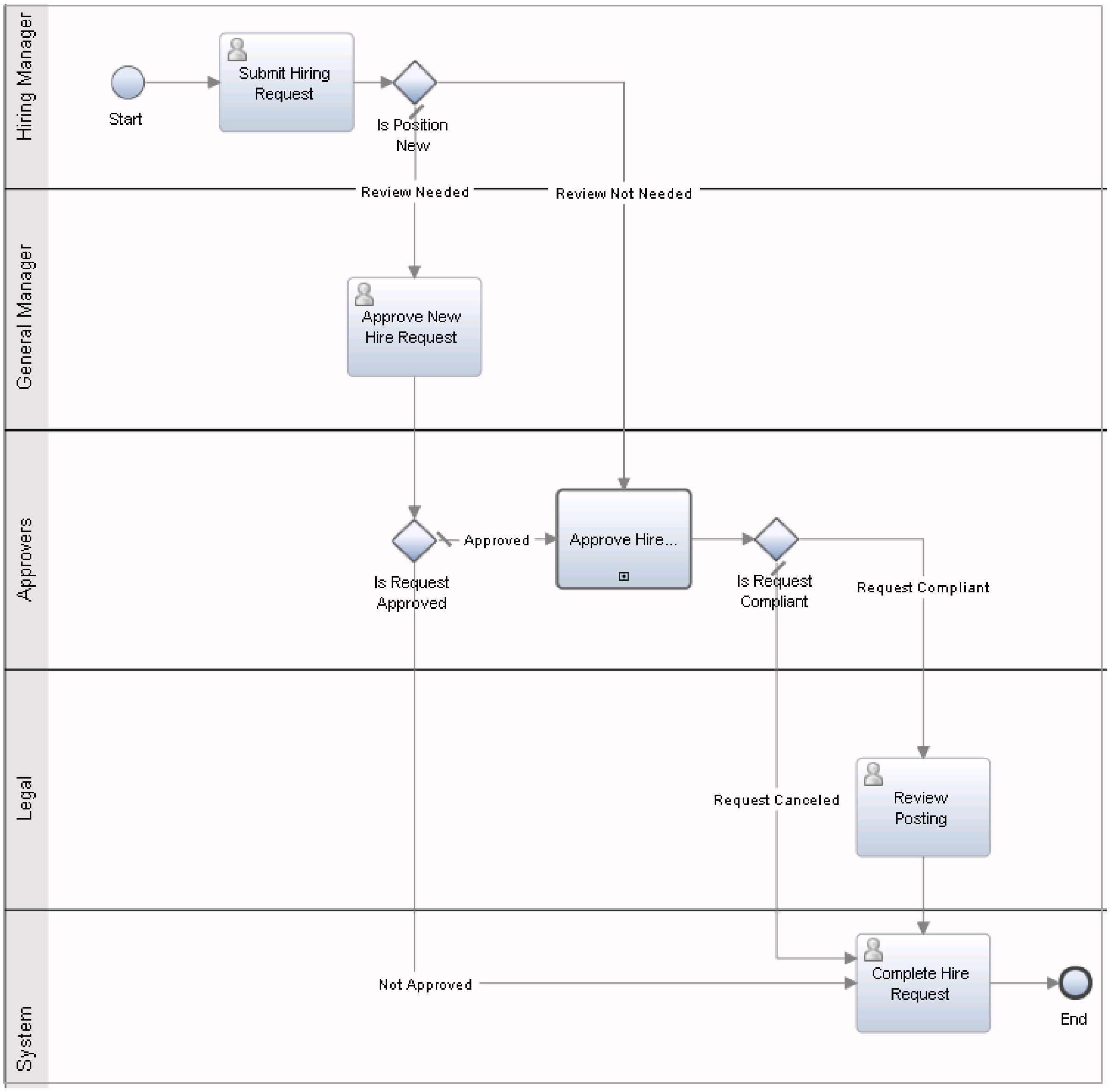


__ 5. Label the flow lines.

__ 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`

Your final process displays as follows:



__ 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 for 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.

Requirements

Successful completion of the previous exercise is required.

Exercise instructions

Part 1: Implement the intermediate timer event

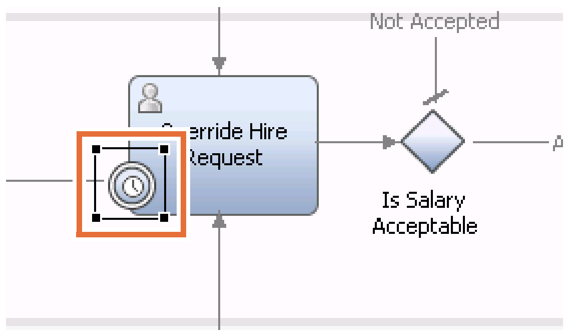
In Playback 0 of the Hiring Request Process, an attached timer intermediate event was modeled. The HR Administrator has 4 hours to complete the review. If the review is not completed within 4 hours, an email is sent to the HR Administrator. The email notifies the HR Administrator of the missed deadline. 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 web Process Designer and open the **Approve Hire Request** process.

__ b. Return to the **Definition** tab.

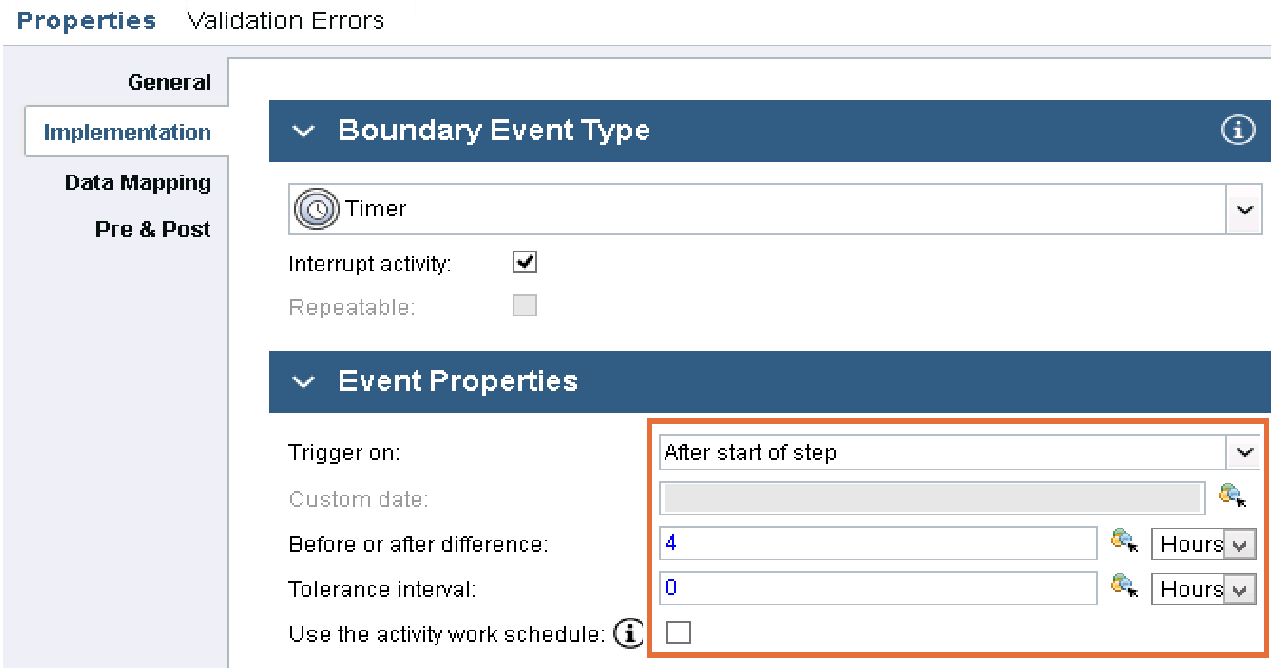
__ c. Click the **timer intermediate event** that is attached to the `Override Hire Request` activity.



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

__ 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`



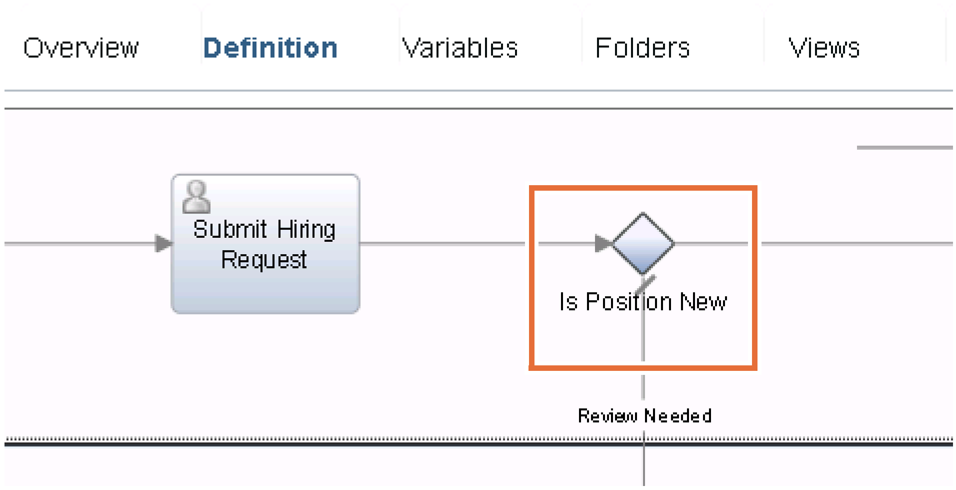
__ f. Save your work.

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

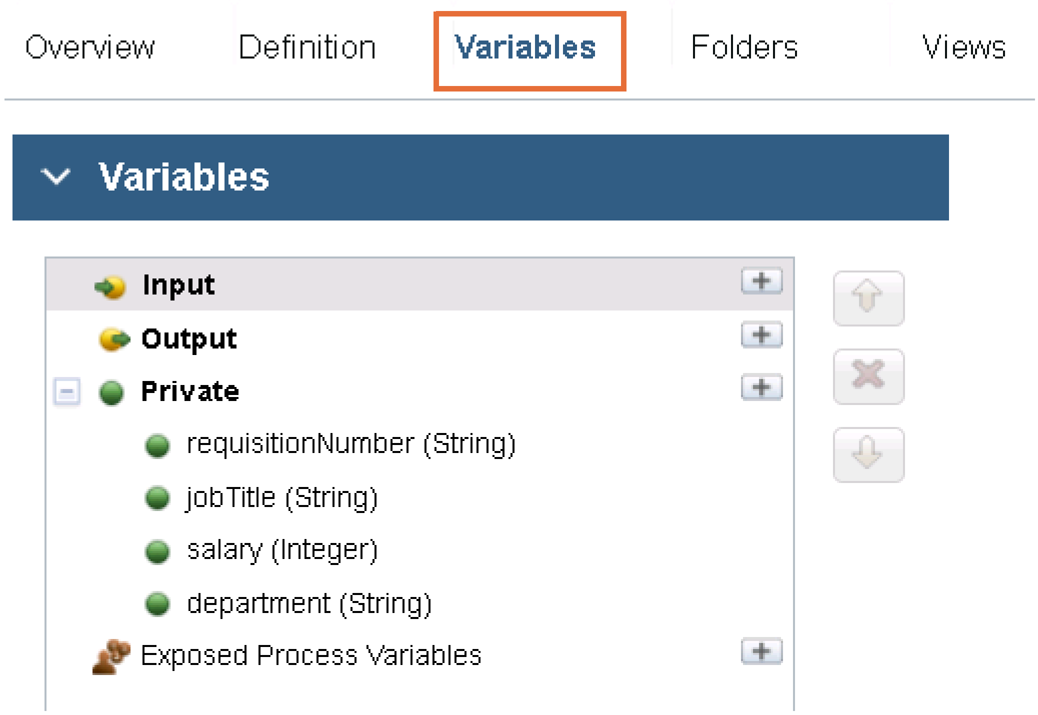
The process model for the **Hiring Request Process** is in place. Process data flow implementation is the one of the goals of Playback 1. Now you must start to implement the process, first by creating simple variables to implement the logic behind the decision gateways on the current process.

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

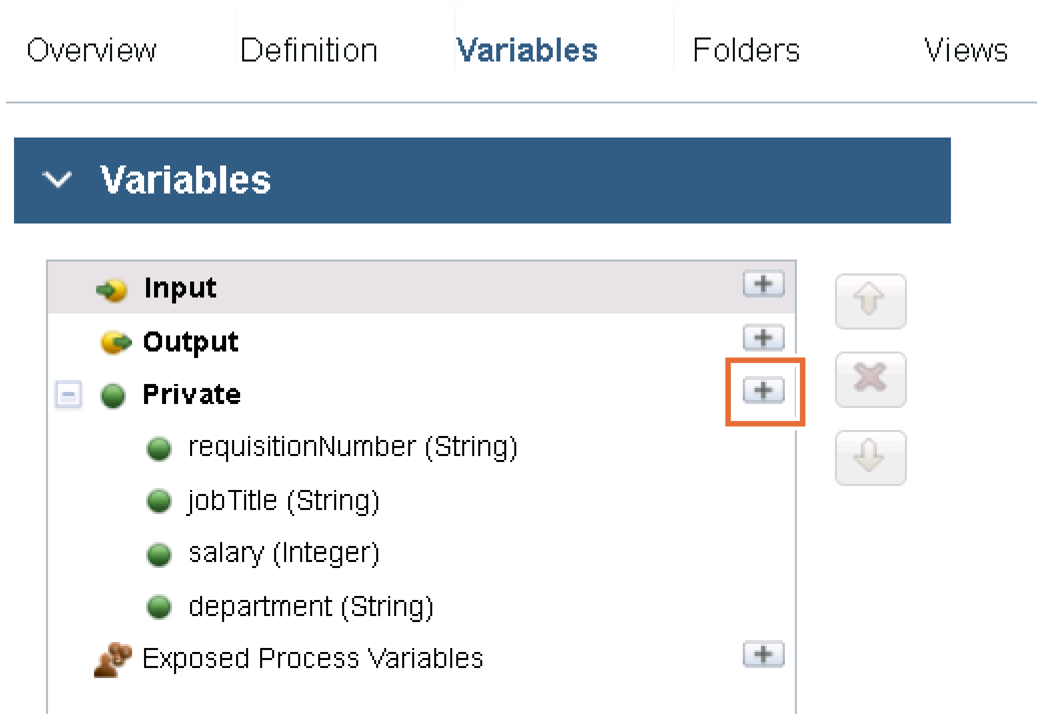
__ 2. Create a private variable to control the **Is Position New** exclusive gateway.



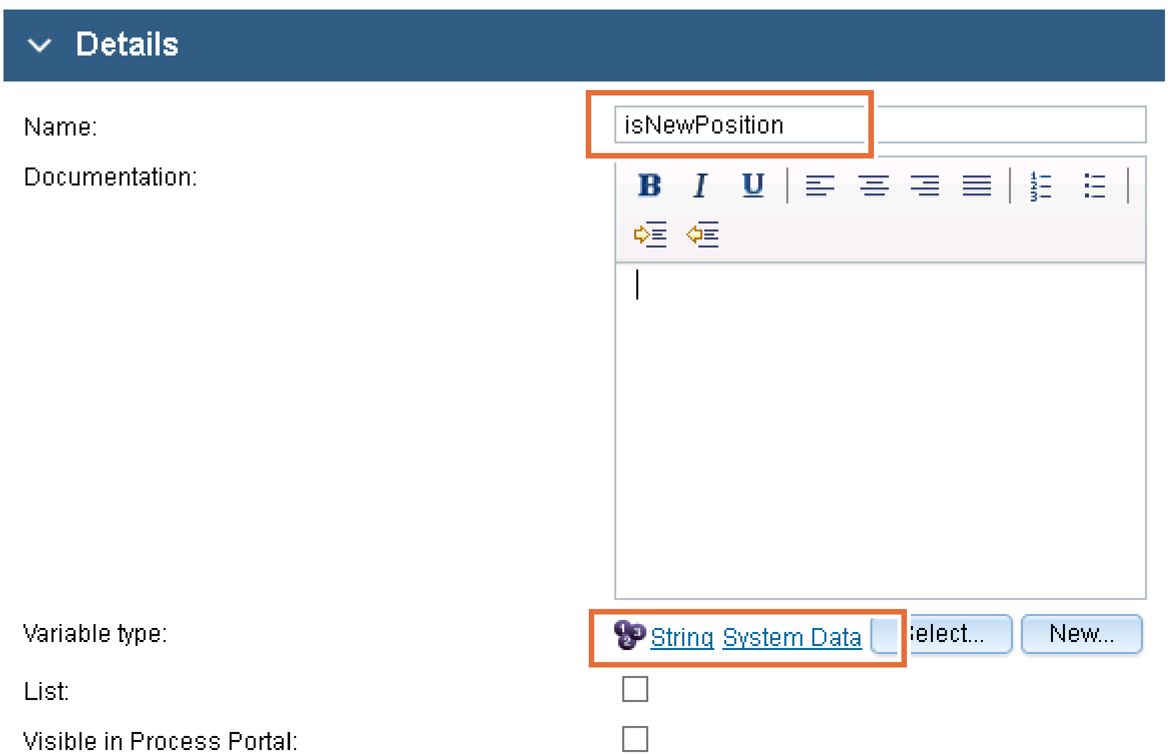
__ a. Click the **Variables** tab.



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



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

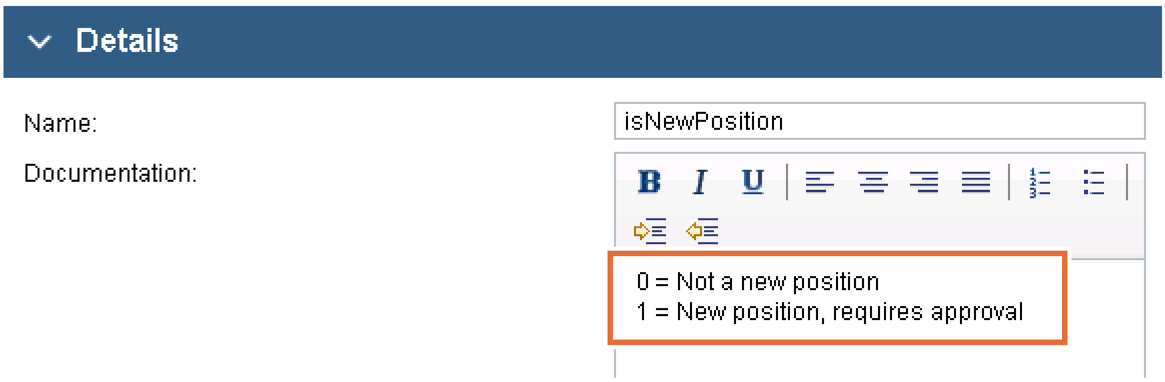


Important

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.

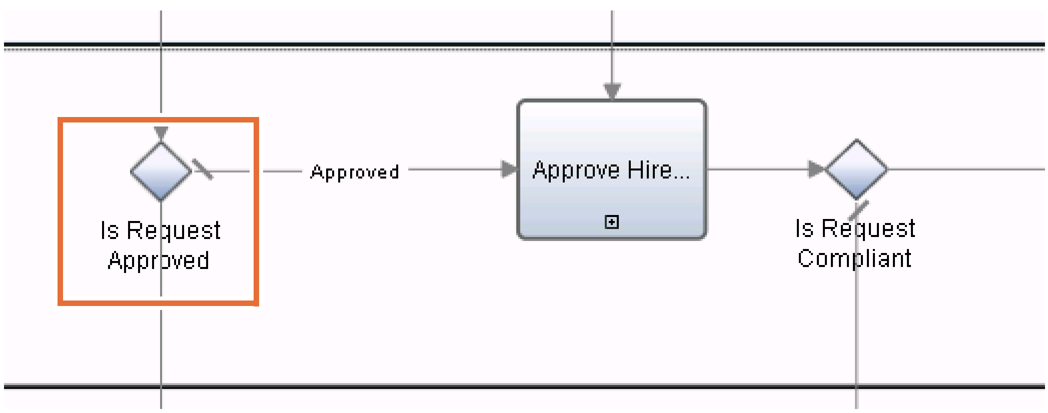
__ d. In the **Documentation** field, insert the following text:

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



__ e. Do not change the other options. Save your changes.

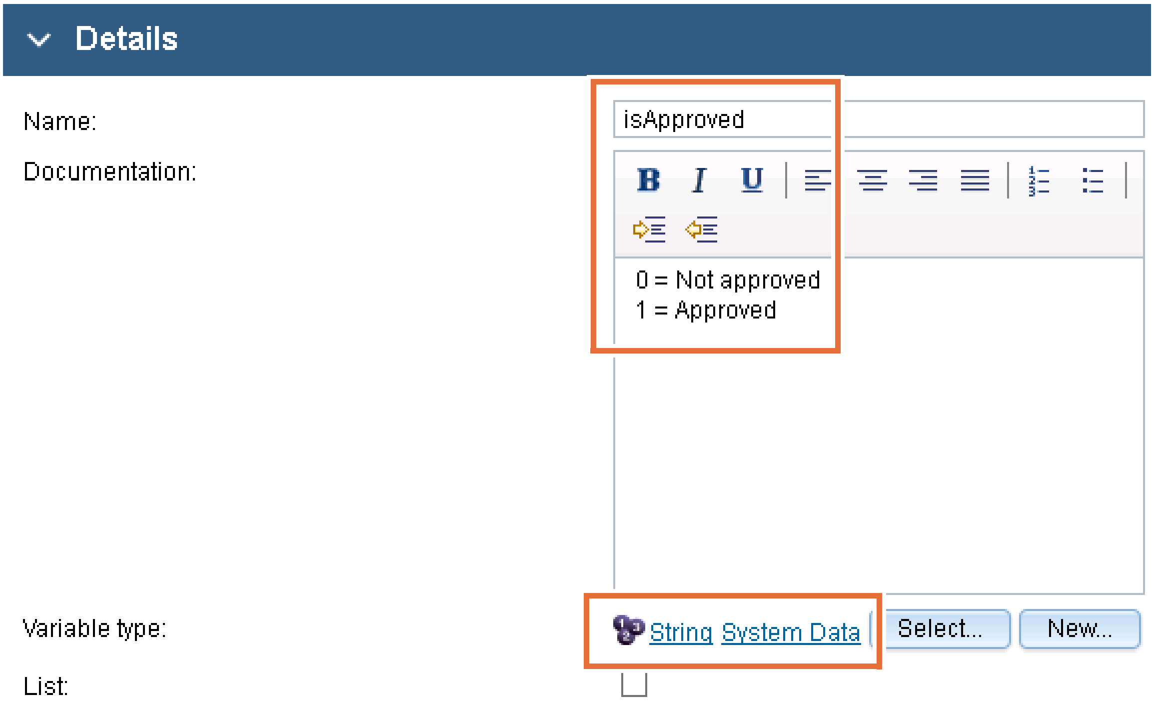
__ 3. Create a variable to control the **Is Request Approved** exclusive gateway.



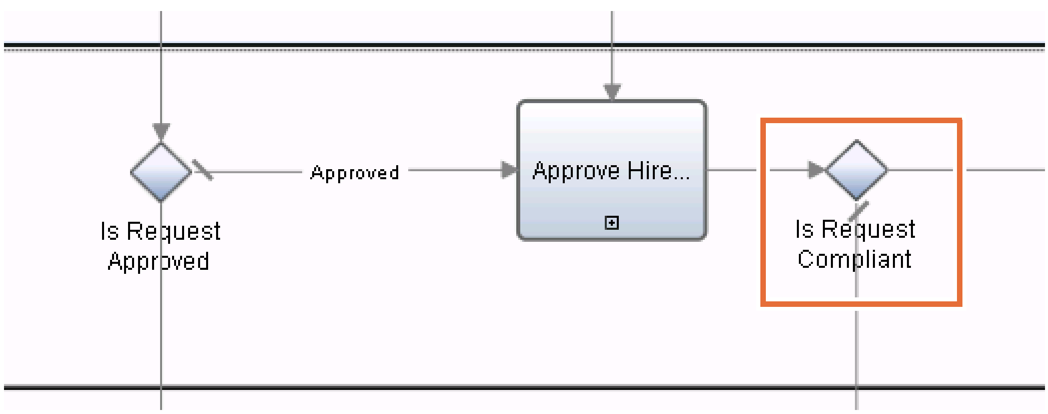
__ a. On the **Variables** tab, create another Private variable: `isApproved (String)`

__ 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.



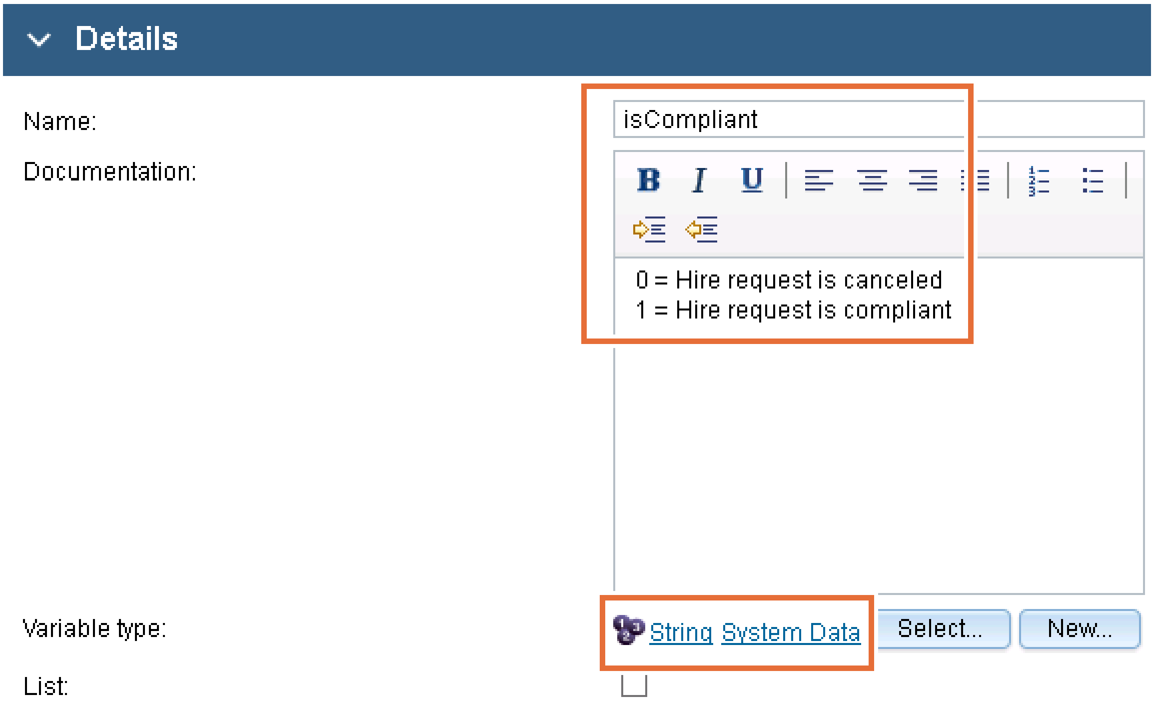
__ a. On the **Variables** tab, create another Private variable: `isCompliant (String)`

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
```



__ c. Save your changes.

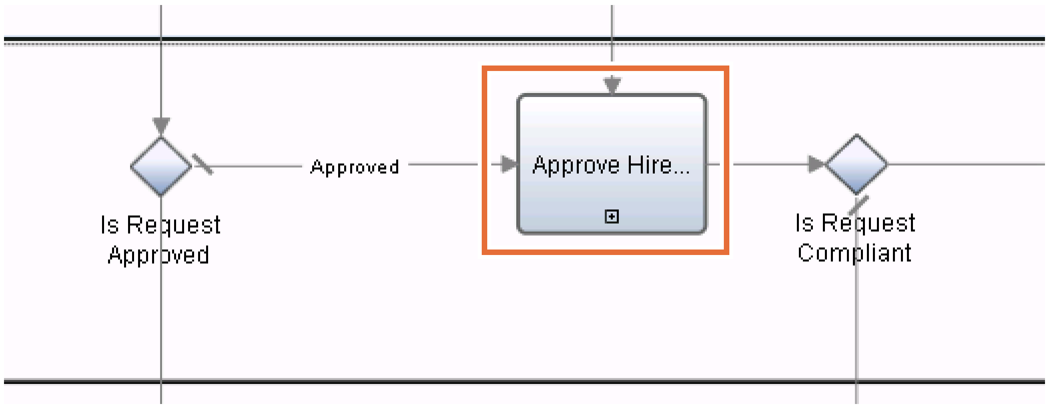
You created three private variables in this exercise.

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.

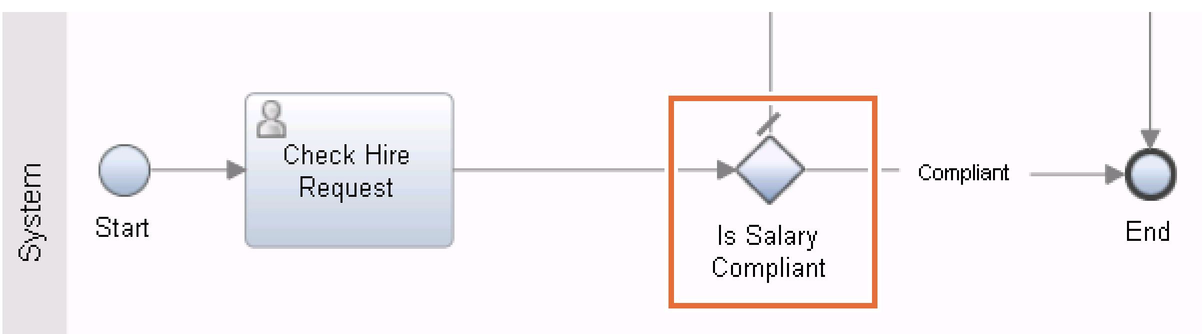
__ b. Open the **Approve Hire Request** linked process by double-clicking the linked process in the diagram.



Important

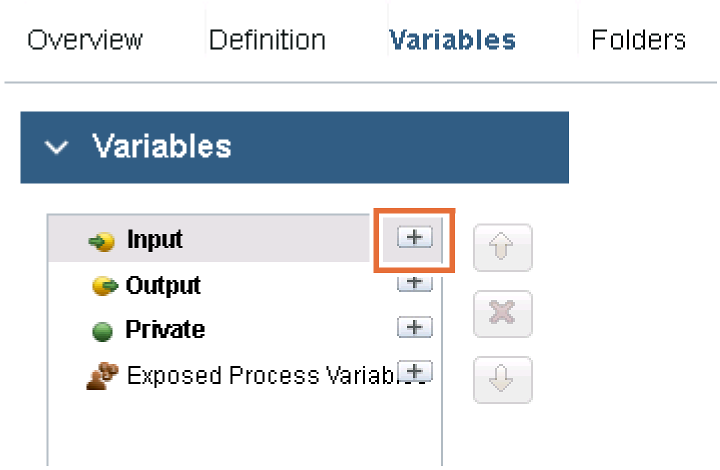
Be sure that you select the **Approve Hire Request** linked process, not the **Approve New Hire Request** activity. Look for the linked process icon on the bottom center.

__ 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.

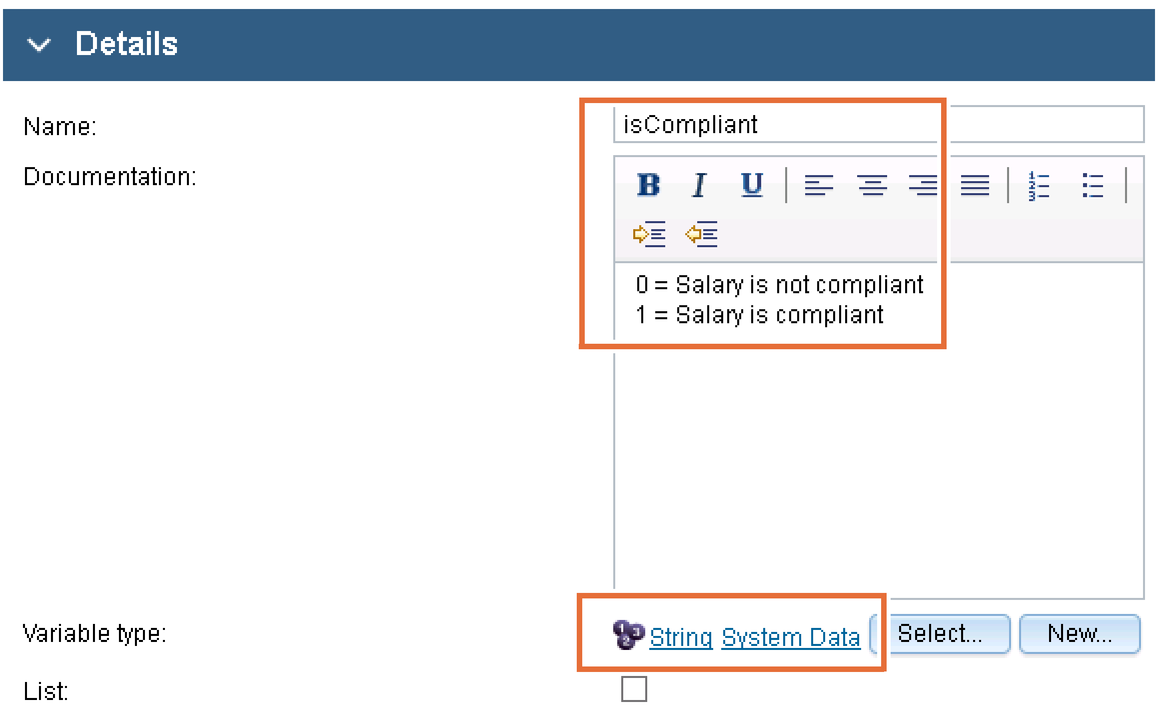
__ e. Click the (**+**) plus sign next to Input.



__ f. Name the input variable: `isCompliant`
Verify that the Variable Type is **String**.

__ g. Insert the following text in the documentation box:

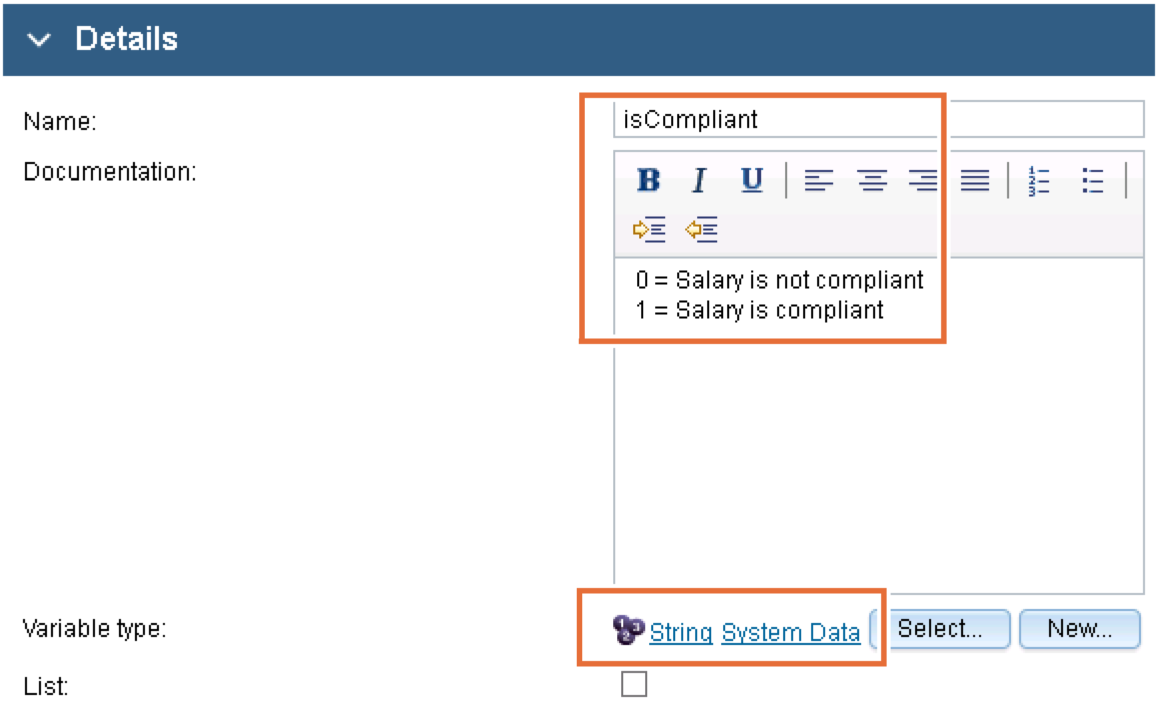
```
0 = Salary is not compliant
1 = Salary is compliant
```



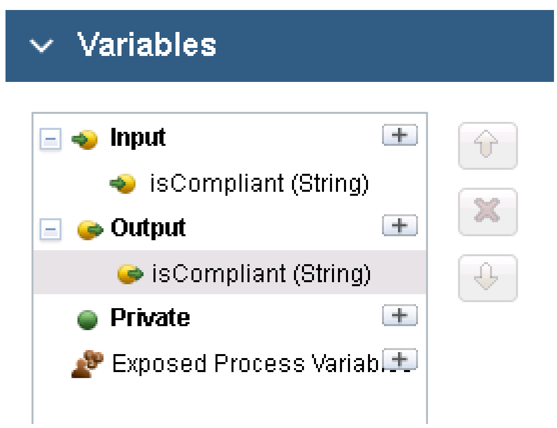
__ h. Create an output variable: `isCompliant (String)`

__ i. Insert the following text in the documentation box:

```
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

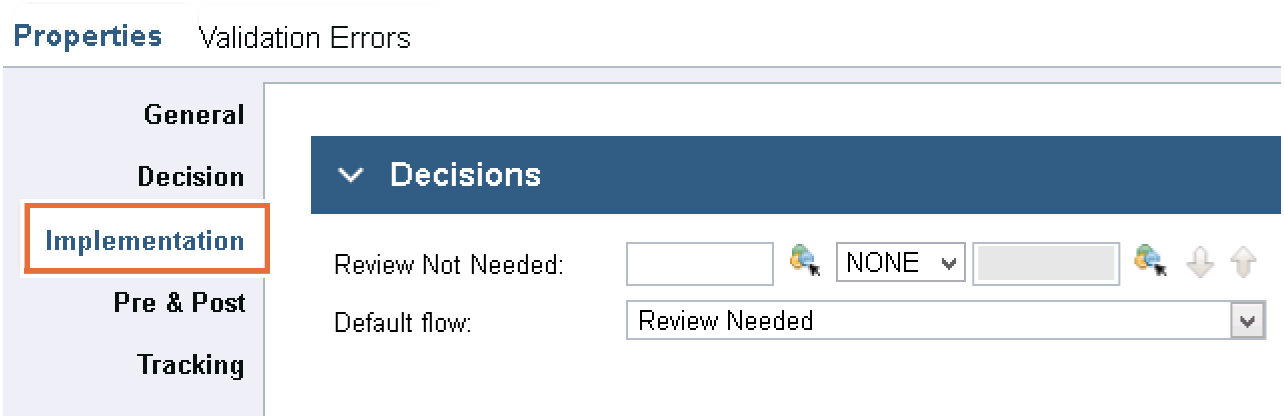
When the position is new, the process must flow to the Approve New Hire Request activity down the Review Needed flow.

__ a. Open the **Hiring Request Process**.

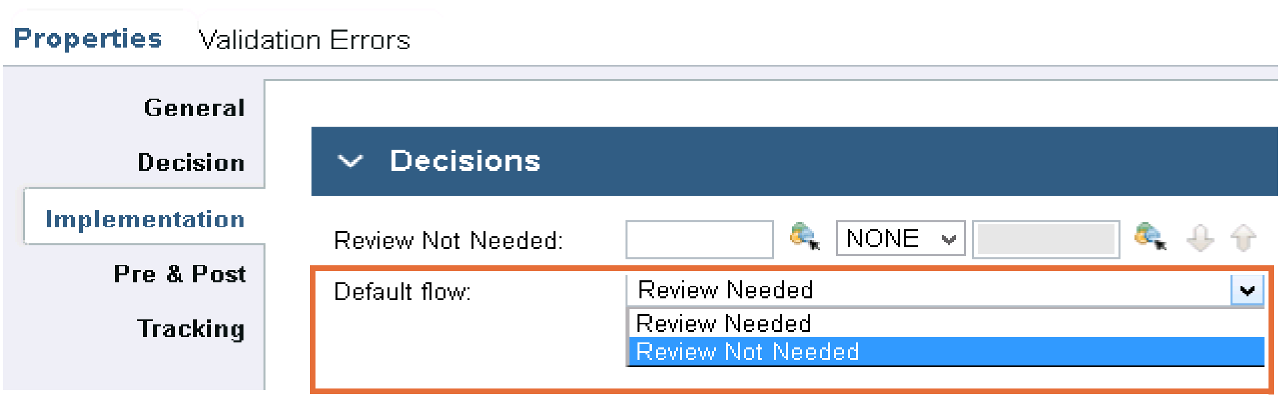
__ b. Switch to the **Definition** tab.

__ c. Click the **Is Position New** exclusive gateway.

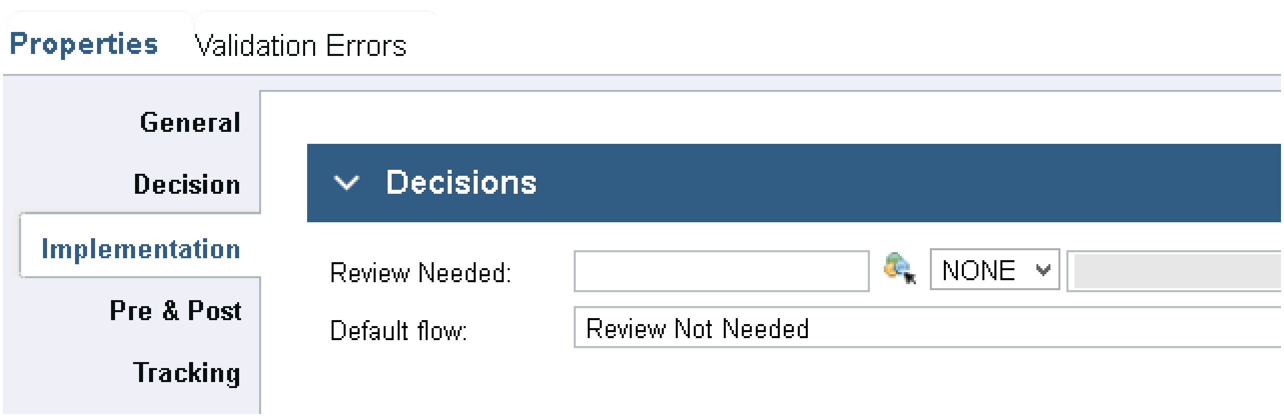
__ d. Open the **Properties > Implementation** menu.



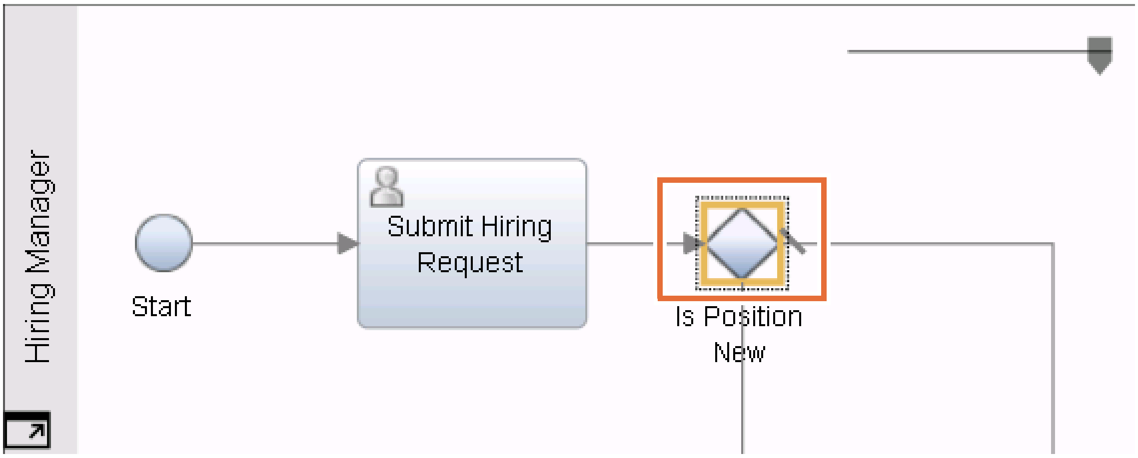
__ e. In the **Decisions** section, open the **Default flow** menu.



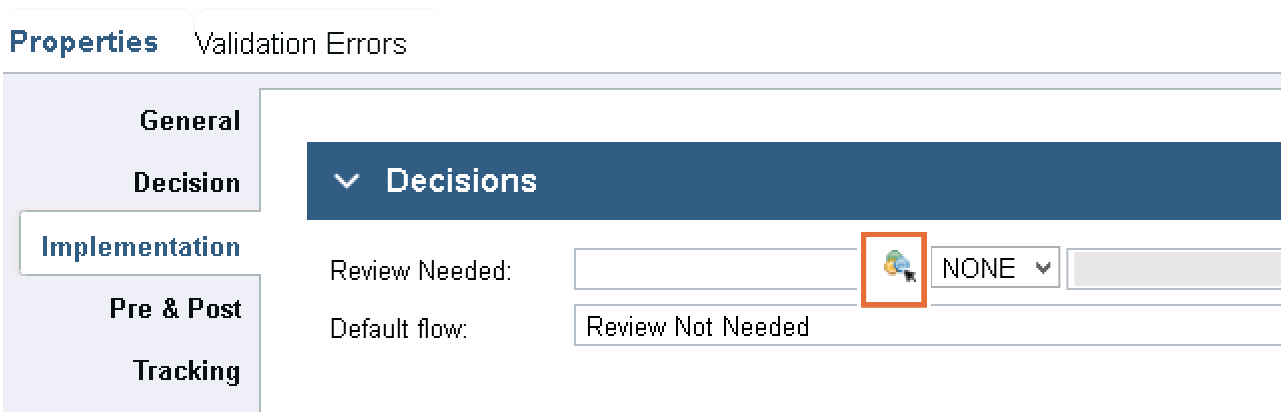
__ f. Select **Review Not Needed**. This selection makes **Review Not Needed** the default sequence flow. The order of sequence flow starts with **Review Needed** at the top and **Review Not Needed** on the bottom.



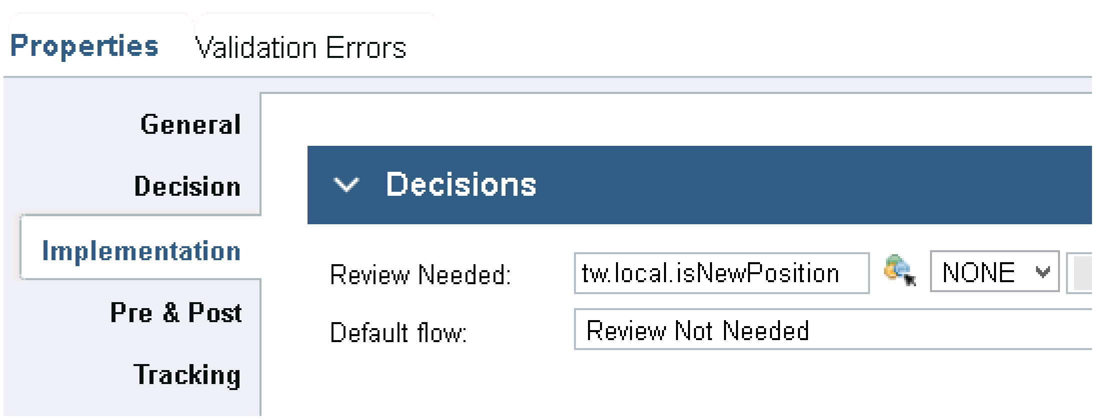
__ g. Verify that the default flow is now changed for the **Is Position New** gateway.



__ h. In the first field to the right of the **Review Needed** label, click the selection icon.

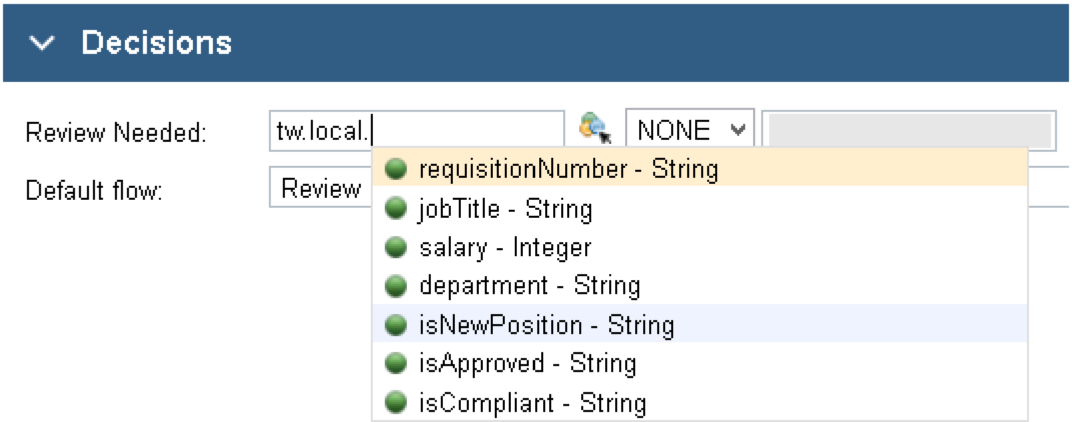


__ i. Select **isNewPosition (String)**. The field is populated with the variable: `tw.local.isNewPosition`



Hint

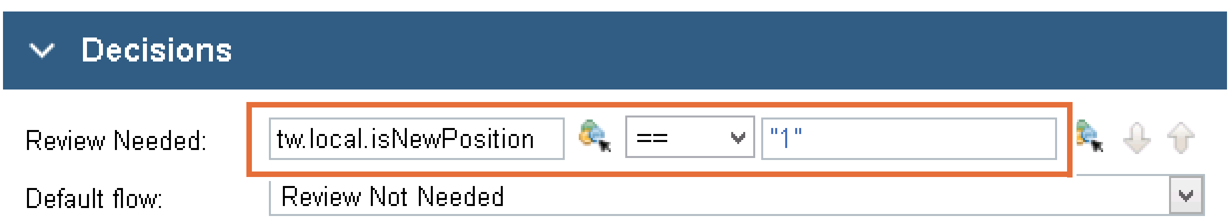
You can also type this variable and as you type and press Ctrl + Spacebar, the auto-complete feature suggests different options to you that match what you already typed. The options are filtered as you complete your entry.



The 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"`



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

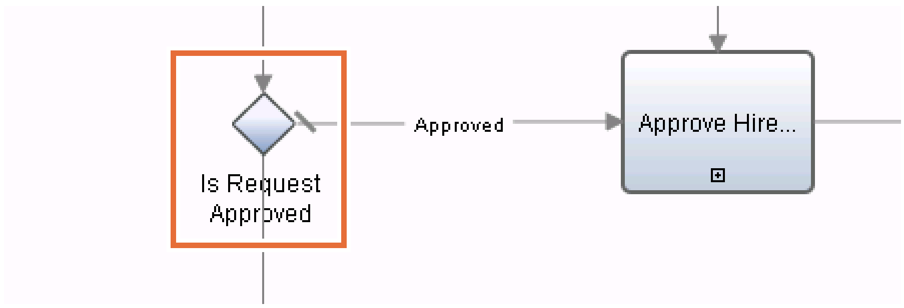
The condition is set for the **Is Position New** gateway. If the first condition is not met, the default condition is **Review Not Needed**.

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

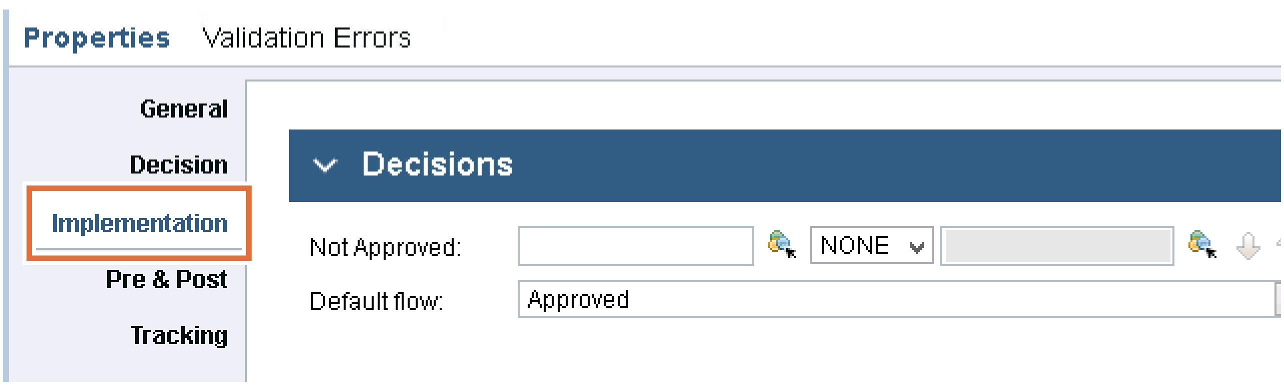
Reminder

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.

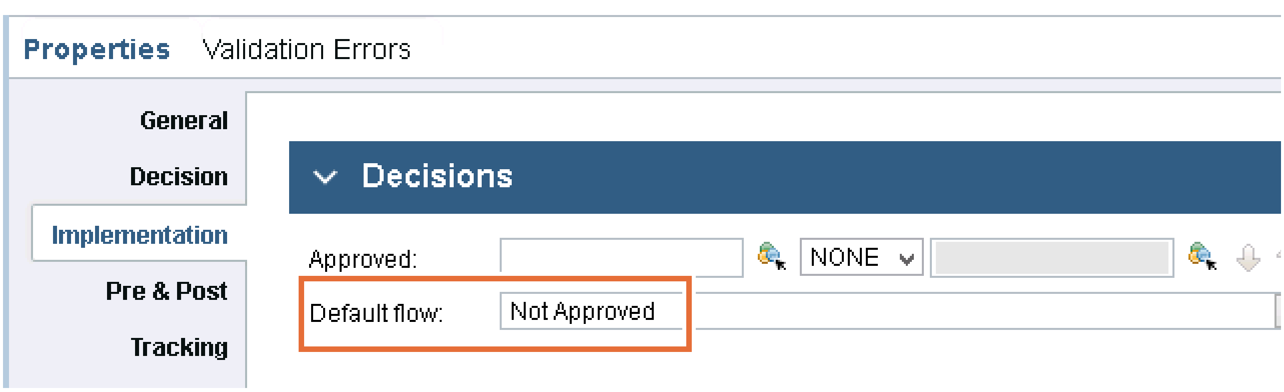


__ b. Open the **Properties > Implementation** menu.



__ c. In the **Decisions** section, open the Default flow menu.

__ 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.

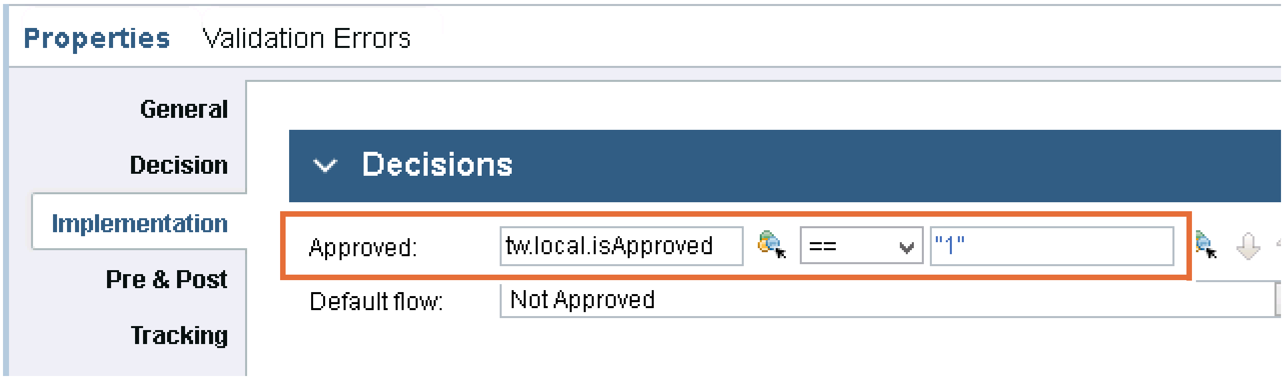


__ e. In the first field to the right of **Approved**, map to:

`tw.local.isApproved`

__ f. Next, change the drop-down menu value to: ==

__ g. In the last field, type: "1"



__ 3. Implement the **Is Request Compliant** exclusive gateway.

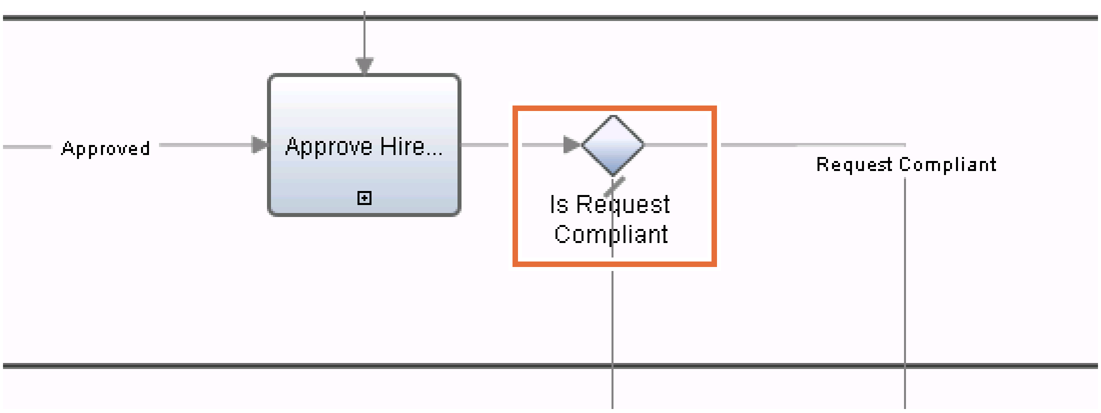
Reminder

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

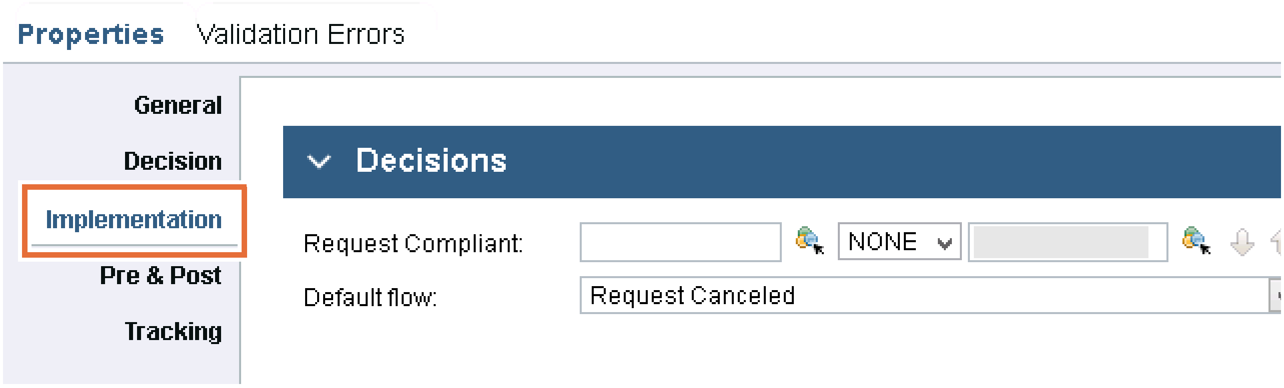
Note

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.

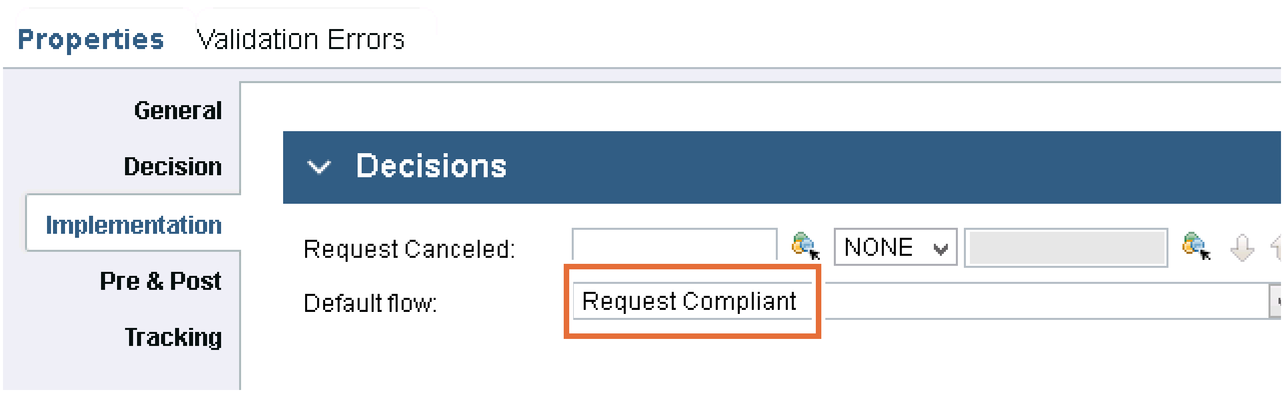


__ b. Click the **Properties > Implementation** menu.



__ c. In the **Decisions** section, open the Default flow menu.

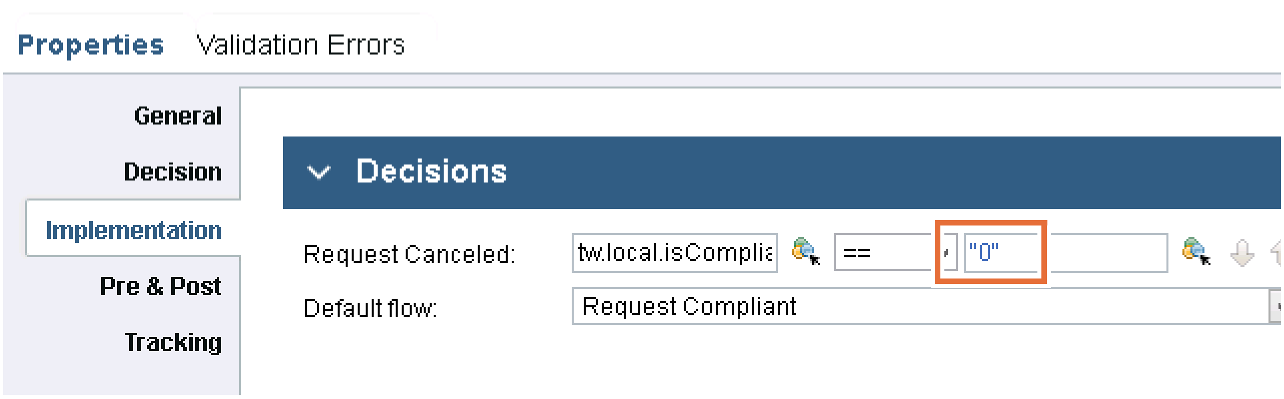
__ d. Select **Request Compliant**. This arrangement makes **Request Compliant** the default sequence flow. The order of sequence flow expresses that the **Request Canceled** option is first and that the **Request Compliant** option is second.



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

__ f. Change the drop-down value to: ==

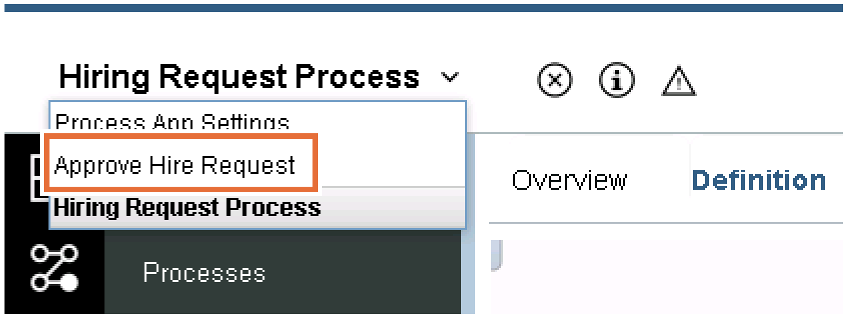
__ g. In the last field to the right of **Compliant**, type: `"0"`



__ h. Save your work.

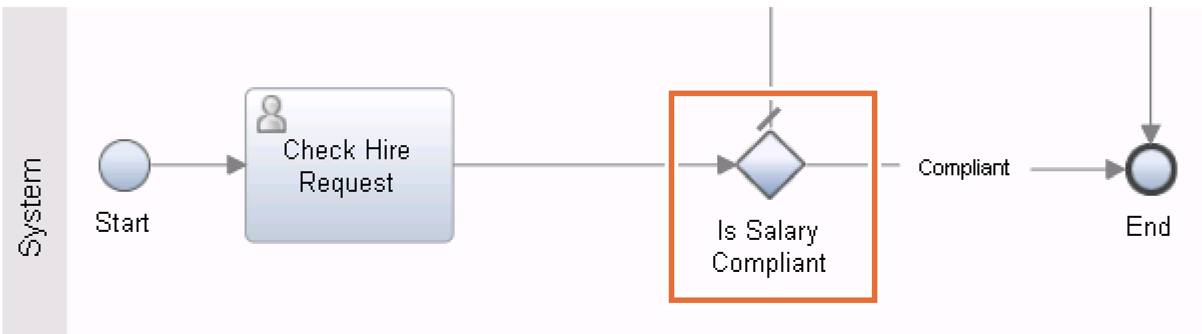
__ 4. Implement the **Is Salary Compliant** exclusive gateway in the **Approve Hire Request** linked process. When the salary is compliant, the flow must move to the end event down the compliant flow.

__ a. Open the **Approve Hire Request** linked process from the History menu.



__ b. Return to the **Definition** tab.

__ c. Select the **Is Salary Compliant** gateway.



Reminder

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

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

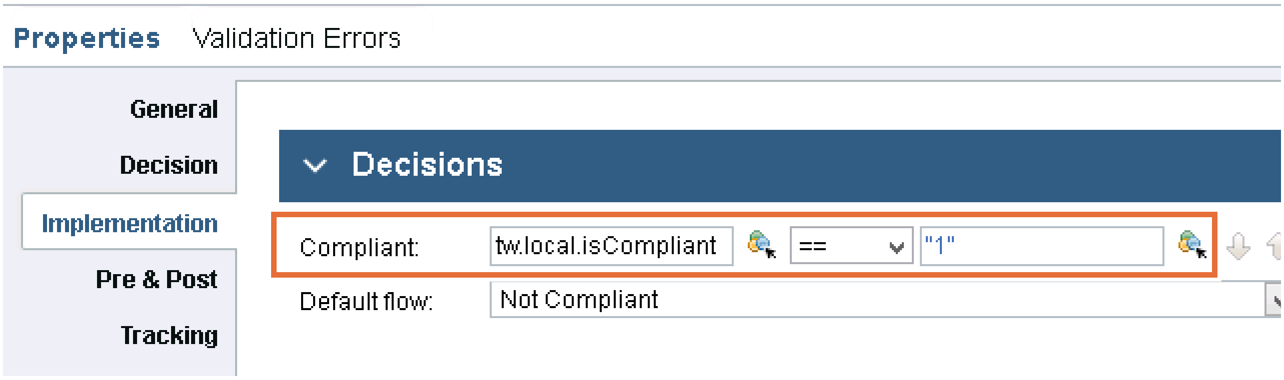
__ e. In the **Decisions** section, open the Default flow menu.

__ f. Select **Not Compliant**. This arrangement makes **Not Compliant** the default sequence flow. The order of sequence flow in the **Implementation** section expresses that the **Compliant** option is first and that the **Not Compliant** option is second.

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

__ h. Change the drop-down value to: ==

__ i. In the last field to the right of **Compliant**, type: "1"



__ j. 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.

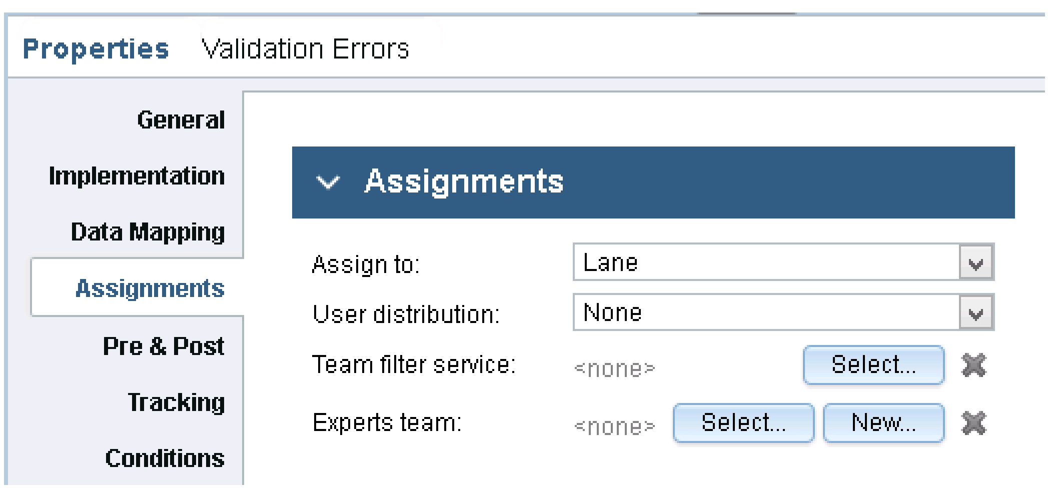
__ 1. Route the Submit Hiring Request activity in the Hiring Request Process to **Lane** and distribution to **Last User**.

__ a. Open the **Hiring Request Process**.

__ b. Switch to the **Definition** tab.

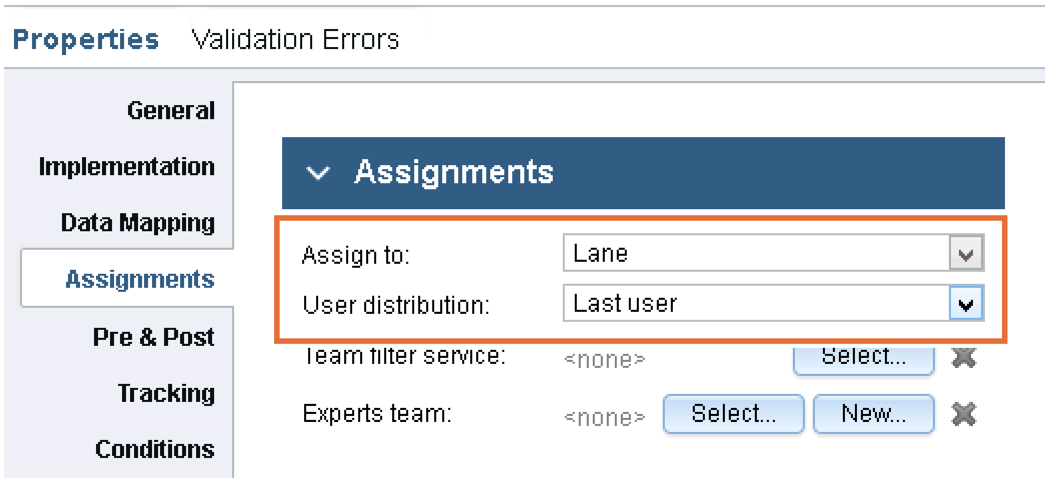
__ c. Click the **Submit Hiring Request** activity on the **Hiring Request Process**.

__ d. Open the **Properties > Assignments > Assignments** section.



__ e. Verify that the default assignment for **Assign to** is: Lane

__ f. Change the **User distribution** to: `Last user`



Information

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

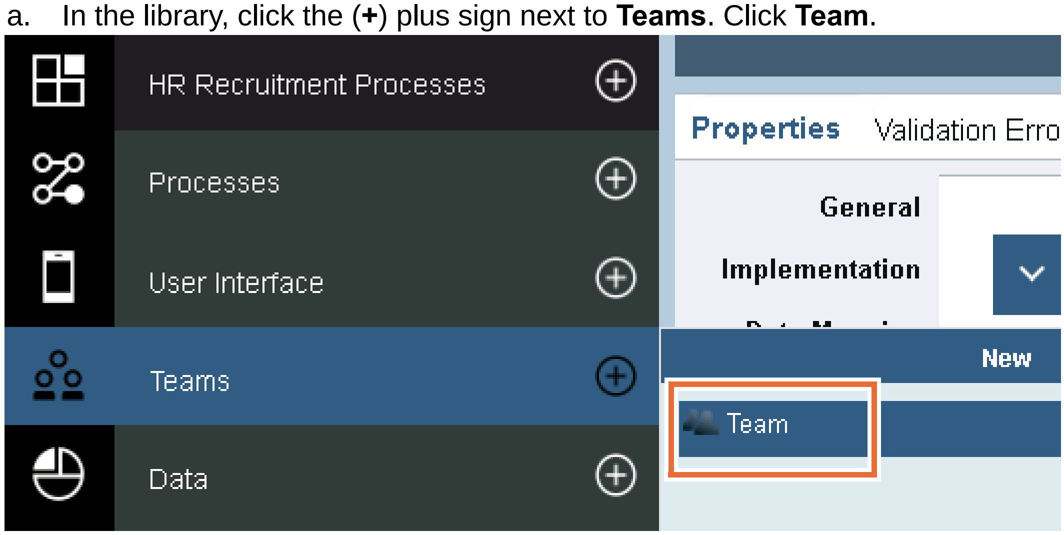
- If the assignment option is set to **Lane**, the task is assigned to the user who completed the activity that immediately precedes the selected activity in the lane.
- If the assignment option is set to **Team**, the task is assigned to the user who completed the last task that was assigned to the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

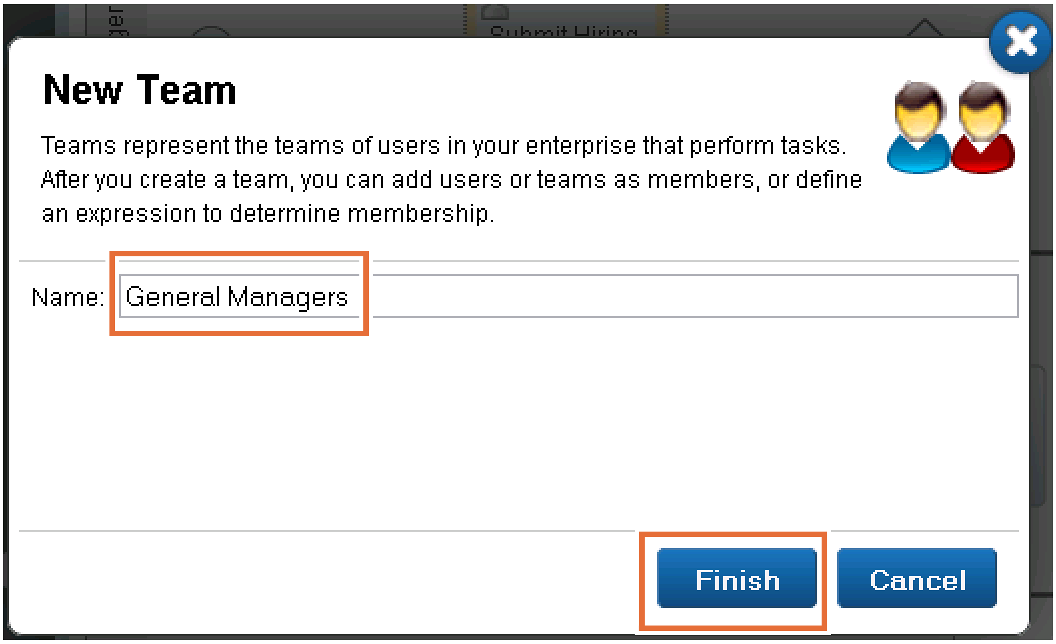
__ g. Save your changes.

__ 2. Create a team and add a member.

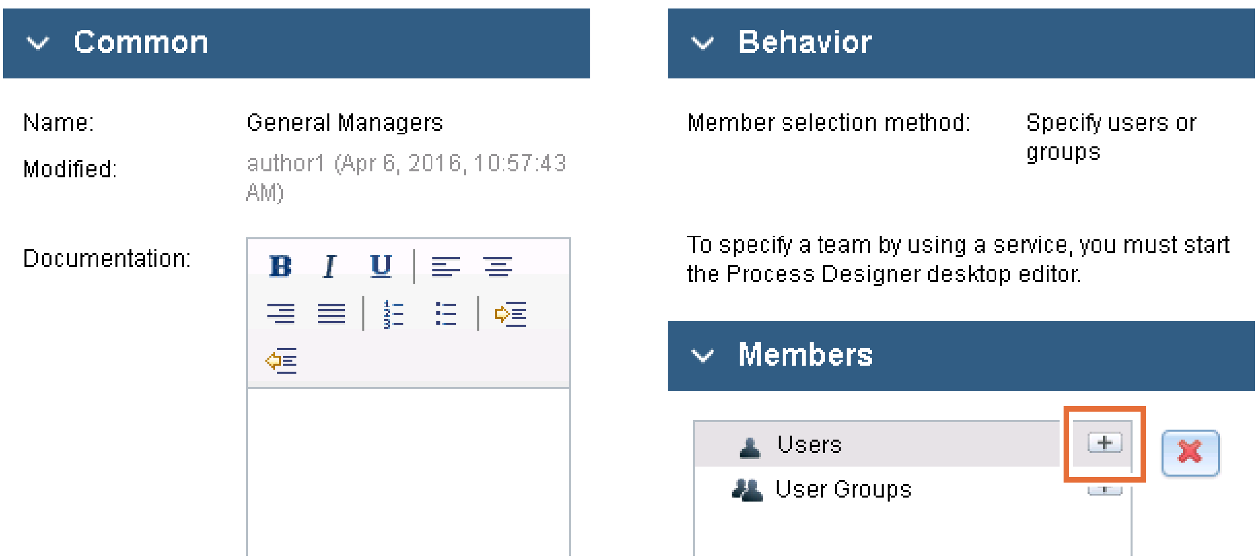
__ a. In the library, click the (**+**) plus sign next to **Teams**. Click **Team**.



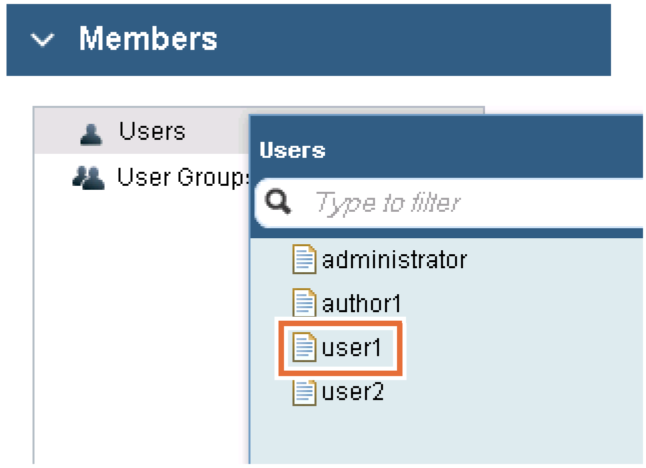
__ b. Name the team `General Managers` and click **Finish**.



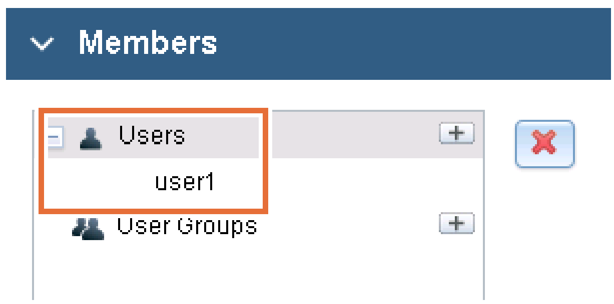
__ c. In the **Members** section, click the (**+**) plus sign next to Users.



__ 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.



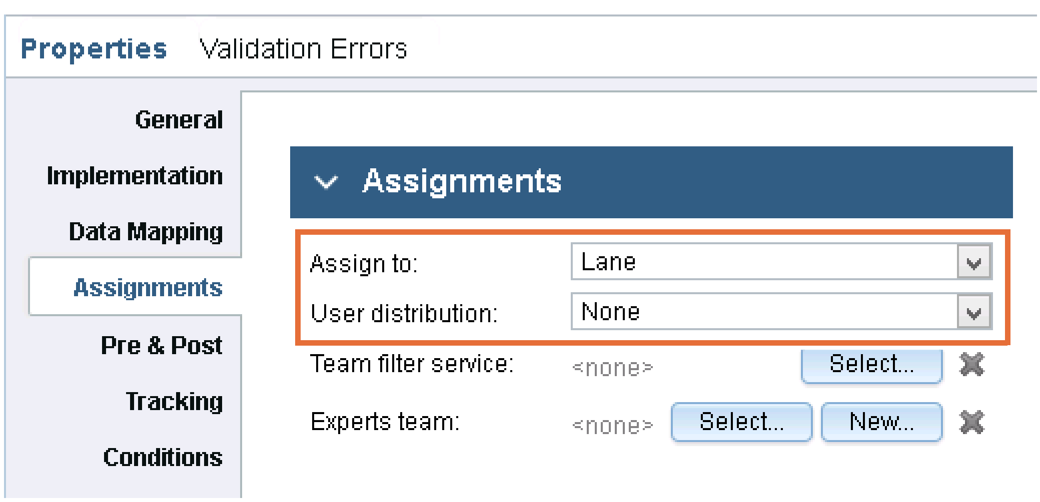
__ e. Save your work.

__ 3. Implement the routing for **Approve New Hire Request**.

__ a. Open the **Hiring Request Process**.

__ b. In the Definition tab, click the **Approve New Hire Request** activity.

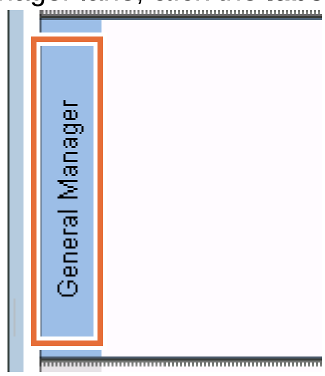
__ c. In the **Properties > Assignments > Assignments** section, leave the value of **Assign To:** as Lane and the value of **User distribution** as None.



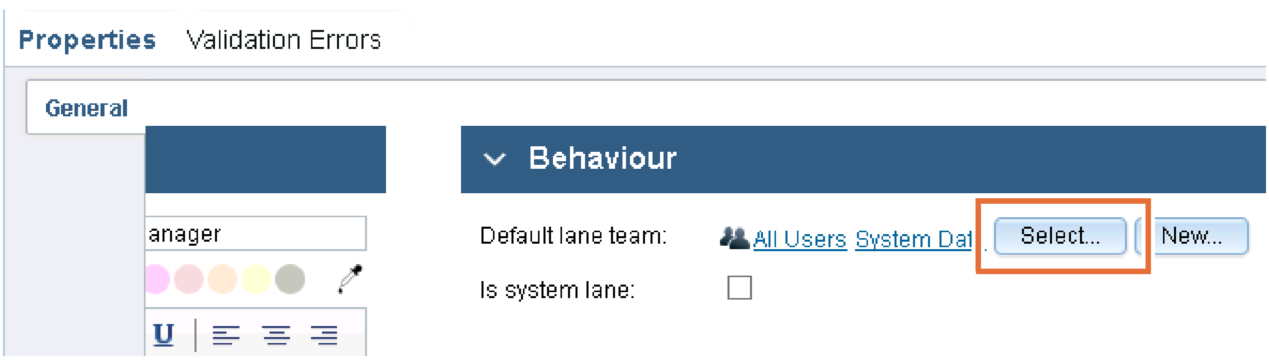
Information

The default **User distribution** option is None. IBM Business Process Manager assigns the runtime task to all the potential users (the default setting).

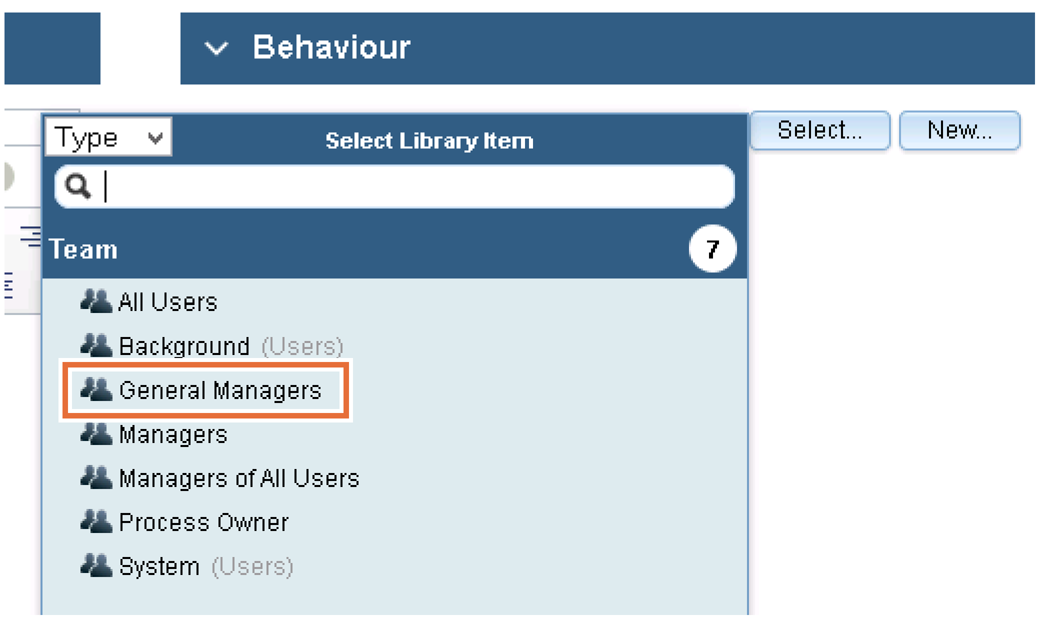
__ d. On the General Manager lane, click the **label** on the left.



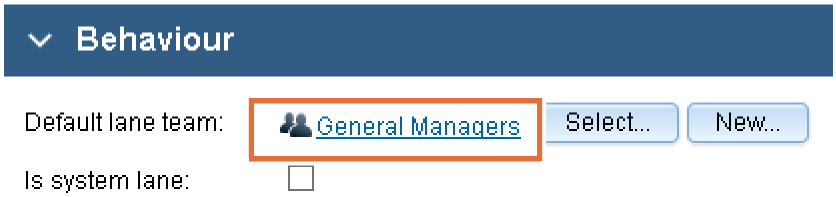
__ e. From the **Properties > General > Behaviour** section, click **Select** next to Default Lane Team.



__ f. Click the **General Managers** team to select it.



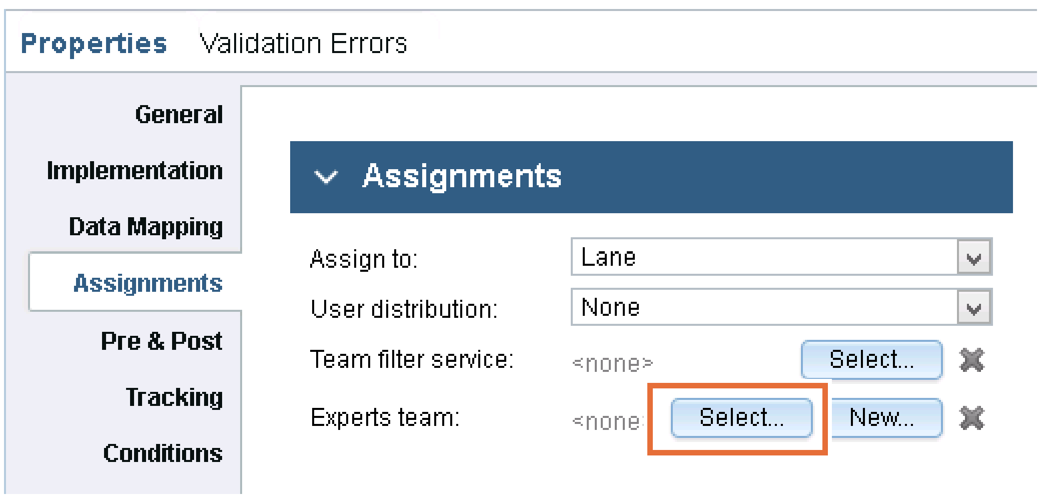
The **General Managers** team is assigned as the default team for the **General Managers** lane.



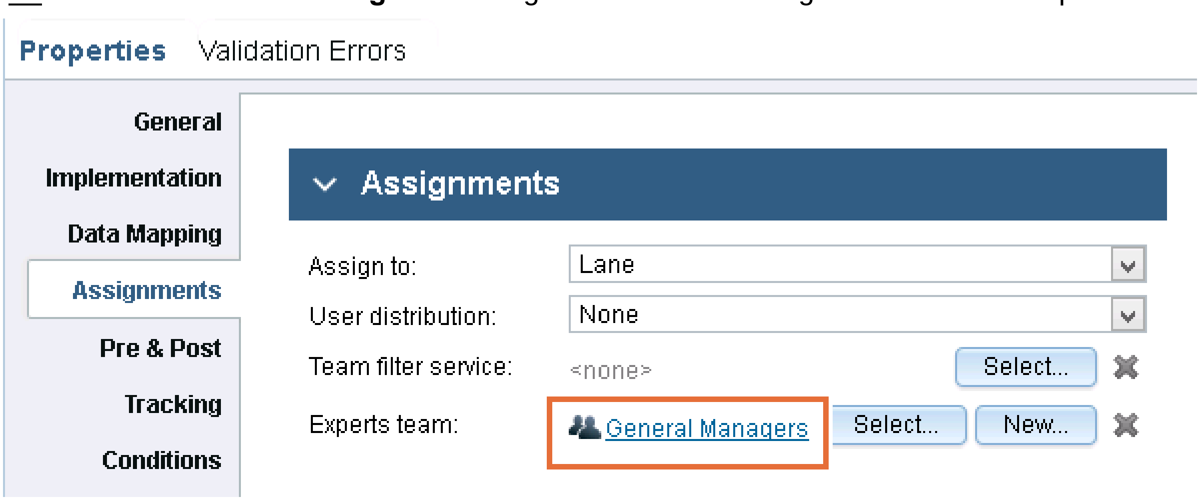
__ 4. Assign the **General Managers** team as the **Experts Group** for the Approve New Hire Request activity.

__ 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.

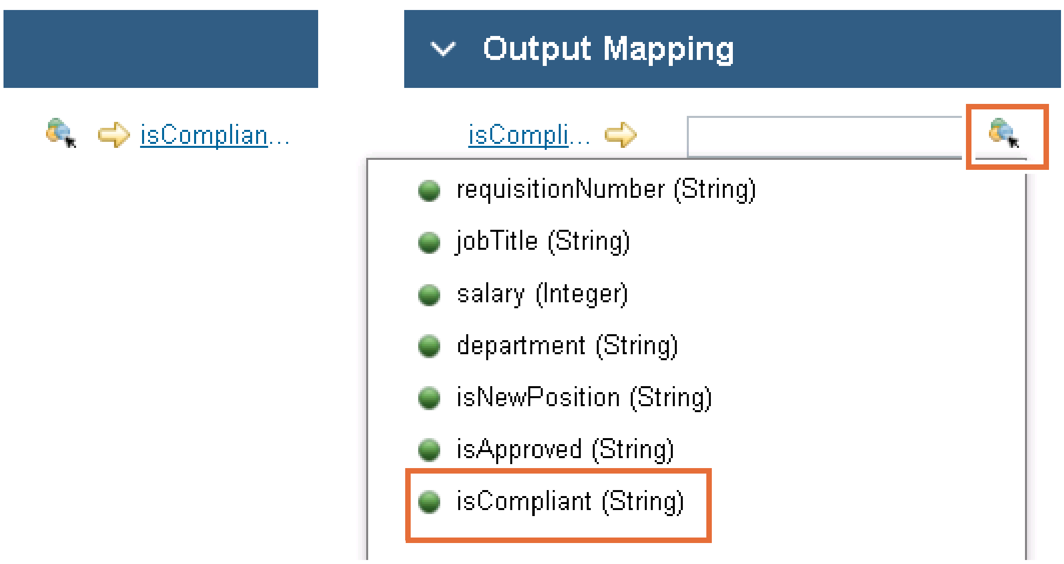
__ a. On the **Hiring Request Process**, click the **Approve Hire Request** linked process.

__ 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.

Hint

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



__ d. Save your work.

Information

If you need more information about conducting the playback, consult 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

00:15

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.

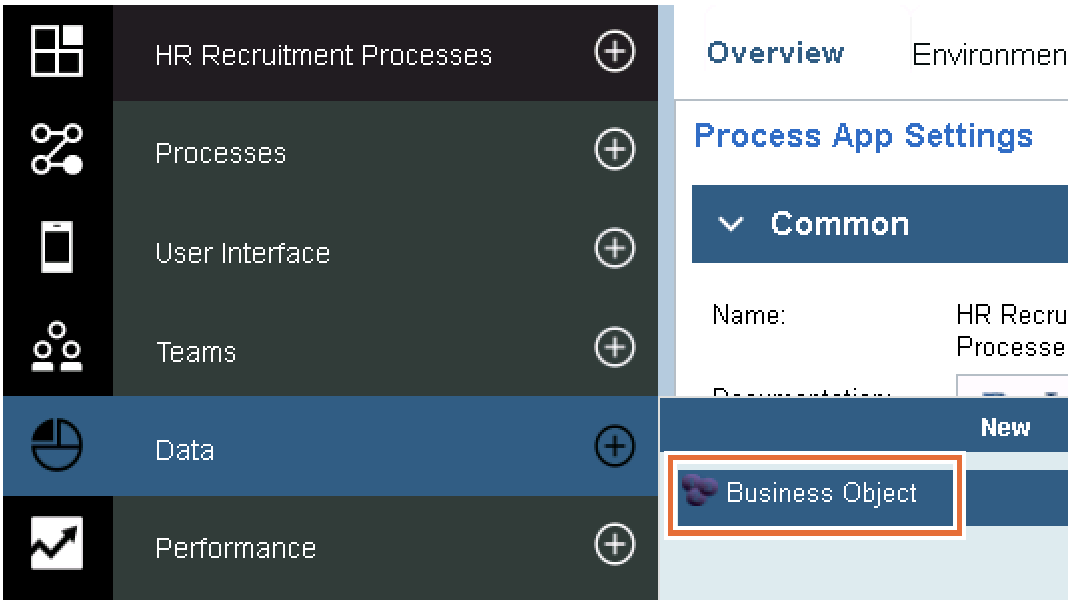
Exercise instructions

Part 1: Build business objects and variables

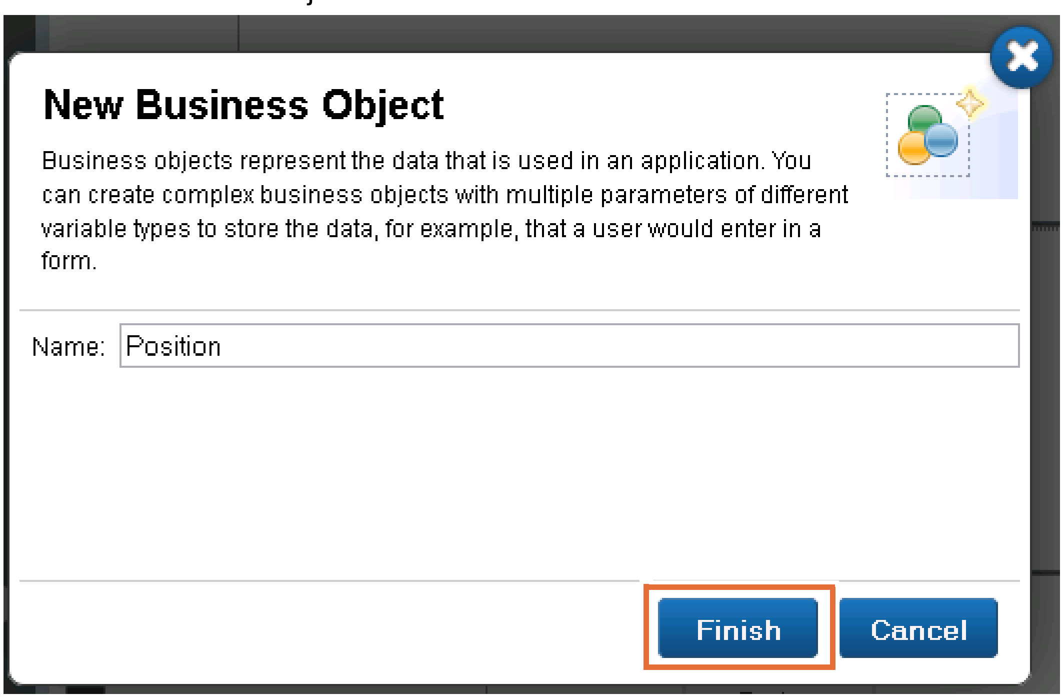
__ 1. Create a complex variable to hold the position data. Use this variable on the coach to allow users to input the position data. Later in the lab, you map these variables to move the data throughout the process.

__ a. In the IBM Web 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**.



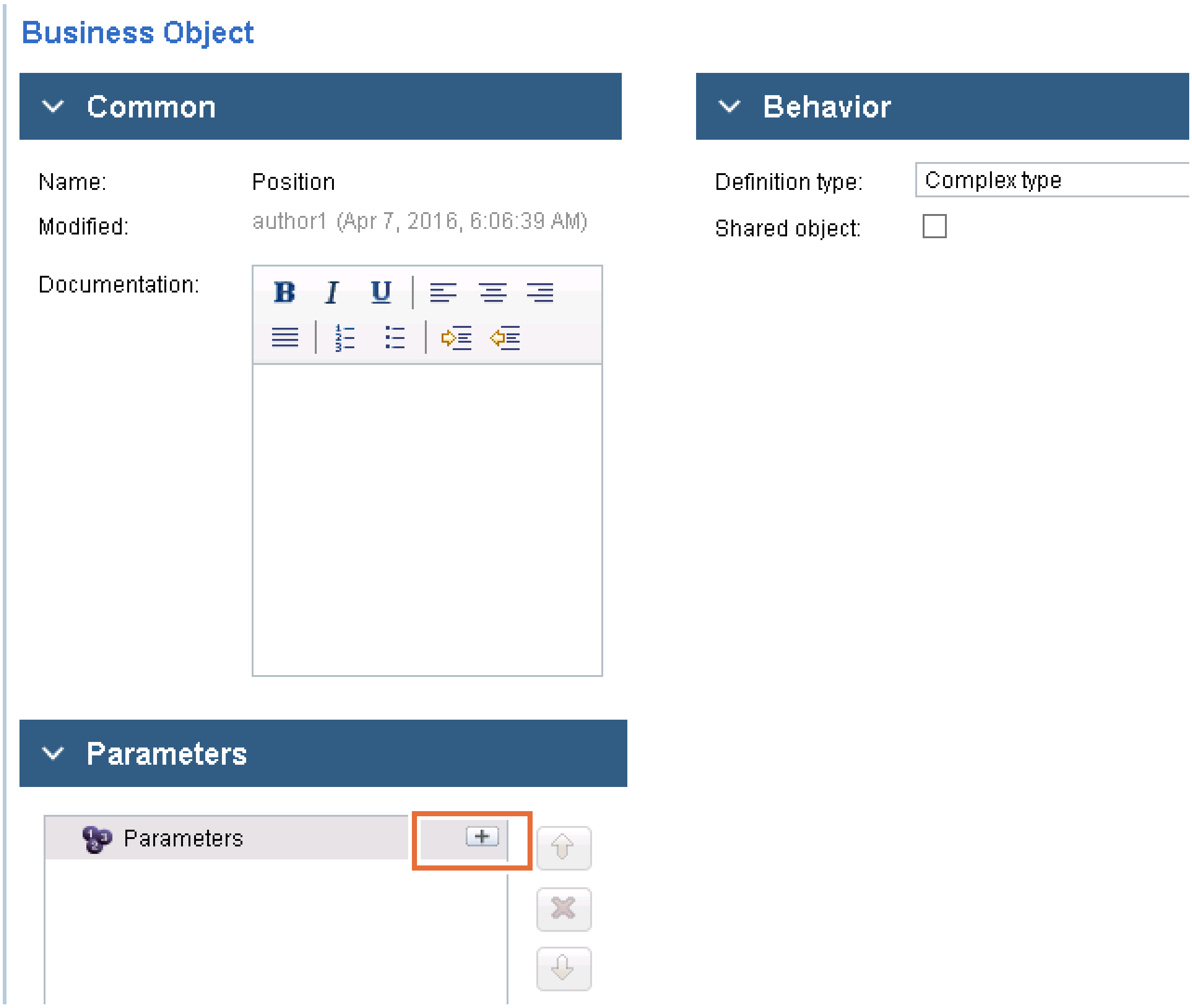
Note

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

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

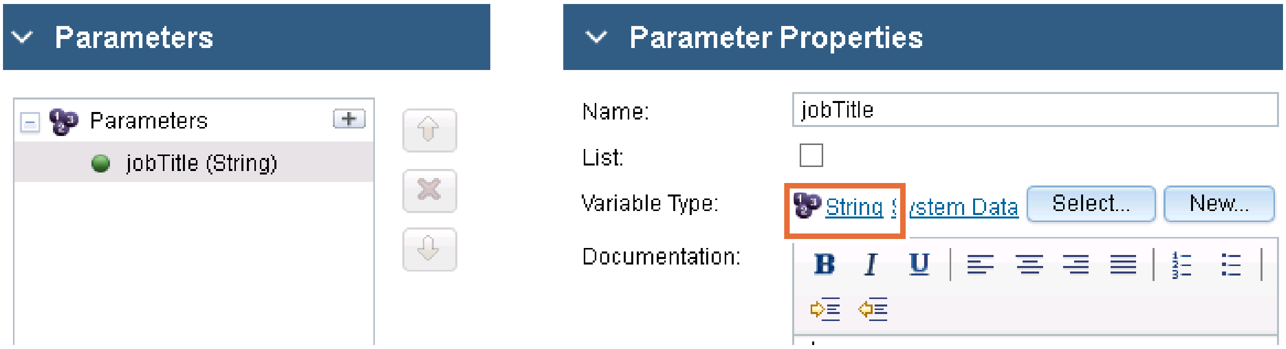
Camel case refers to a word or string of letters that has no space and has an uppercase letter in a position other than the first letter.

__ 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**.

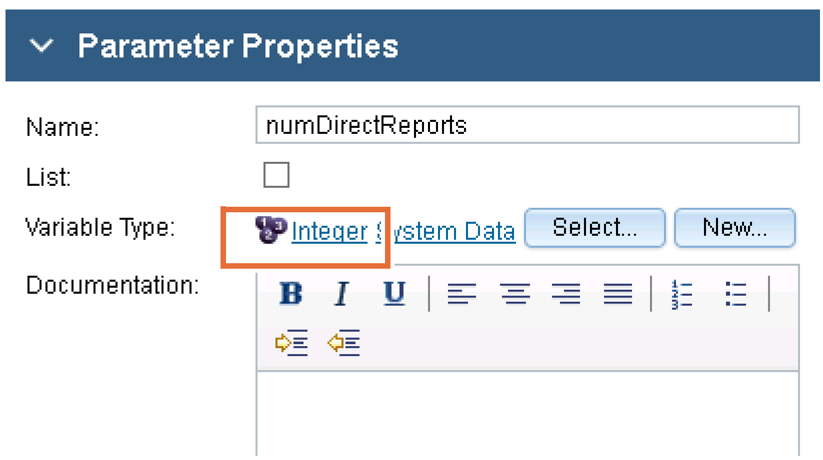


__ g. Repeat steps 1d – 1f to add the following parameters:

- `jobDescription` (String)
- `jobLevel` (String)

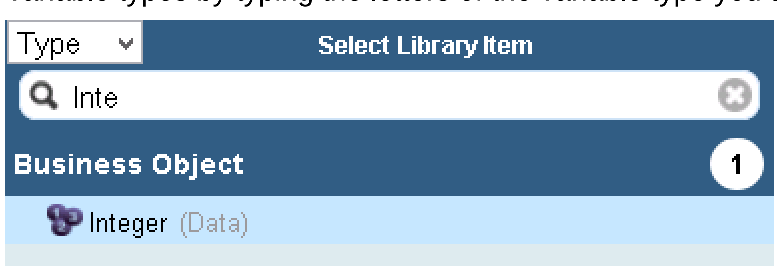
__ h. Add another parameter that is named: `numDirectReports`

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

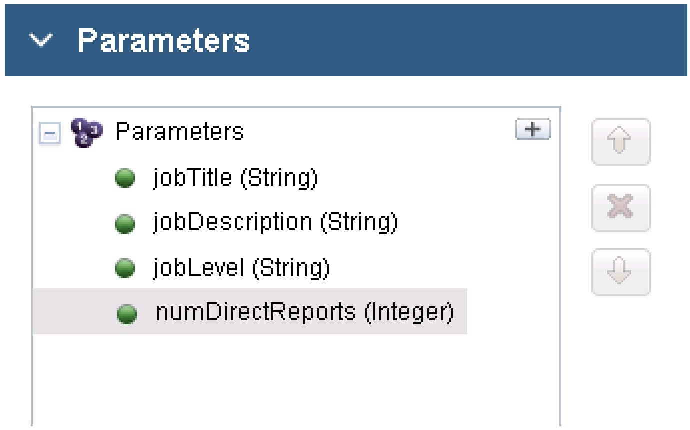


Hint

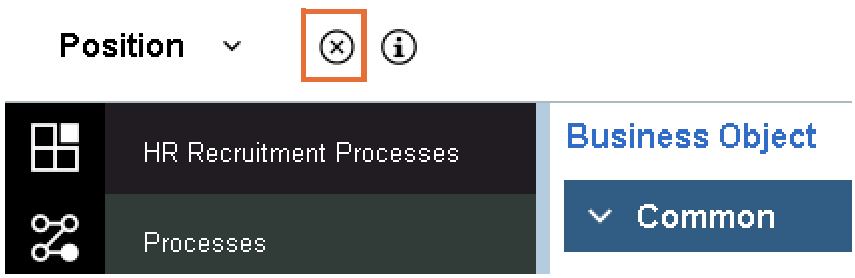
You can filter the variable types by typing the letters of the variable type you are looking for.



__ j. **Save** your process application.



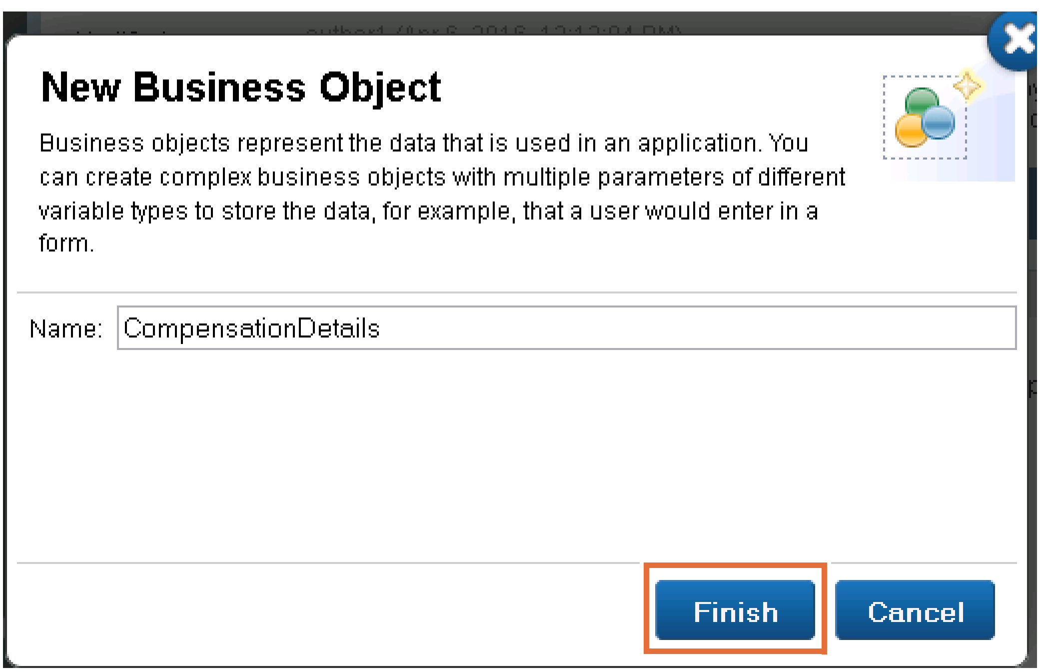
__ k. Close the business object.



__ 2. Create a second business object that is named `CompensationDetails` to allow users to input compensation data on a coach.

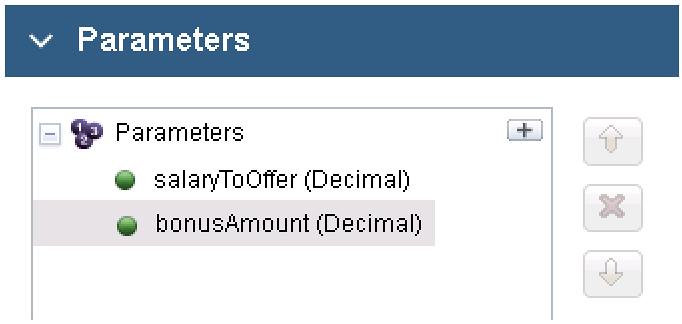
__ a. Click the (**+**) plus sign next to the **Data** category in the library, and select **Business Object**.

__ 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)



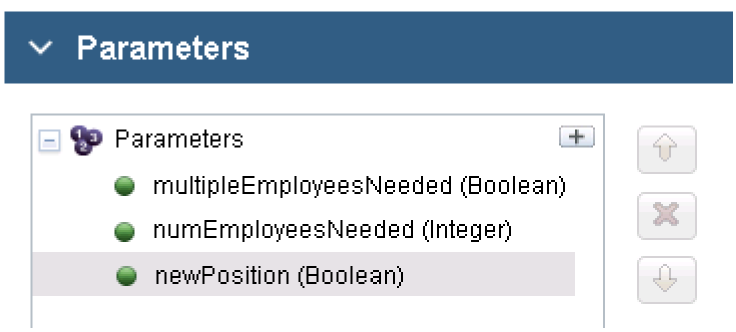
__ d. Save the process app and close the business object.

__ 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)



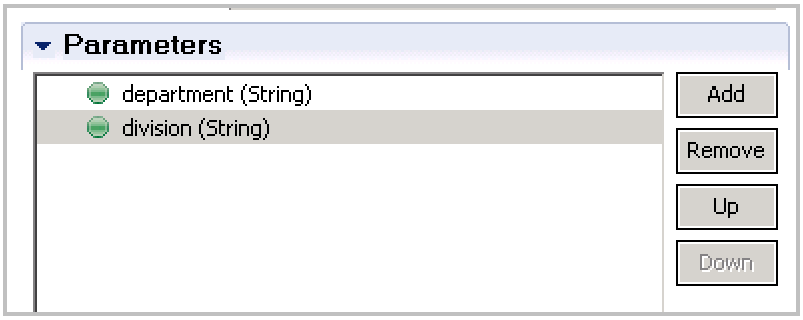
__ c. Save the process app and close the business object.

__ 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)`



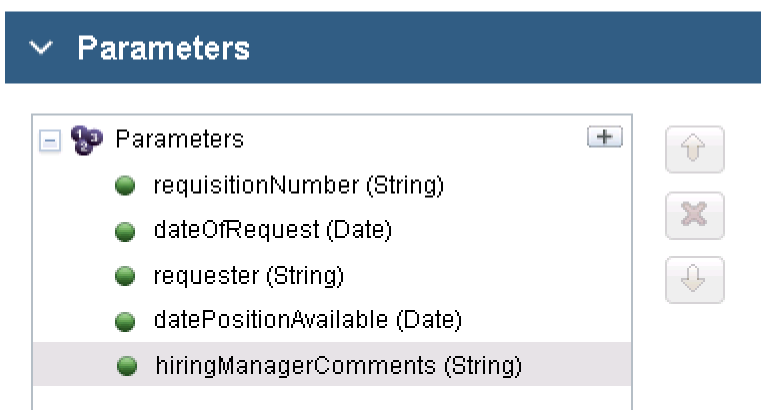
__ c. Save your process application and close the business object.

__ 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 business object to hold all the data for the process.

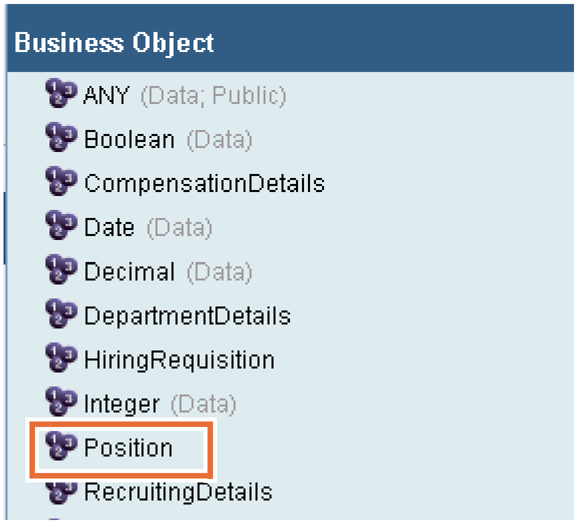
__ a. Create a business object: `HiringRequisition`

__ b. Add the following parameters:

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

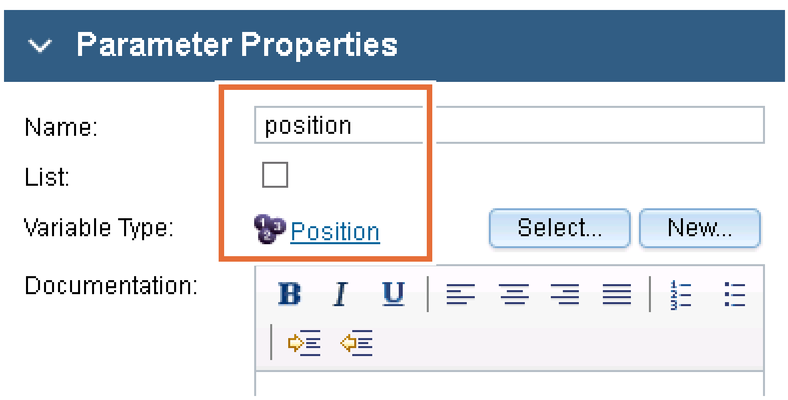


__ c. Add a parameter that is called `position` but this time, click **Select** to the right of **Variable Type** and use the **Position** business object.



Reminder

The variable type is capitalized but the parameter name is lowercase.



__ d. Continue to add the other complex objects:

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

__ e. Save the process app and close the business object.

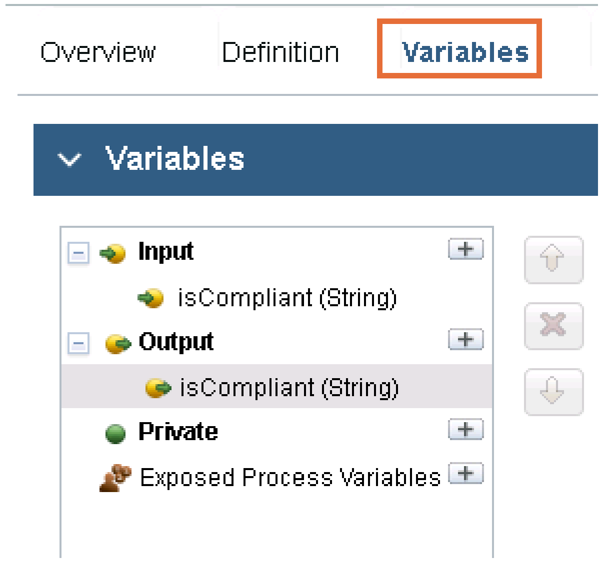
__ 6. Add `requisitionDetails (HiringRequisition)` to the Approve Hire Request linked process as an input and output variable.

Important

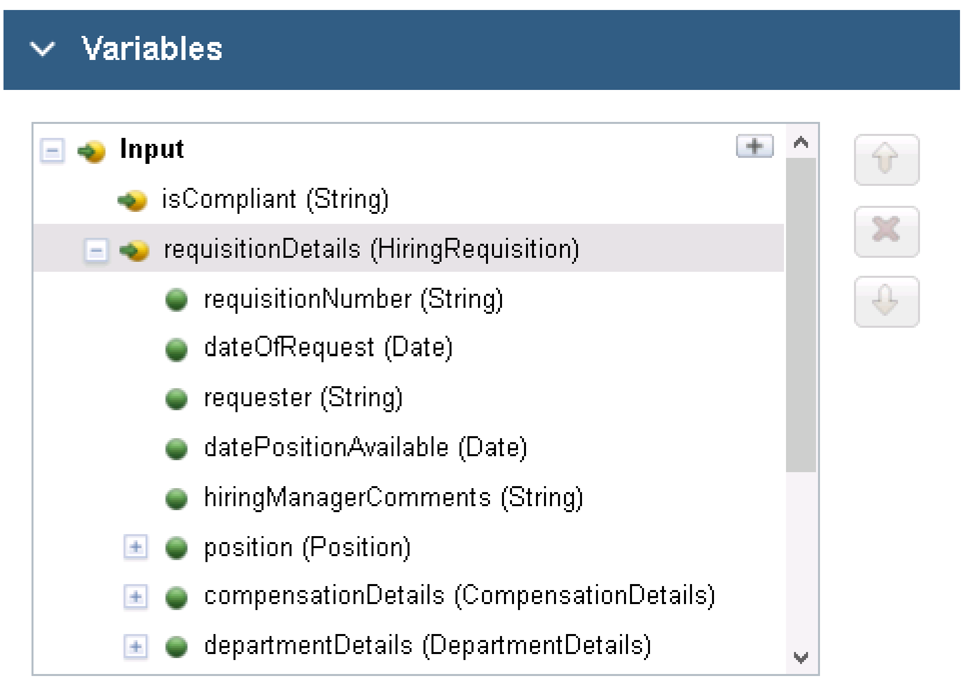
To pass data from the parent process (`Hiring Request Process`) to your nested process (`Approve Hire Request`), you must create and map variables. Because the nested process is a linked process, you must pass the data object to the nested process by way of the input and output variable settings in the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

__ a. Open the **Approve Hire Request** linked process.

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



__ c. Add an Input variable `requisitionDetails (HiringRequisition)` to the process.



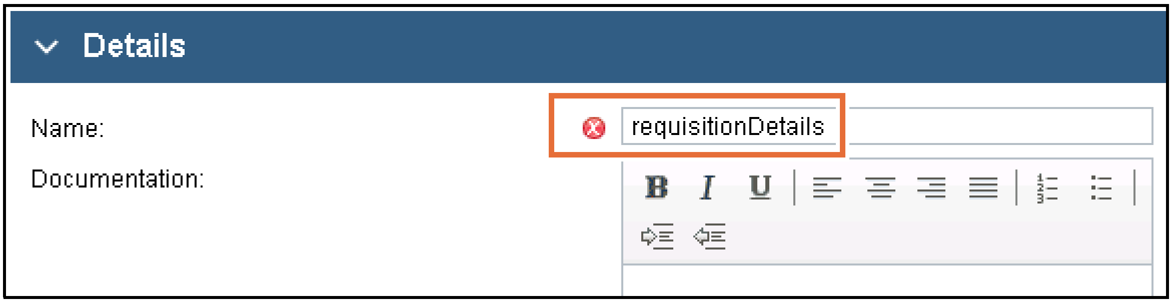
Reminder

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

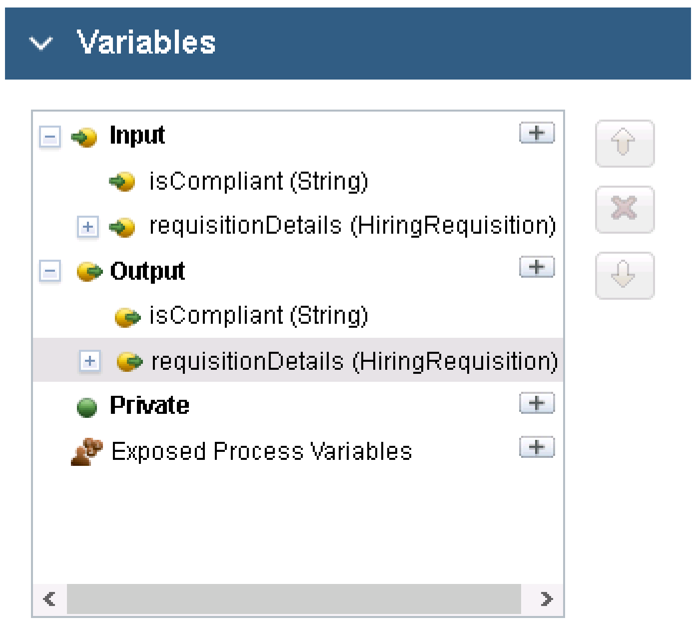
__ 7. Add an output variable `requisitionDetails (HiringRequisition)` to the process.

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`. To clear the error, you set the variable type to match the type of the input variable.



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



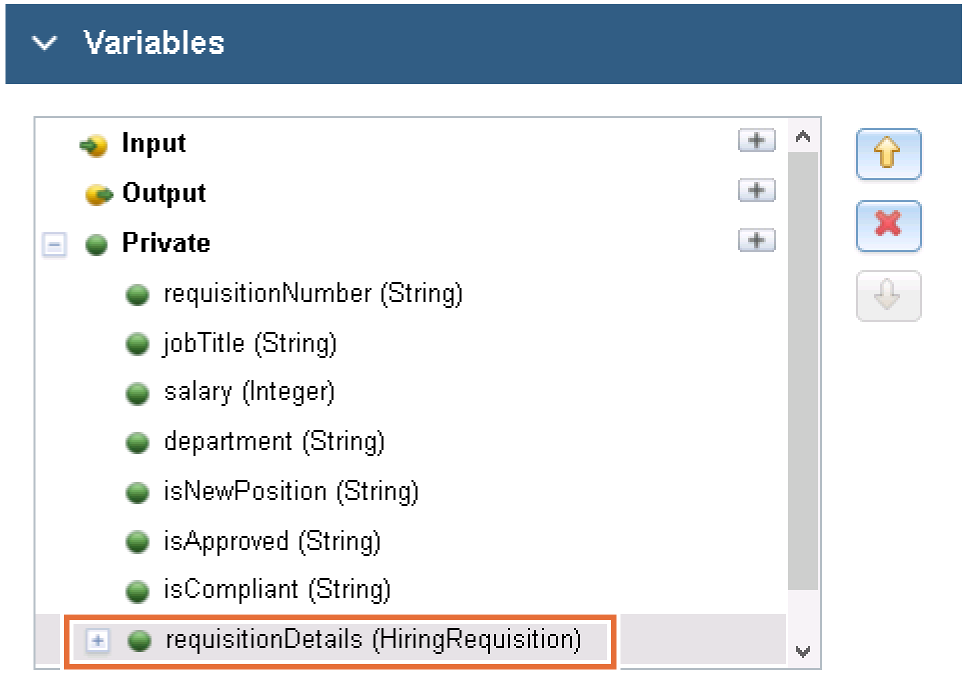
Important

Make sure that the input and output variable names exactly match. If they do not match, you have two different variables in the linked process, and the input data to the process is not sent as output to the parent process.

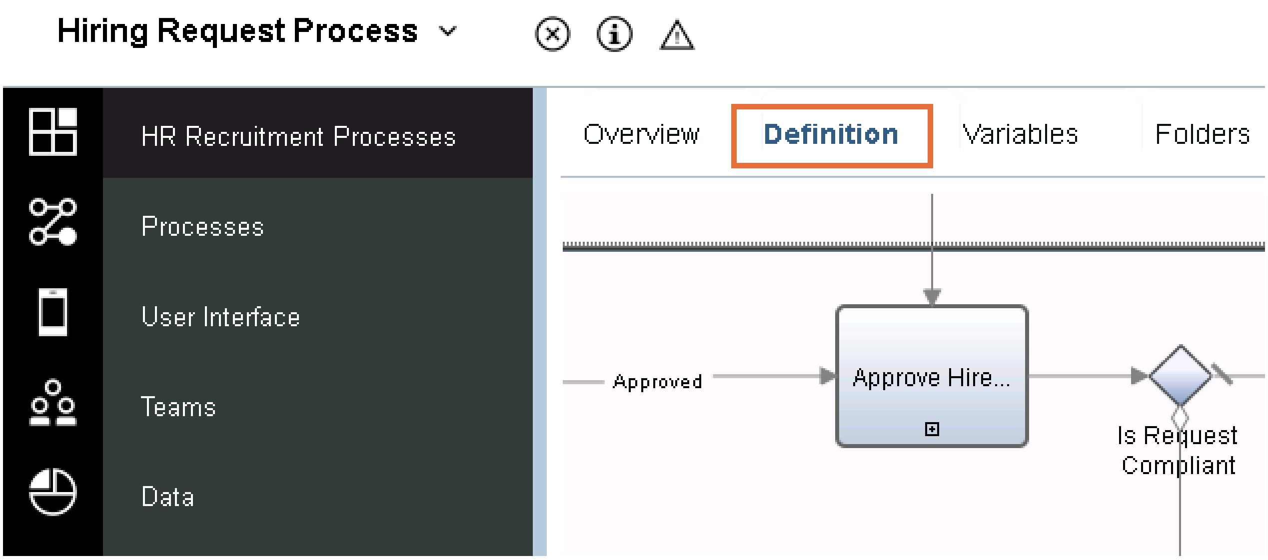
__ 9. Create and map the requisitionDetails variable from the higher-level process to the nested process.

__ a. Open the **Hiring Request Process** and click the **Variables** tab.

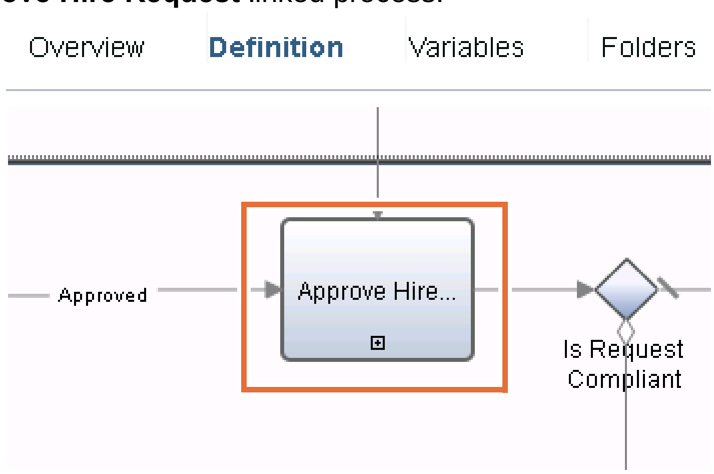
__ b. Create a private variable: `requisitionDetails (HiringRequisition)`



__ c. Click the **Definition** tab.



__ d. Click **Approve Hire Request** linked process.



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

__ 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.

__ g. Save your work.

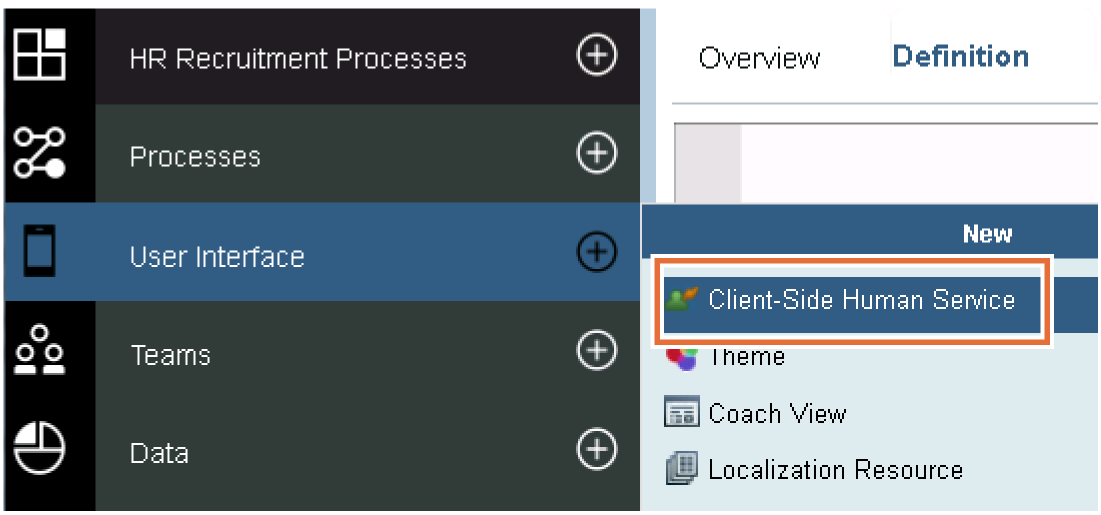
Part 2: Create a reusable client-side human service and define variables

The default human service coach was automatically implemented for each activity when the activity was created 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. Next, 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 data view (read-only).

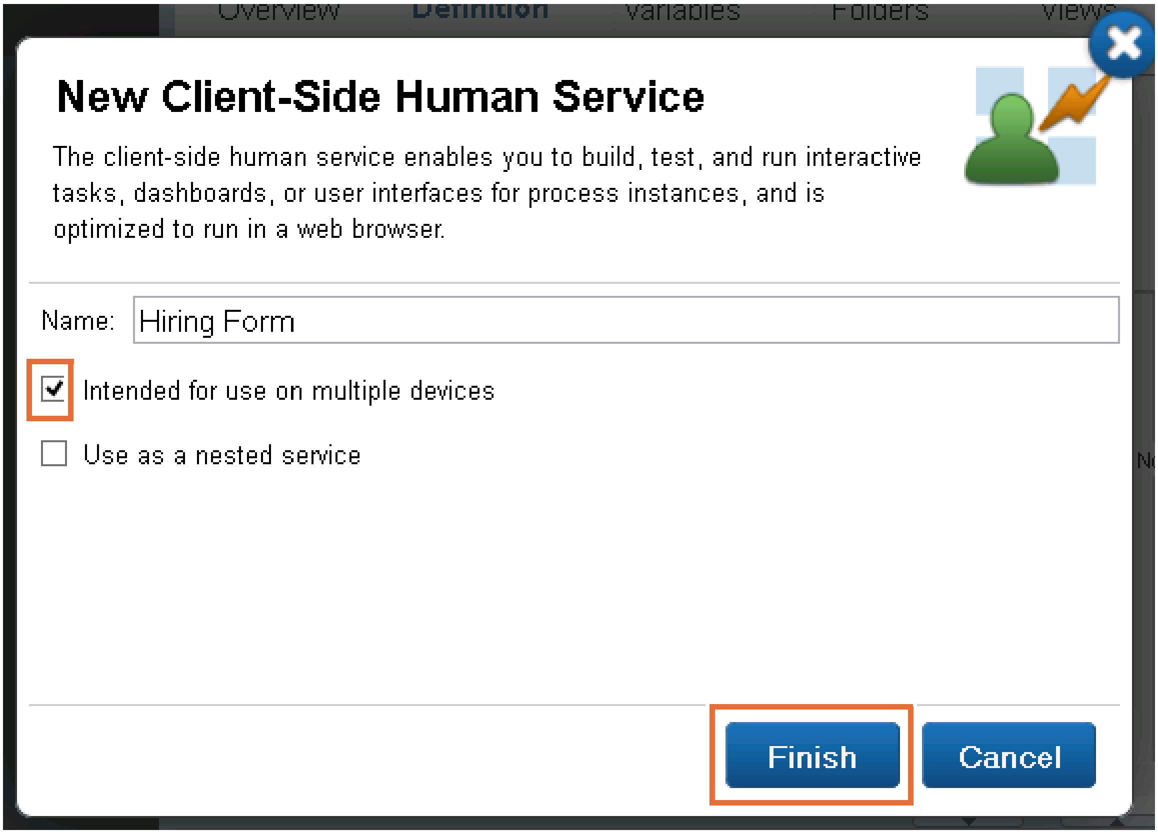
__ 1. Create a client-side human service.

__ a. In the web 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`

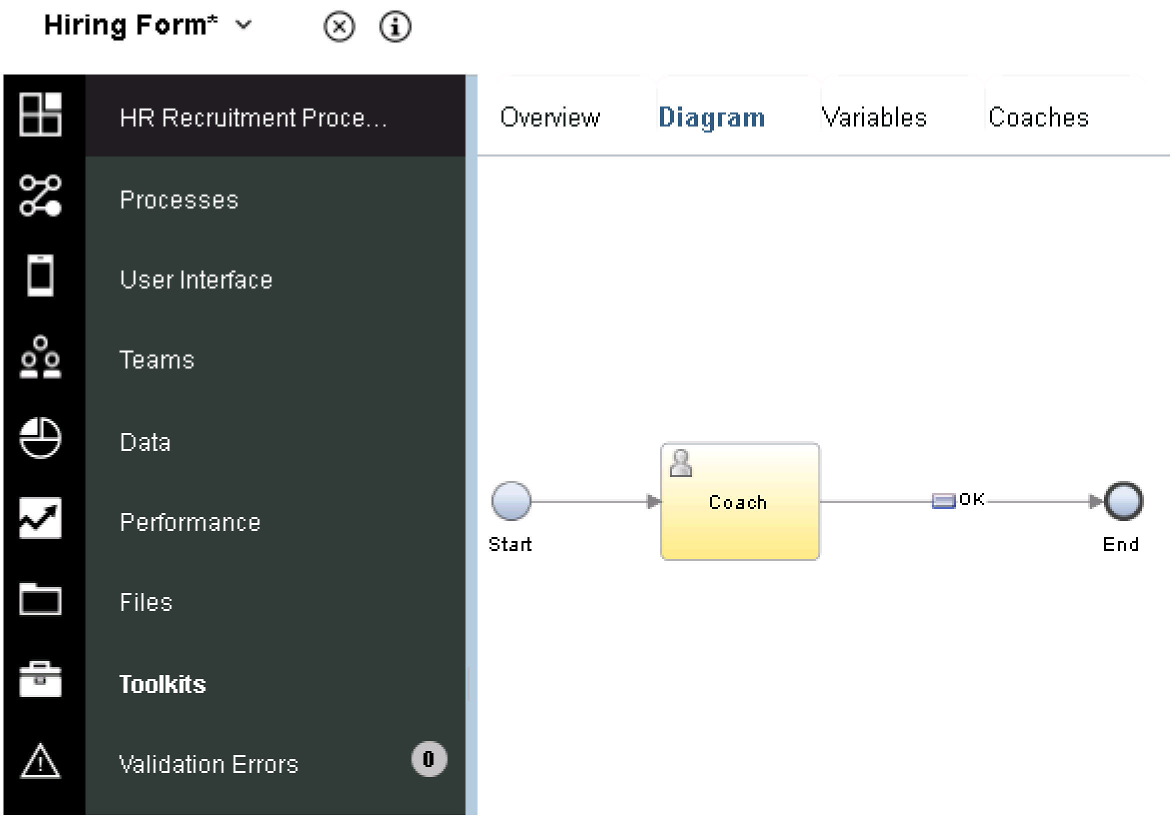
__ c. Enable the **Intended for use on multiple devices** option and click **Finish**.



Note

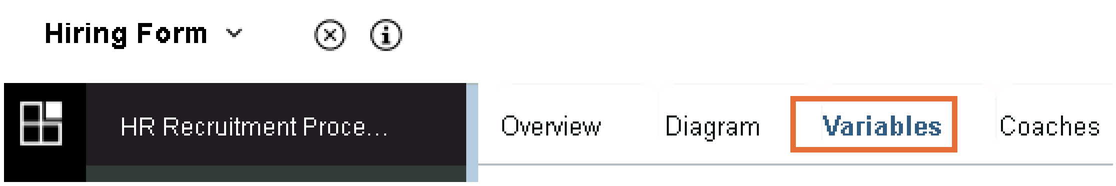
You select **Intended for use on multiple devices** so that the new service can be used on multiple device types. With this option, you can use responsive controls for all the new coaches and coach content that you add to the service.

__ d. The new human service opens in the web Process Designer editor.

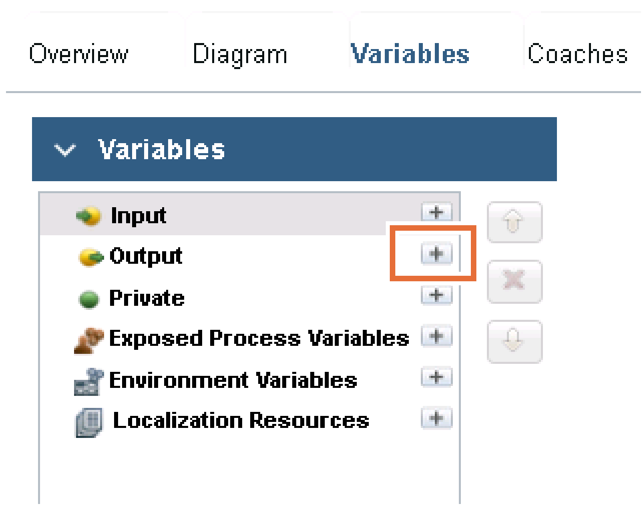


__ 2. Add the `requisitionDetails` output variable to the human service.

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

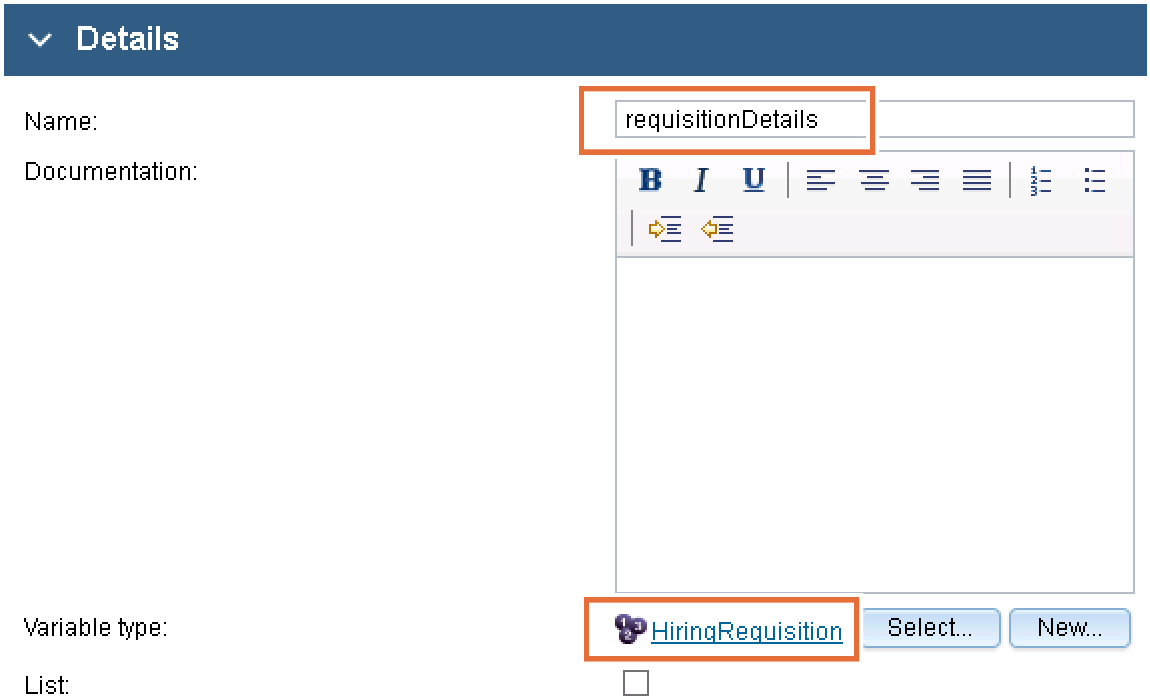


__ b. Click the (**+**) plus sign next to **Output**.



__ c. Name the variable: `requisitionDetails`

__ d. Click **Select** to the right of the **Variable type** option to select **HiringRequisition**.

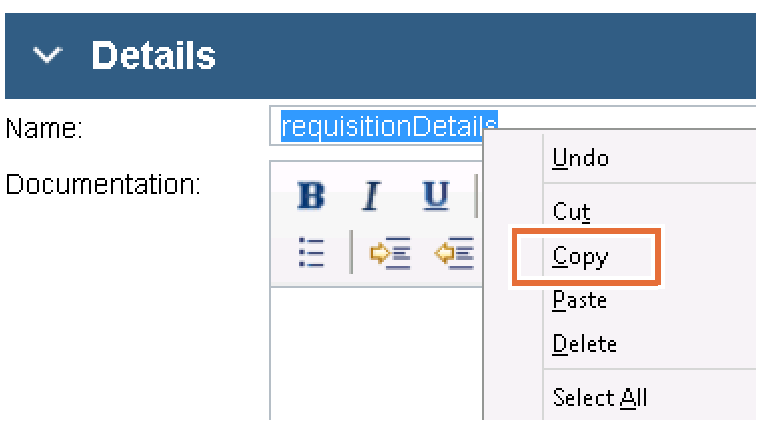


Information

Because you need all of the subobjects of the parent **HiringRequisiton** 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.

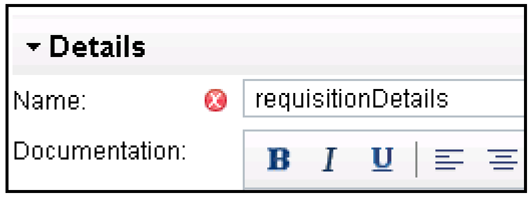
__ a. Click the output variable **`requisitionDetails`**, highlight the **Name**, and copy it.



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

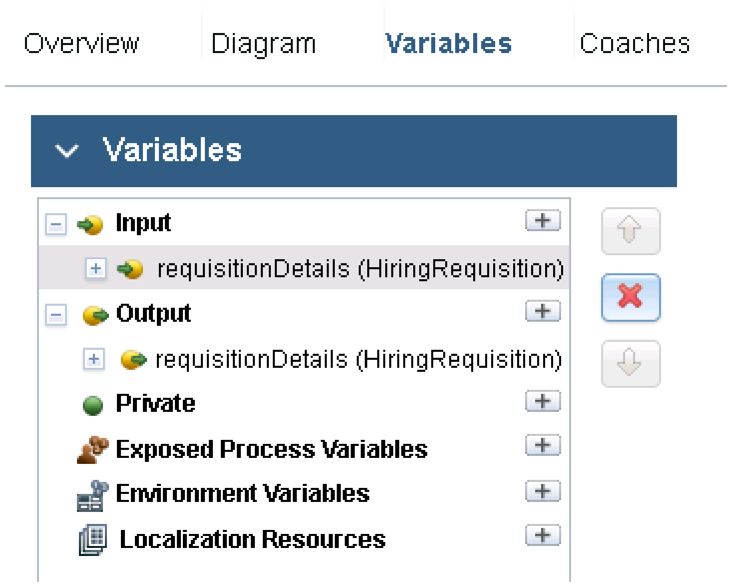
Information

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**.

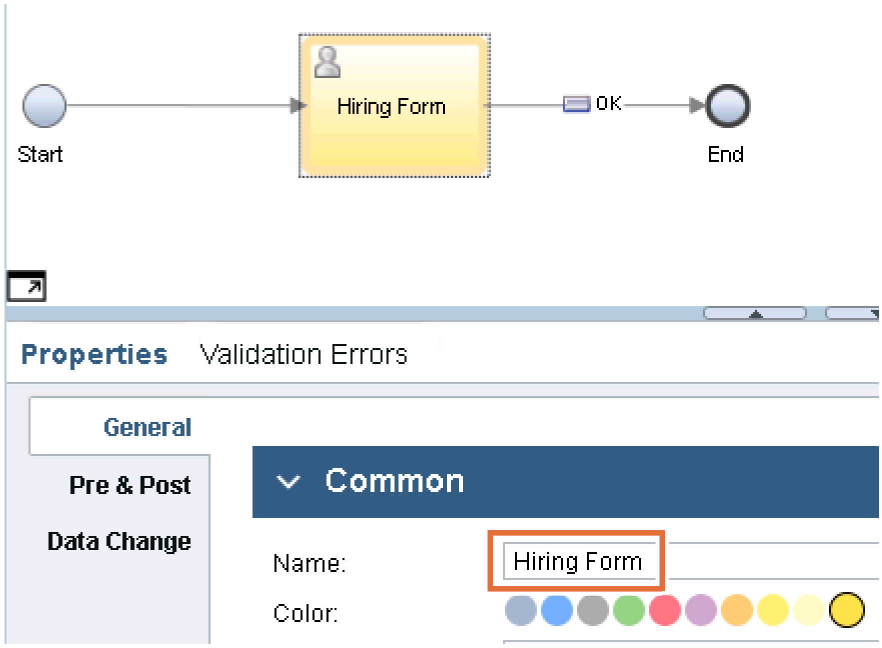


__ e. Save your work.

__ 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.

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

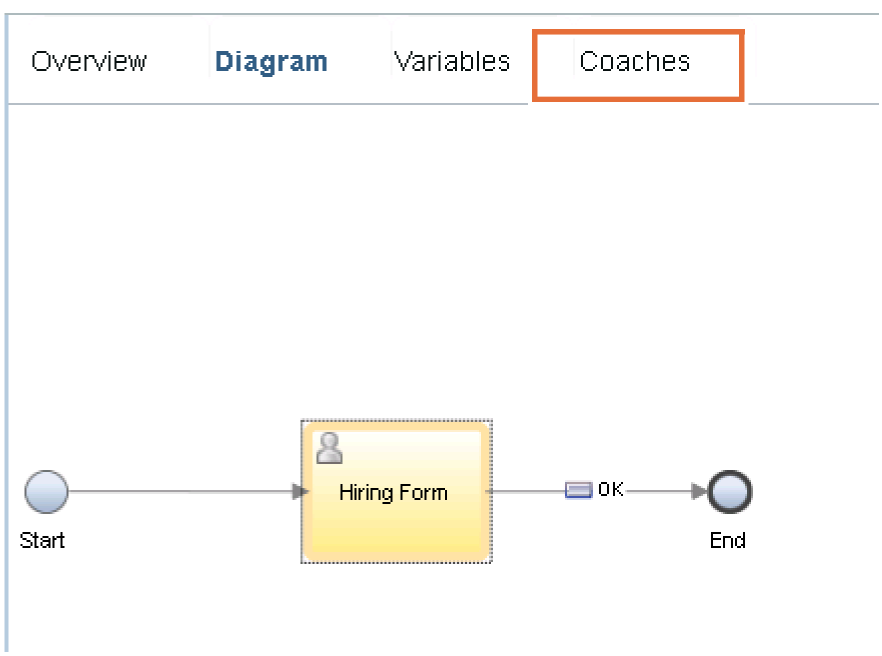


Reminder

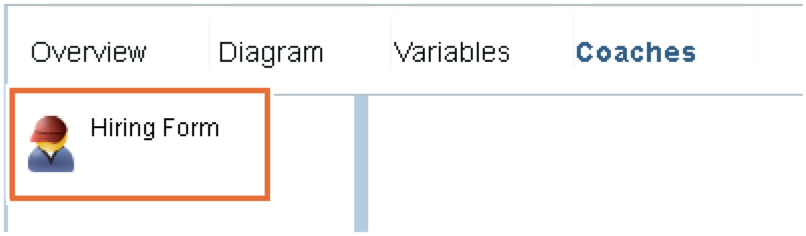
You can also double-click the coach to rename it.

__ 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.



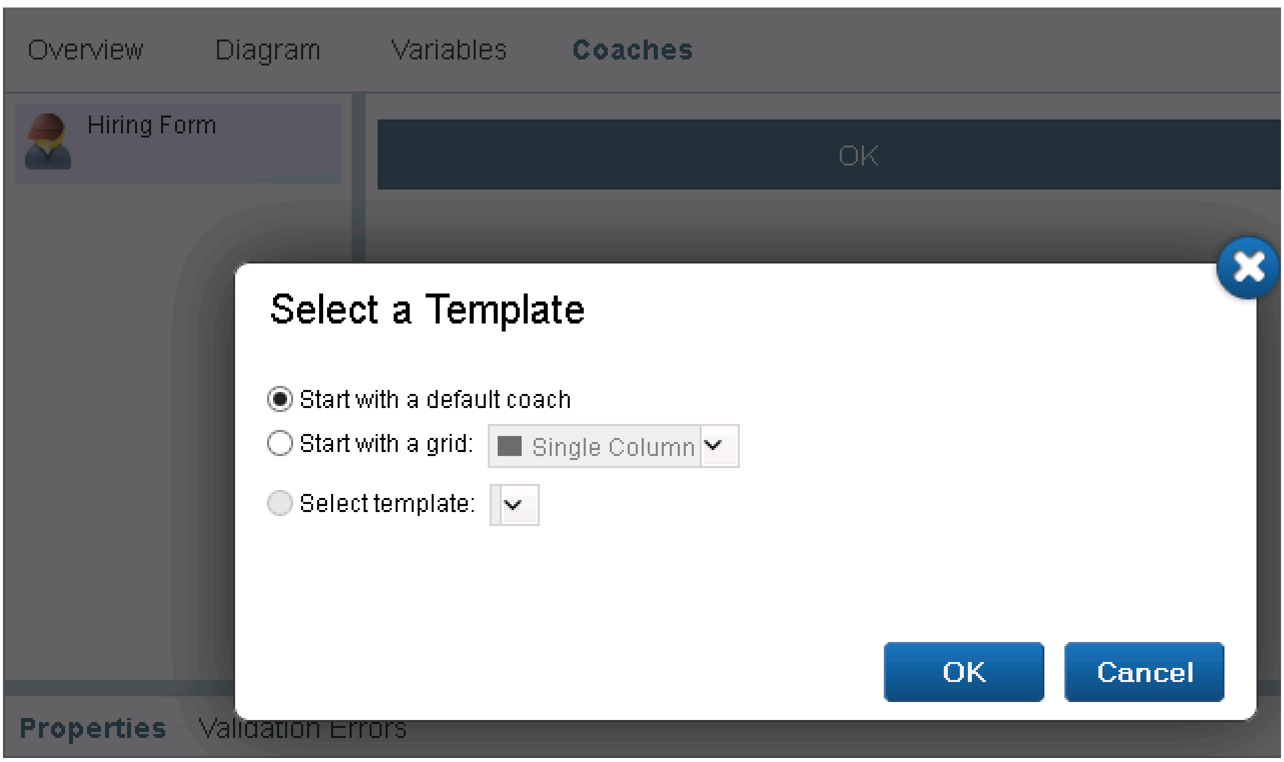
__ b. Click the **Hiring Form** coach.



Hint

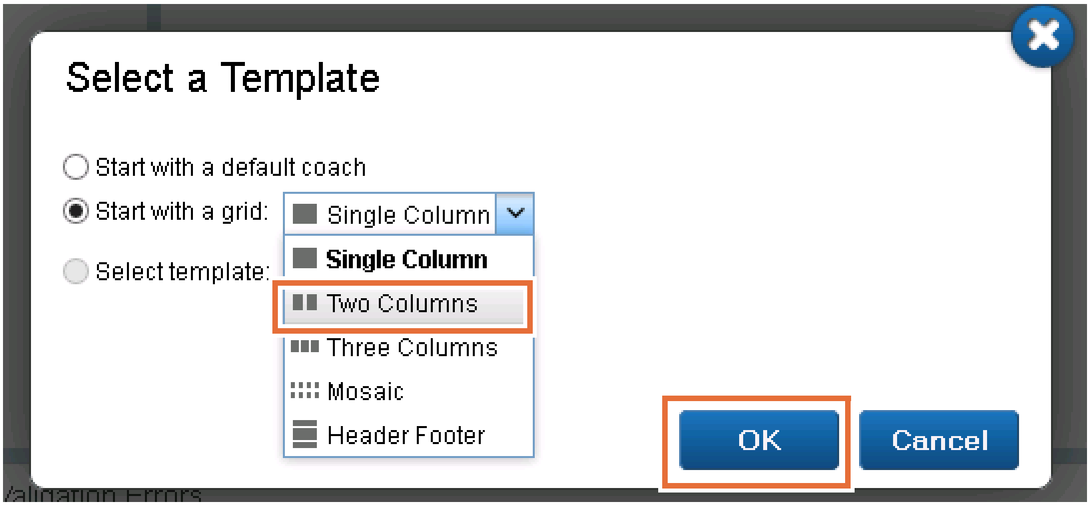
Consider how items are laid out. This consideration determines whether the coach is laid out vertically or horizontally. This coach lays out items vertically except for the position field.

__ c. A message box for selecting a template appears.

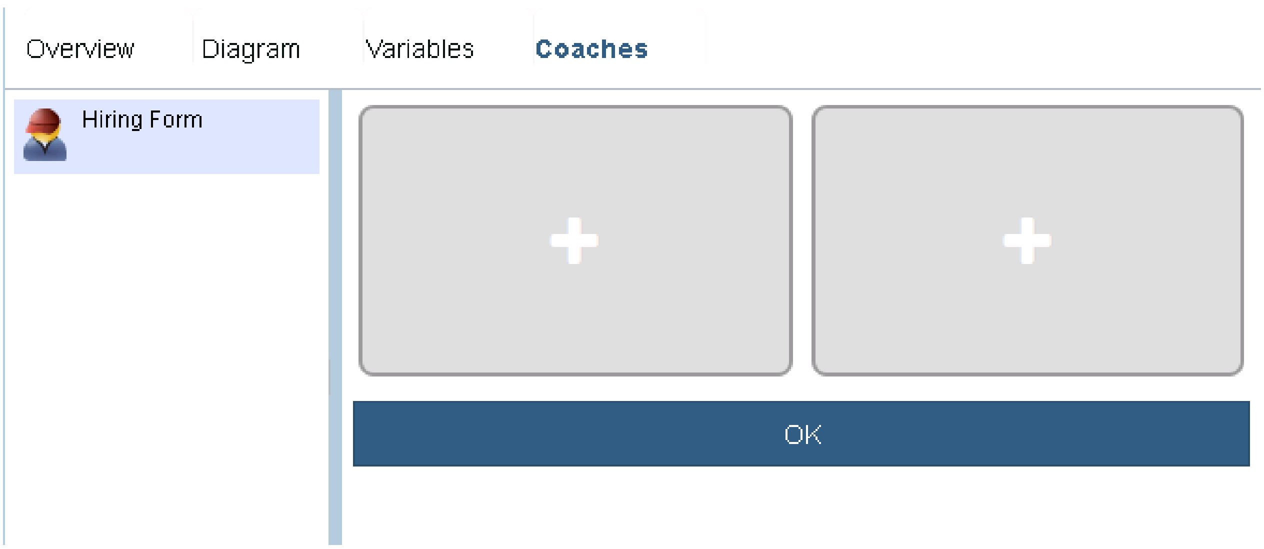


__ d. Select the second option: **Start with a grid**.

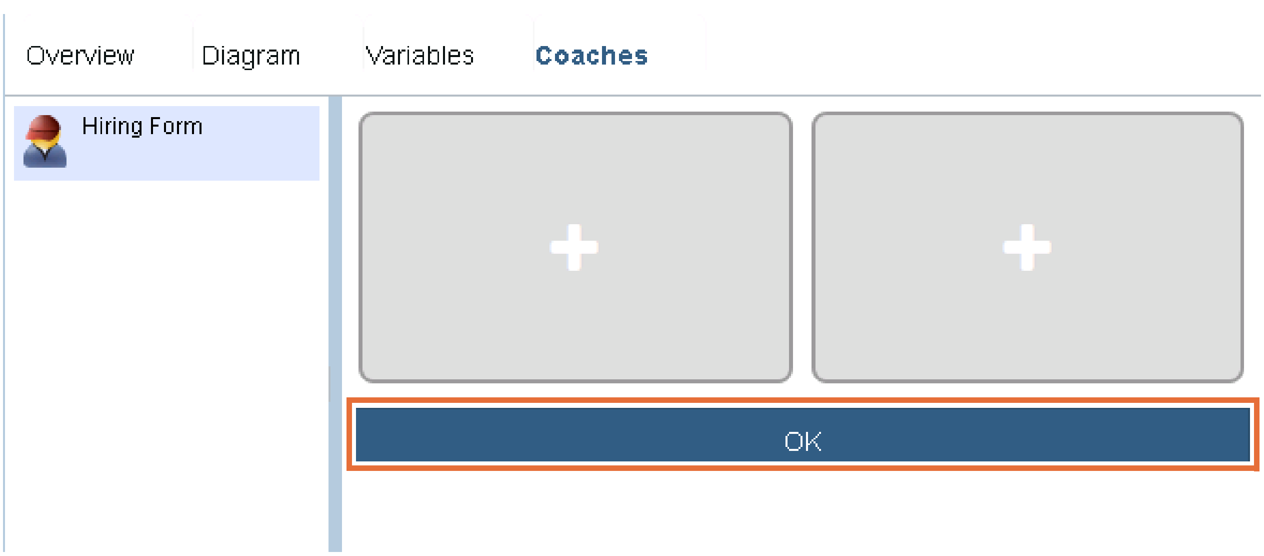
__ e. From the list, select the option **Two Columns** and click **OK**.



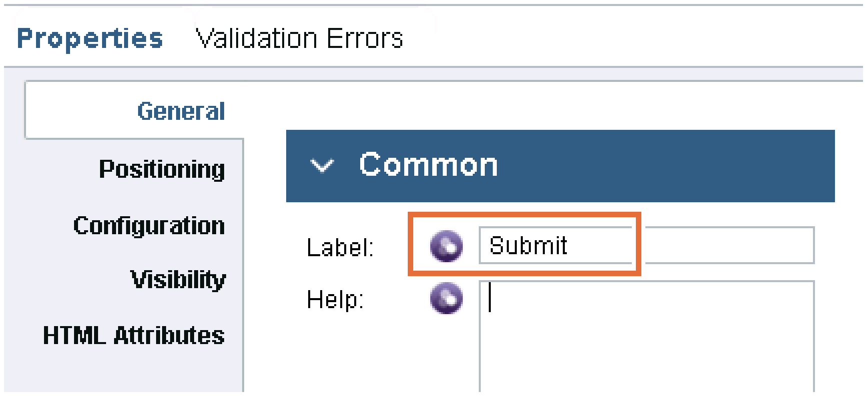
__ f. A two-column grid appears for the **Hiring Form** coach layout.



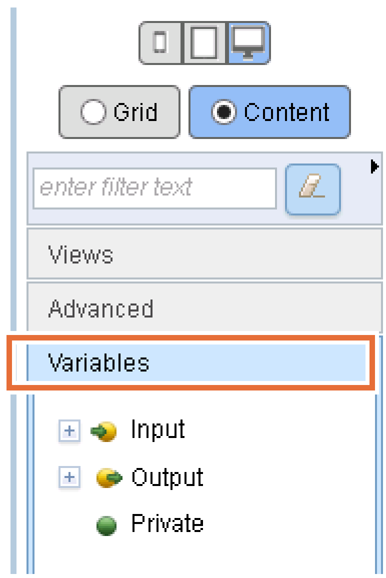
__ g. Click **OK** on the coach canvas.



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



__ i. To the right of the coach canvas in the palette, expand the **Variables** section to view the service variables.

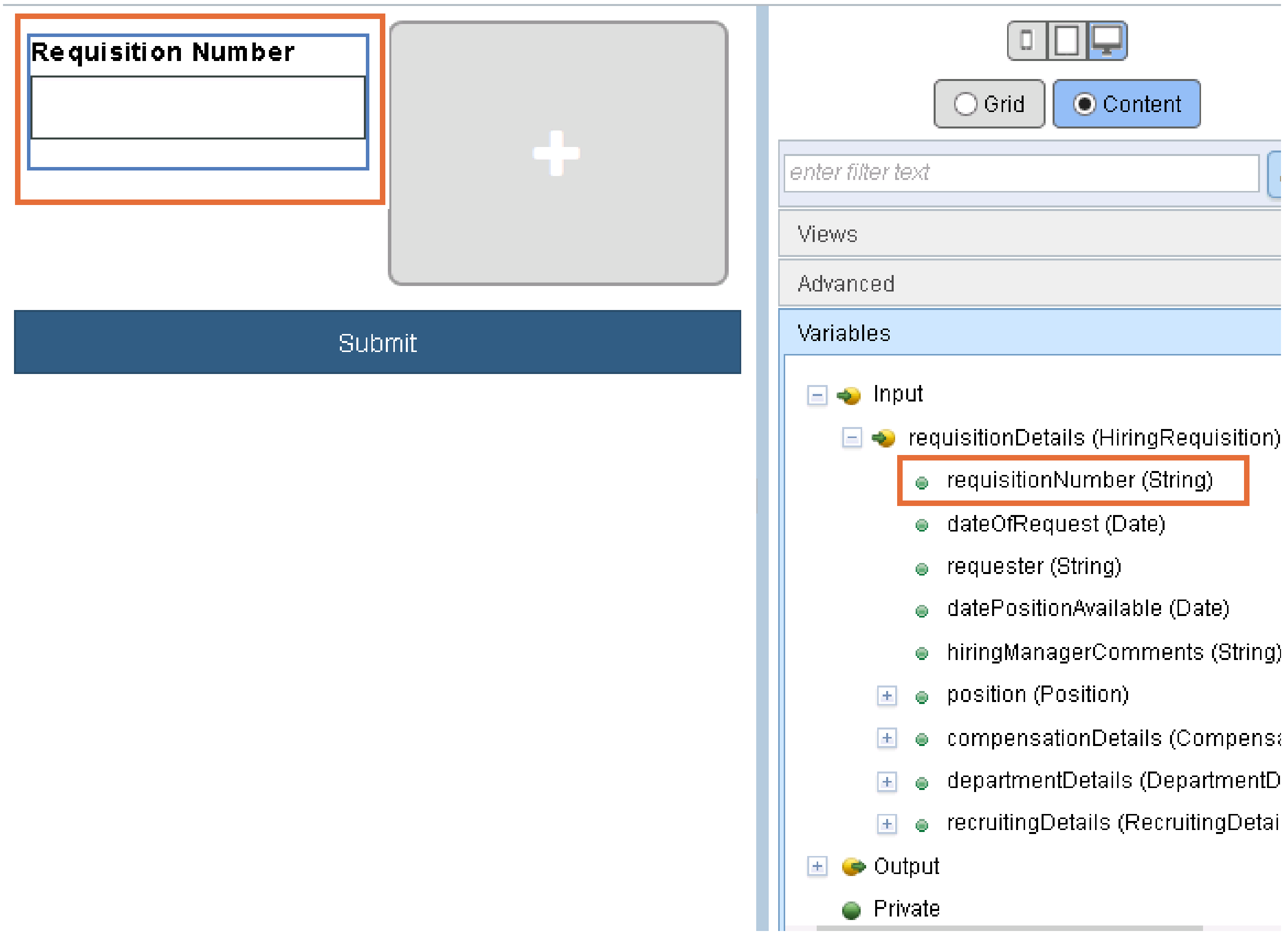


__ j. The first inputs that you add are the components of the `requisitionDetails` variable. From the palette, click the (**+**) plus sign next to **Input** to view the service input variables.

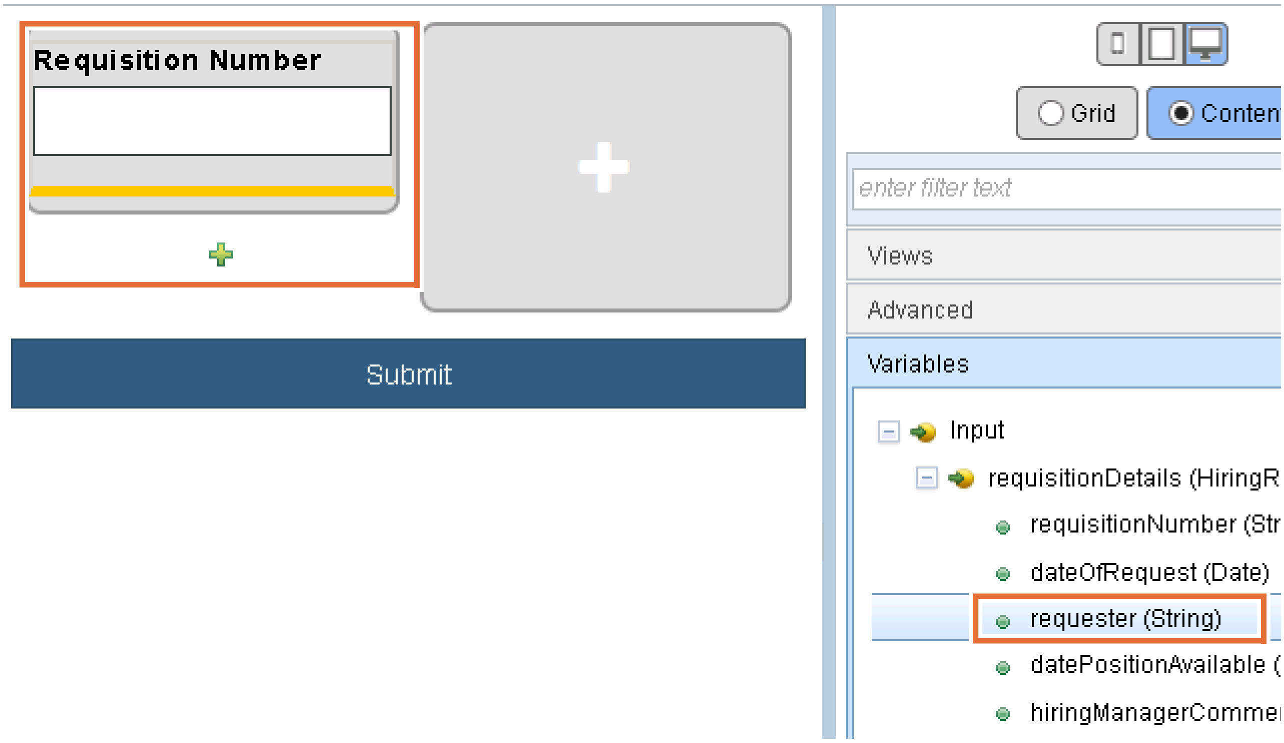
Hint

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

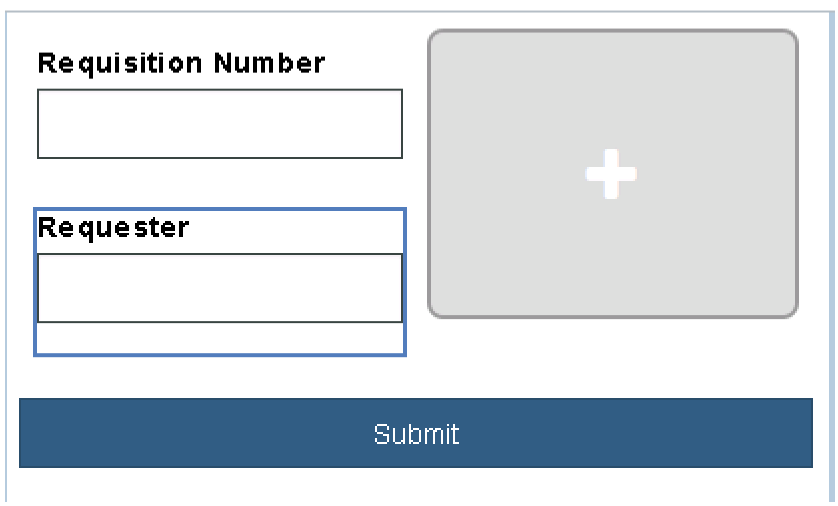
__ k. Drag the **requisitionDetails > requisitionNumber** variable to the left column in the grid above **Submit**.



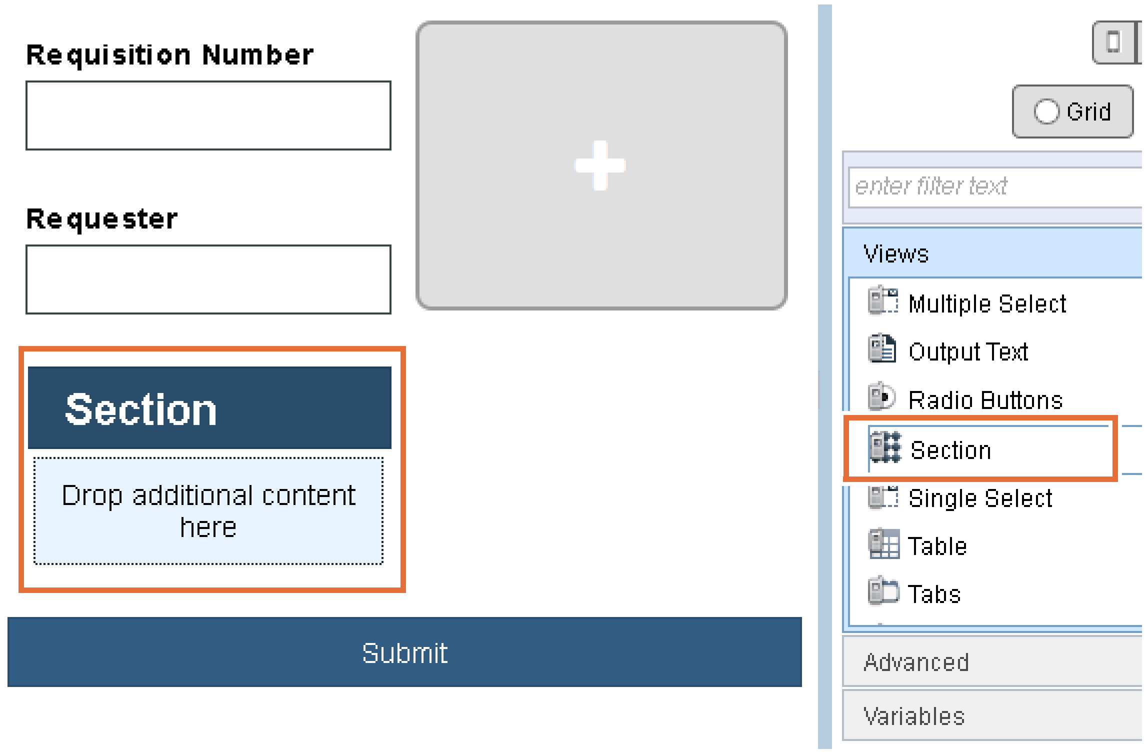
__ l. Drag the **requisitionDetails > requester** variable from the palette to the space underneath **Requisition Number** in the left column. Wait until you see the orange line below the Requisition Number variable in the same grid.



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

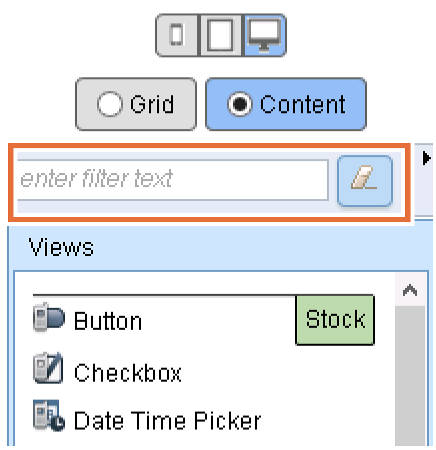


__ n. Expand the **Views** palette tray. Drag a **Section** control in the same grid onto the canvas beneath the **Requester** field.

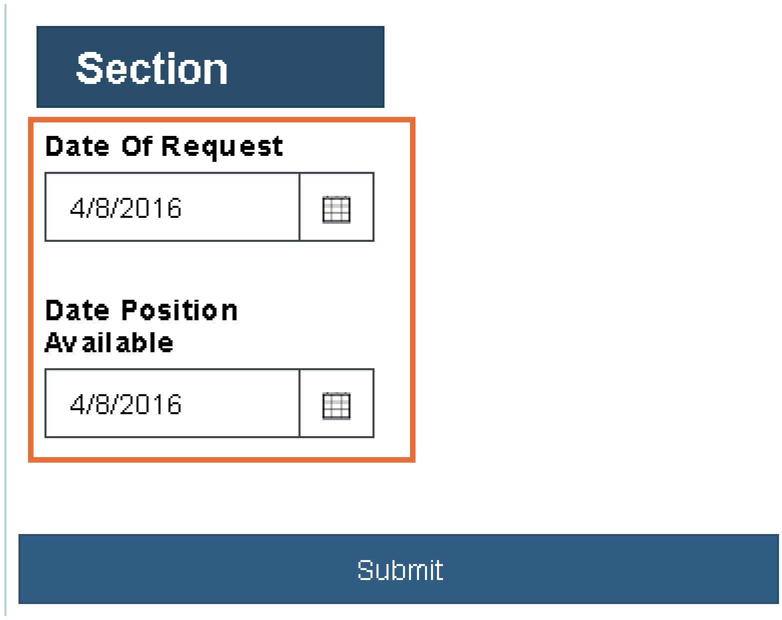


Hint

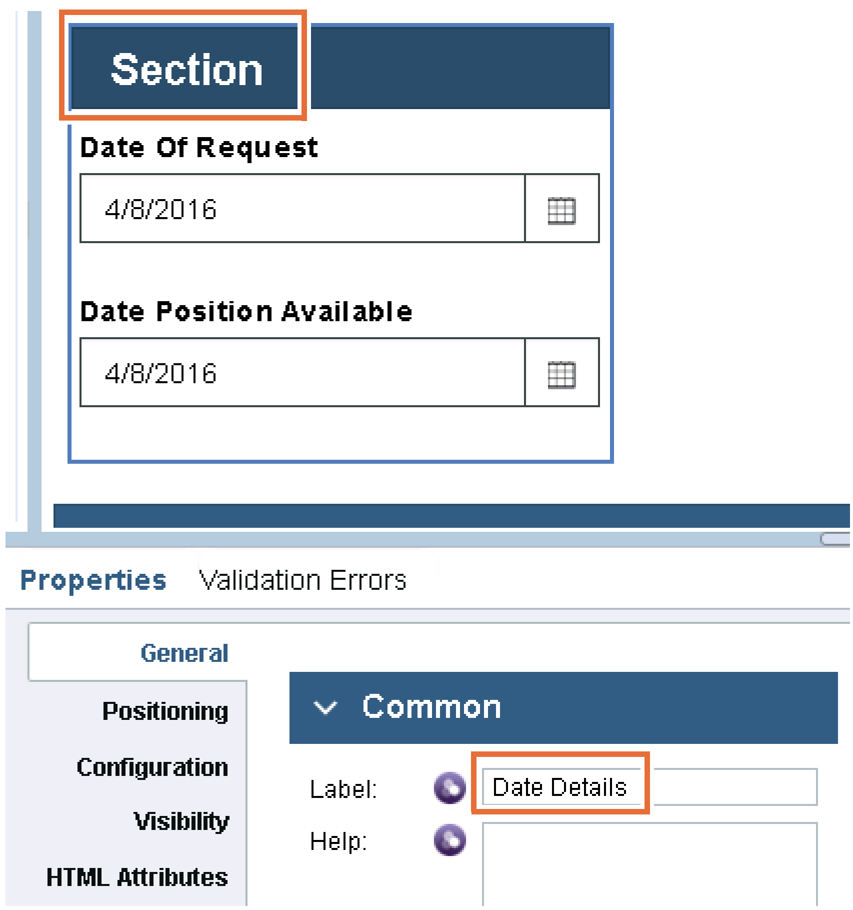
Similar to the filtering capabilities in the web Process Designer, use the input box above the Views palette tray to filter the coach view controls.



__ o. Expand the **Variables** palette tray and drag the date fields (**requisitionDetails > dateOfRequest** and **requisitionDetails > datePositionAvailable**) into the section that is labeled **Drop additional content here**.

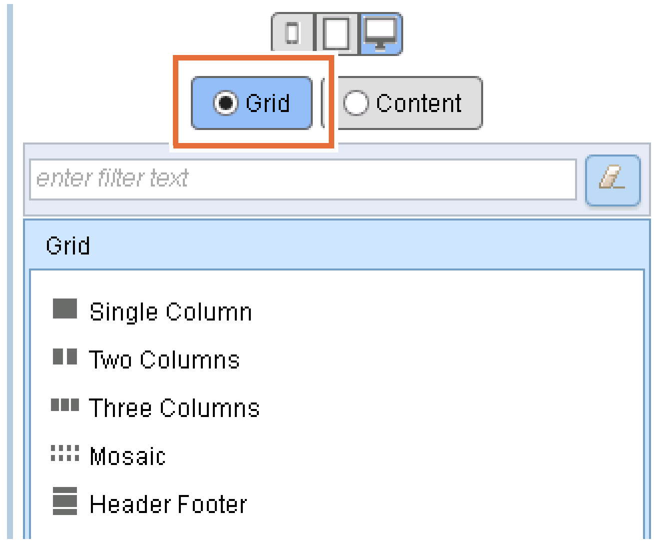


__ p. Select the **Section**. From the **Properties > General > Common** section, change the label to: `Date Details`

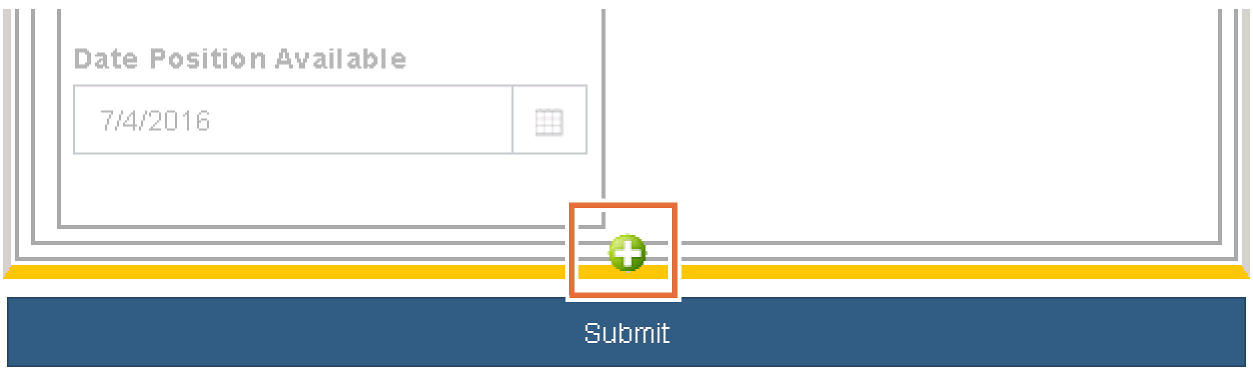


__ 6. Add a grid to add the **requisitionDetails > hiringManagerComments** variable above the **Submit** button.

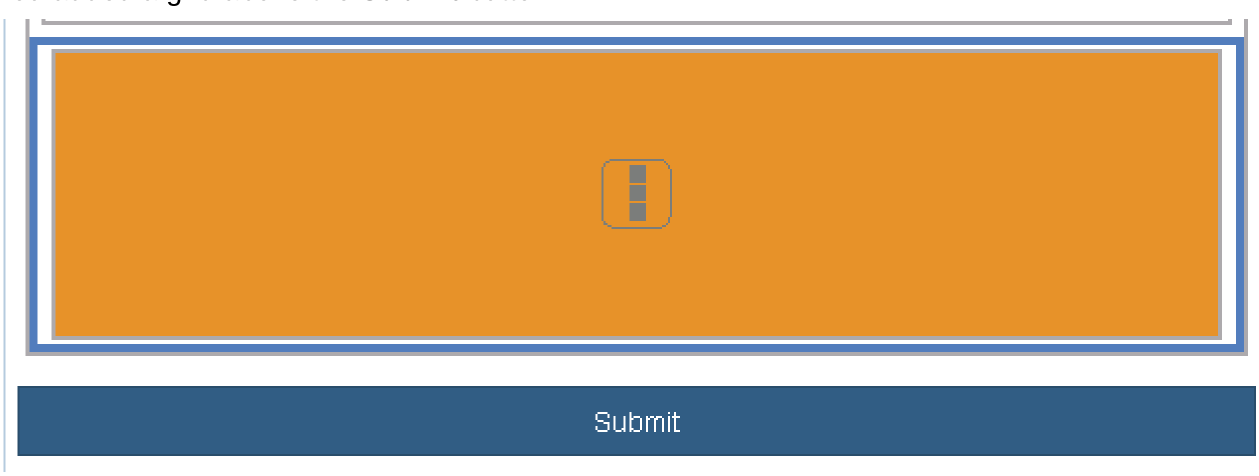
__ a. Enable the **Grid** radio button on the right side above the palette tray.



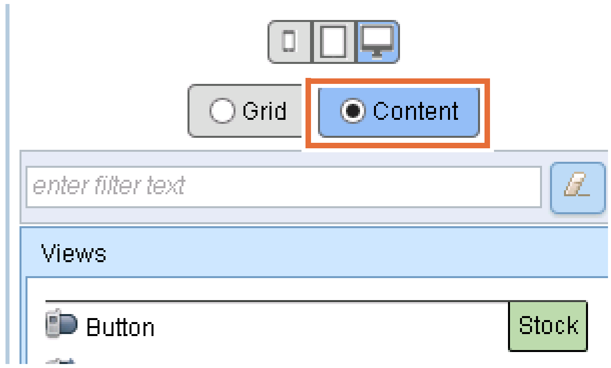
__ b. Click the (**+**) plus icon on the bottom of the grid to add a grid above the Hiring Manager Comments input control.



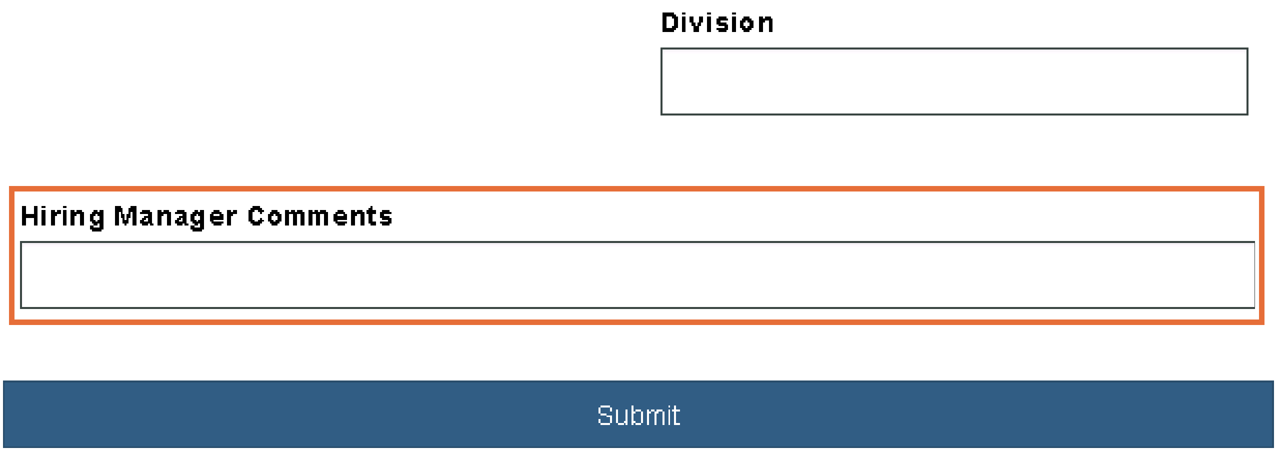
You added a grid above the **Submit** button.



__ c. Click **Content** to enable it.



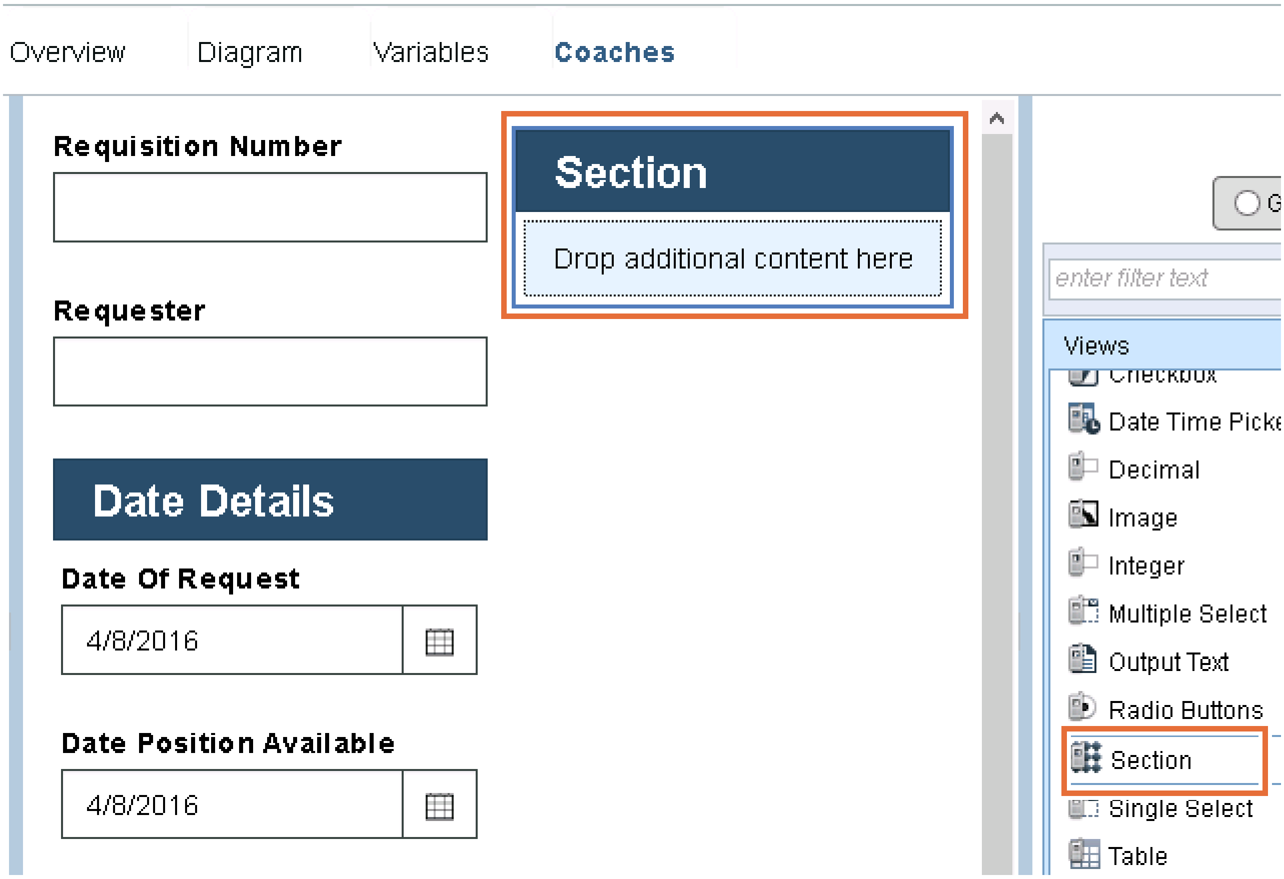
__ d. Drag the **requisitionDetails > hiringManagerComments** variable in the new grid.



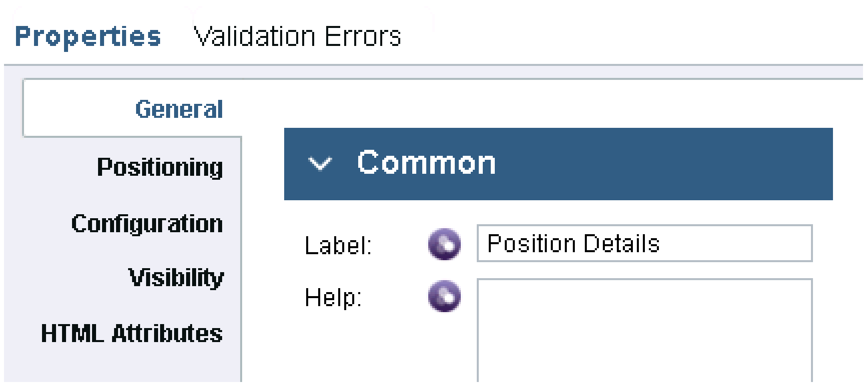
__ e. Save your work.

__ 7. Create a section and populate it with the Position Details business objects.

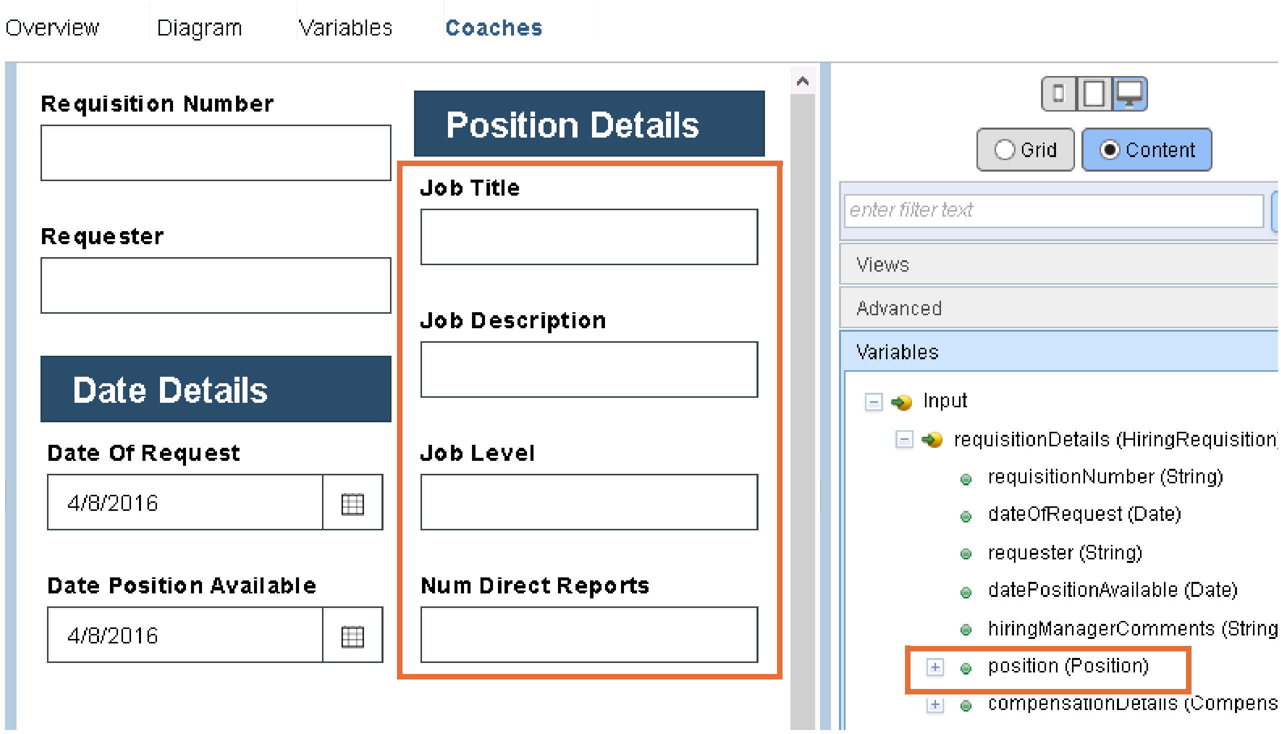
__ a. Expand the **Views** palette tray. Drag a **Section** control onto the canvas in the right column grid beside the Requisition Number and Requester fields.



__ b. In the **Properties > General > Common** section, label the section: `Position Details`

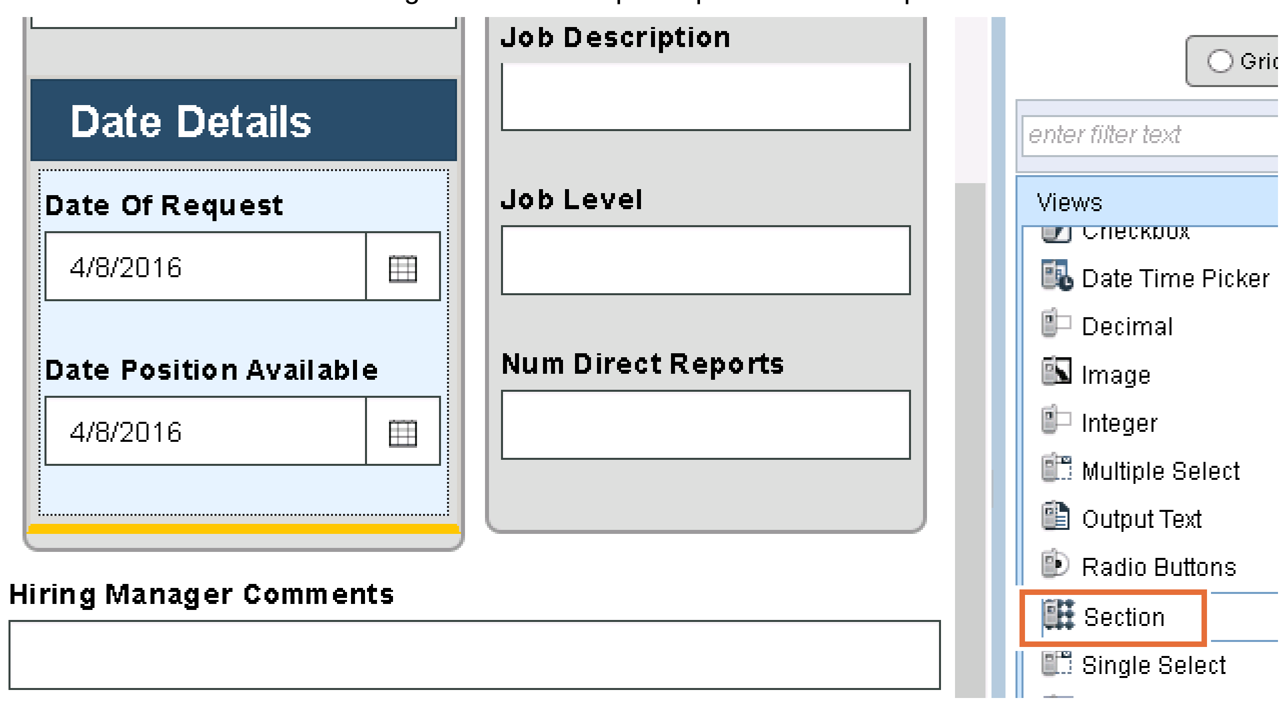


__ c. From the **Variables** palette tray, drag the **requisitionDetails > position** complex variable into the new vertical section that is labeled `Drop additional content here`. The section populates with the subvariables of the Position variable.

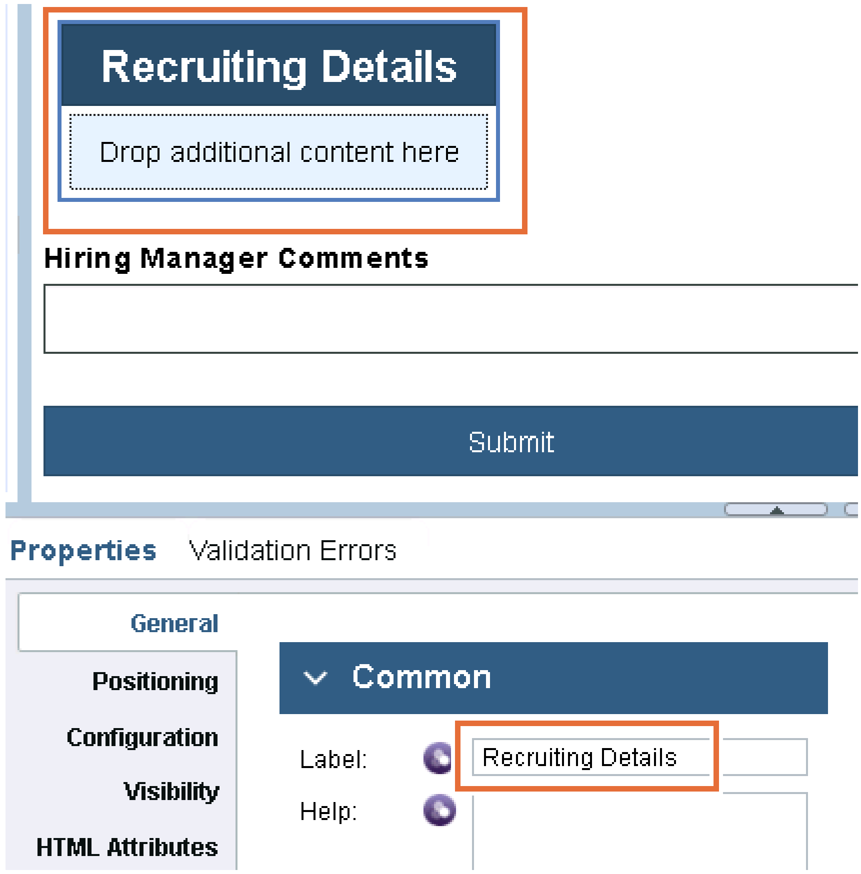


__ 8. Group the `recruitingDetails` variables together, setting them apart from the other data on the screen in a section.

__ a. Drag a section onto the canvas in the first column grid below the **Date Details** section. Wait to see the orange line at the required position and drop the section there.



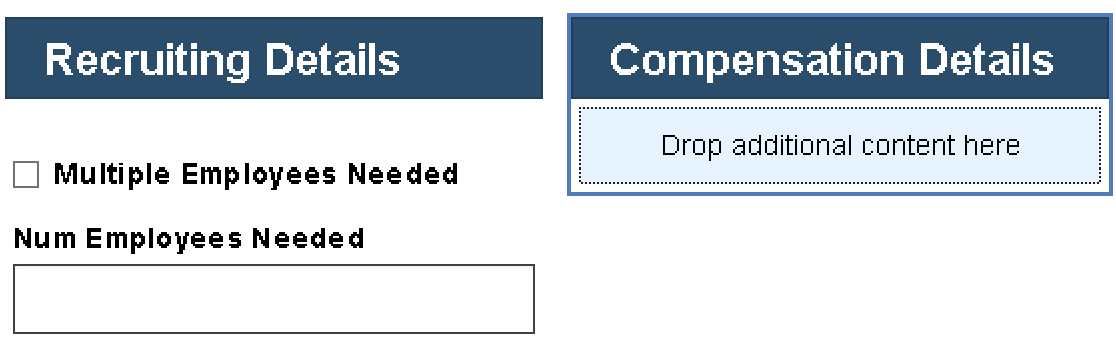
__ b. In the **Properties > General > Common** section, label the section: `Recruiting Details`



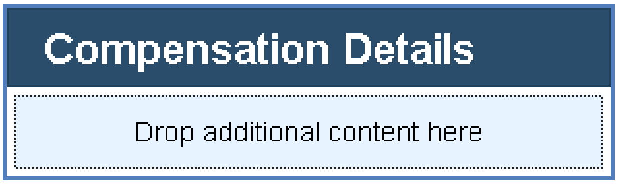
__ c. From the **Variables** palette tray, drag the **requisitionDetails > recruitingDetails** complex object to the new section.

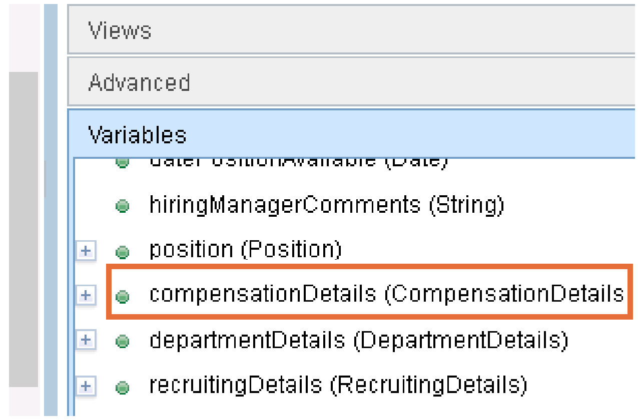
__ 9. Add the Compensation Details to the coach.

__ a. Expand the **Views** palette tray. Drag a **Section** control onto the canvas in the second column grid beside the **Recruiting Details** section, and label the new section: `Compensation Details`



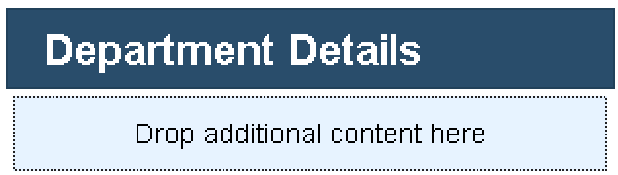
__ b. Add the **requisitionDetails > compensationDetails** complex variable to the new section.



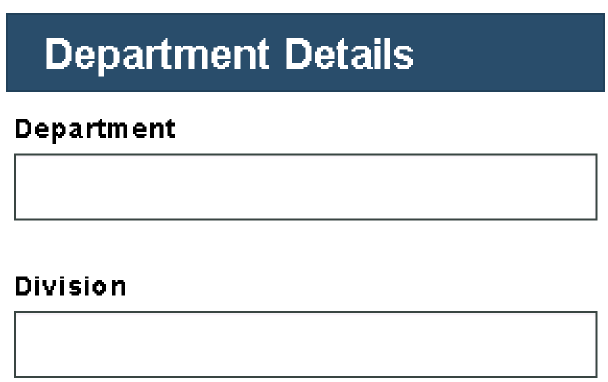


__ 10. Finally, add the Department Details section to the coach.

__ a. For the final section, drag a section in the first column grid below the **Recruiting Details**, and label the new section: `Department Details`

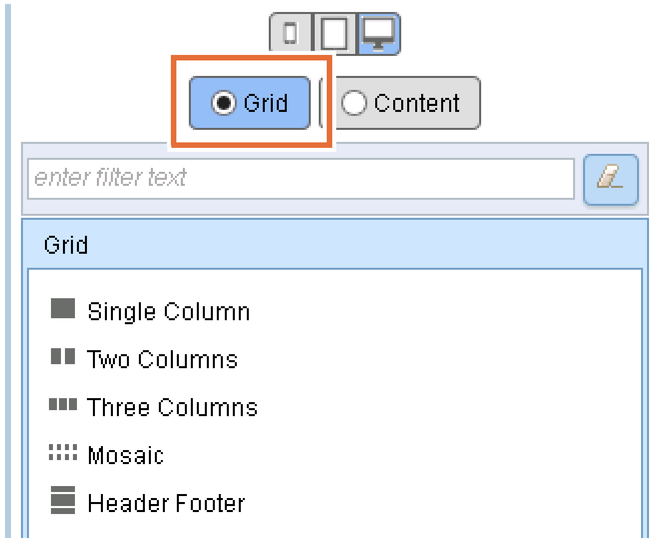


__ b. From the **Variables** palette tray, drag the **requisitionDetails > departmentDetails** complex object to the new section.

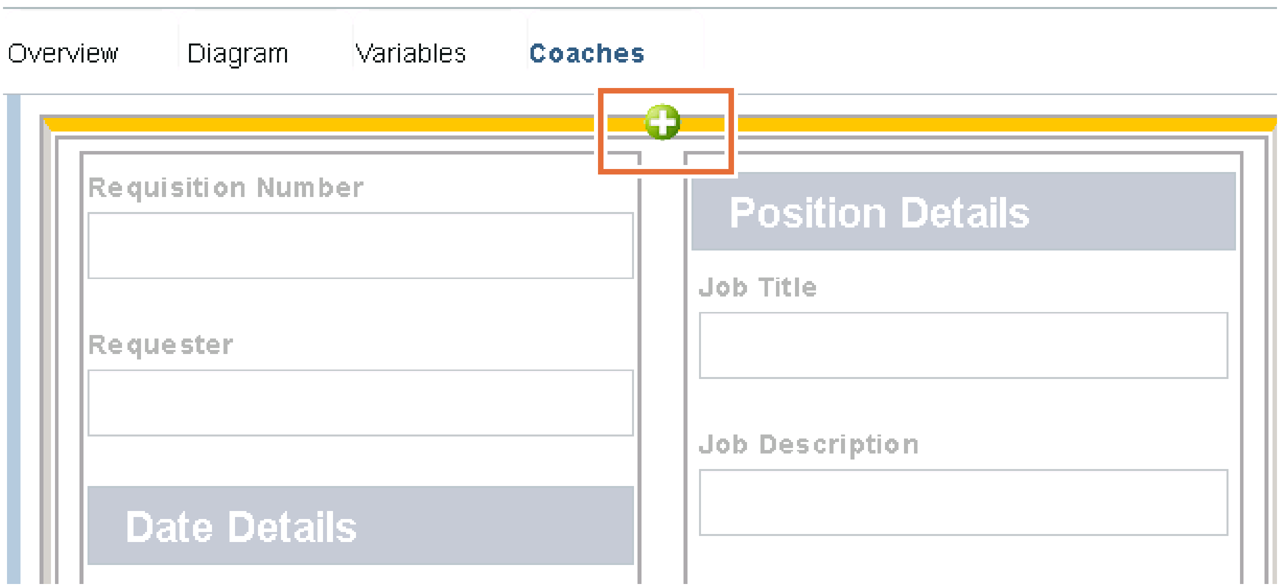


__ 11. Add a grid in the coach to realign the sections.

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

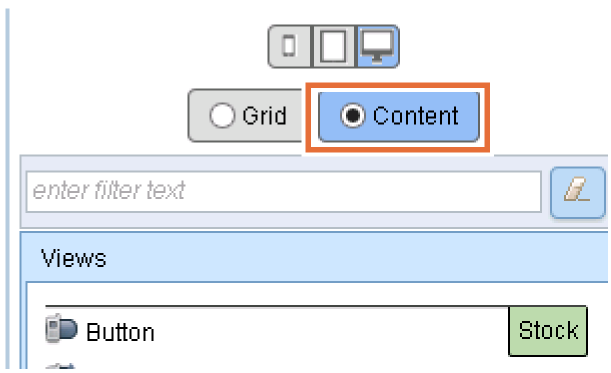


__ b. Click the (**+**) plus icon on top of the grid to add a grid at this location.



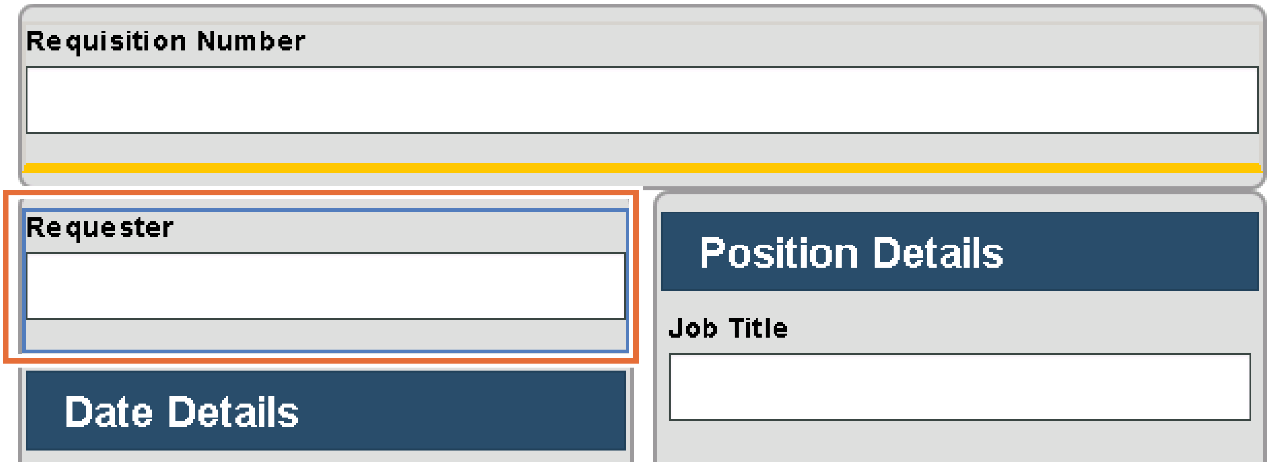
You added a grid at the top.

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

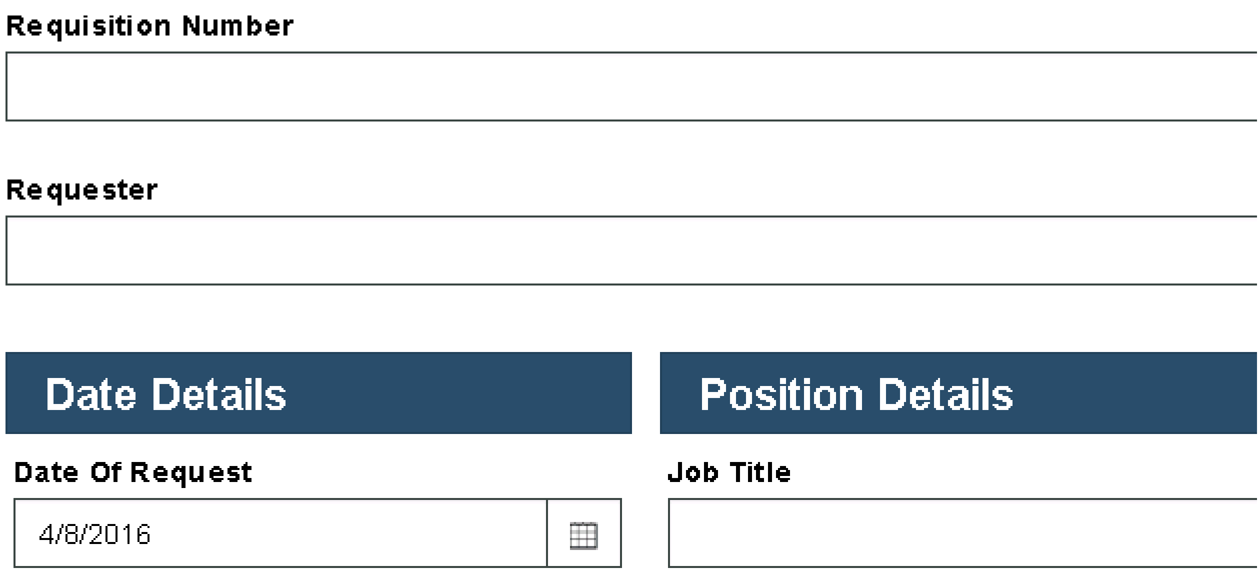


__ d. Drag the **Requisition Number** input control that is in the left grid to the new grid at the top.

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

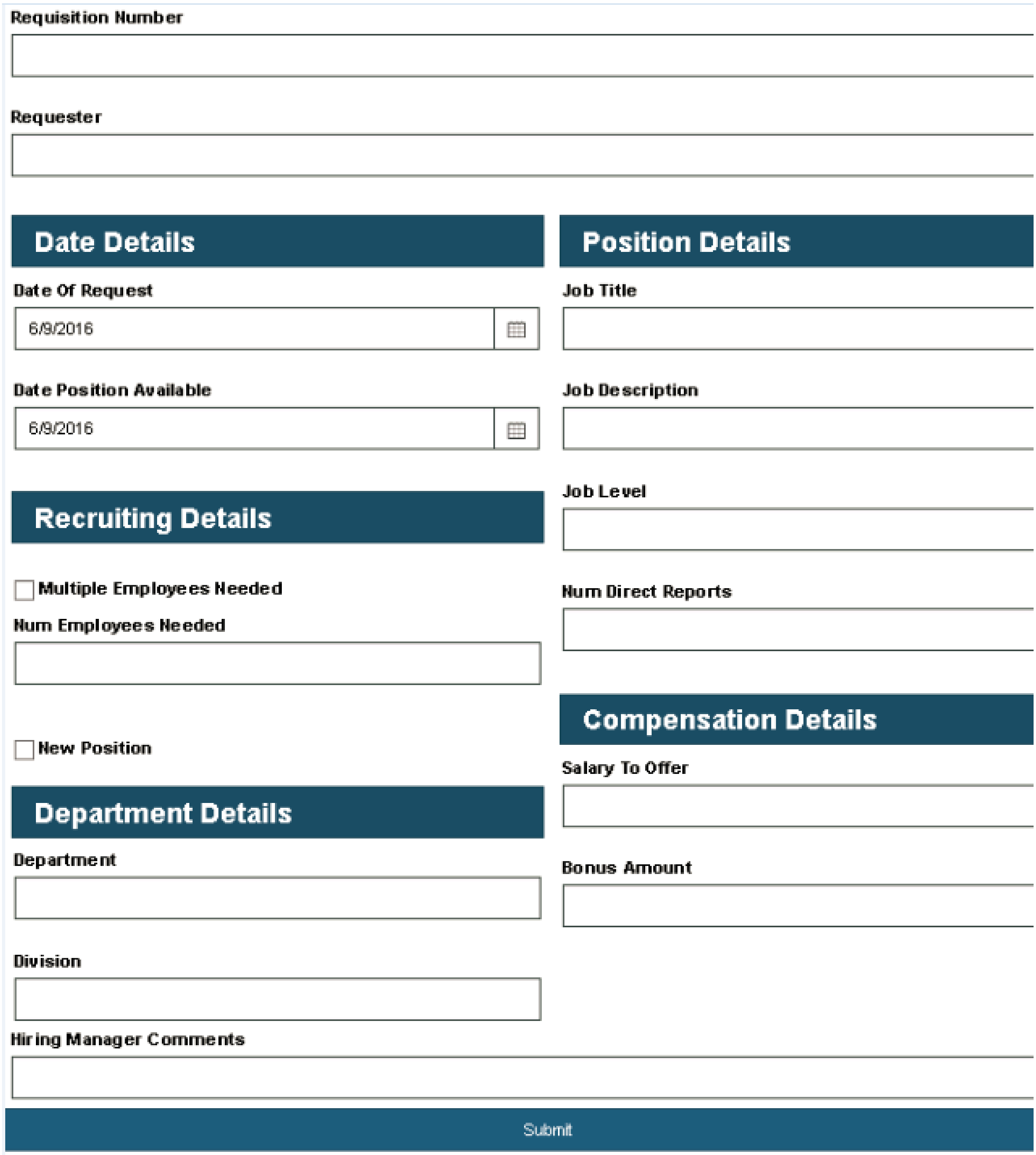


__ f. `Requisition Number` and `Requester` are now part of this new grid.



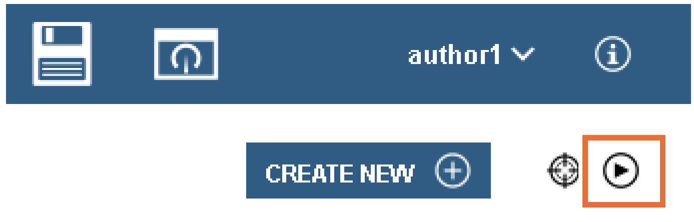
__ g. Save your work.

Now the coach layout looks similar to the following coach.



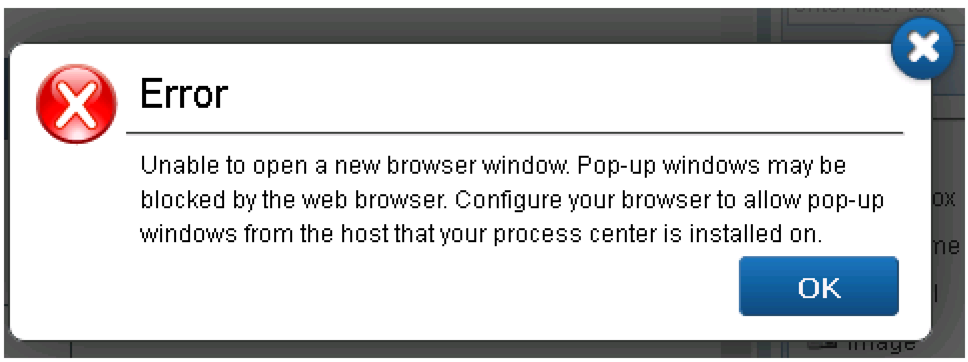
__ 12. Run the coach.

__ 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.

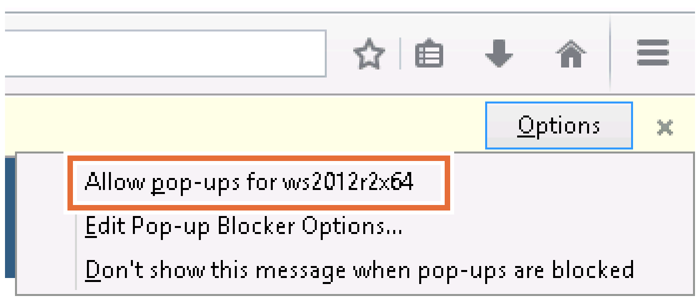


Note

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

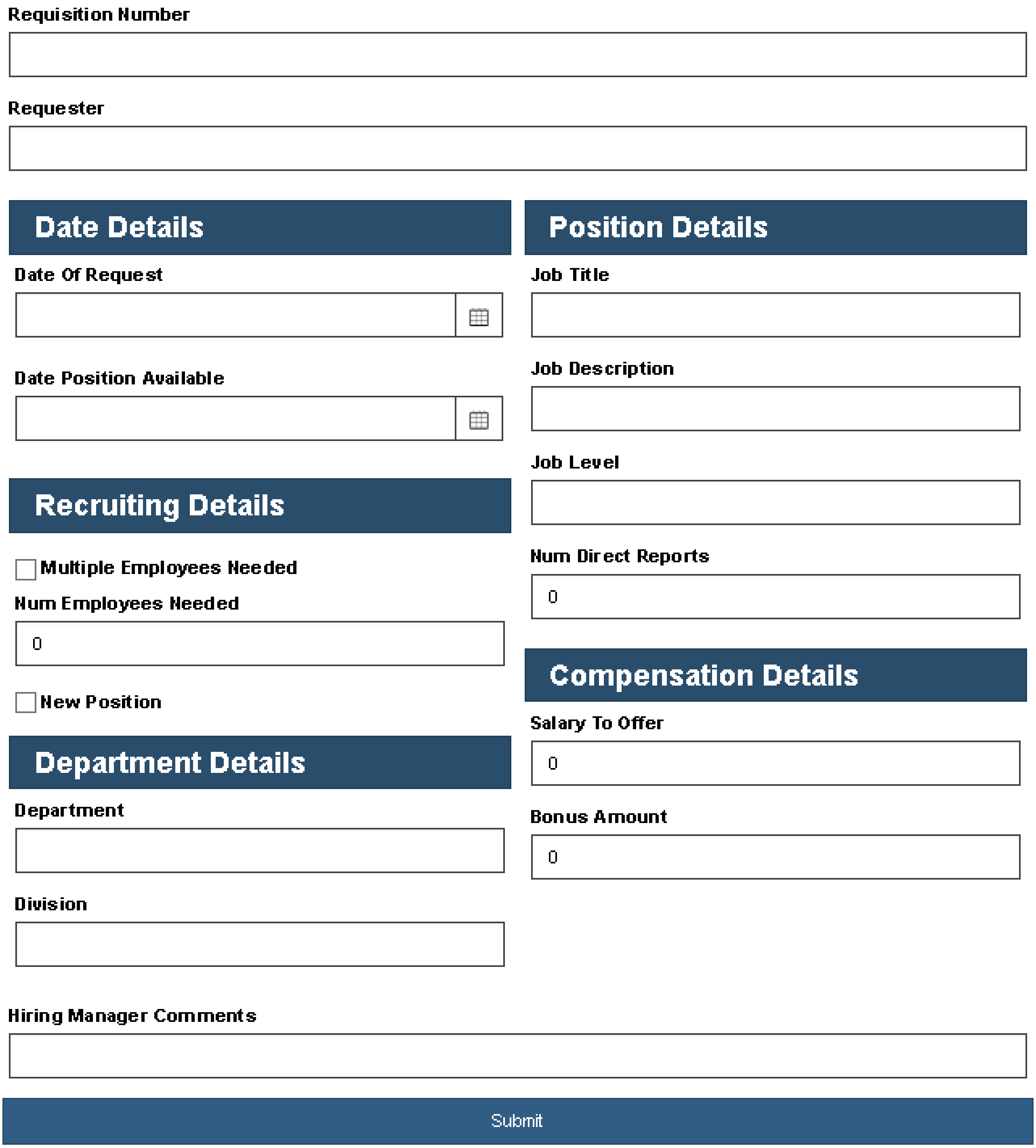


To correct the problem, in the yellow bar at the top of the browser, click **Options > Allow pop-ups for ws2012r2x64**.



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.



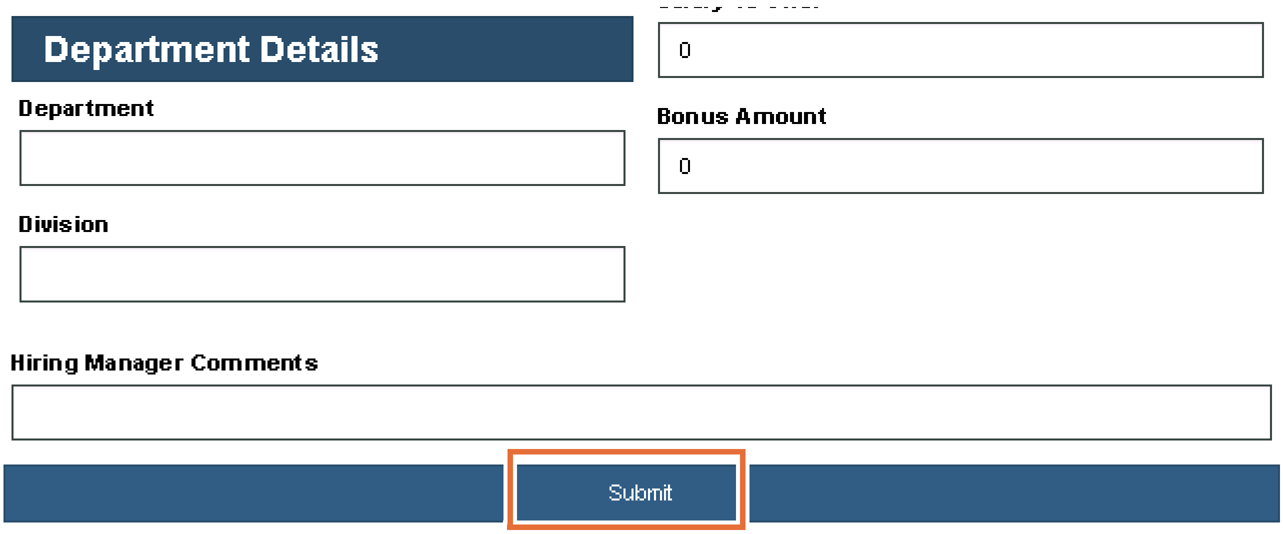
Important

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

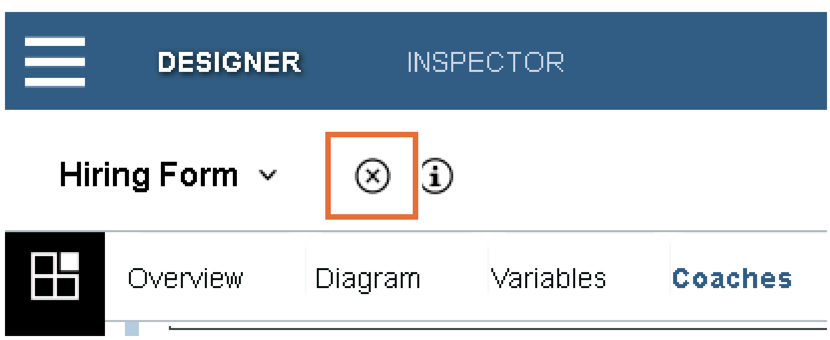
This coach is not meant to be an attractive, fully functioning front end; you add more functional and visual features later on in this playback. The intent of this playback is to start data flow.

__ c. Click the calendar button next to 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** to close the coach.



__ 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.



Part 3: Create human services

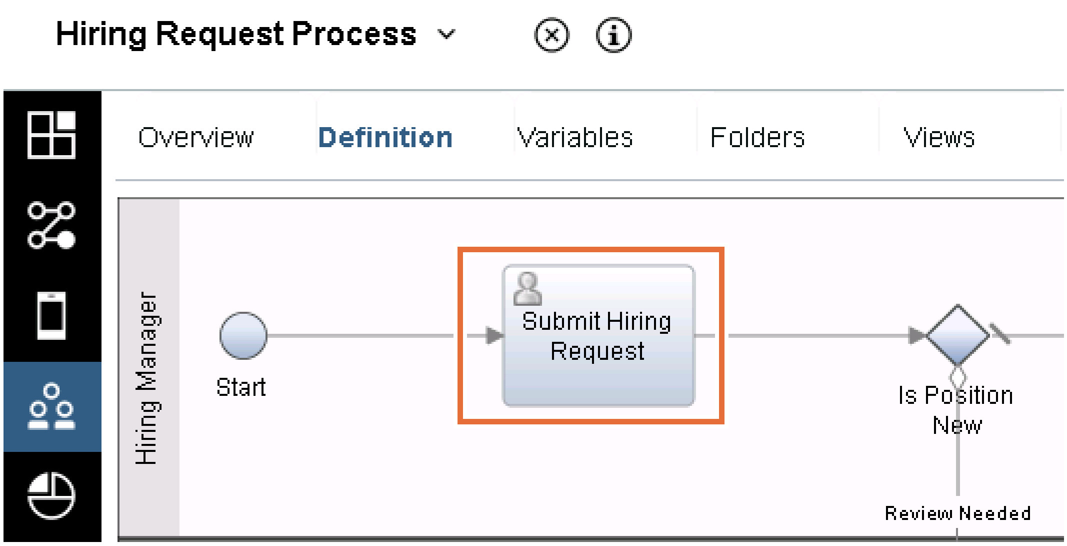
Because the **Approve New Hire Request** activity in the **Hiring Request Process** reviews the same data that is entered in the last human service, you can reuse the Hiring Form human service. Add a service to an activity.

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.

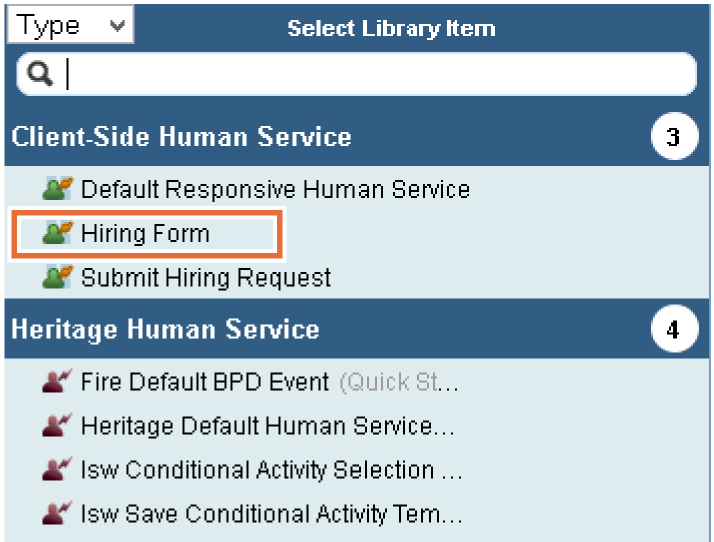
__ a. In the web Process Designer, open the **Hiring Request Process** process.

__ b. Click the **Submit Hiring Request** activity.



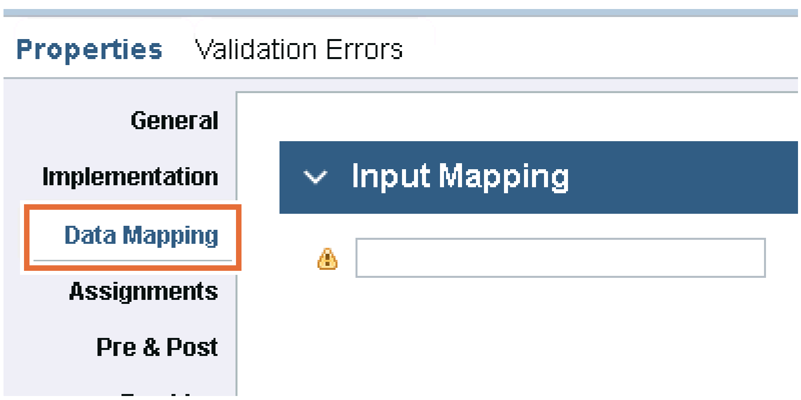
__ c. In the **Properties > Implementation > Implementation** section, click **Select** next to Implementation to select the service that implements the user task.

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



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

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

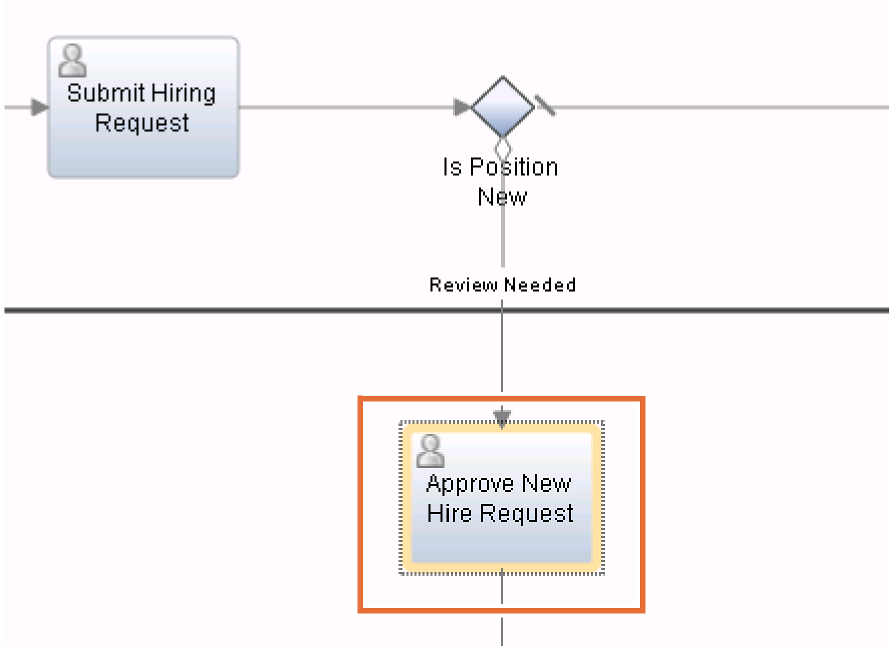


__ b. Map `tw.local.requisitionDetails` to both **Input Mapping** and **Output Mapping**.

__ c. Save your work.

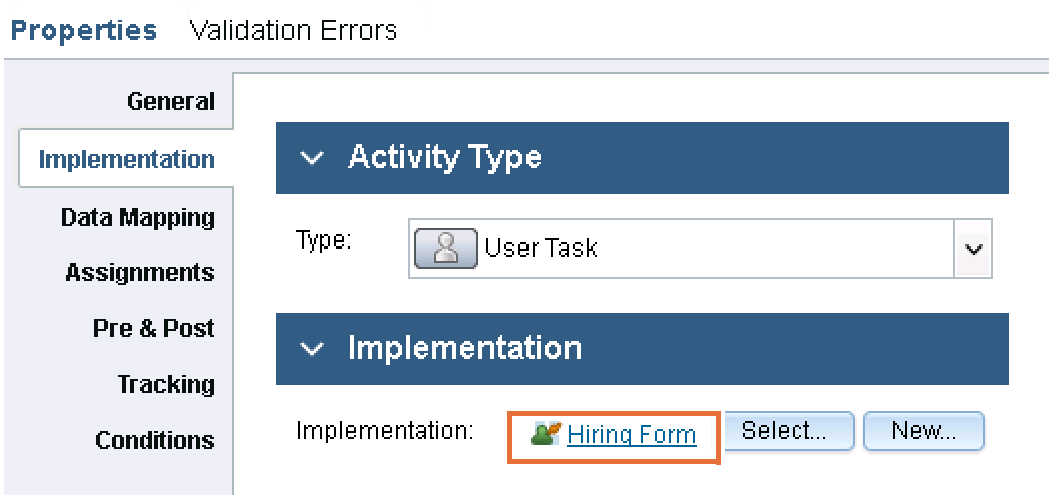
__ 3. Implement the **Approve New Hire Request** activity.

__ 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.

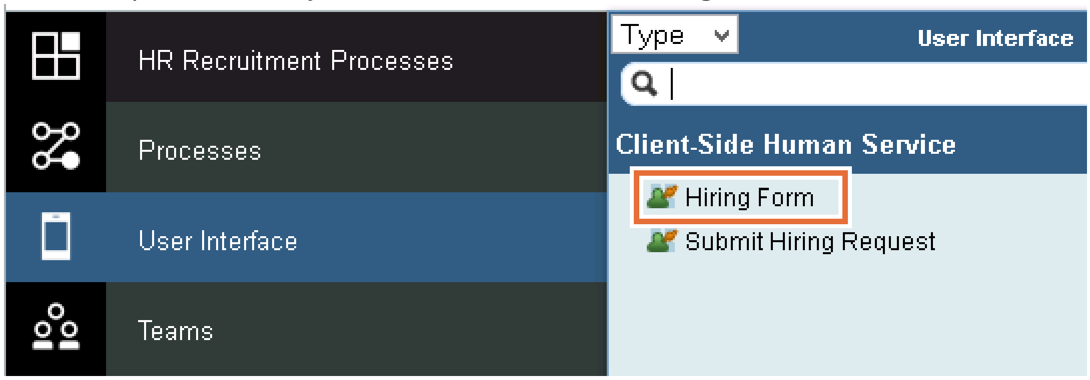
__ a. Open the **Properties > Data Mapping** menu for the **Approve New Hire Request** activity.

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

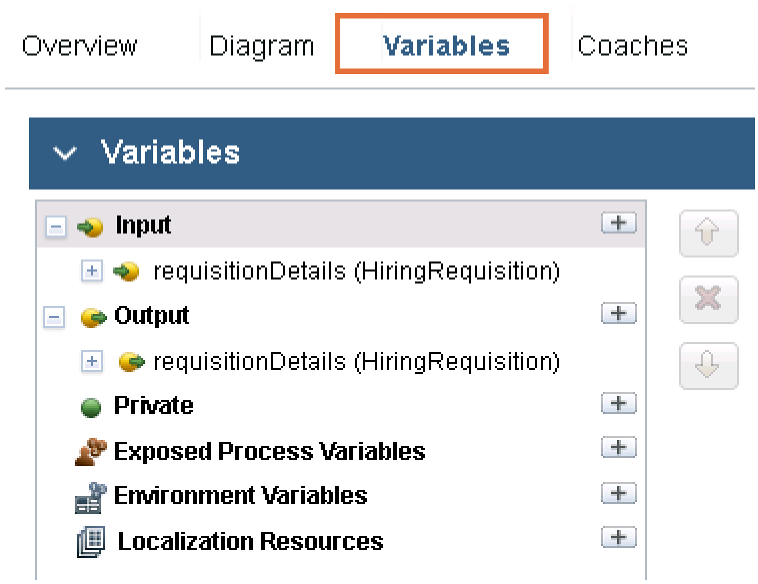
__ c. Save your work.

__ 5. To change the process flow, you can assign business data to a process flow object. Assign the `NewPosition` variable to the `IsNewPosition` variable.

__ a. In the process library, click **User Interface > Hiring Form**.

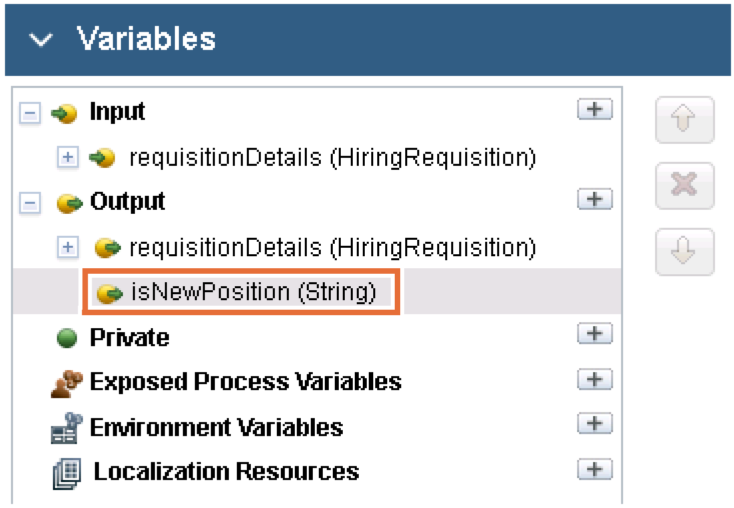


__ b. Click the **Variables** tab.



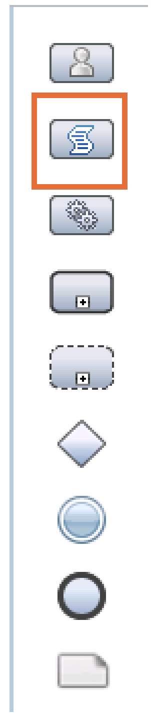
__ c. Click the (**+**) plus sign next to Output.

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

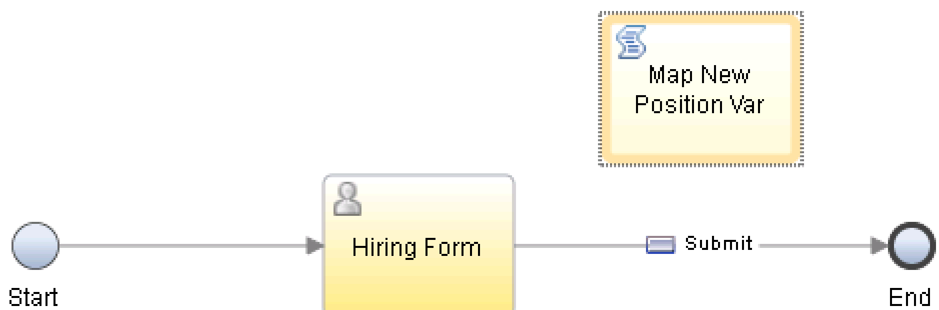


__ e. Click the **Diagram** tab.

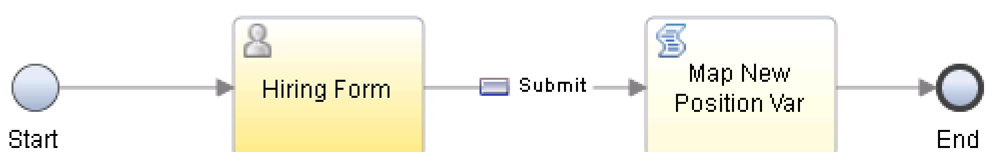
__ f. Drag a **Client-Side script** between the **Hiring Form** coach and the **End** event.



__ g. Name the new client-side-script: `Map New Position Var`

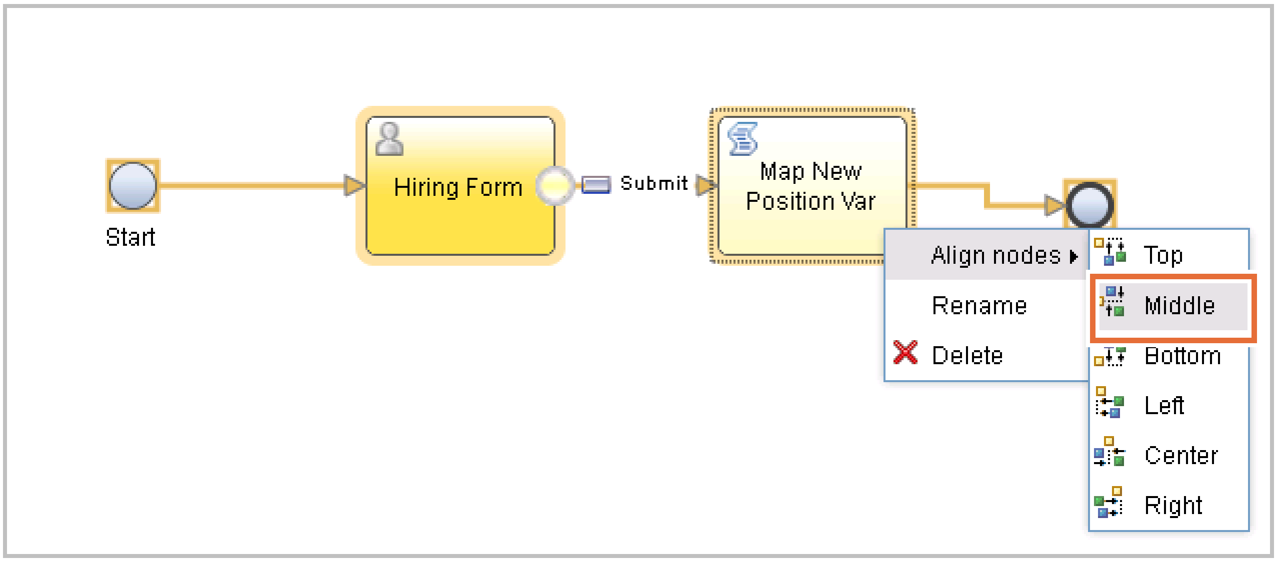


__ h. Connect the flow between the **Hiring Form** and **Map New Position Var** script and then to the **End** event.



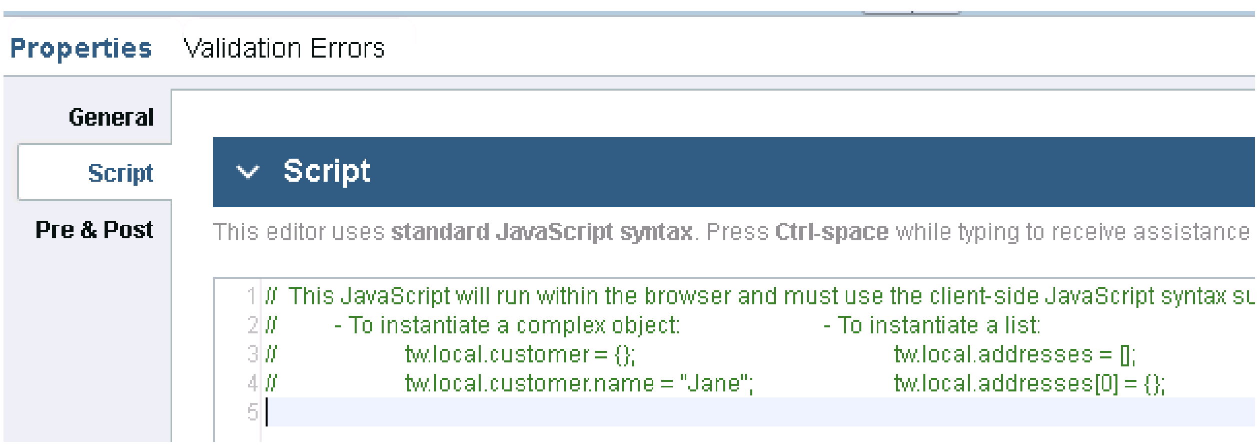
Hint

To straighten the lines when building a service, click **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**.



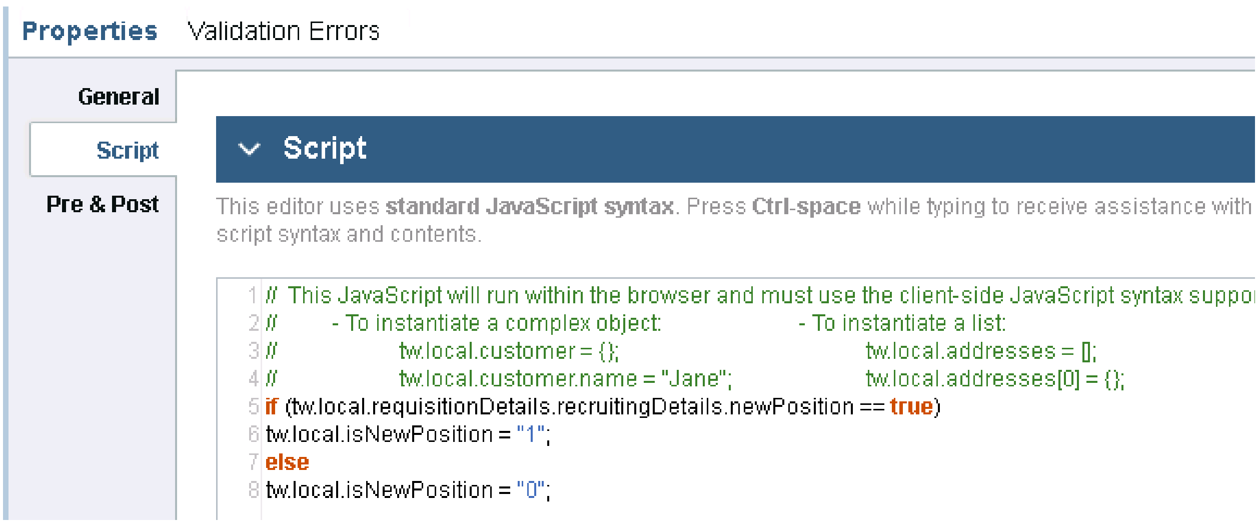
__ i. Click the **Map New Position Var** server script.

__ j. Open the **Properties > Script** menu.



__ k. Copy and paste the following code in the Script box. 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";
```



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.

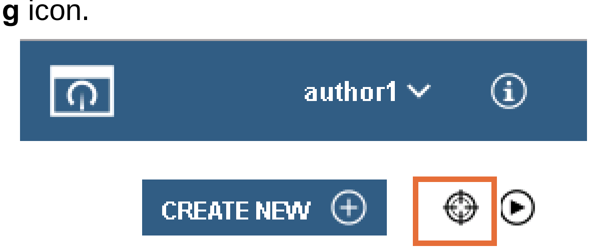
Note

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

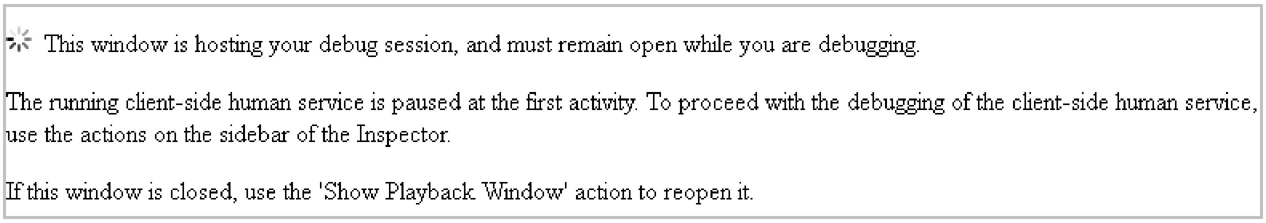
__ l. Save your work.

__ 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.

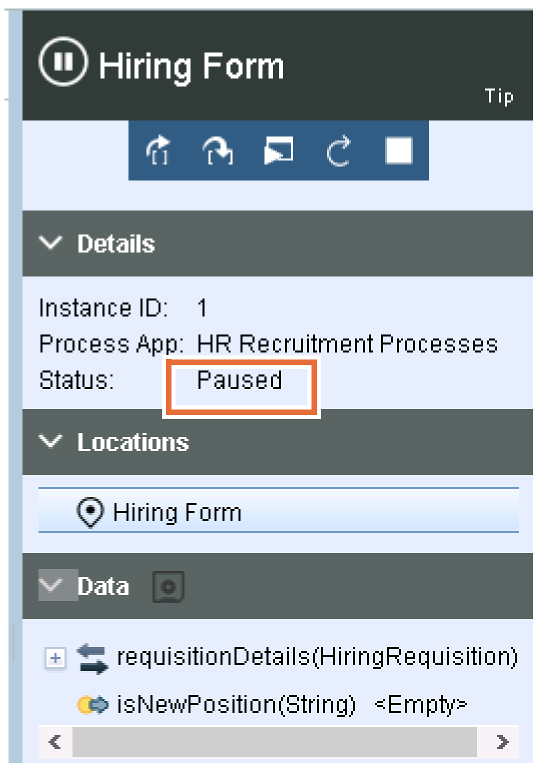
The running client-side human service is paused at the first activity. To proceed with the debugging of the client-side human service, use the actions on the sidebar of the Inspector.

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

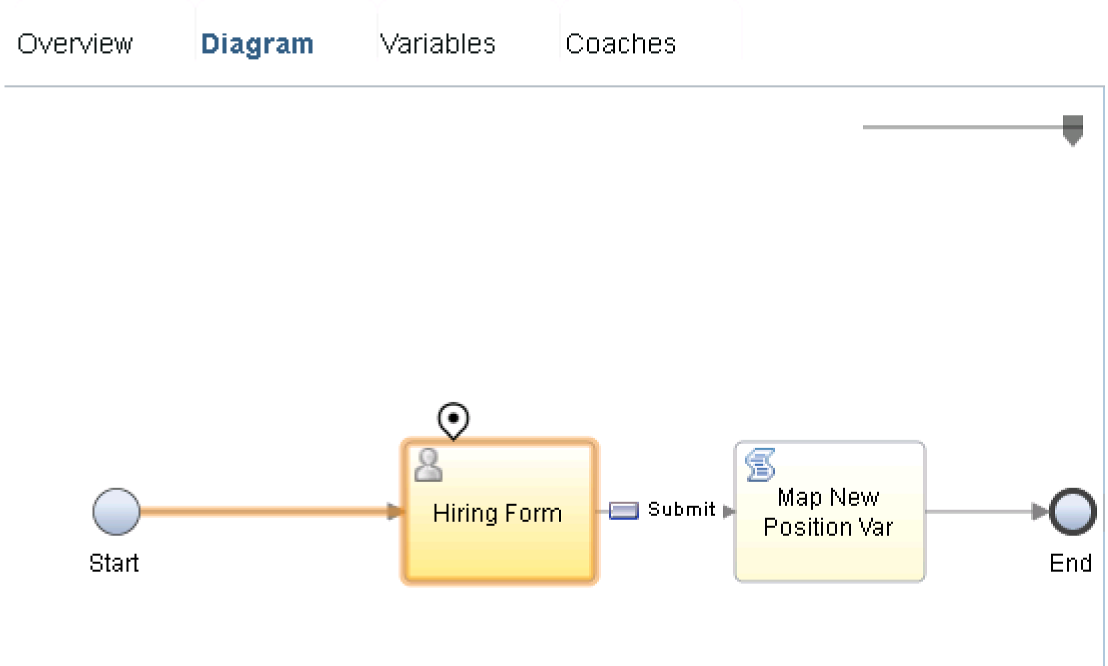
__ c. Minimize this browser window to view the client-side human service designer window.

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

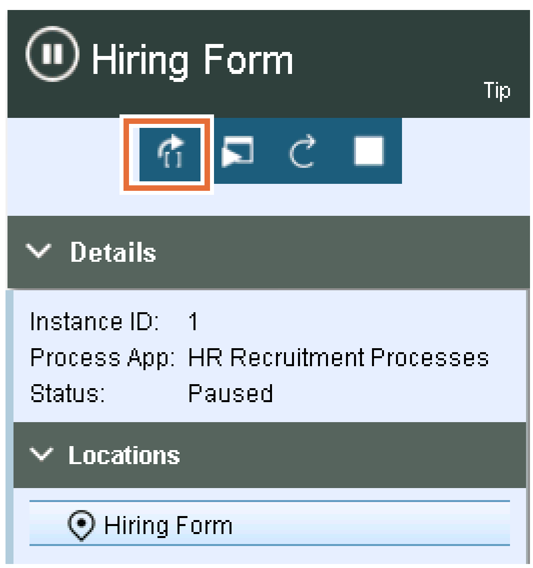
__ e. The inspector section is displayed on the right side. The status shows that the execution is Paused.



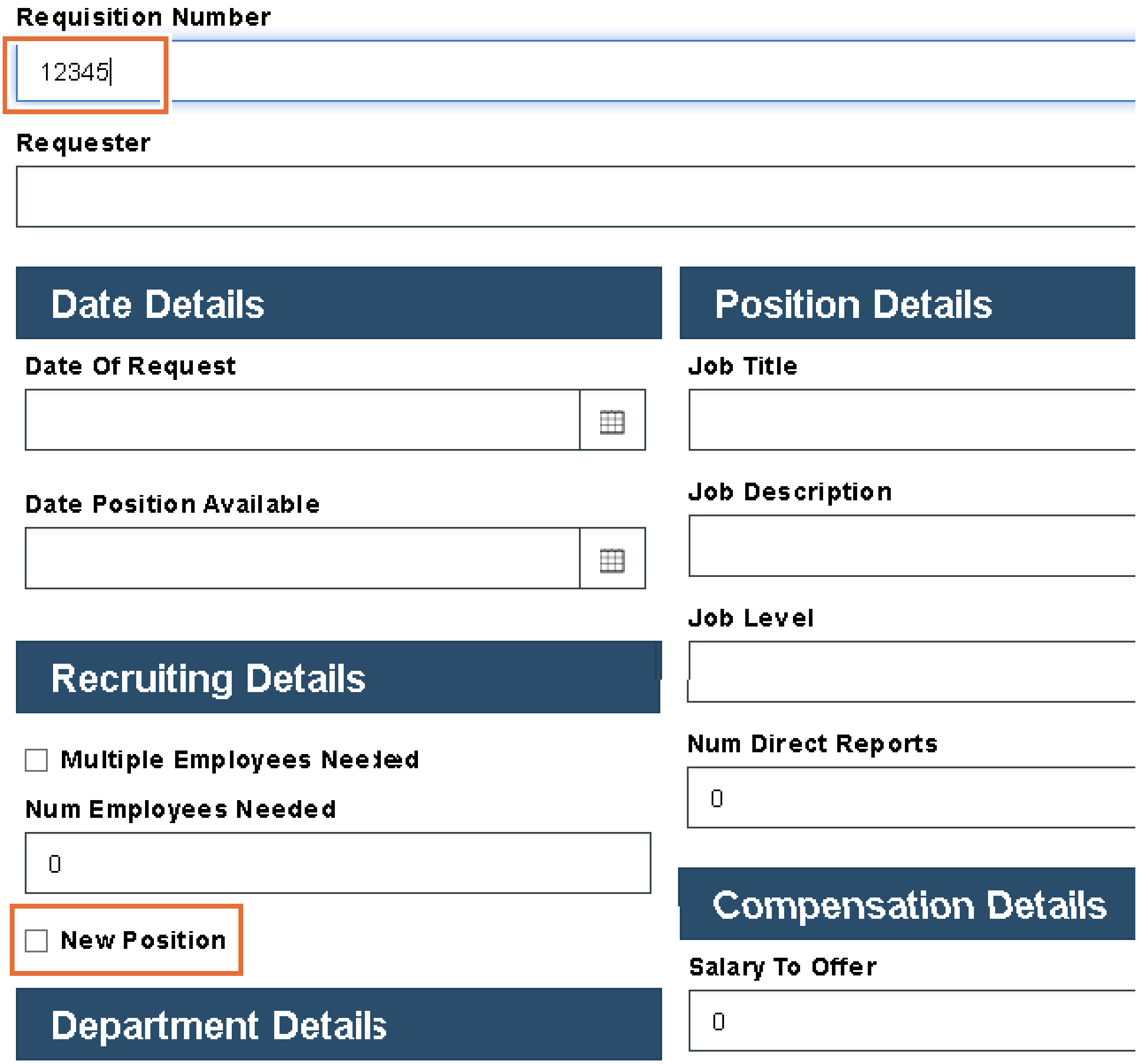
__ f. If not already selected, click the **Diagram** tab. The inspector shows a token that is waiting to execute this step in the process.



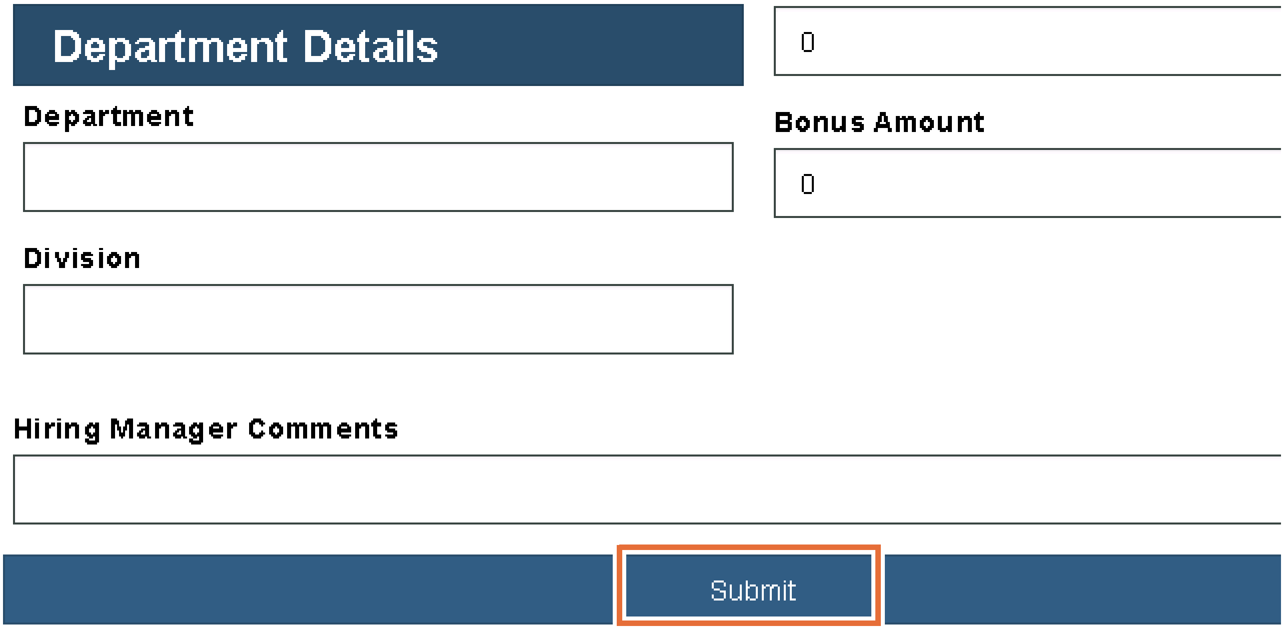
__ g. Click the **Step over** icon for the Hiring Form coach.



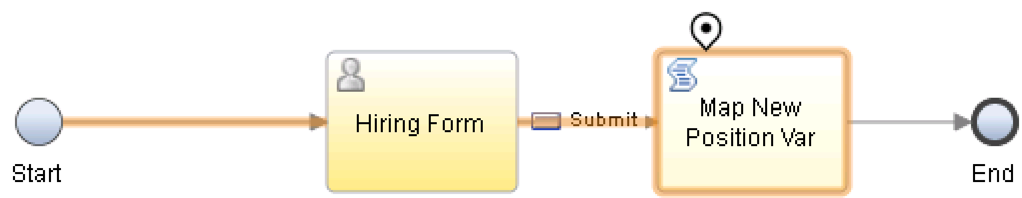
__ h. The second browser window renders the coach. Enter 12345 as the Requisition Number. Ensure that the **NewPosition** check box is not selected.



__ i. Click **Submit**.

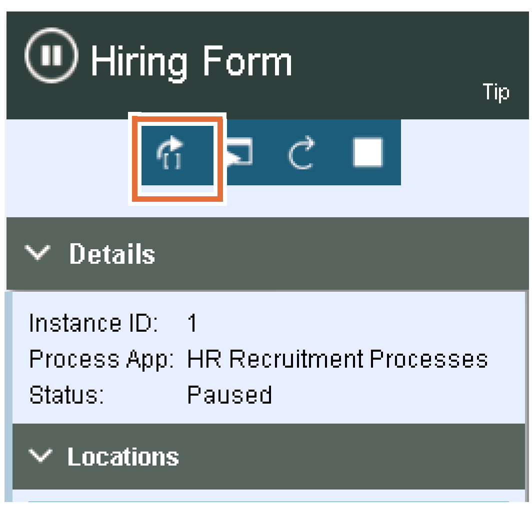


__ j. Minimize the coach browser window and return to the designer browser window. The Hiring Form coach is complete, and the token moves to **Map New Position Var**.



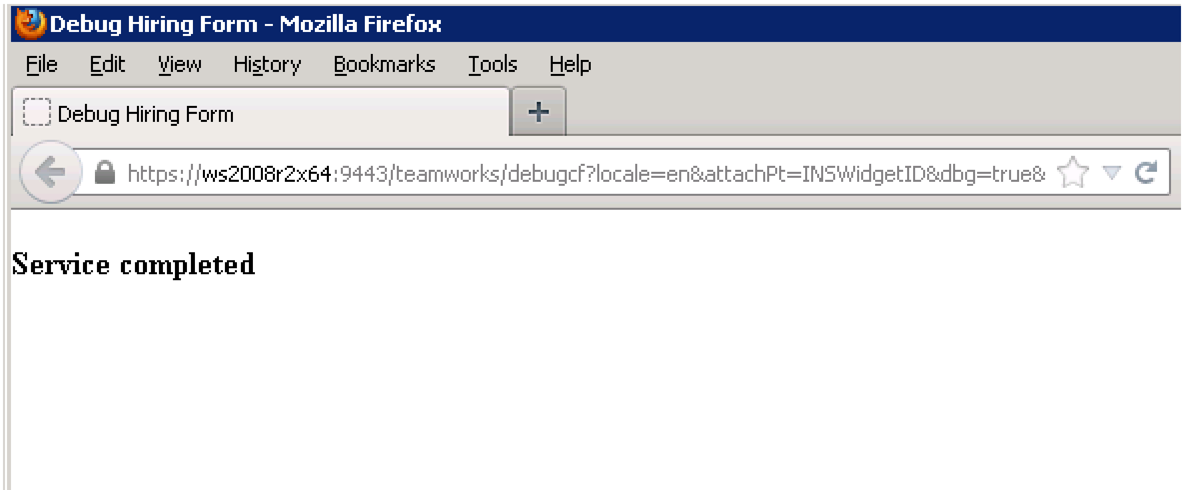
__ k. Expand the **Data** section and click the (**+**) plus sign next to **requisitionDetails** to see the updated value for the `requisitionDetails.requisitionNumber` variable. Verify that the updated value for the `requisitionDetails.requisitionNumber` variable is 12345.

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

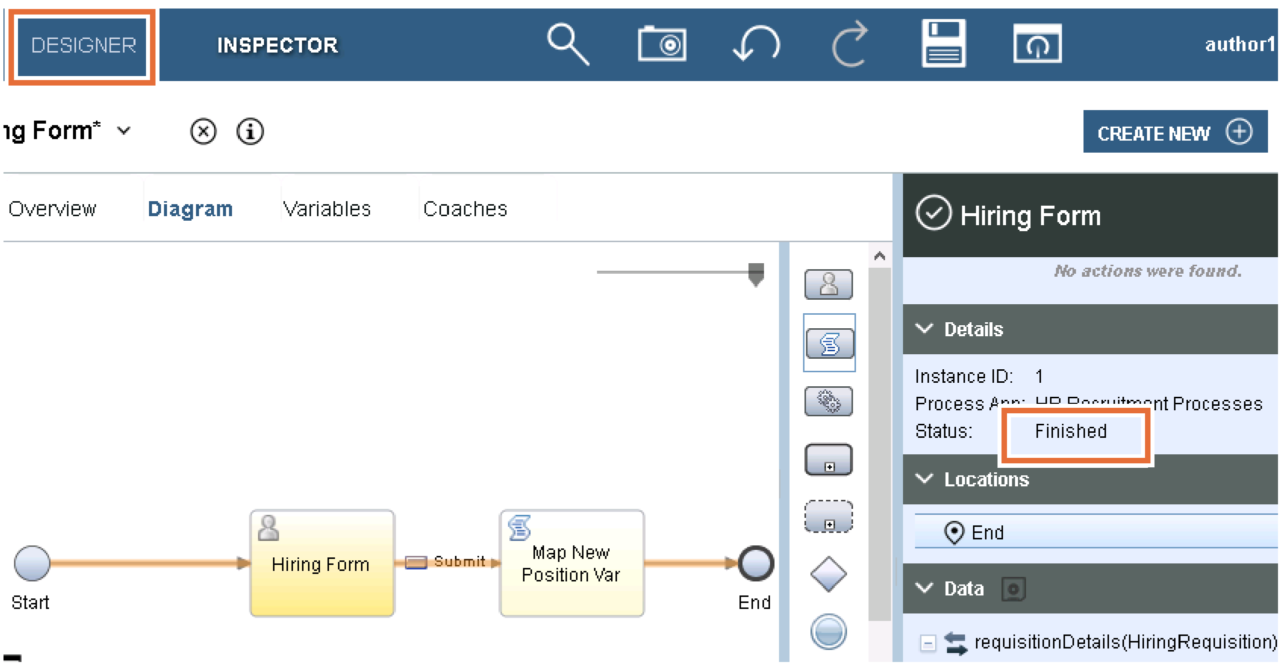


__ m. The status changes to **Finished** and the token disappears because it reached the end event. Verify that the Map New Position Var script set the `isNewPosition` value to 0. The data is flowing correctly.

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

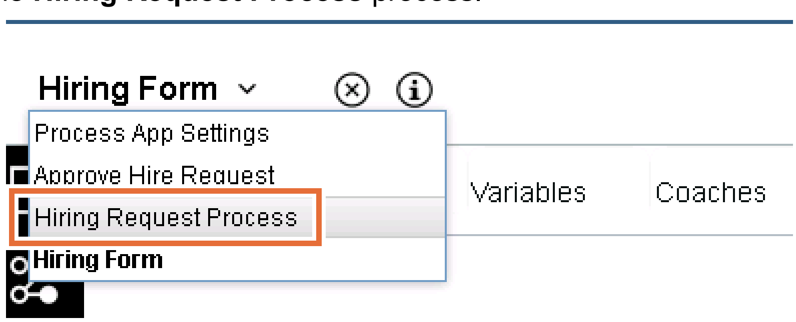


__ o. The Status is **Finished**, and you are currently in the **INSPECTOR** tab. Return to the **DESIGNER** tab. The activity and flow highlights disappear in the Designer mode.



__ 7. Map the new variable in the process.

__ a. Go to 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.

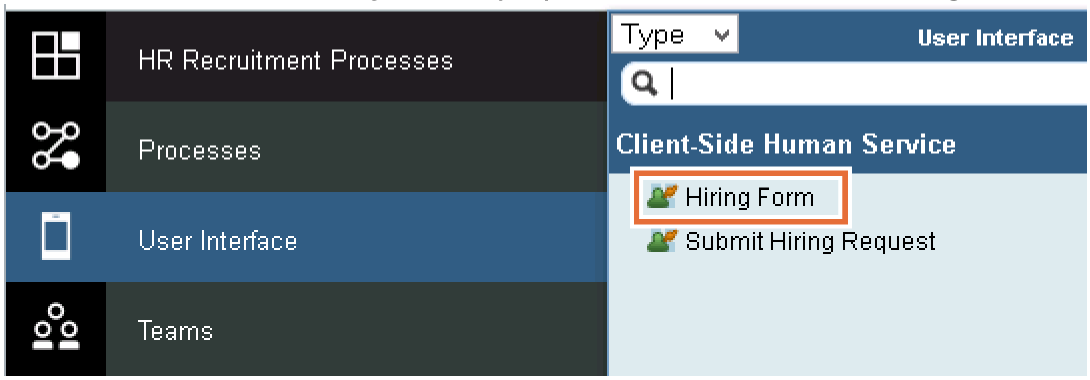
Exercise instructions

Part 1: Group controls into tabs on a coach

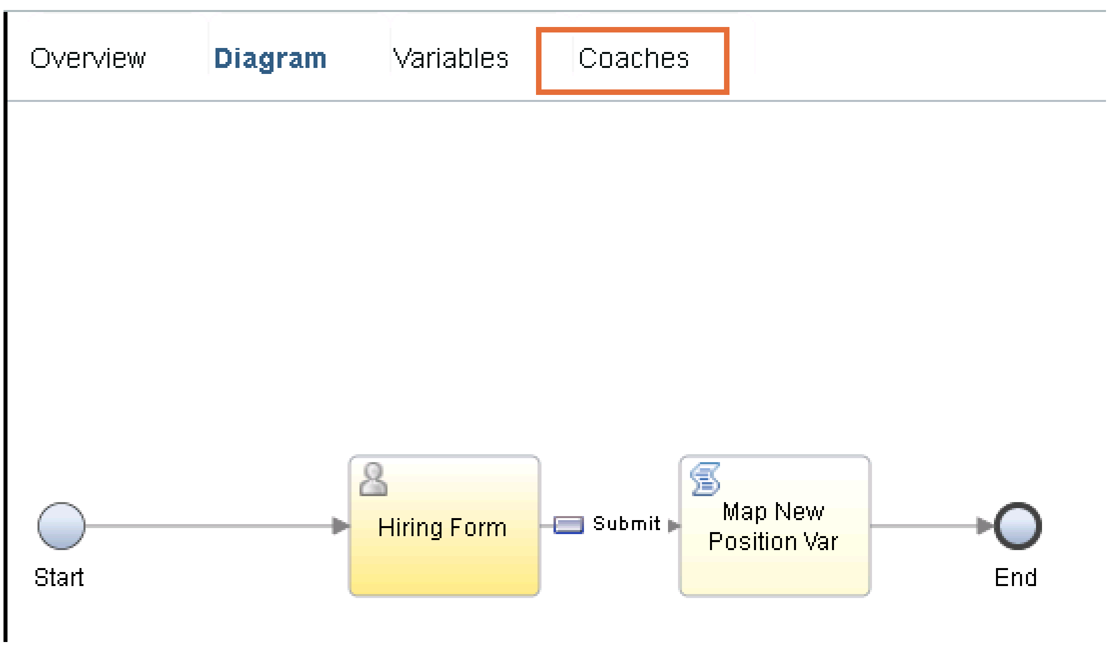
You completed different stages of playback in the development cycle. In one of the previous playbacks, you provided a functional look at the coaches in the **Hiring Request Process**. Enhance the coaches with more features and functions. Start with creating a polished look to the coach by grouping similar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

__ 1. Create tabs to hold the grouped sections on the coach.

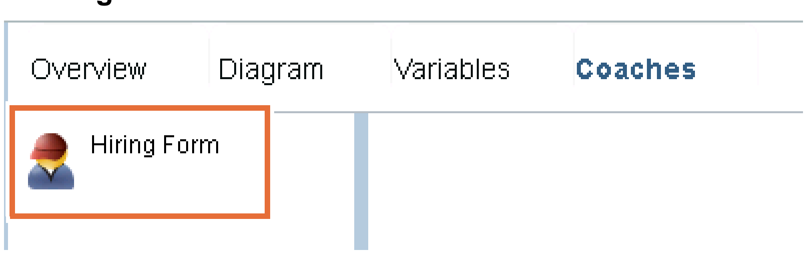
__ a. In the web Process Designer library, open the **User Interface > Hiring Form**.



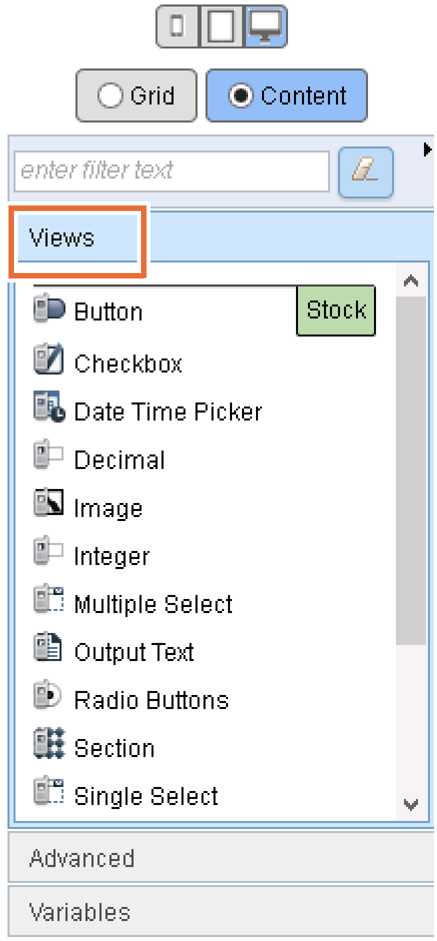
__ b. Click the **Coaches** tab.



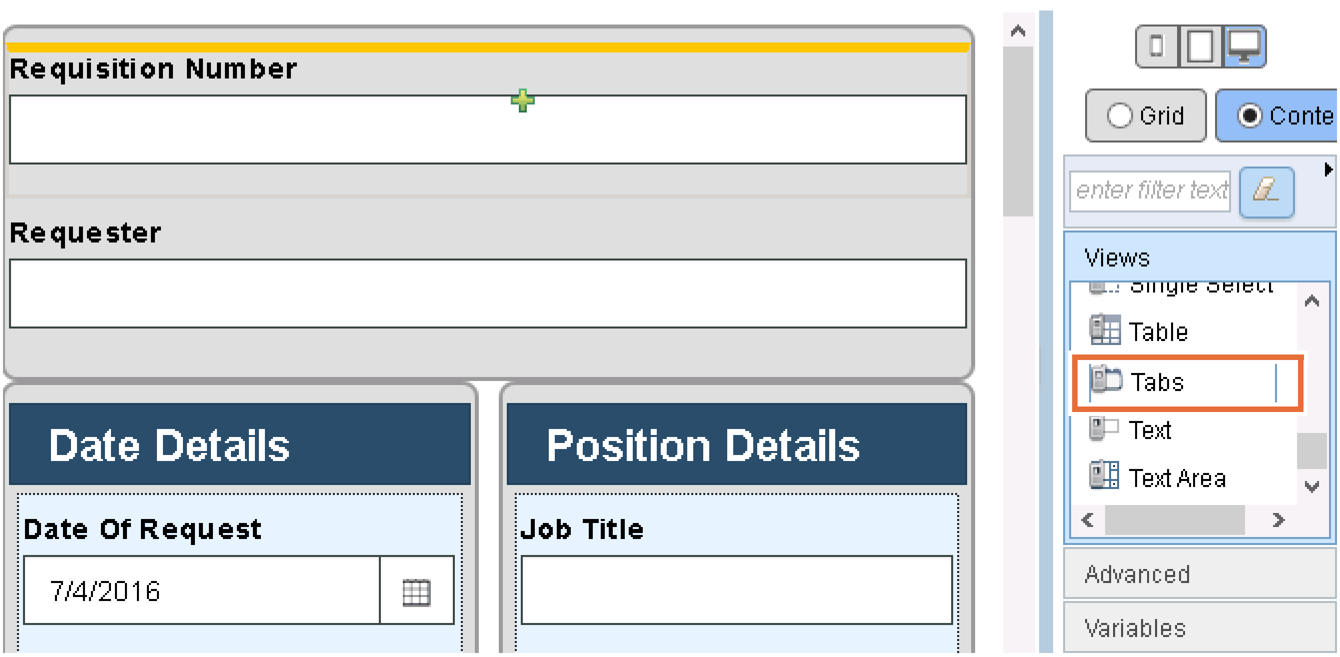
__ c. Click the **Hiring Form** coach.



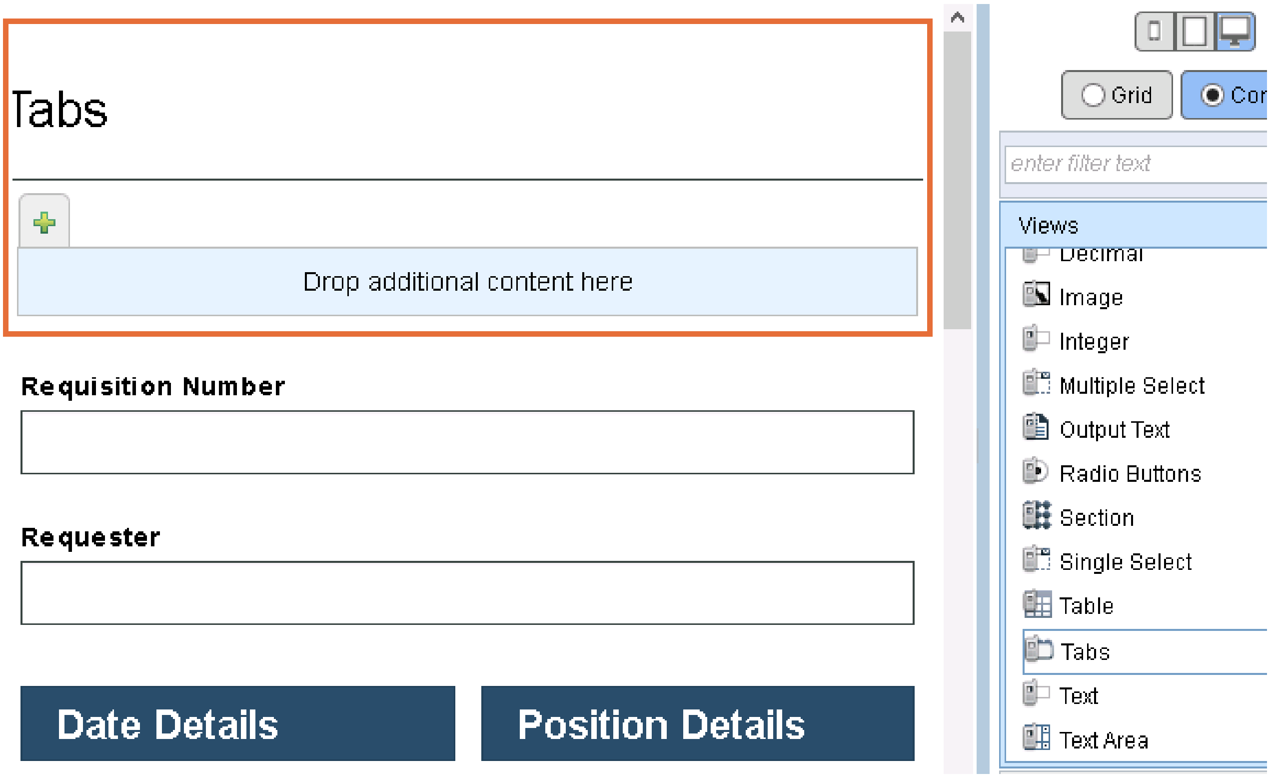
__ d. Expand the **Views** palette tray on the right.



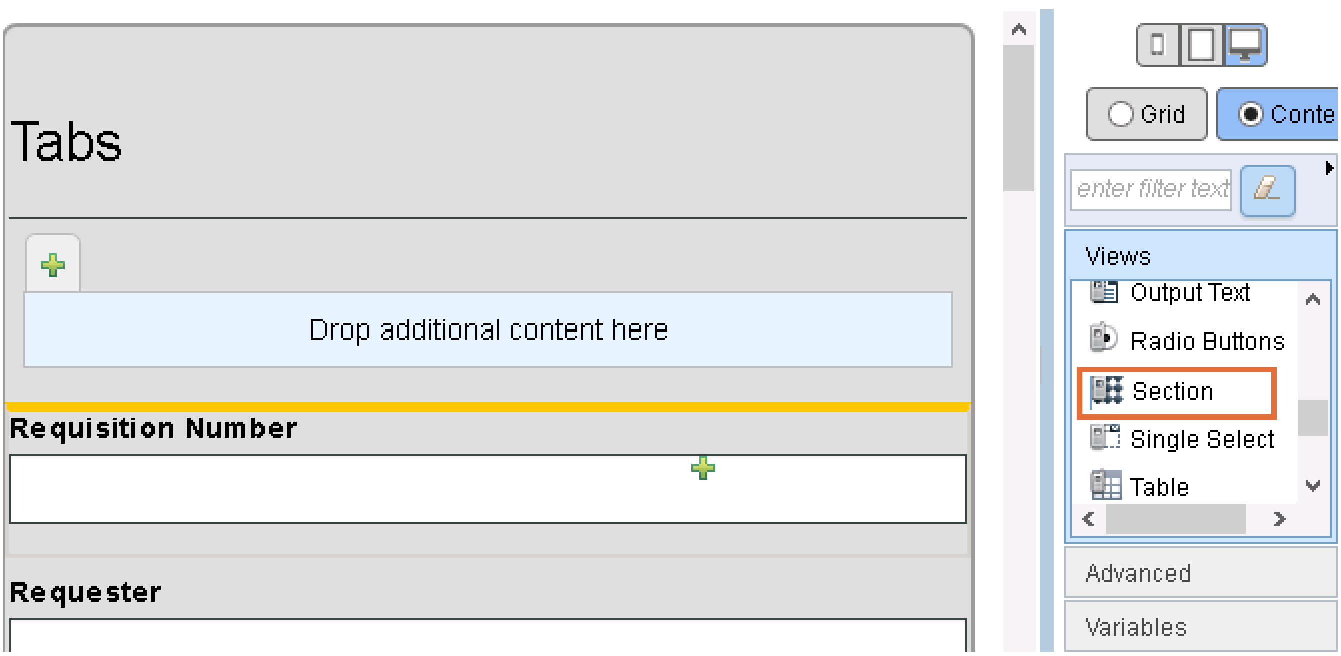
__ e. Drag a **Tabs** control onto the canvas in the top grid of the coach.



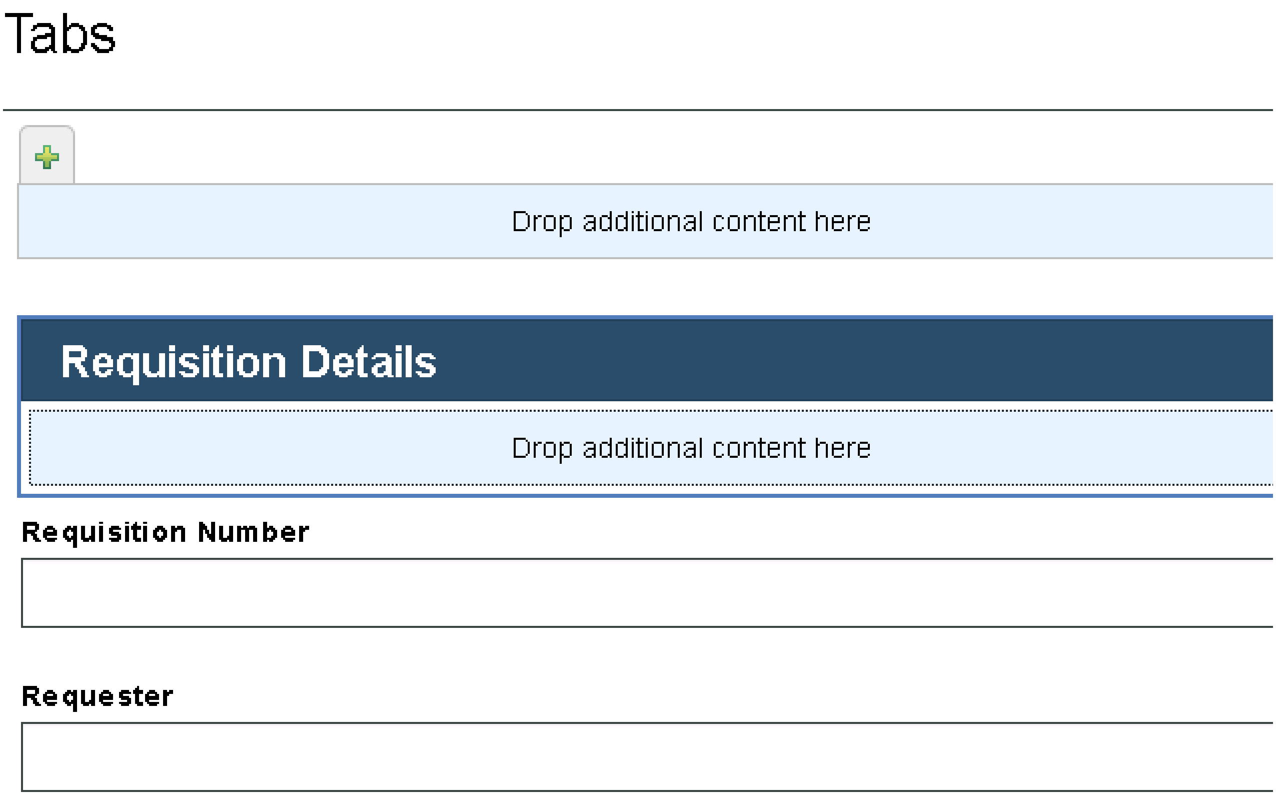
You added the Tabs control in the top grid of the coach.



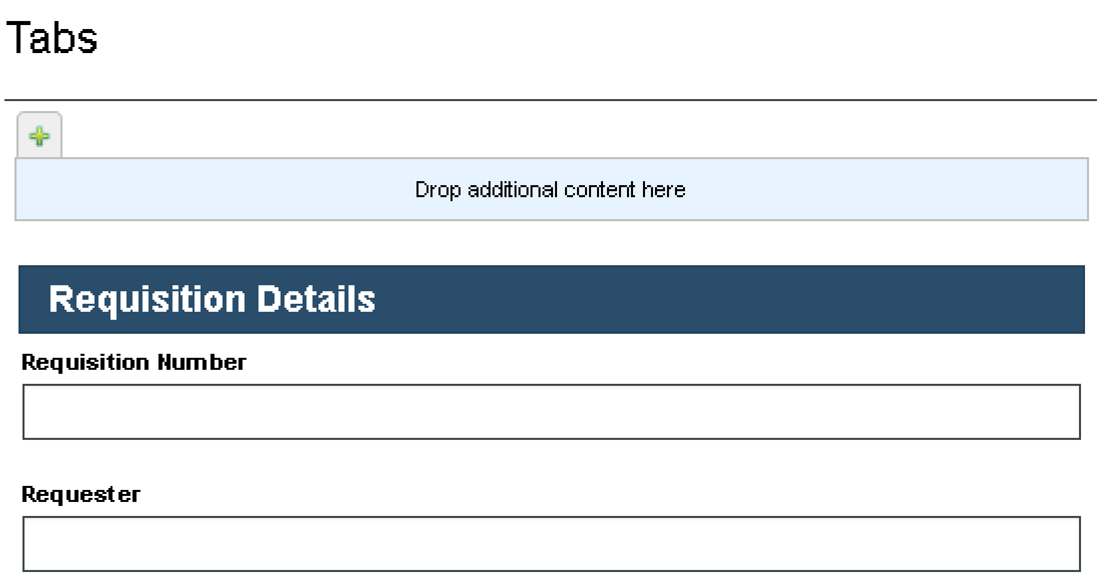
__ f. From the **Stock** section in the palette, drag a **Section** control above the Requisition Number input control.



__ g. Rename the section to: `Requisition Details`

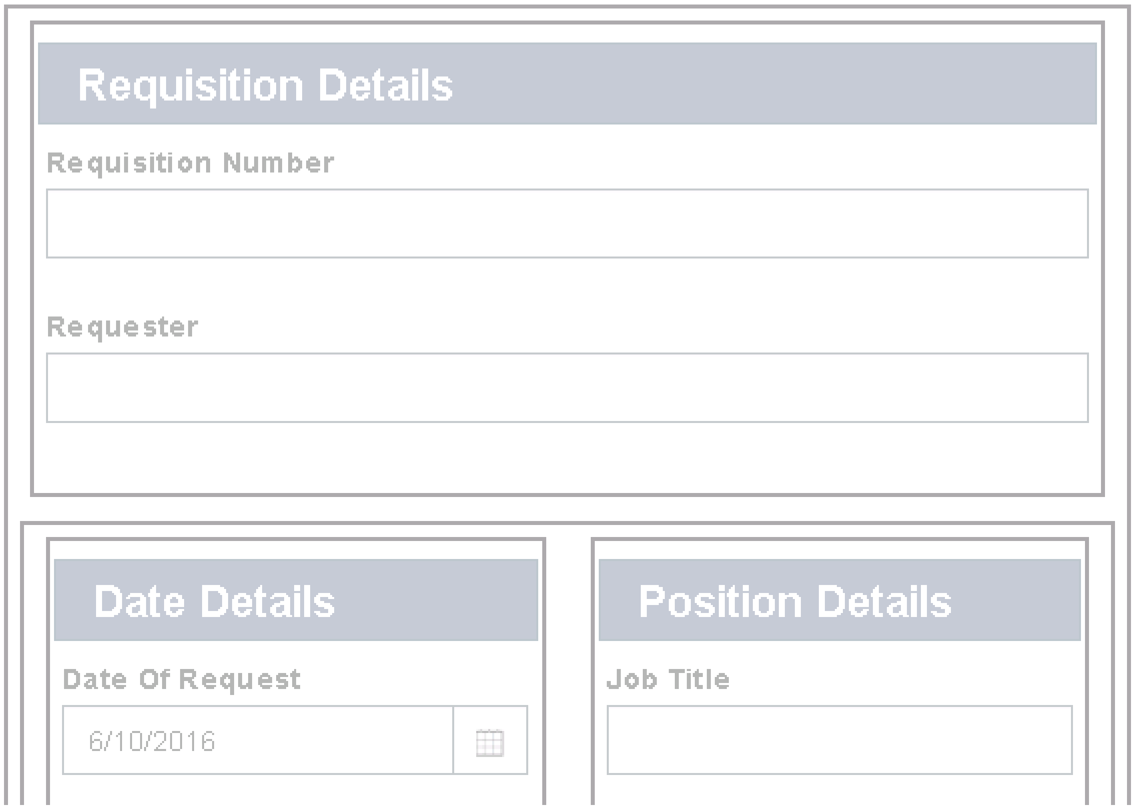


__ h. Drag the **Requisition Number** and **Requester** controls from the coach onto the area that is labeled **Drop additional content here** under Requisition Details.

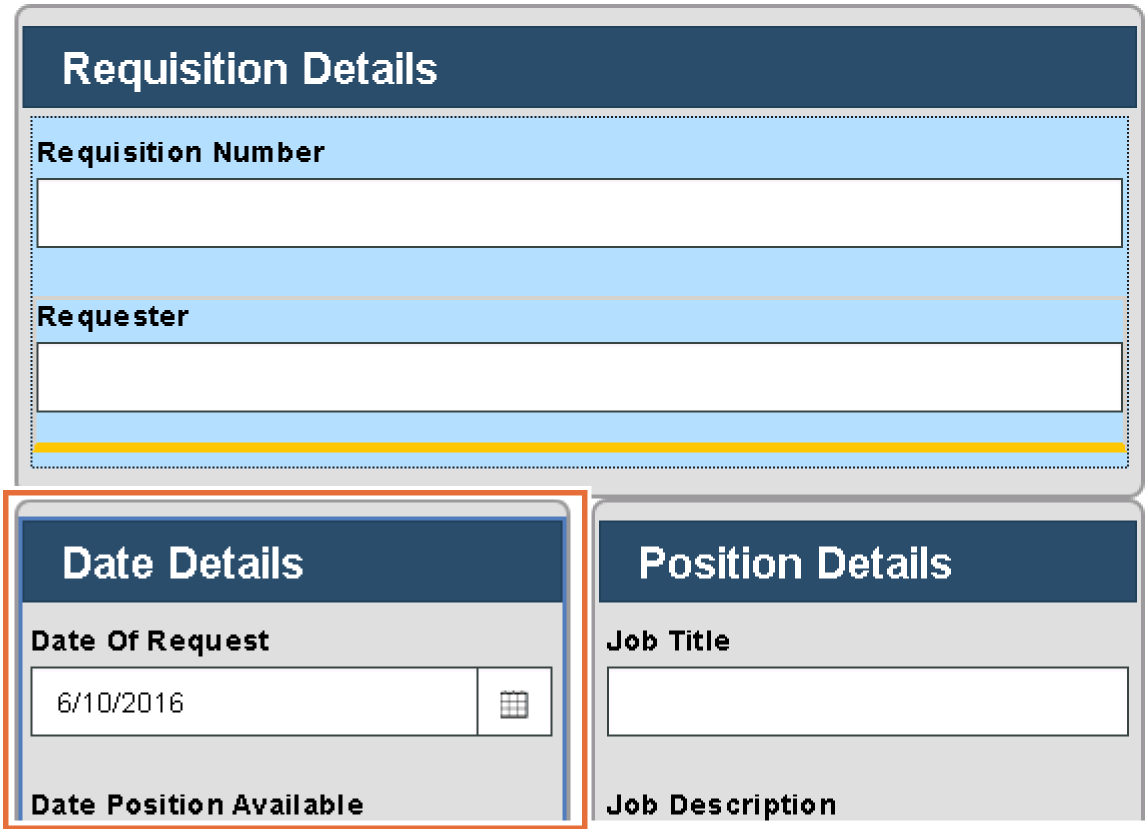


Hint

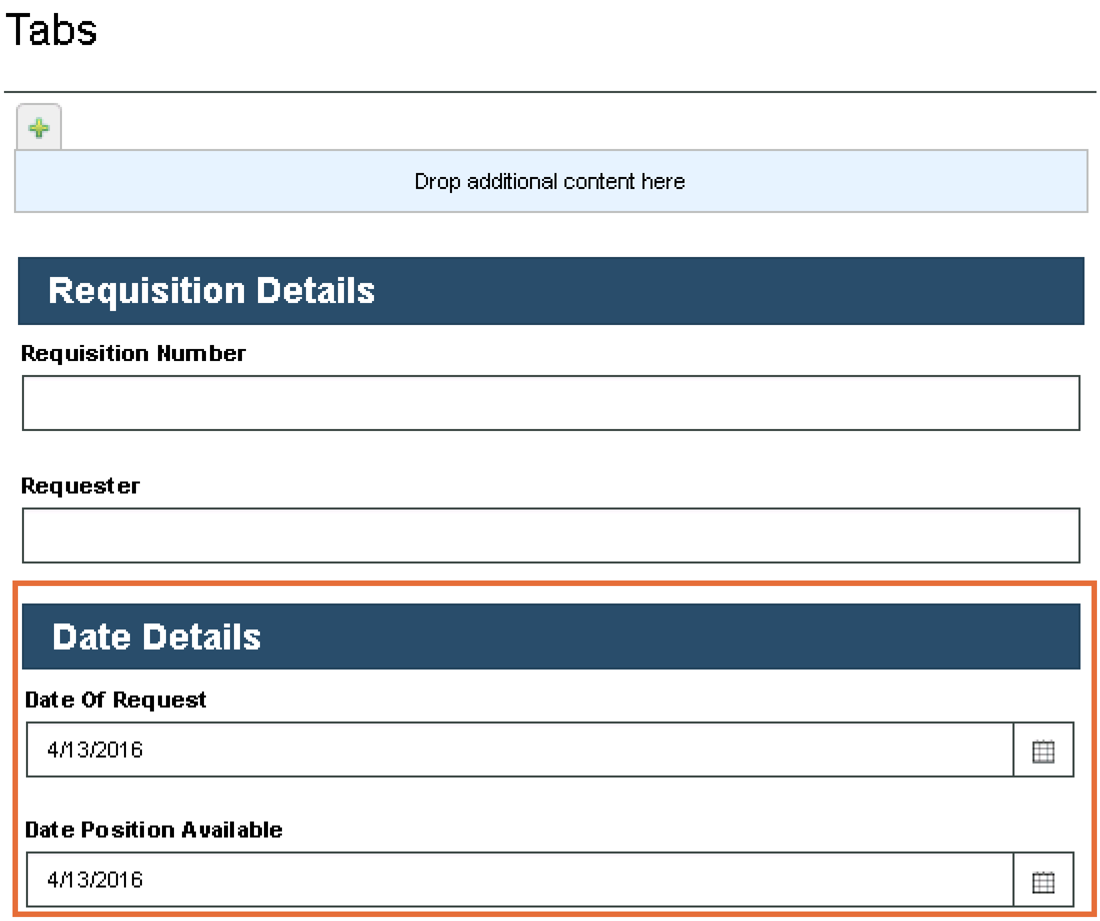
You can switch to the Grid view to verify the position of the input controls. The Requisition Number and Requester controls are part of Requisition Details section.



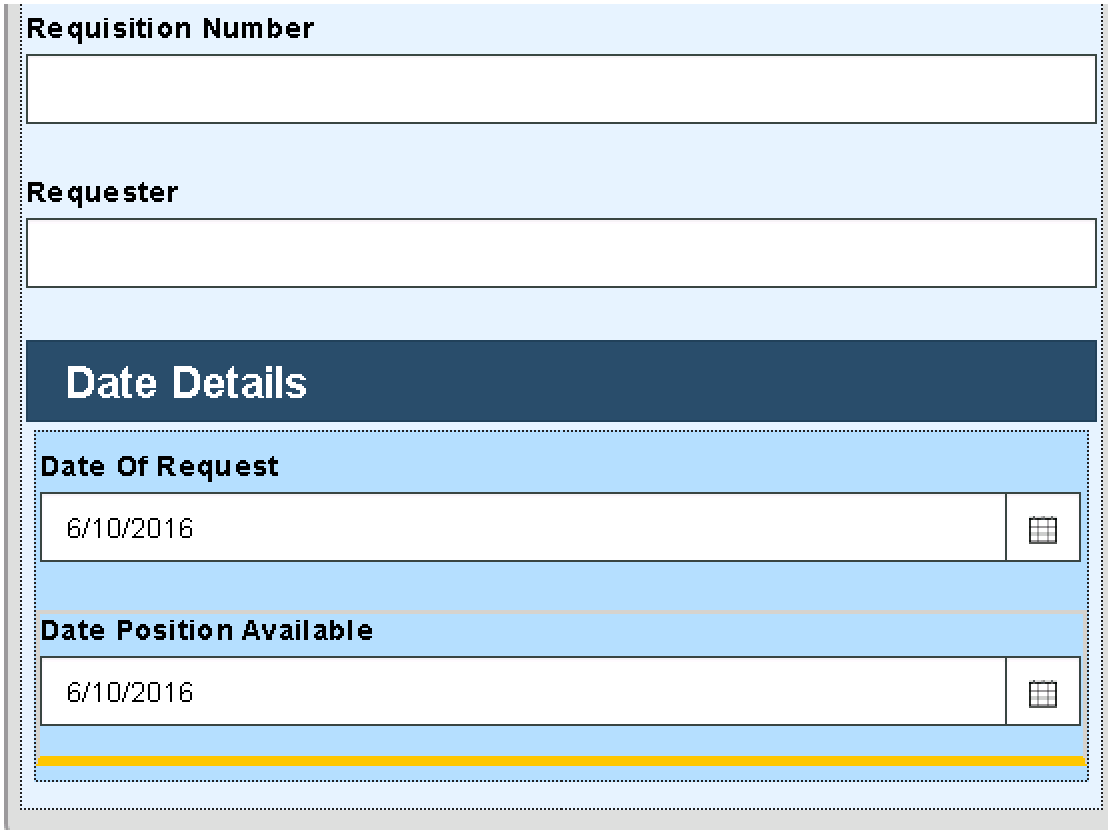
__ i. Drag the **Date Details** section that contains the **Date Position Available** and the **Date of Request** inputs into the editable area of the section under **Requester**.



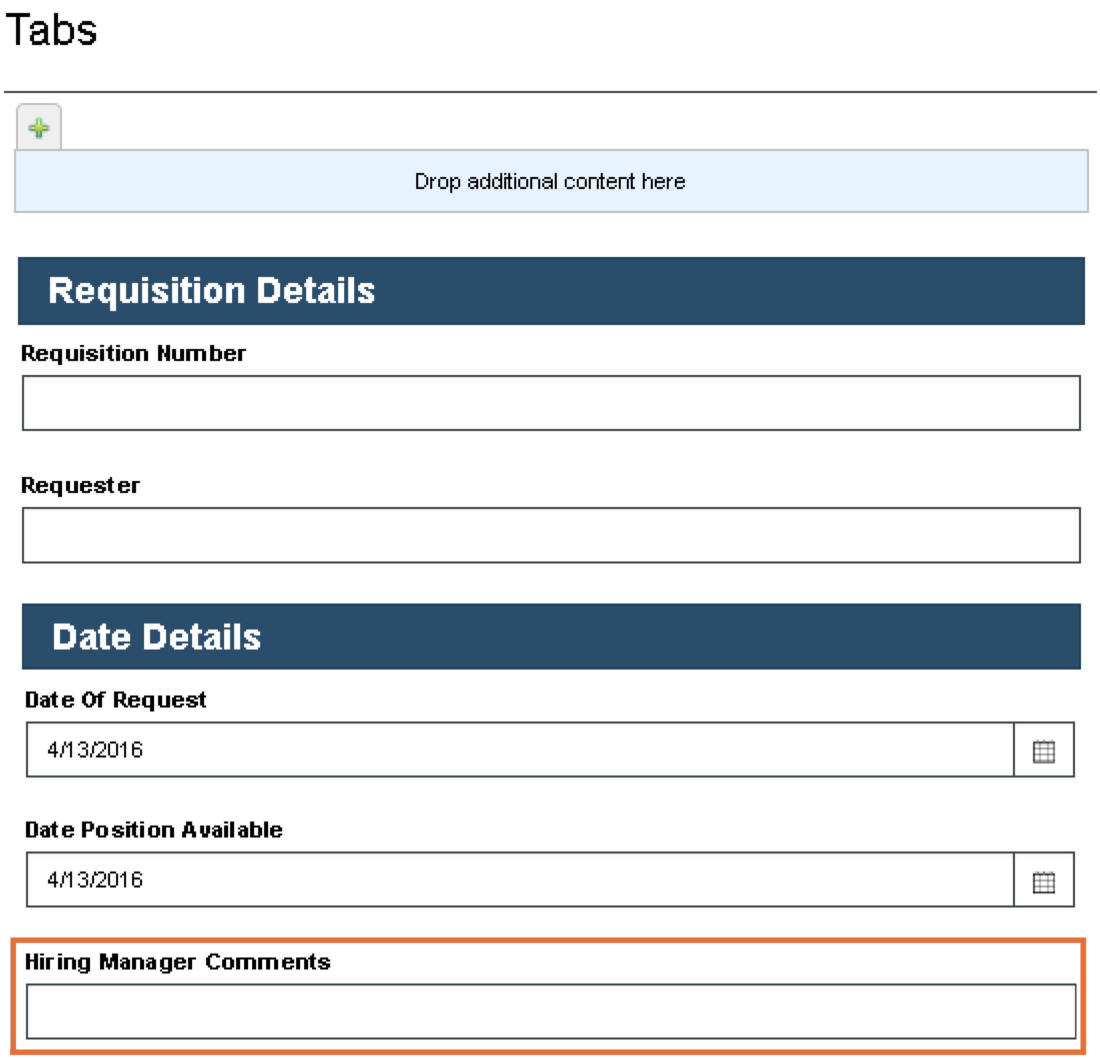
You added the Date Details section.



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



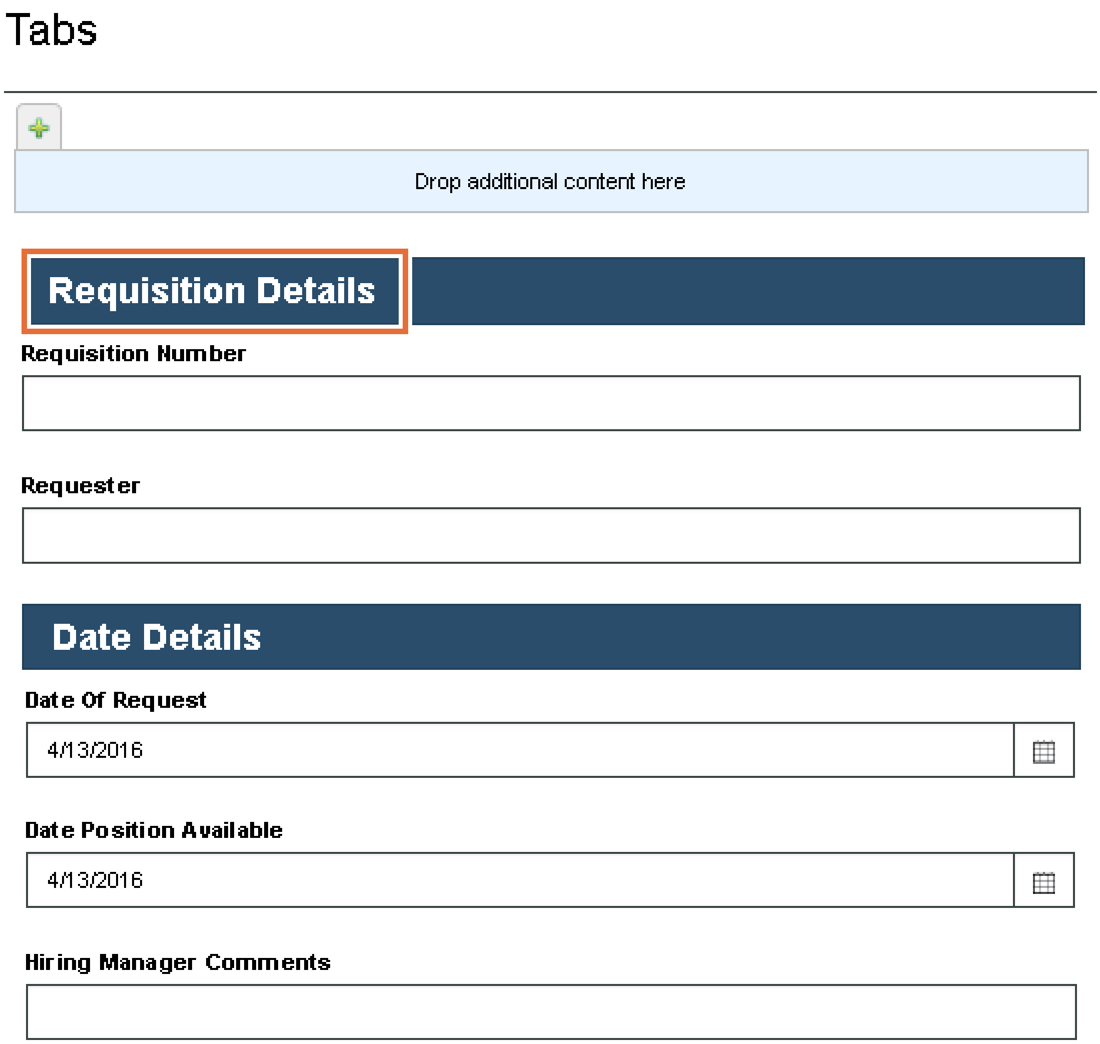
You added Hiring Manager Comments control.



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 and 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**.

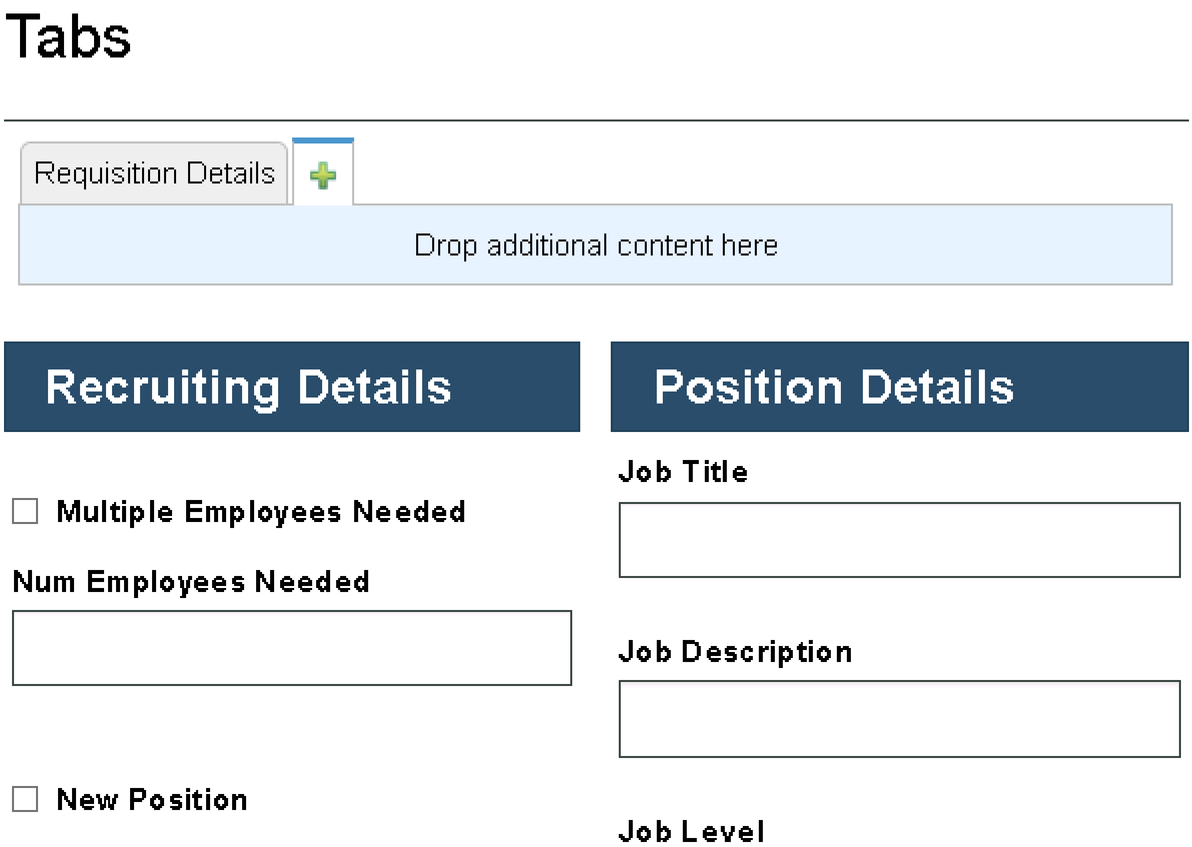


Hint

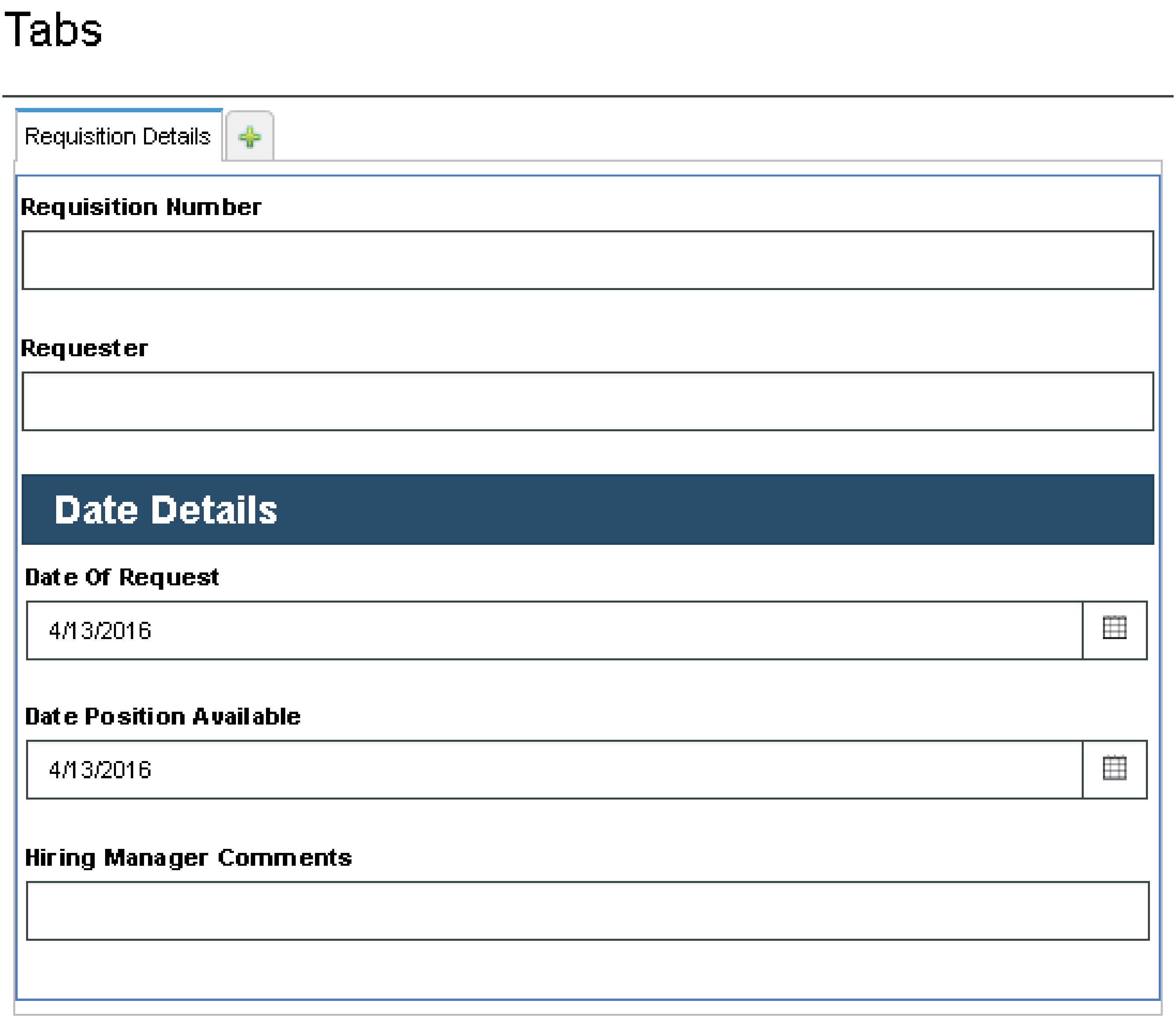
When you drag the Requisition Details header section, the input controls that are left behind were not part of Requisition Details section.

Note

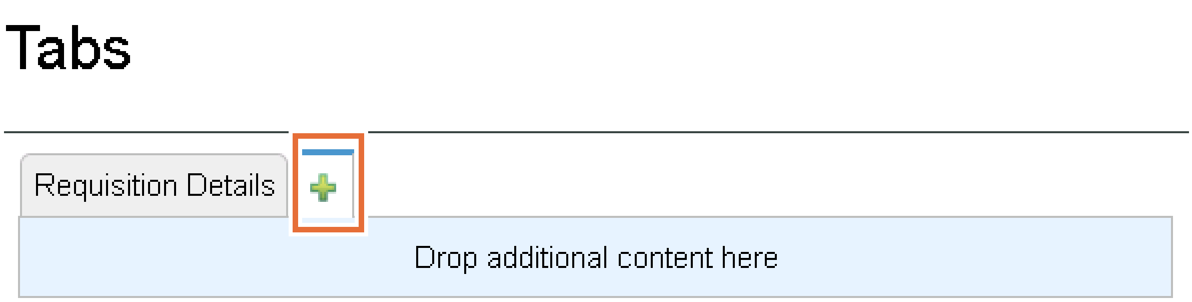
The label on the tab changes to match the label that is applied to the section, and a second tab opens.



__ l. Click to view the **Requisition Details** tab. The first tab is complete.



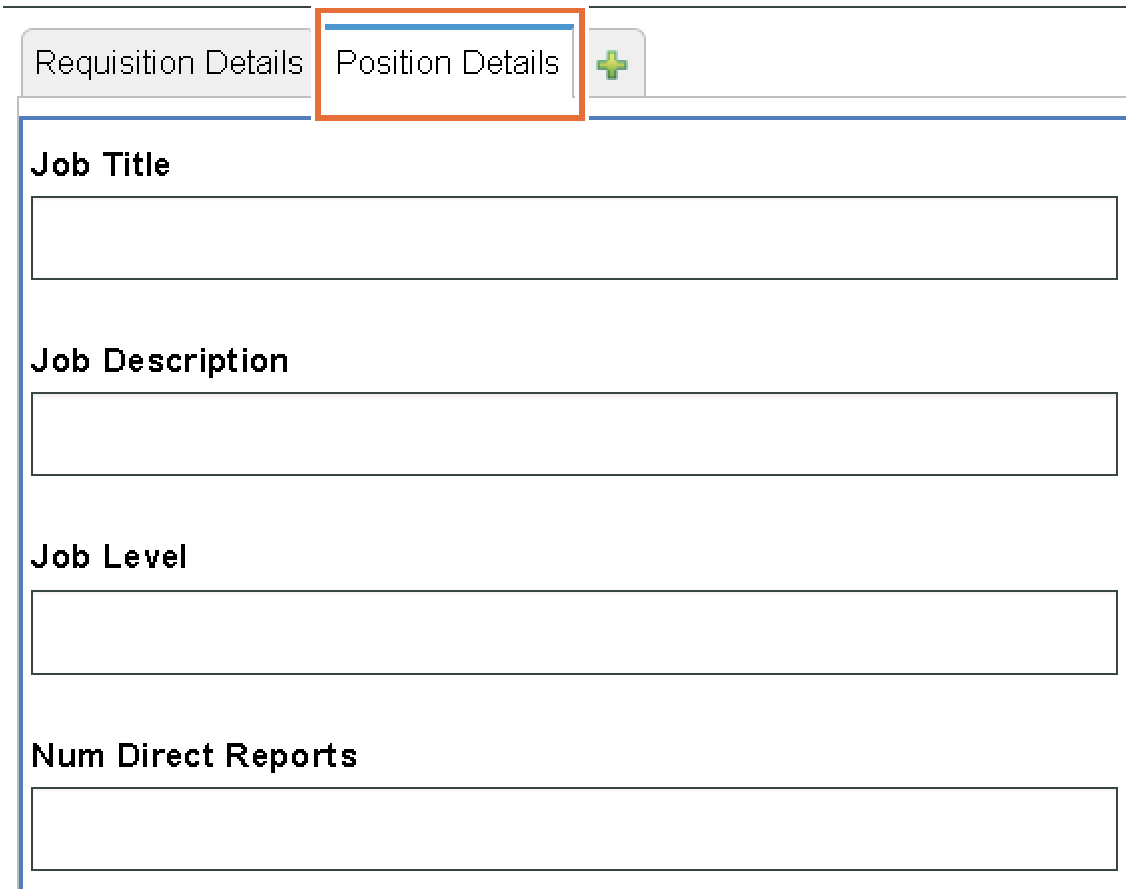
__ m. Click the (**+**) plus sign on the tab next to the **Requisition Details** tab.



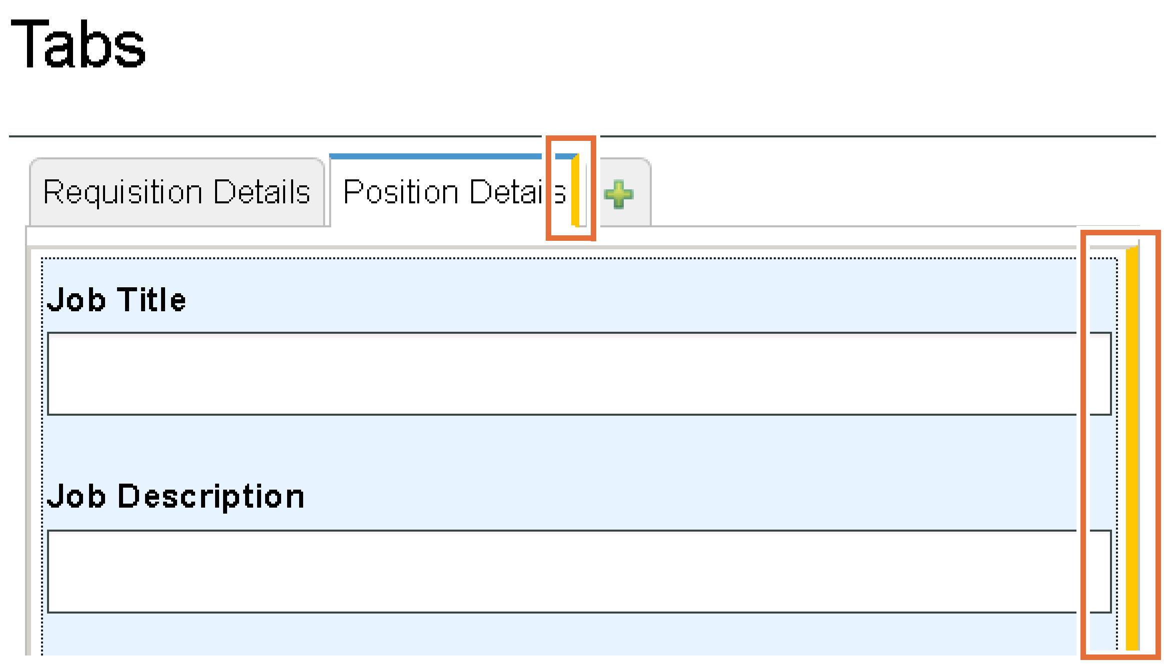
Note

The section that you added in the Requisition Details tab is still there, but that disappears as soon as you click other tab. That appears again when you click the Requisition Details tab.

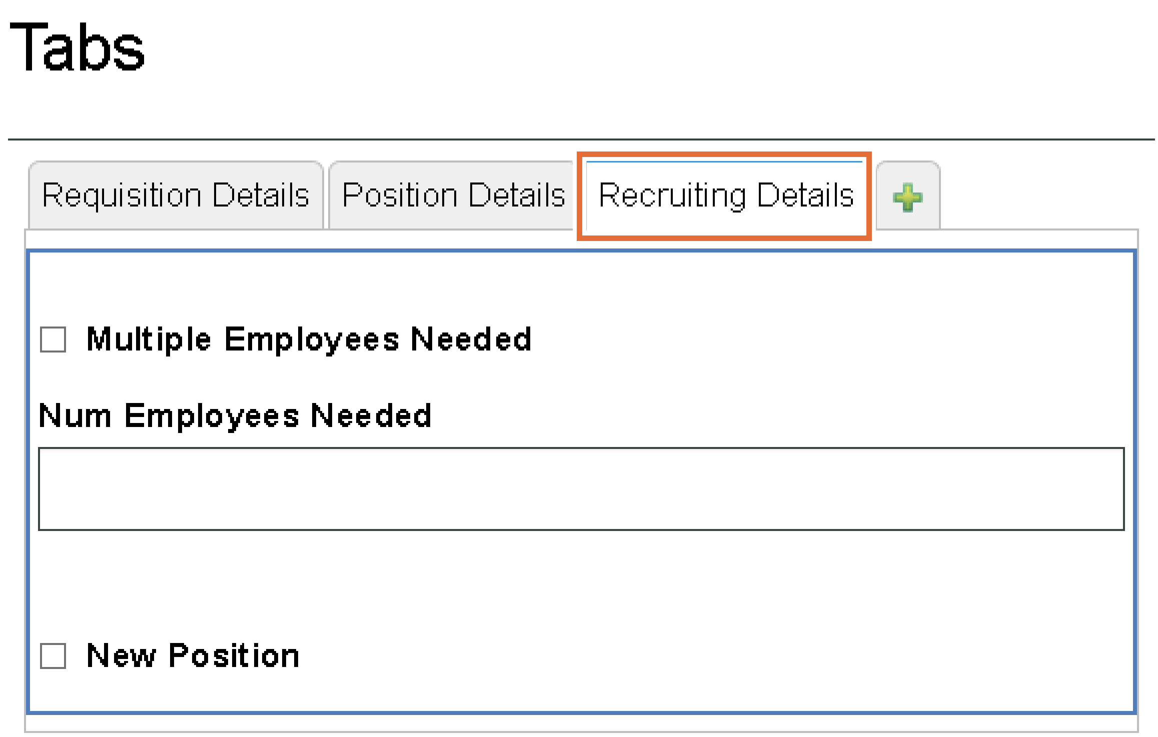
__ n. Move the **Position Details** section onto the tab. This one is easier because all the controls are already in a section.



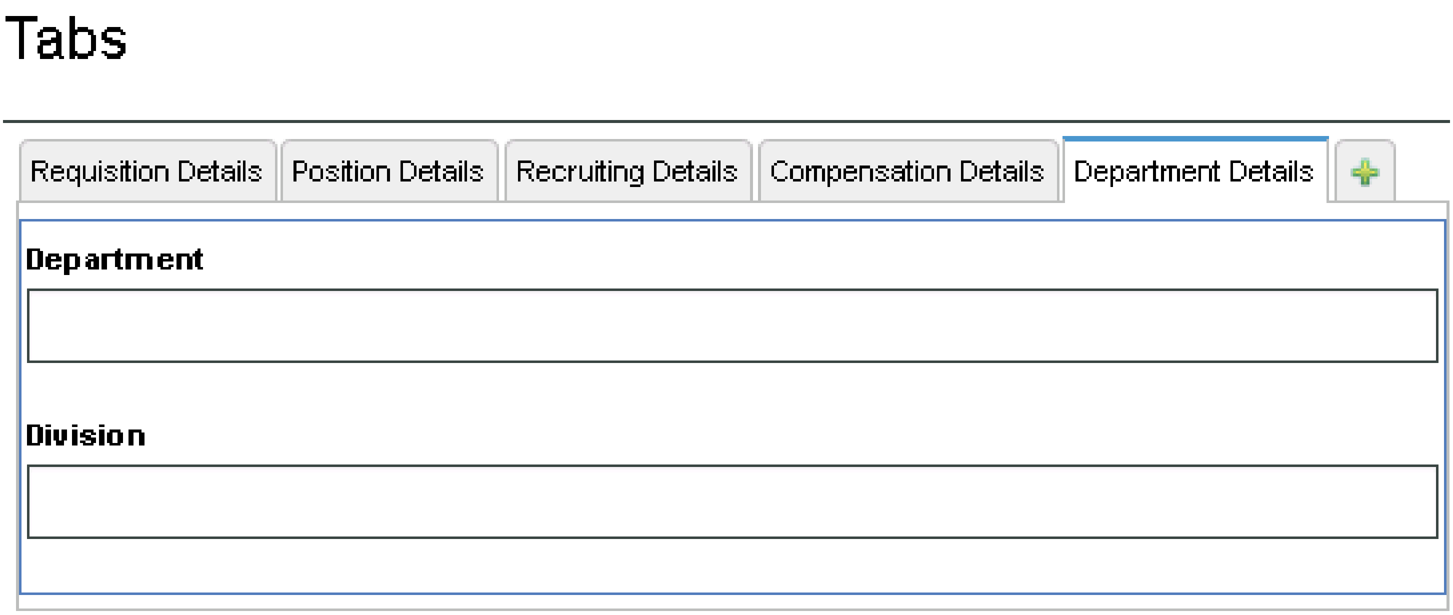
__ o. Create a tab for the **Recruiting Details**. This time, drag the header for the **Recruiting Details** section to the right of the `Position Details` tabs. Notice that an orange line indicates where the new tab is placed when you release the mouse button.



__ p. Release the mouse button, and the section becomes a tab next to the **Position Details** tab.



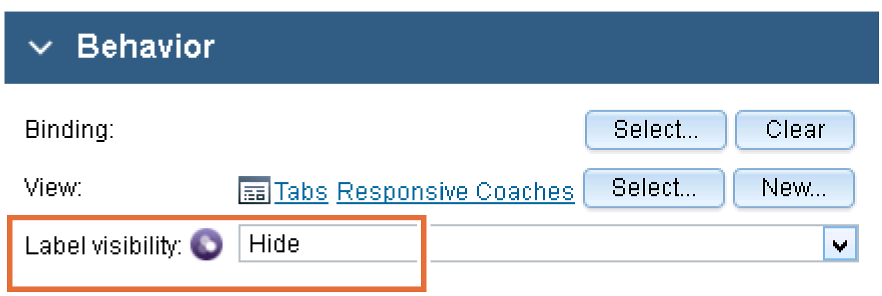
__ q. Continue to create tabs for the **Compensation Details** and **Department Details**. Drag each of the respective sections onto their tabs.



__ 2. Hide the Tabs label.

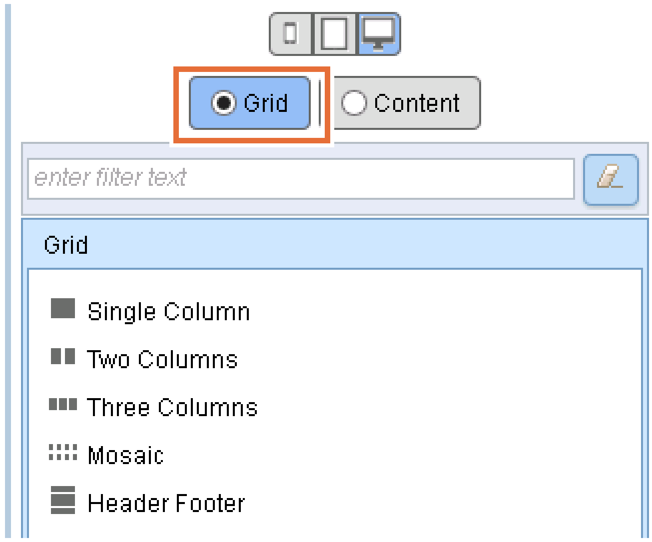
__ a. Click the **Tabs** label on the top.

__ b. In the **Properties > General > Behavior** section, select **Hide** for **Label visibility**.

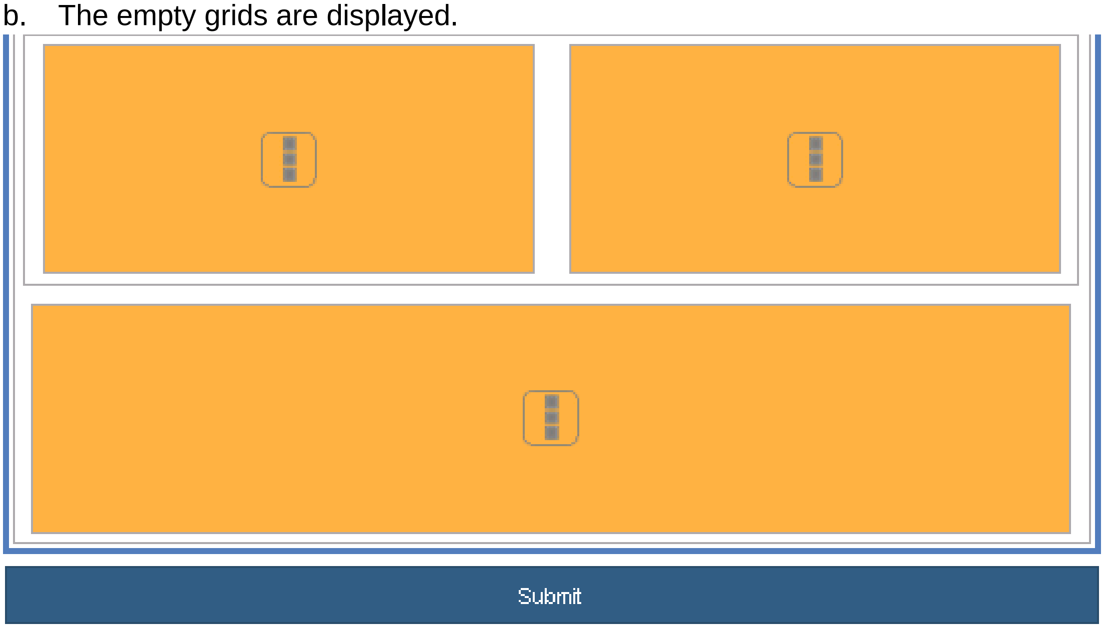


__ 3. Rearrange the empty grids.

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

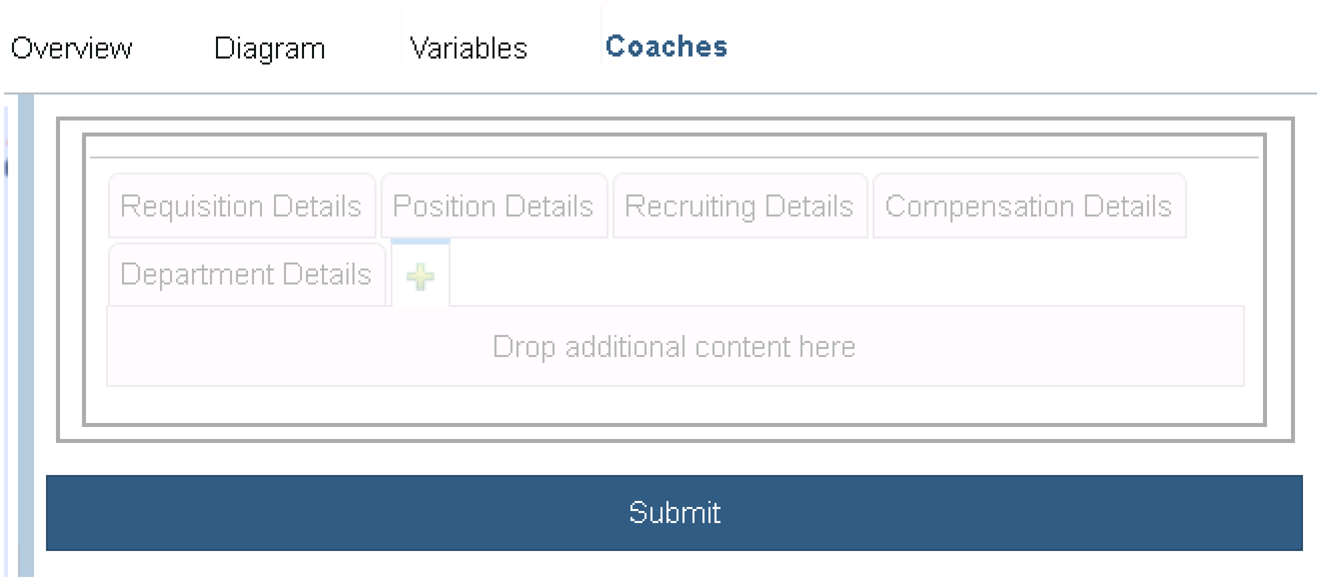


__ b. The empty grids are displayed.



__ c. Right-click each empty grid and click **Delete**.

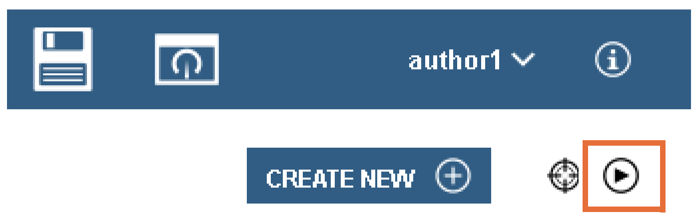
__ d. Now only the grid with the tabs is left with the Submit button at the bottom.



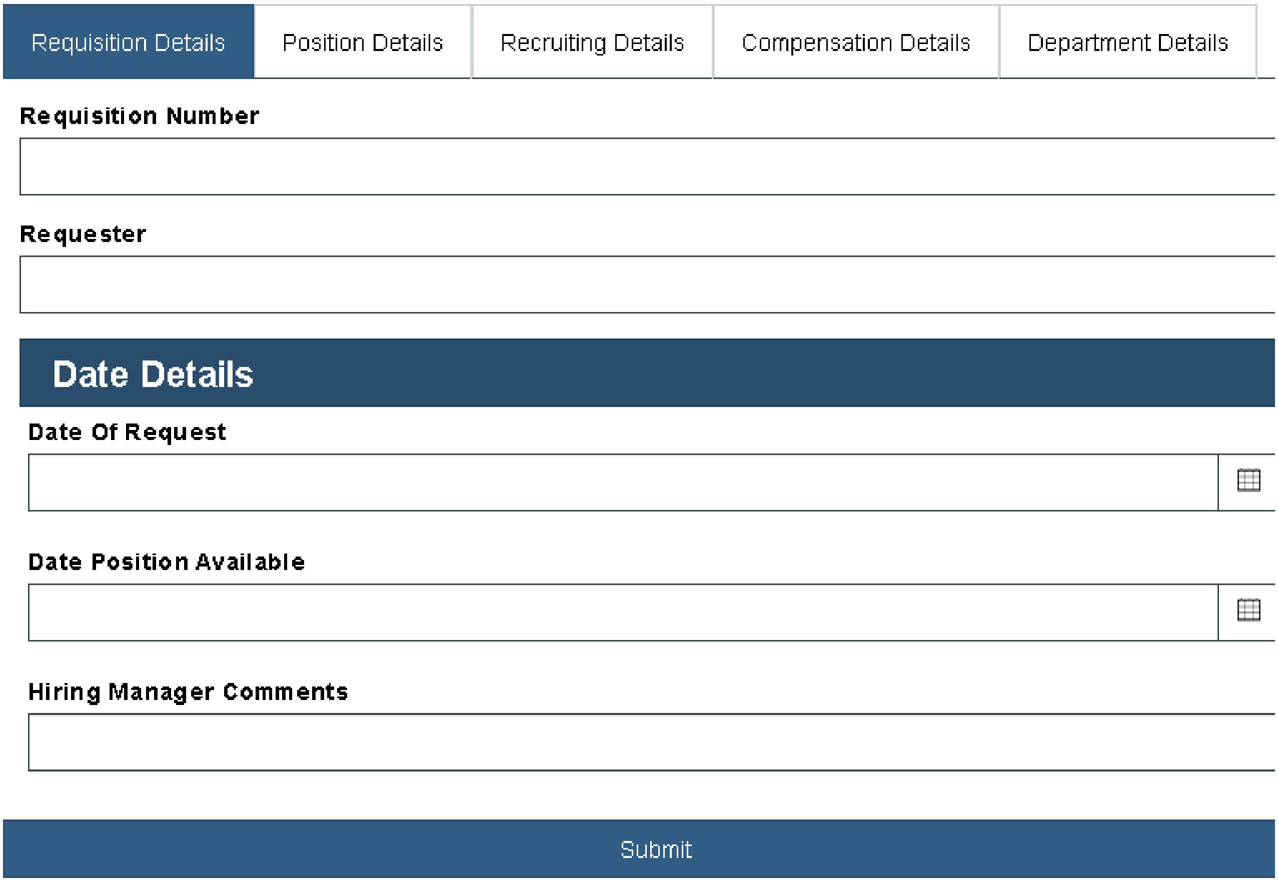
__ e. Save your work.

__ 4. 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 in each tab.



__ c. Close the browser.

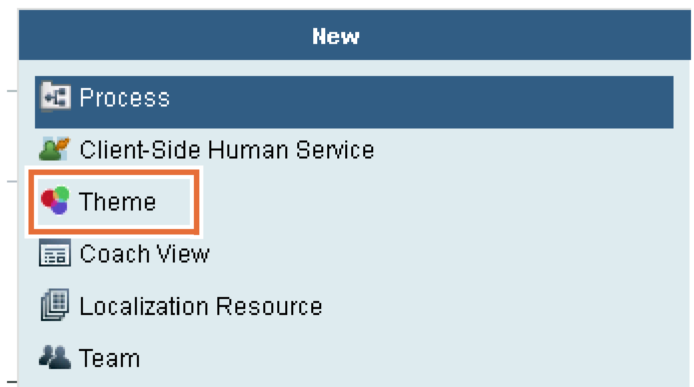
Part 2: Change the appearance of the coach by applying a custom theme

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 a complete set of 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.

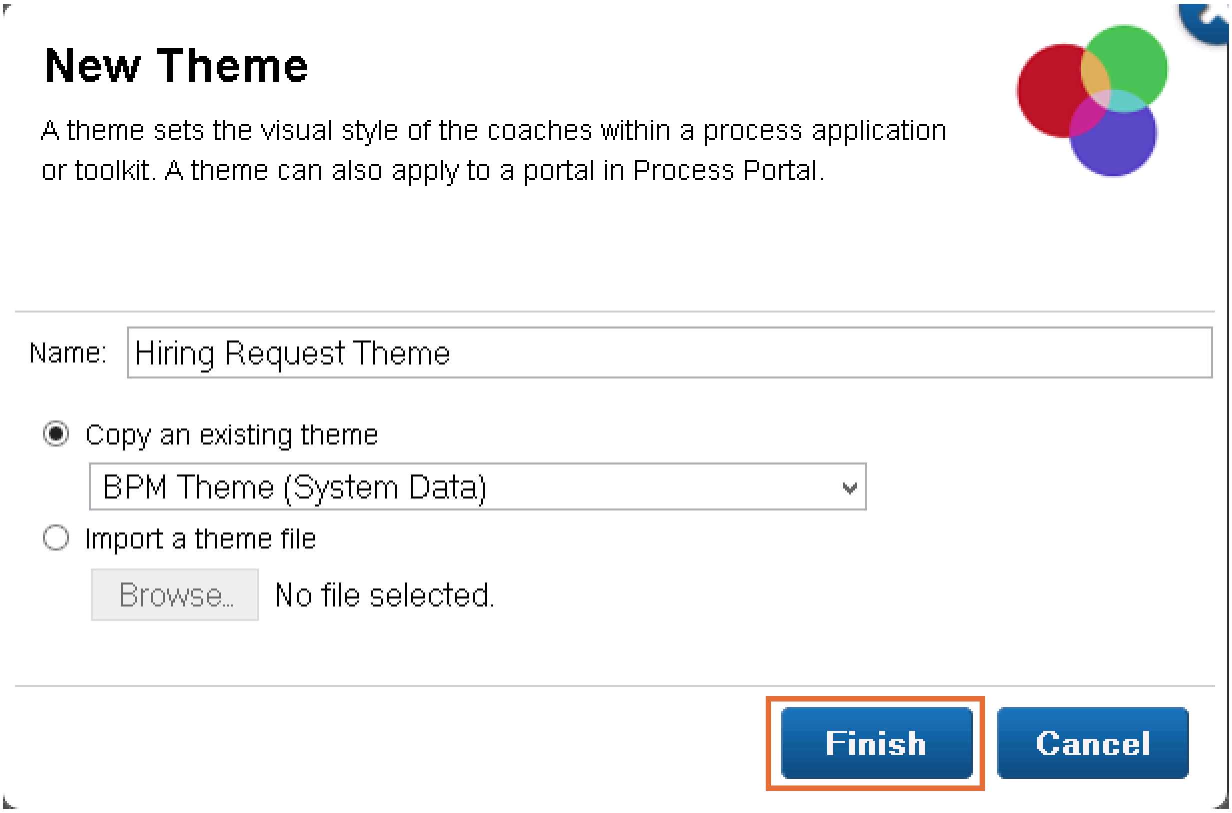
__ 1. Create a custom theme for the HR Recruitment Processes process application.

__ a. Click **Create New** on the upper-right corner of the web Process Designer.

__ b. Click **Theme**.



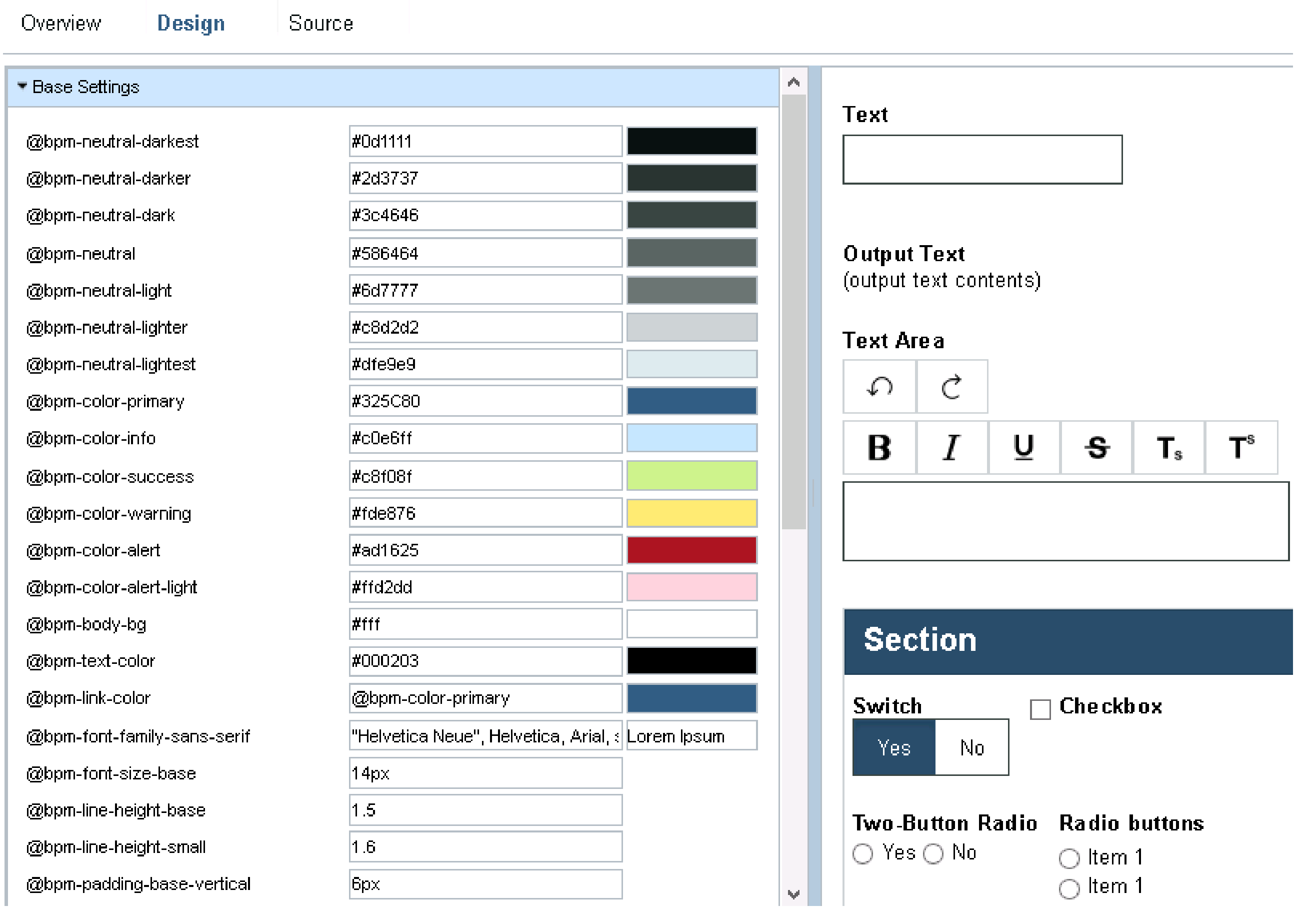
__ c. Enter the name: `Hiring Request Theme`
Leave the option **Copy an existing theme: BPM Theme (System Data)** selected and click **Finish**.



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.

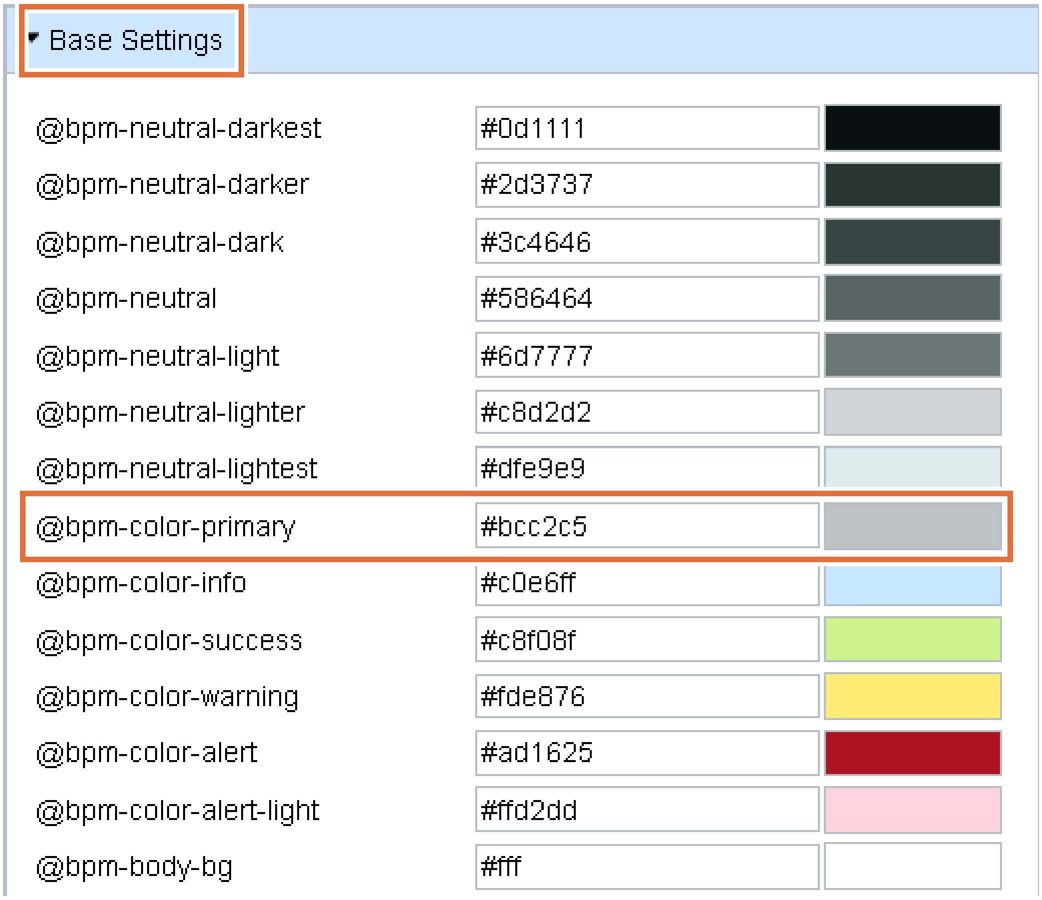
__ d. The editor opens the new theme.



__ 2. Change the primary color of the theme

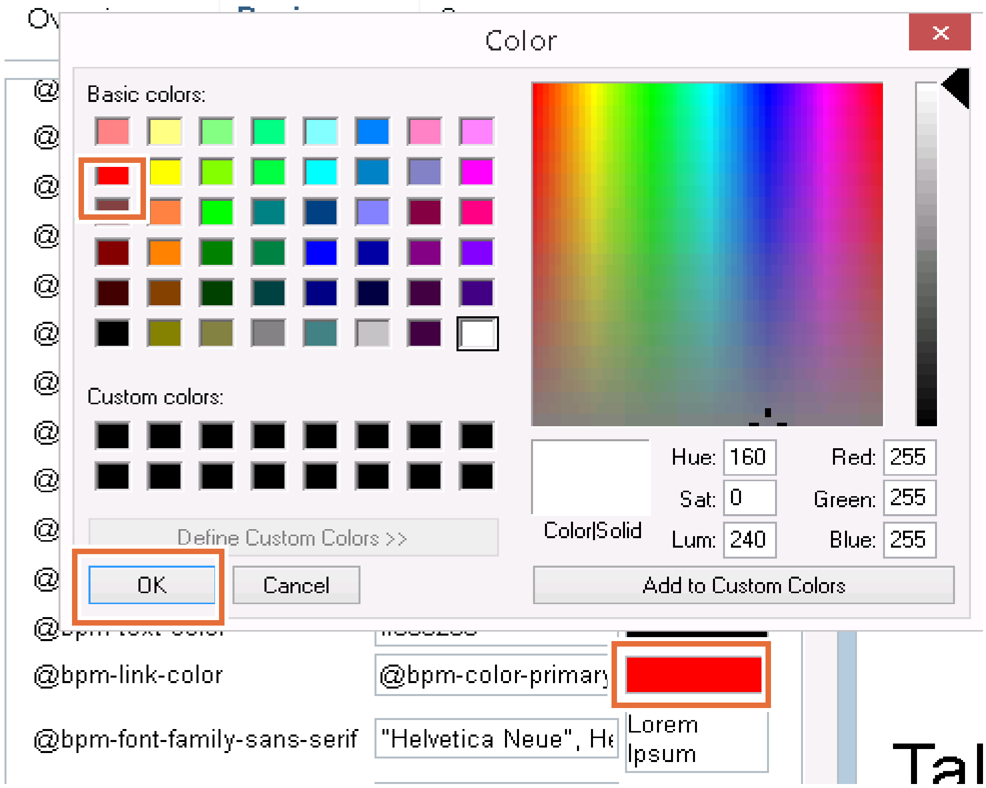
Management wants to change the color scheme to match the corporate color scheme. Change the responsive controls of the default BPM Theme and apply the new theme to the process.

__ a. In the Design page of the theme editor, change the variable `@bpm-color-primary` in the **Base Settings** section from `#325C80` to: `#bcc2c5`

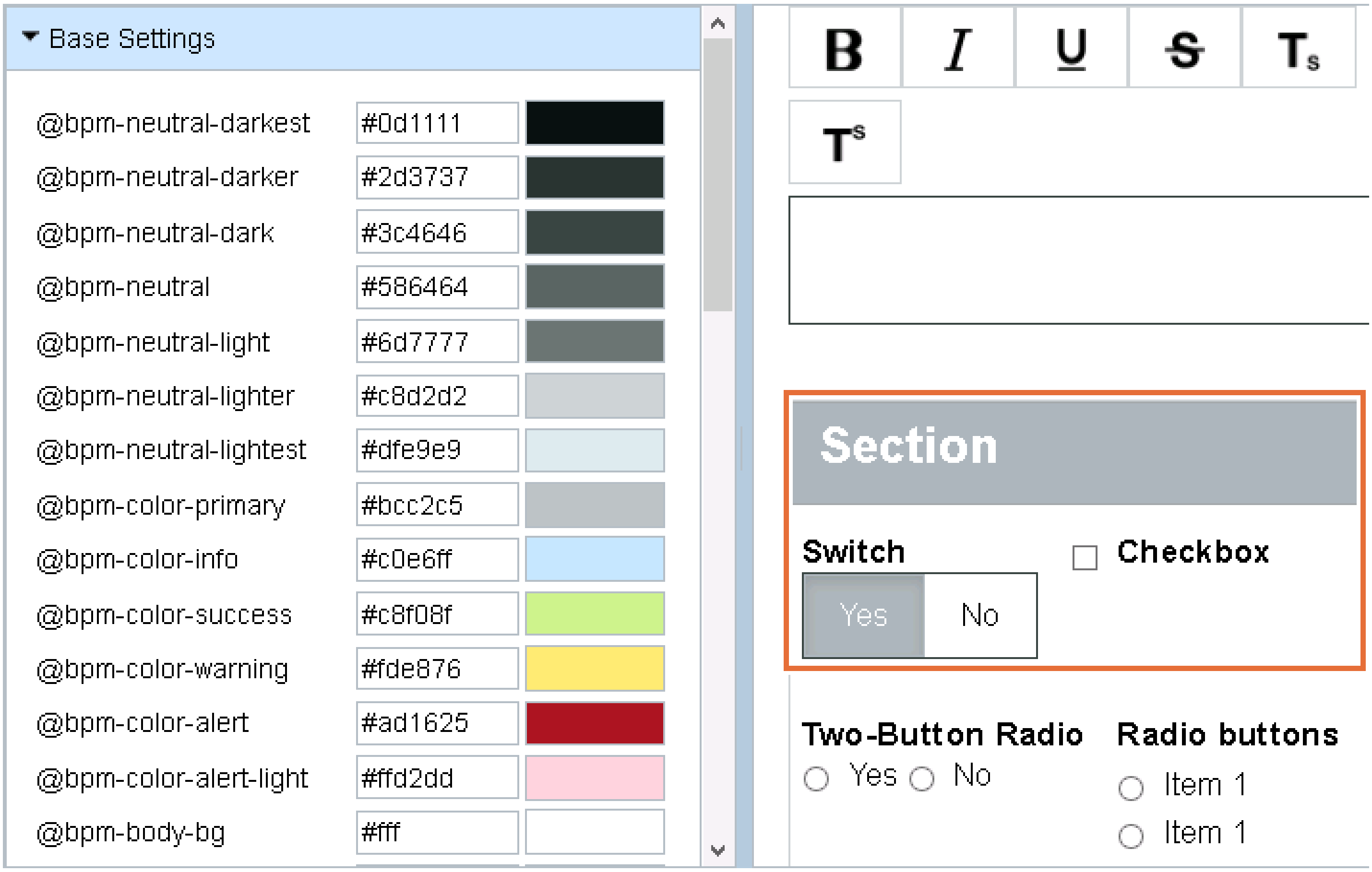


Note

Many variables also have a swatch that you can click, then choose a value directly from a picker, and click **OK**.

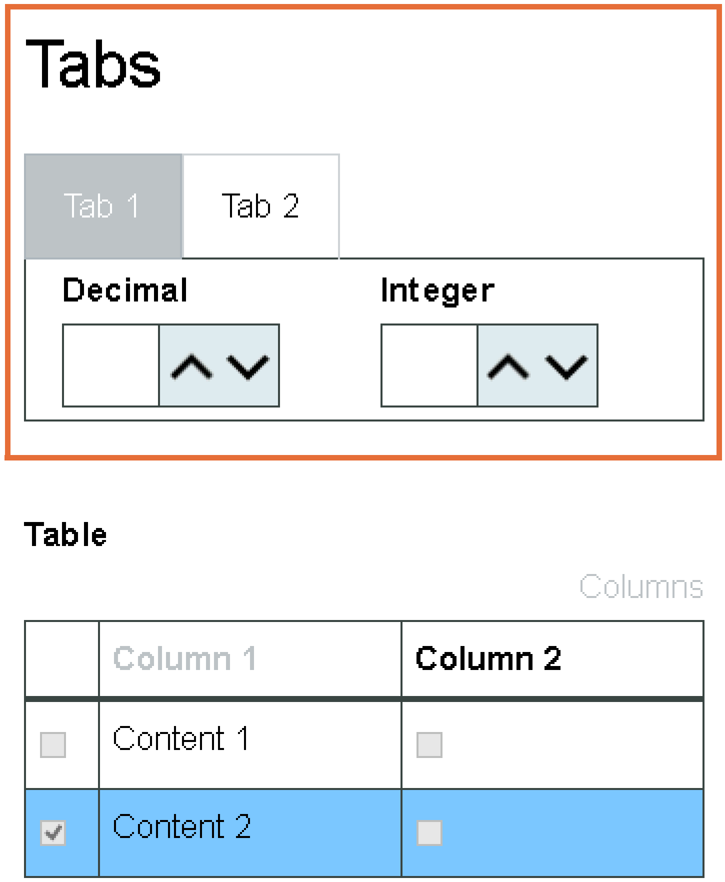


__ b. Click somewhere on the example window on the right, and you immediately see the color change for the affected controls.

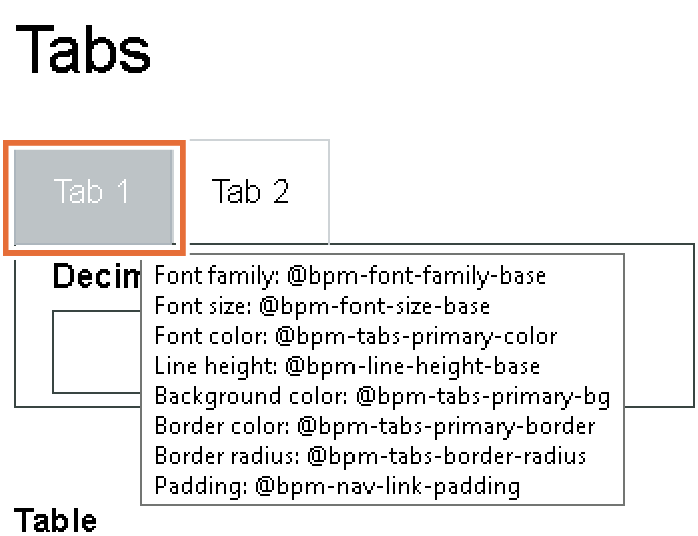


__ 3. Change the font color of the tabs.

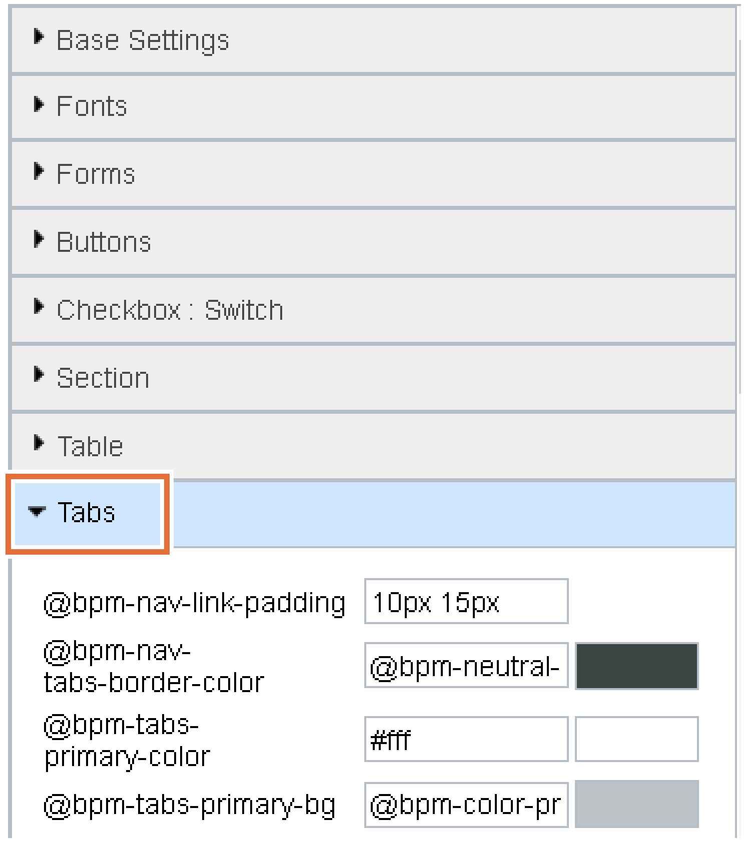
__ a. Scroll down the example frame on the right and look for the **Tabs** control.



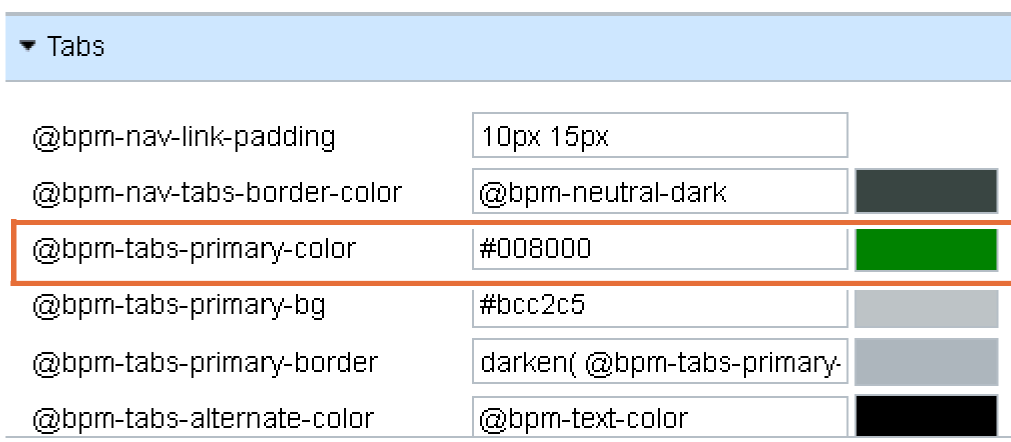
__ b. Hover over the **Tabs** control, and it shows all the variables that are associated with this control. For example, to change the font color of the tabs, you need to change the value of the `@bpm-tabs-primary-color` variable.



__ c. To change the font color, collapse the Base Settings section and expand the **Tabs** section.



__ d. Look for the variable `@bpm-tabs-primary-color` and change the color to: `#008000`



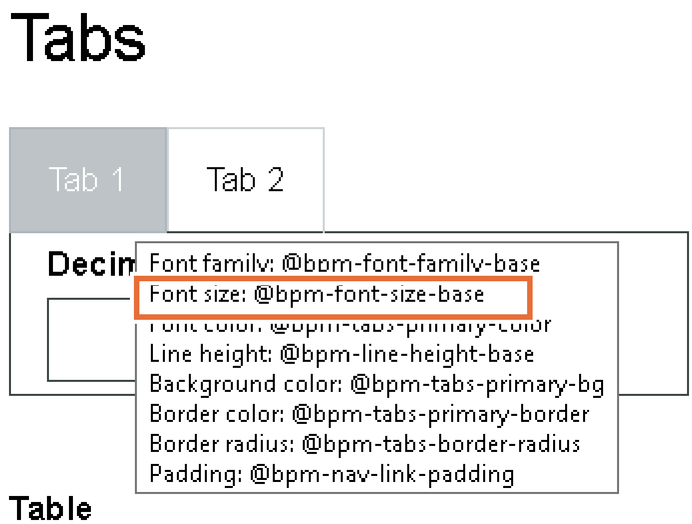
Hint

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 it from there.

__ 4. Change the font size.

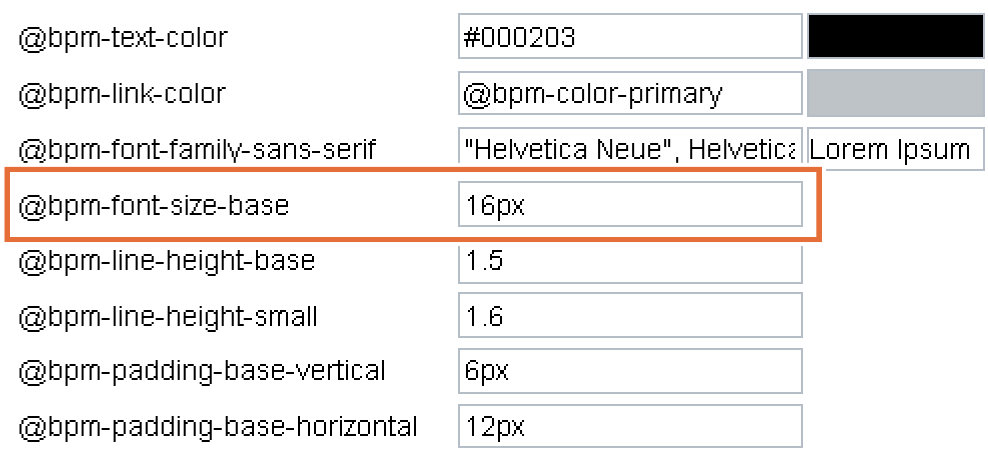
Hint

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.

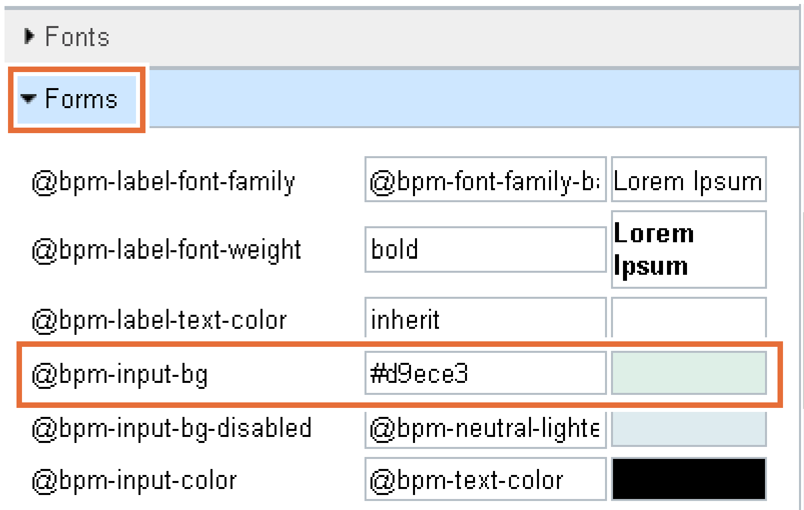
__ b. Look for the variable `@bpm-font-size-base` and change the value to: `16px`



__ 5. Change the background color of the input controls.

__ a. Expand the **Forms** section.

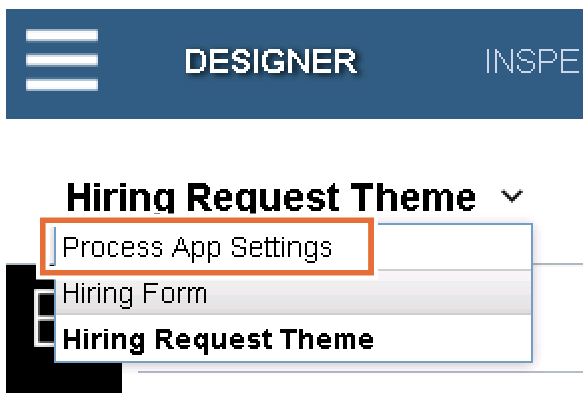
__ b. Change the variable `@bpm-input-bg` to: #d9ece3



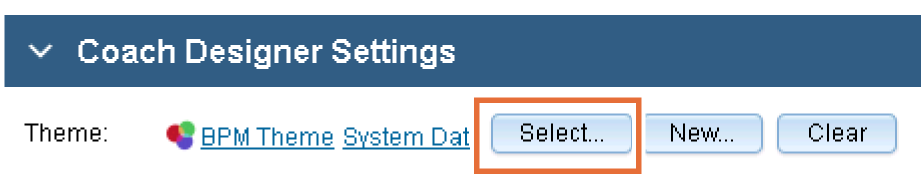
__ 6. Save your new theme.

__ 7. Apply the new theme to the `HR Recruitment Processes` process application.

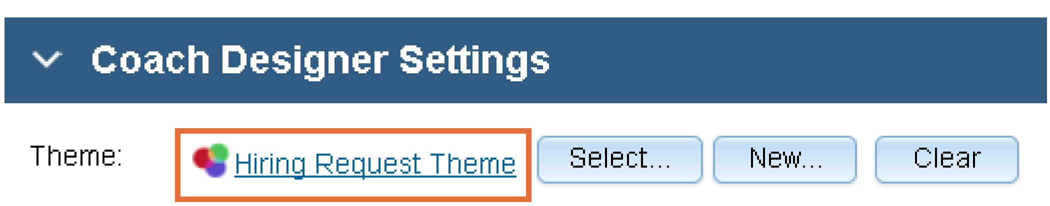
__ a. Open the **Process App Settings** from the History menu on the top.



__ b. In the Coach Designer Settings section, click **Select**.



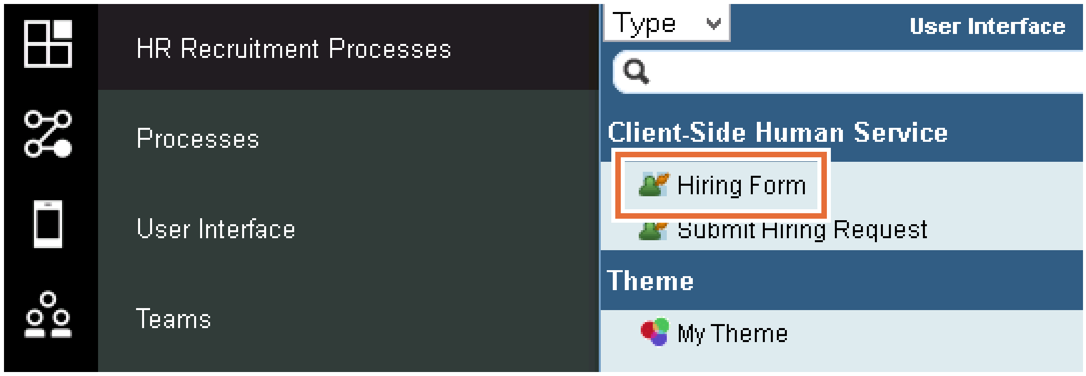
__ c. Select **Hiring Request Theme**. The theme is now changed to the new theme.



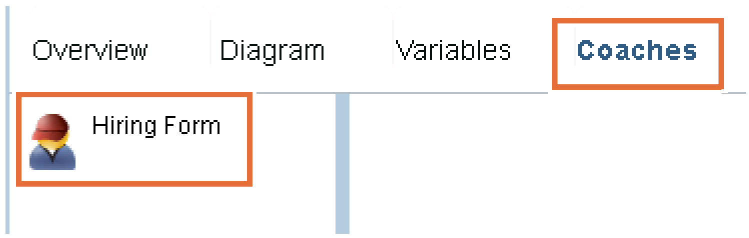
__ d. Save your changes. The save generates the CSS that the browser uses to display the controls in the layout.

__ 8. Verify the theme changes

__ a. Open the **Hiring Form** Client-Side Human Service.

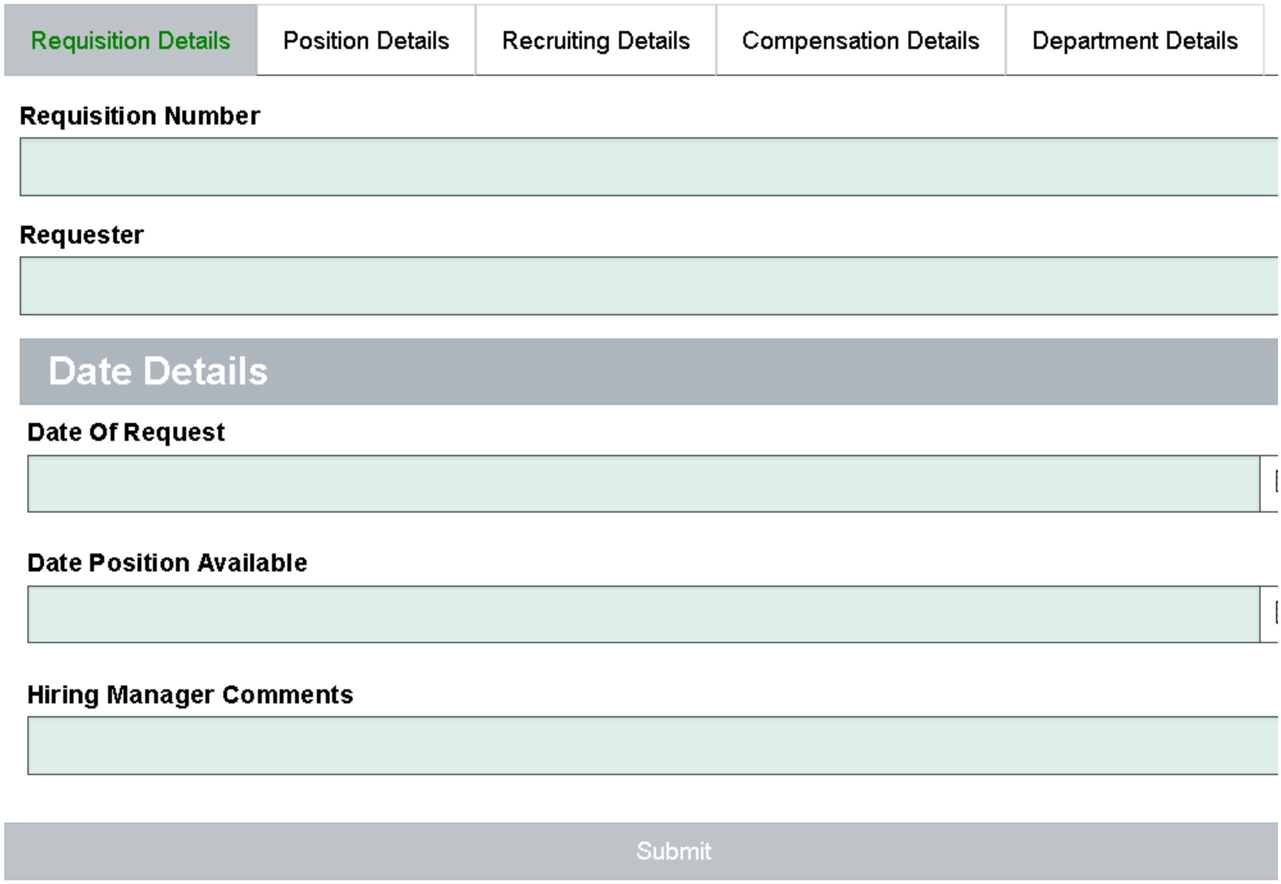


__ b. Click **Coaches > Hiring Form**.



__ 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.



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.

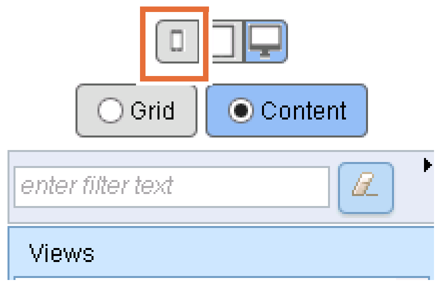
__ d. Close the browser window.

Part 3: 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 different screen size formats.

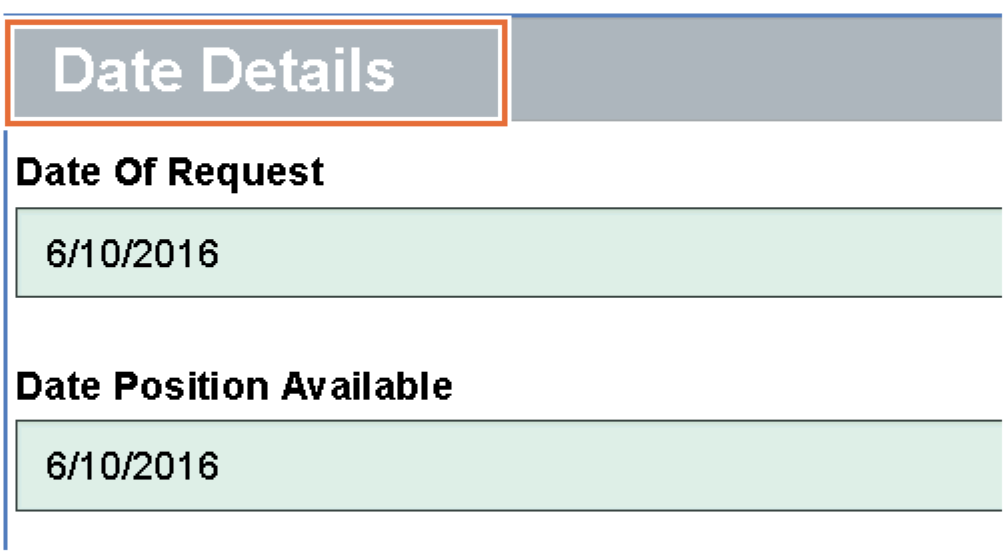
__ a. Return to the **Coaches** tab in the service and click the **small screen** icon above the palette.



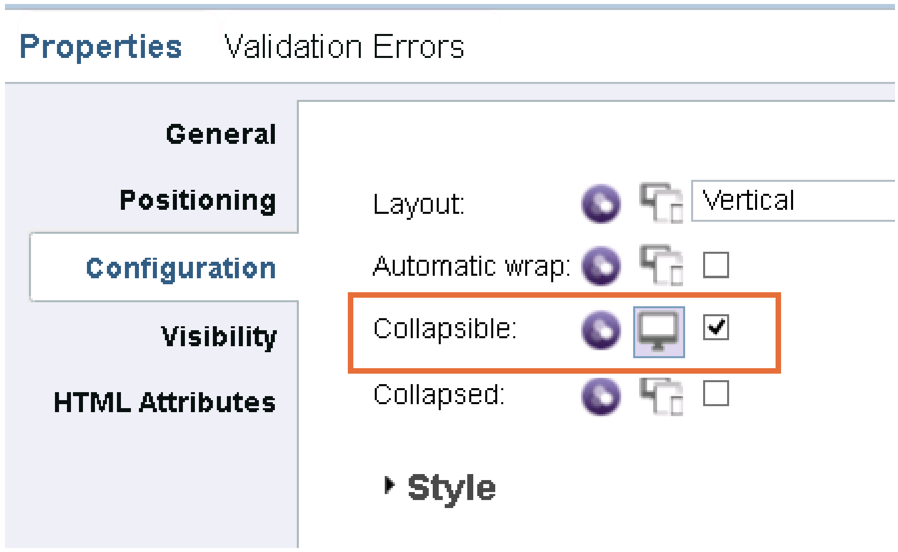
Note

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.

__ b. In the Requisition Details tab, click the **Date Details** section header.

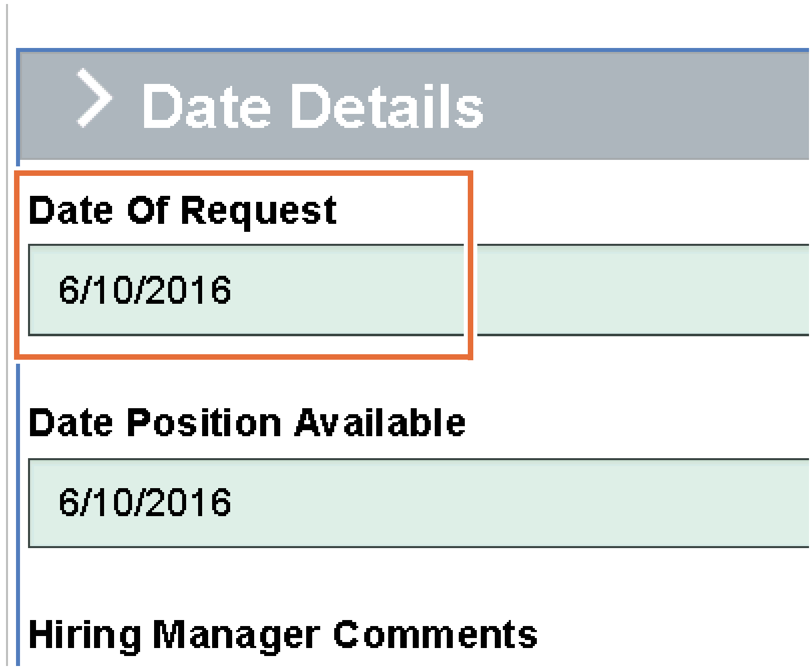


__ c. Click the **Properties > Configuration** menu. Select **Collapsible**.

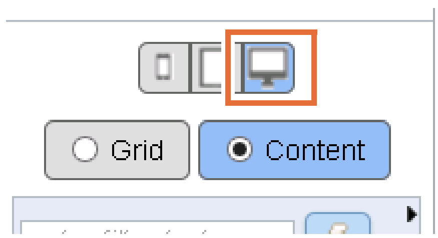


__ d. You immediately see that the collapsible icon is added next to the **Date Details** section. You can collapse and expand your Date Details section by using this icon.

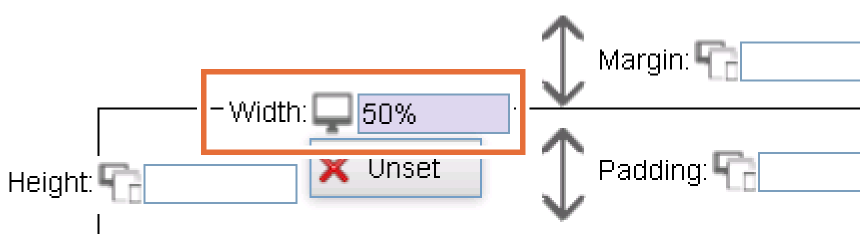
__ e. Select the **Requisition Details > Date Details > Date of Request** input control.



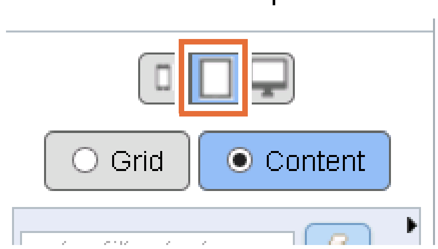
__ f. Click the **large screen** icon above the palette.



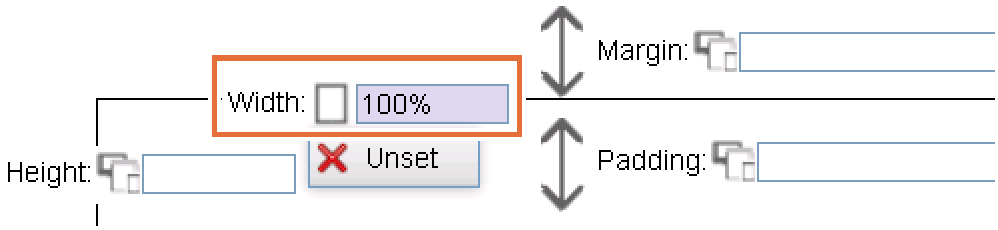
__ g. Set the **Properties > Positioning > Width** to 50%. The entered width is applicable only for large screen devices.



__ h. Click the **medium screen** icon above the palette.



__ i. Set the **Properties > Positioning > Width** to `100%`. Click the **select screen** icon next to Width to display the medium screen icon. The entered width is applicable only for medium screen devices.

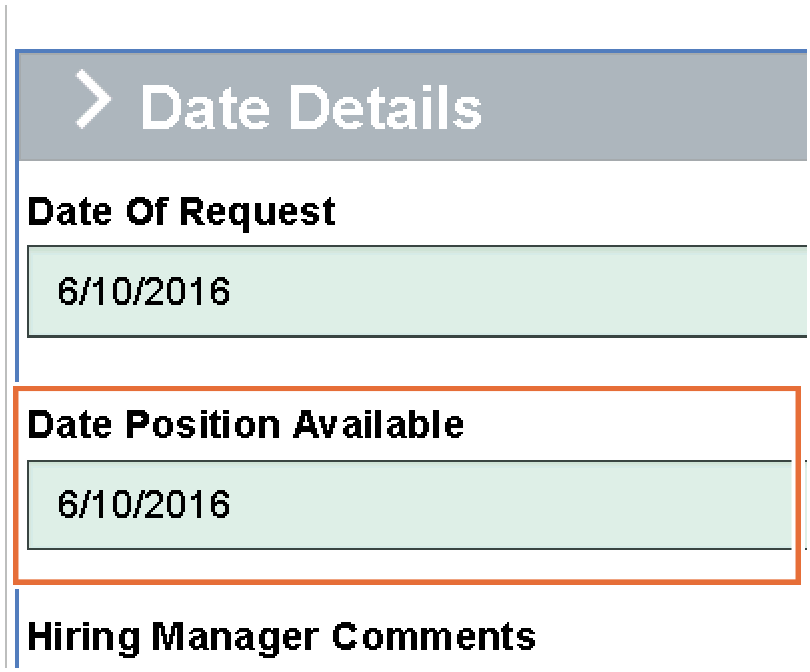


Note

When you set the tablet setting (medium screen), the small screen automatically is set to `100%` and you see that it is inherited from the tablet format setting, so you do not have to set it to `100%`.

__ j. Save your work.

__ k. Select the **Requisition Details > Date Details > Date Position Available** input control.

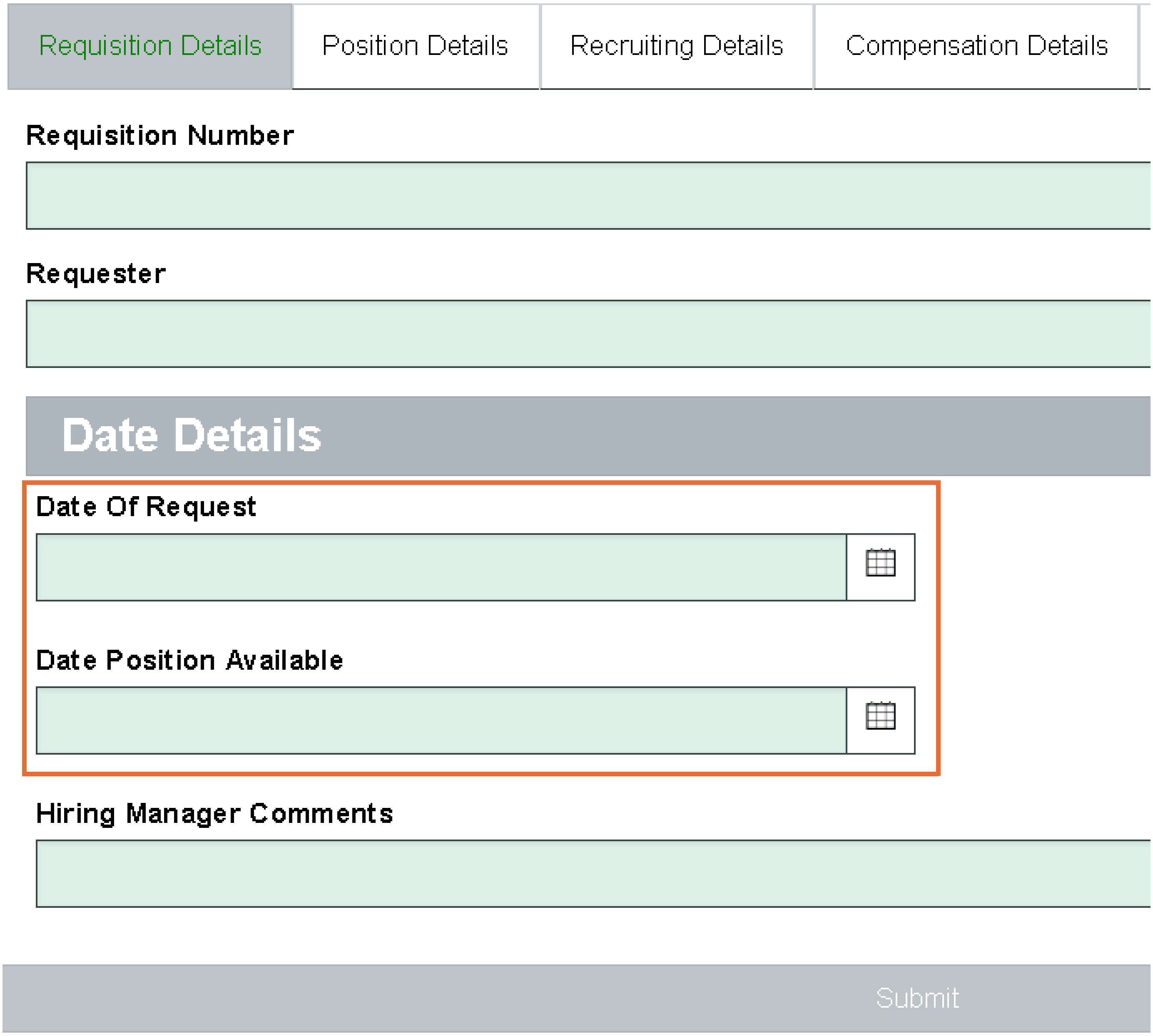


__ l. Repeat the steps Part 3, 1.f to 1.j to set the width of the **Date Position Available** input control for large, medium, and small screen devices.

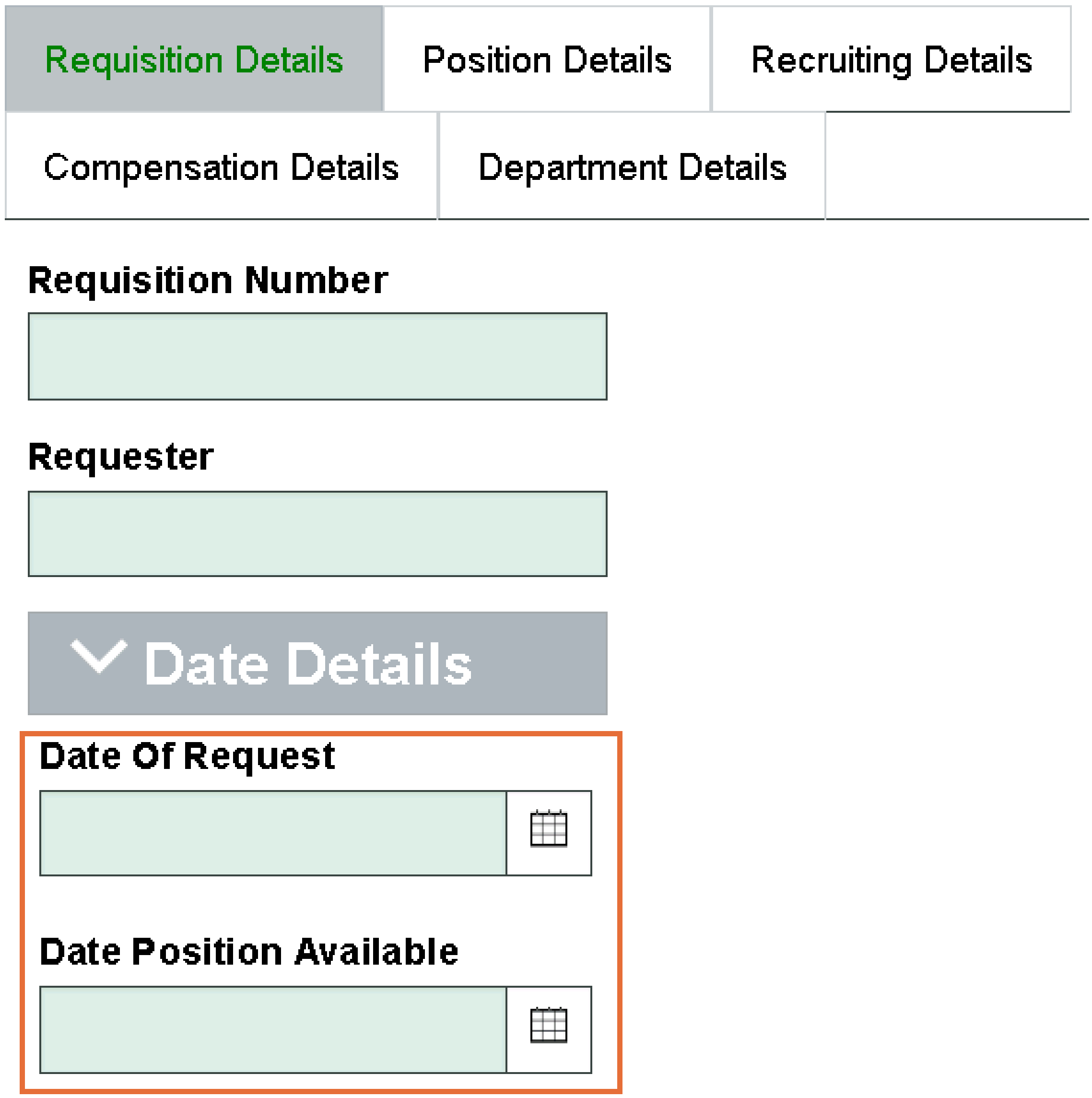
__ 2. Verify the coach display for different screen sizes

__ a. Click the **Run** button.

__ b. A new browser window opens with the coach displayed. Either expand the width of the browser to the full width of your screen, or maximize the pop-up browser window to see the **Date of Request** control. This control is reduced to `50%` width because you set this width for the large screen (default) devices.



__ c. Slowly reduce the width of the browser. The **Date Details** section changes, and the **Date Of Request** control in the section changes its width to `100%` when the browser width passes the threshold between the tablet and desktop browser window width setting.



__ d. Close the browser window.

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?

If you need more information about conducting the playback, consult 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

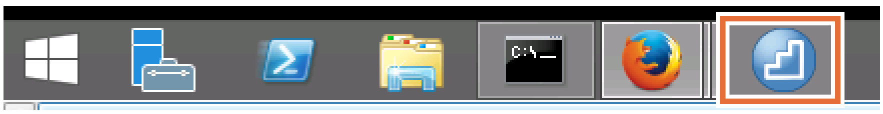
Part 1: Demonstrate the Review Needed path

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.

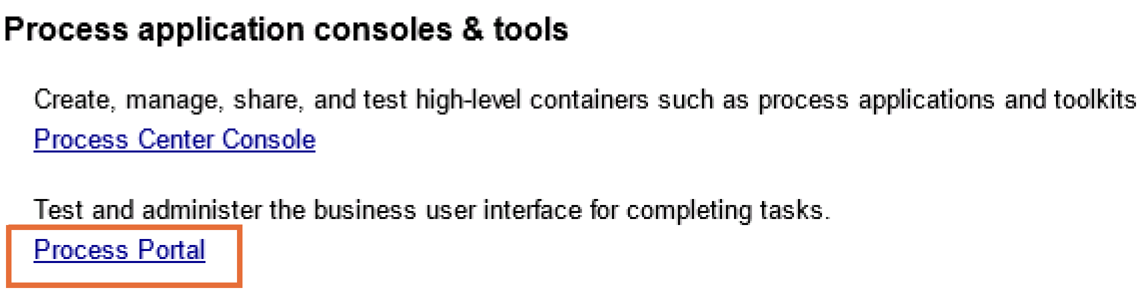
You accomplish a process playback session just as you would do when you seek consensus to move the project to the next playback phase. You log in as different users of the process to demonstrate tasks that different users accomplish in the process, demonstrating coaches that you created in the current 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.

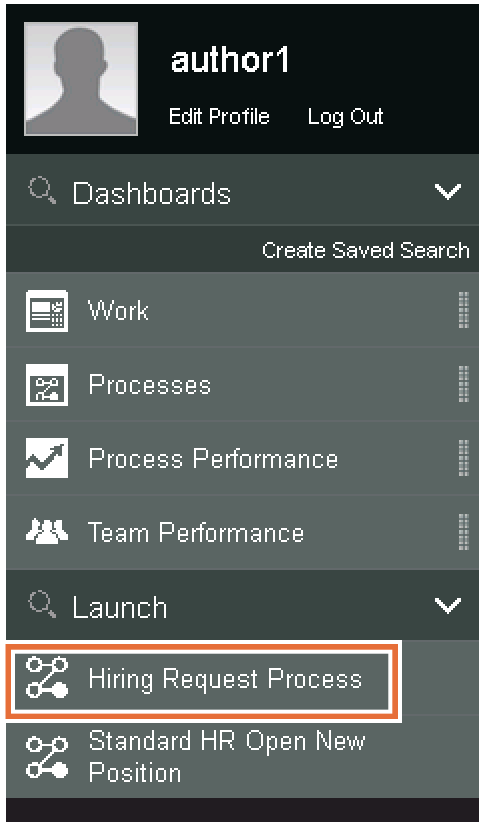


Note

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.

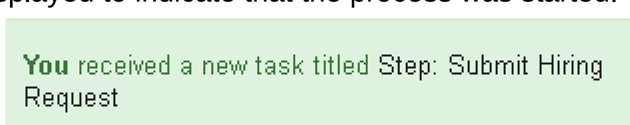
__ a. In the left frame, click **Hiring Request Process** to start an instance of the process.



Note

If the library is not displayed, click the **library** icon on the left side of the Work tab.

__ b. A notice is displayed to indicate that the process was started.

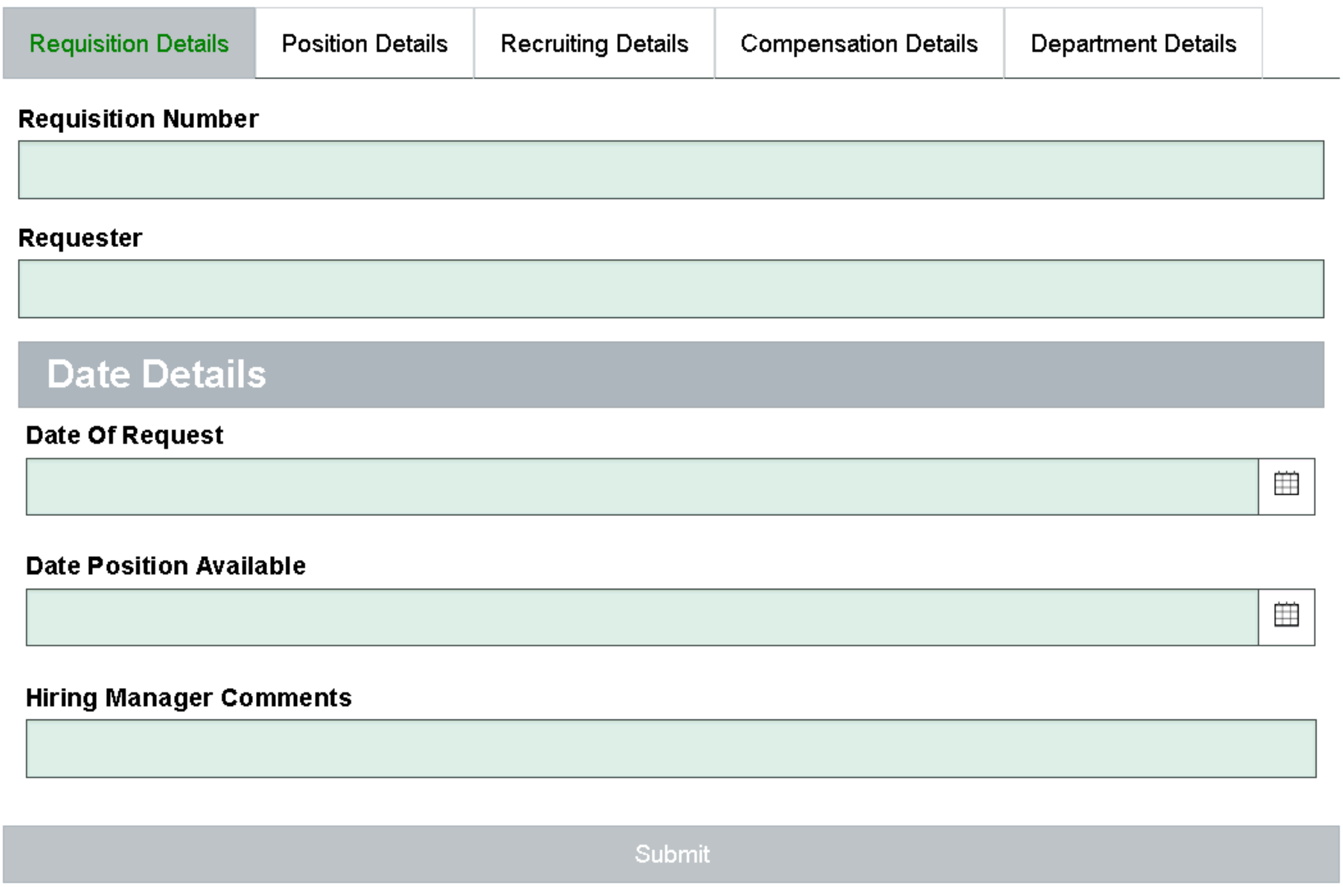


Information

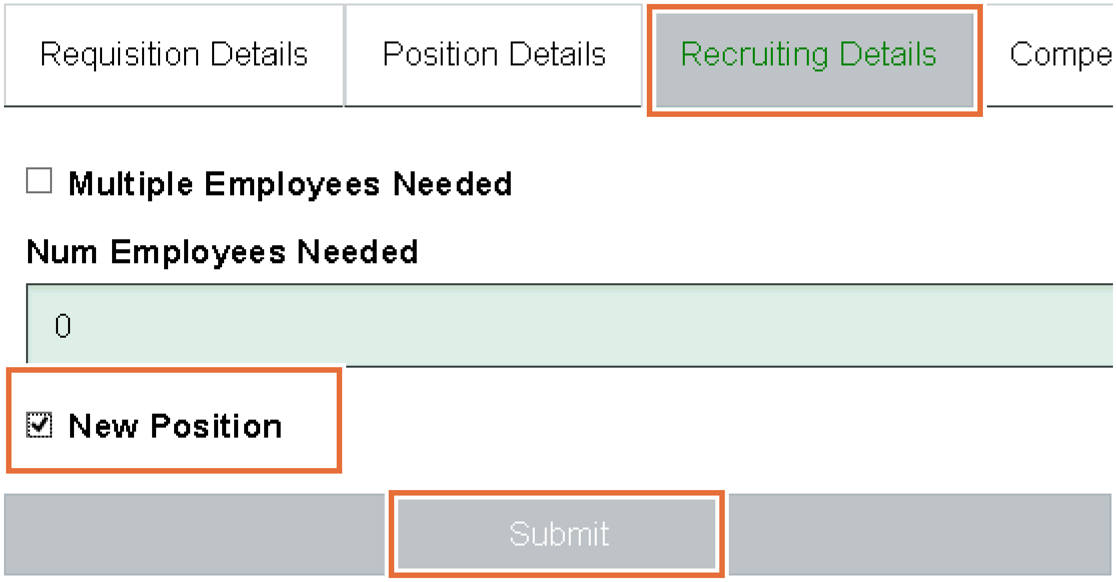
The first task starts automatically in the portal because you assigned the first task to **Lane** and **Last User**. If the task does not start automatically, return to the steps in Exercise 5, Part 5.1 to configure the first activity to start automatically.

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.

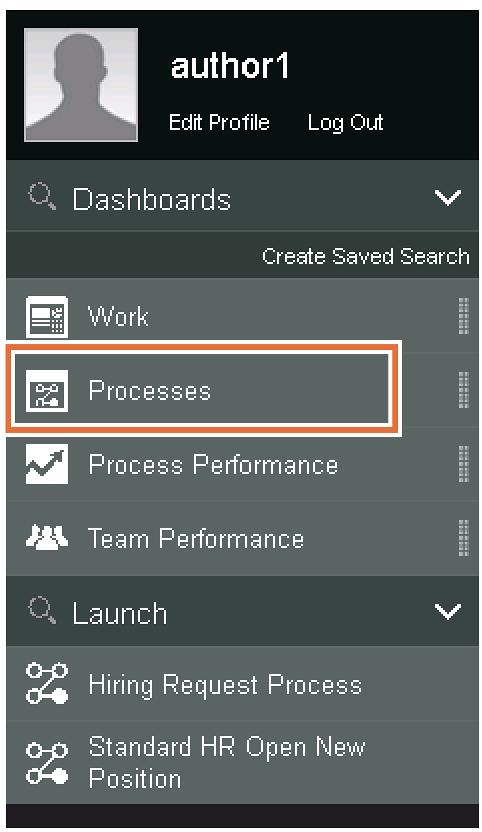


__ c. Select the **Recruiting Details** tab, and enable the **New Position** check box so that it follows the Review Needed path. Click **Submit**.

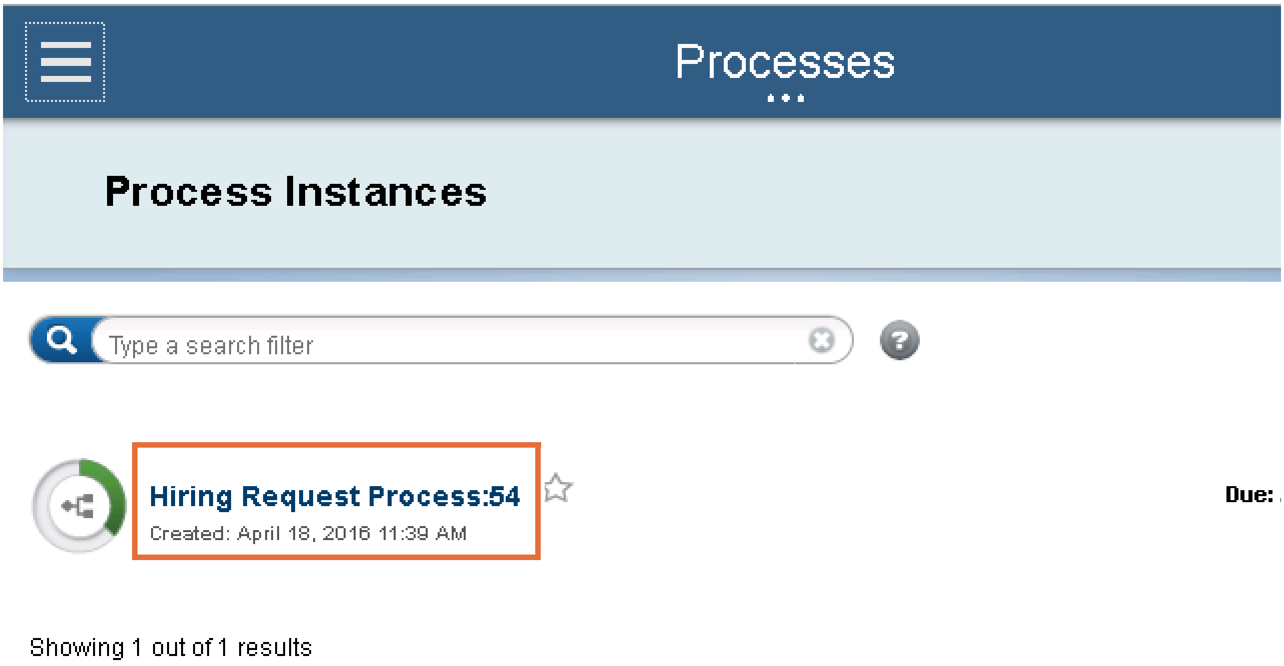


__ 3. Show the task that is assigned to the next activity, `Approve New Hire Request`, in the process.

__ a. In the Library click **Processes**.



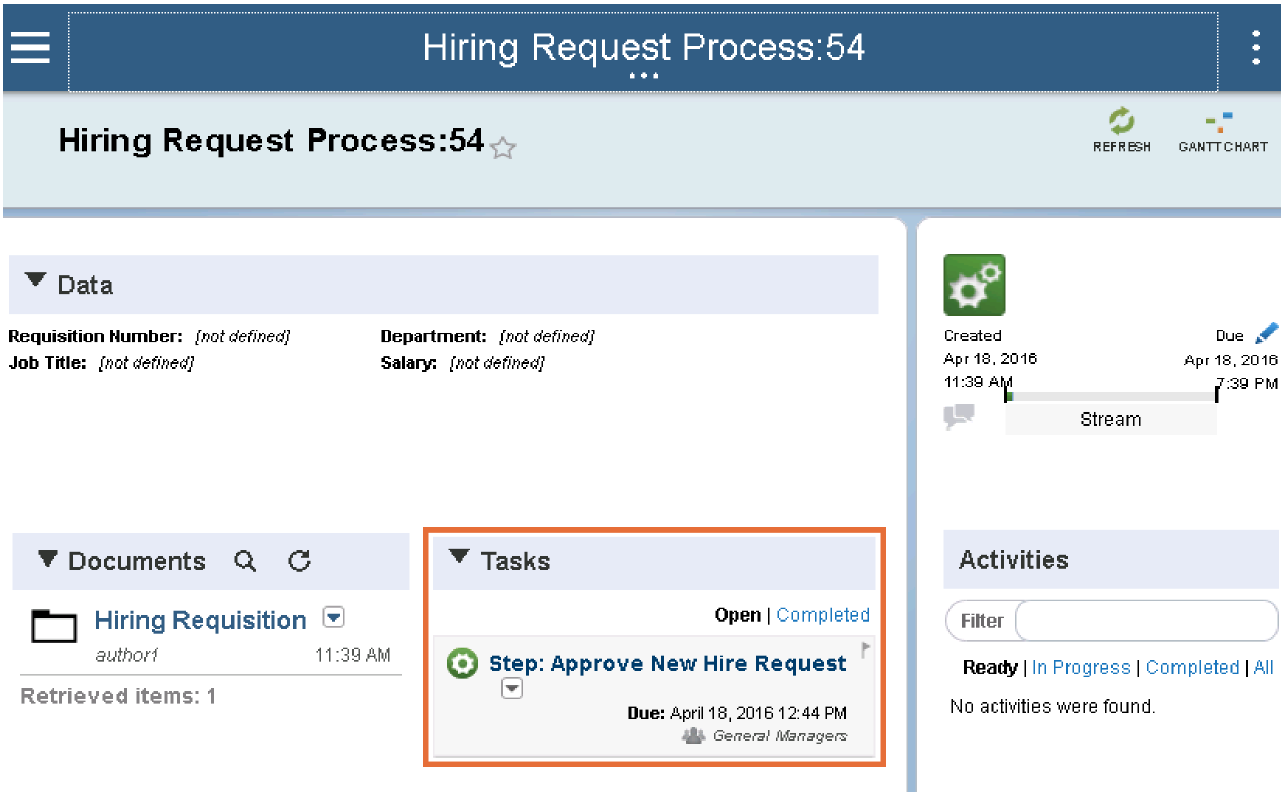
__ b. The Processes page opens. Click **Hiring Request Process**.



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).

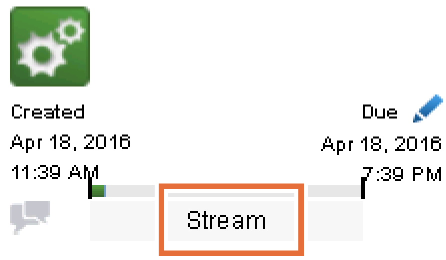
__ c. The process tab displays data about the instance. The bottom section that is labeled **Tasks** shows the next incomplete task in the process.



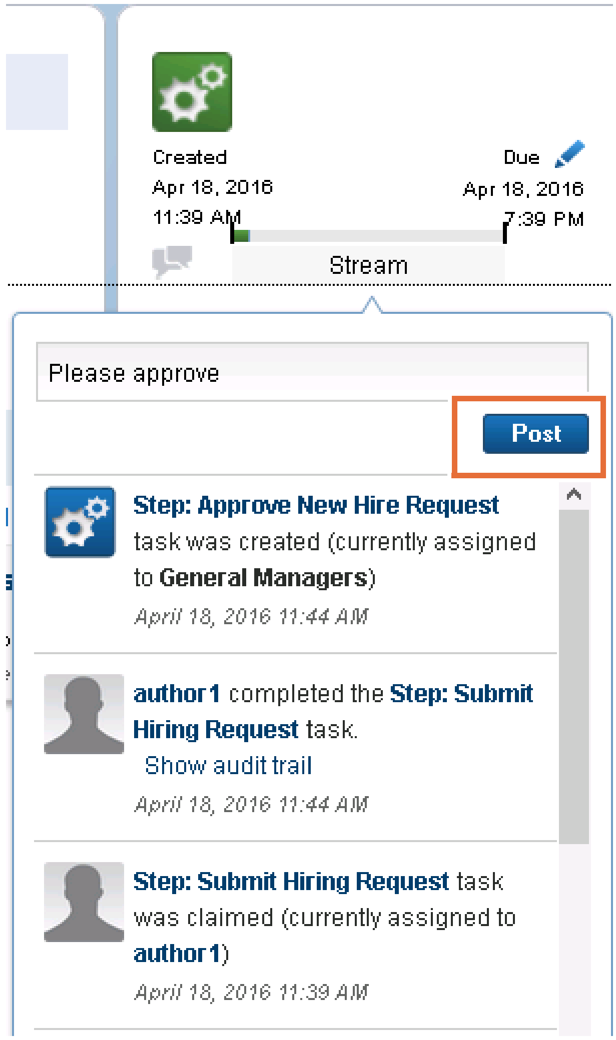
Note

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.

__ 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** link to view the actions that occur for this instance.

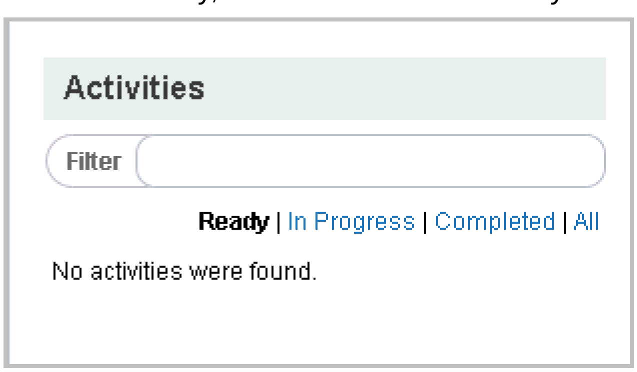


__ e. To add to the stream, type `Please approve` into the stream field and click **Post**. Your comment immediately appears as the most recent entry in the stream.

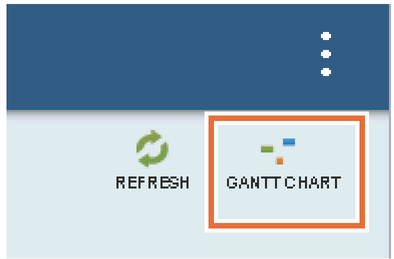


Reminder

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.



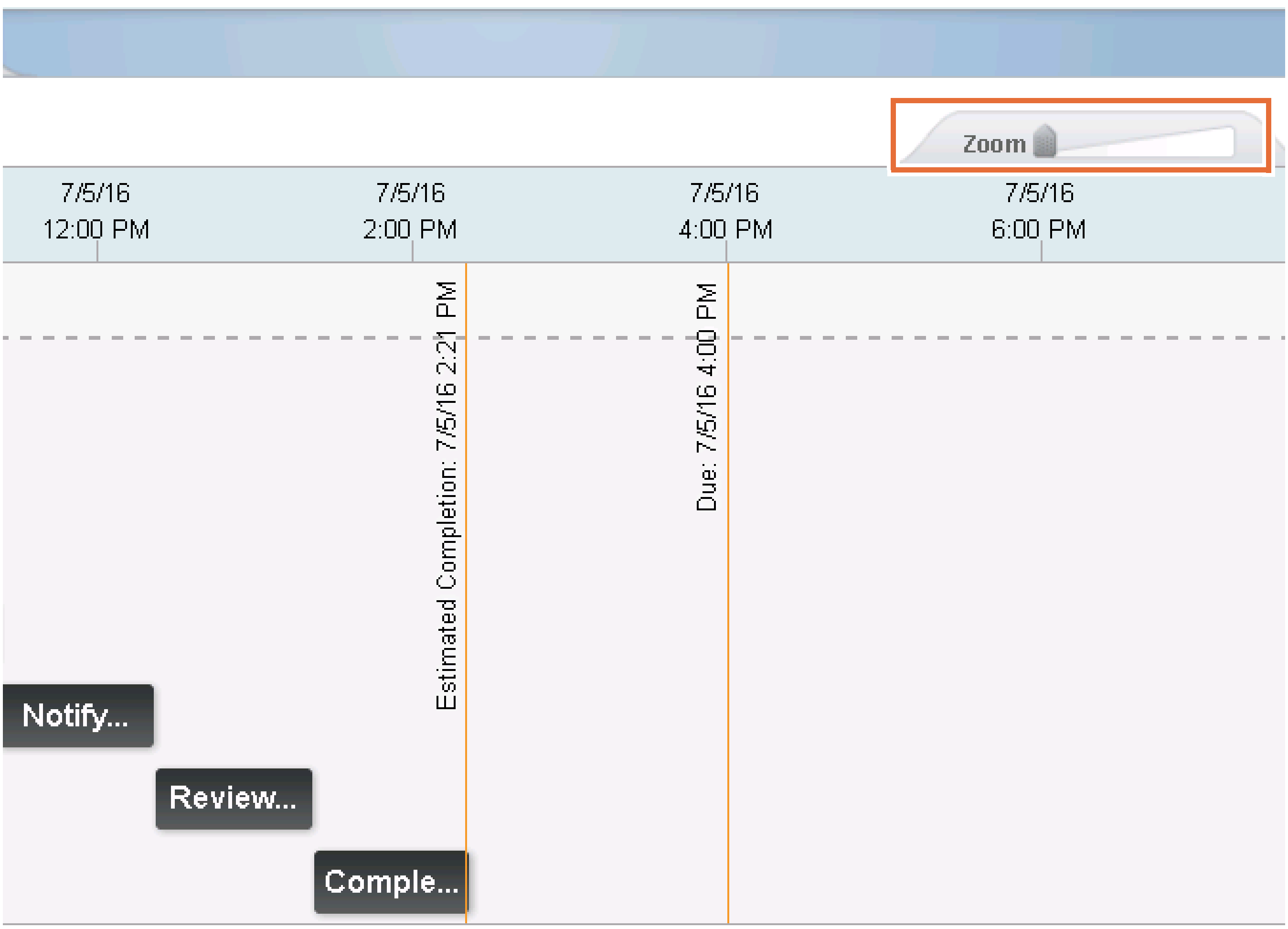
__ f. Click the **GANTT CHART** icon in the upper-right corner to go to the Gantt chart view.



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.

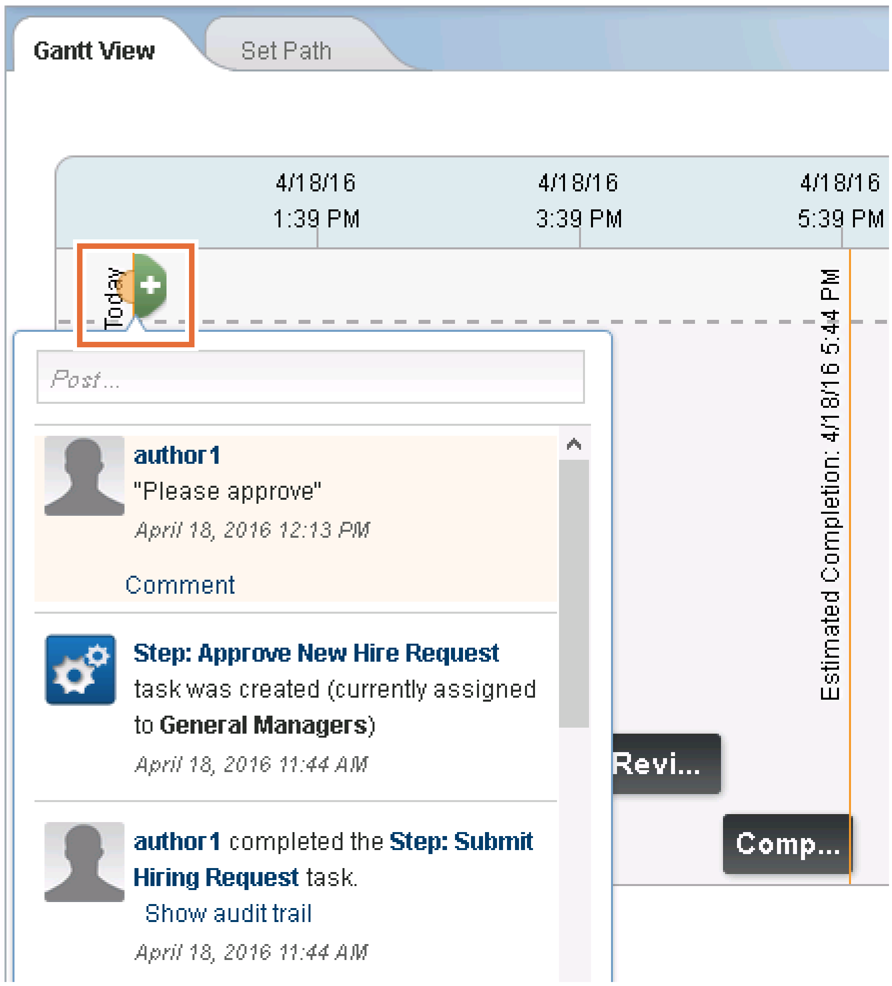
__ g. When the Process Performance page opens, move the slider in the **Zoom** to zoom in on the Gantt chart view data.



Important

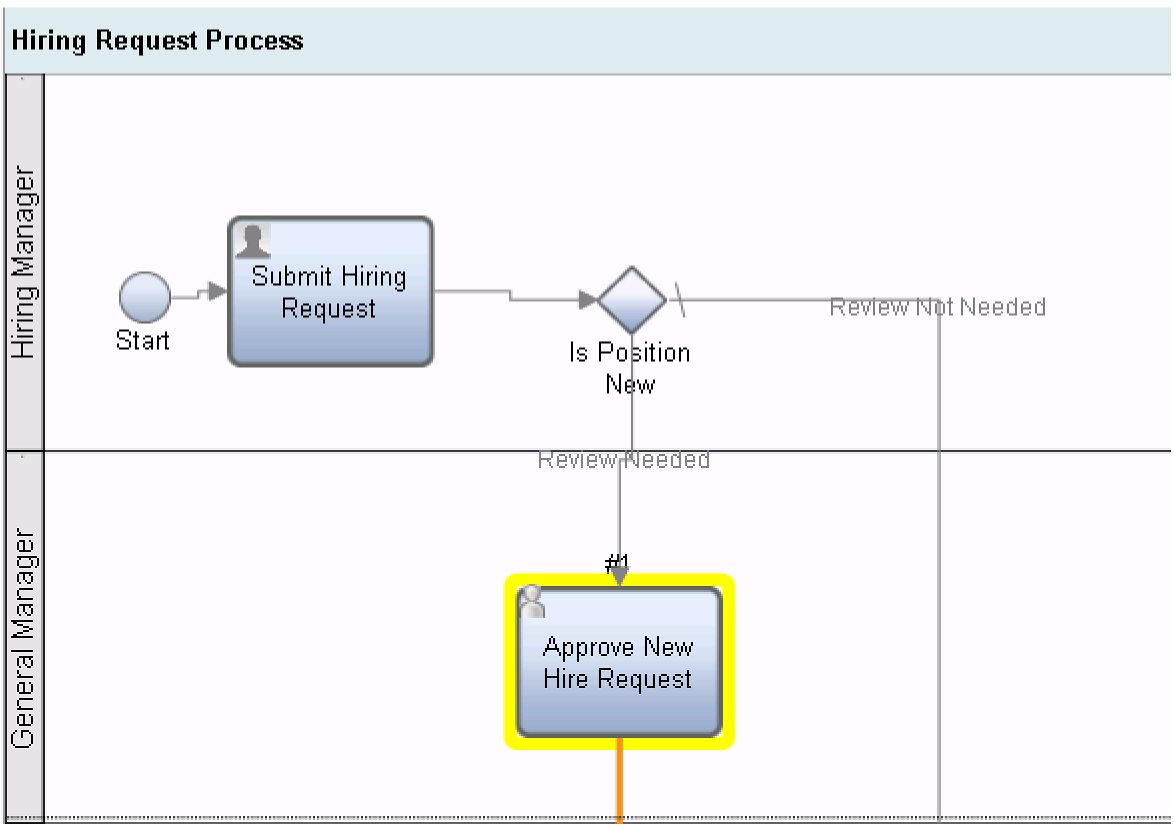
The Gantt chart lists all the instance tasks that are part of the process and the expected durations of the activities as defined by the due dates of the activities. The next task is assigned to the General Managers team. This result is because you set the default value of the first gateway to `"1"` (`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 **green circle** next to the **Today** label, and it turns yellow and shows the stream.



__ i. Click the **Set Path** tab for a model view of the same process instance.

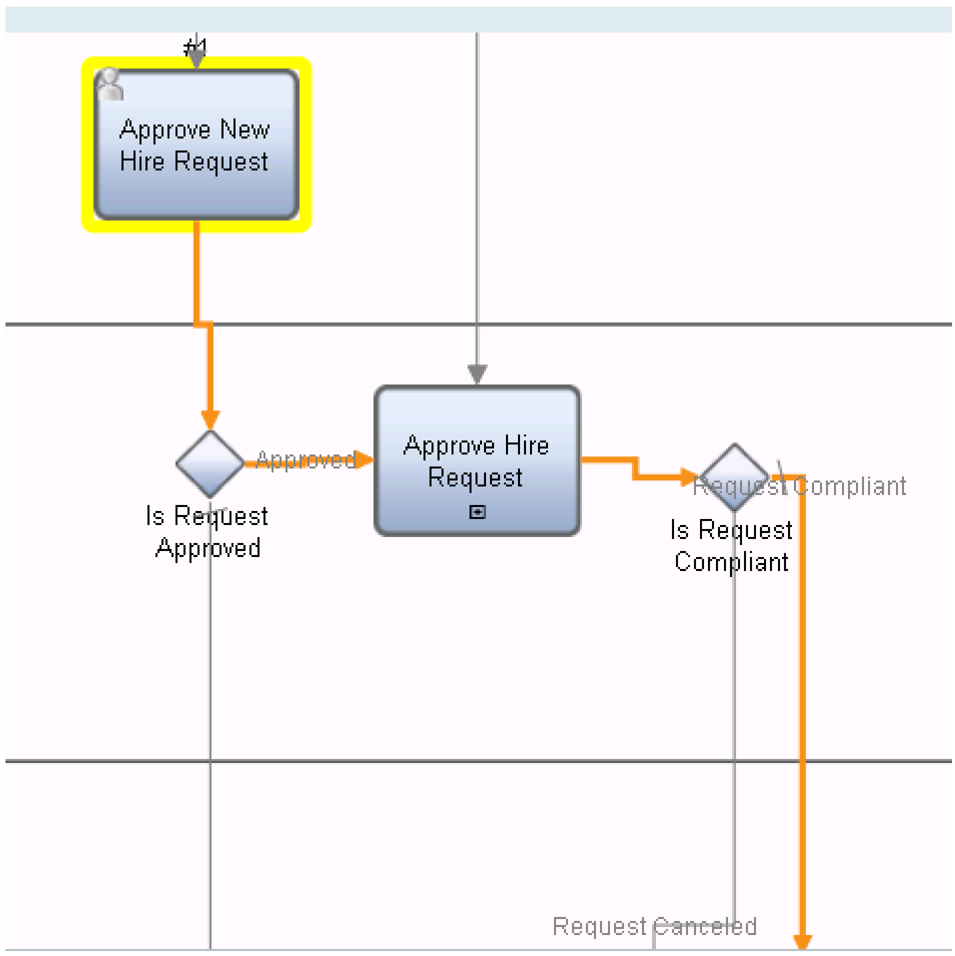
__ j. The model view of the process instance is displayed.



Important

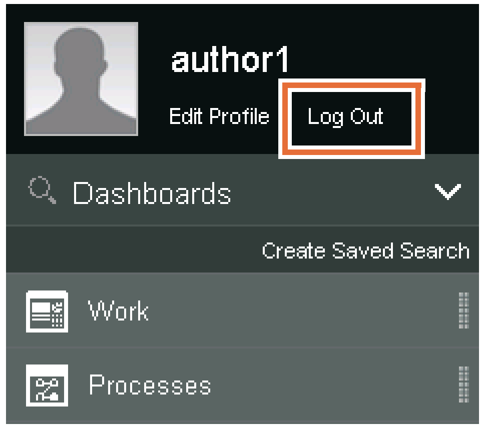
This view shows the process with the Approve New Hire Request task that is highlighted for the **Review needed** path, and in the next part of this exercise you work on the **Review not needed** path.

The same process performance information is given to the user as before, only this time in a process model (or diagram) view. This view provides a different way to set the path that is taken through gateways. The path that is shown in orange is the default path.

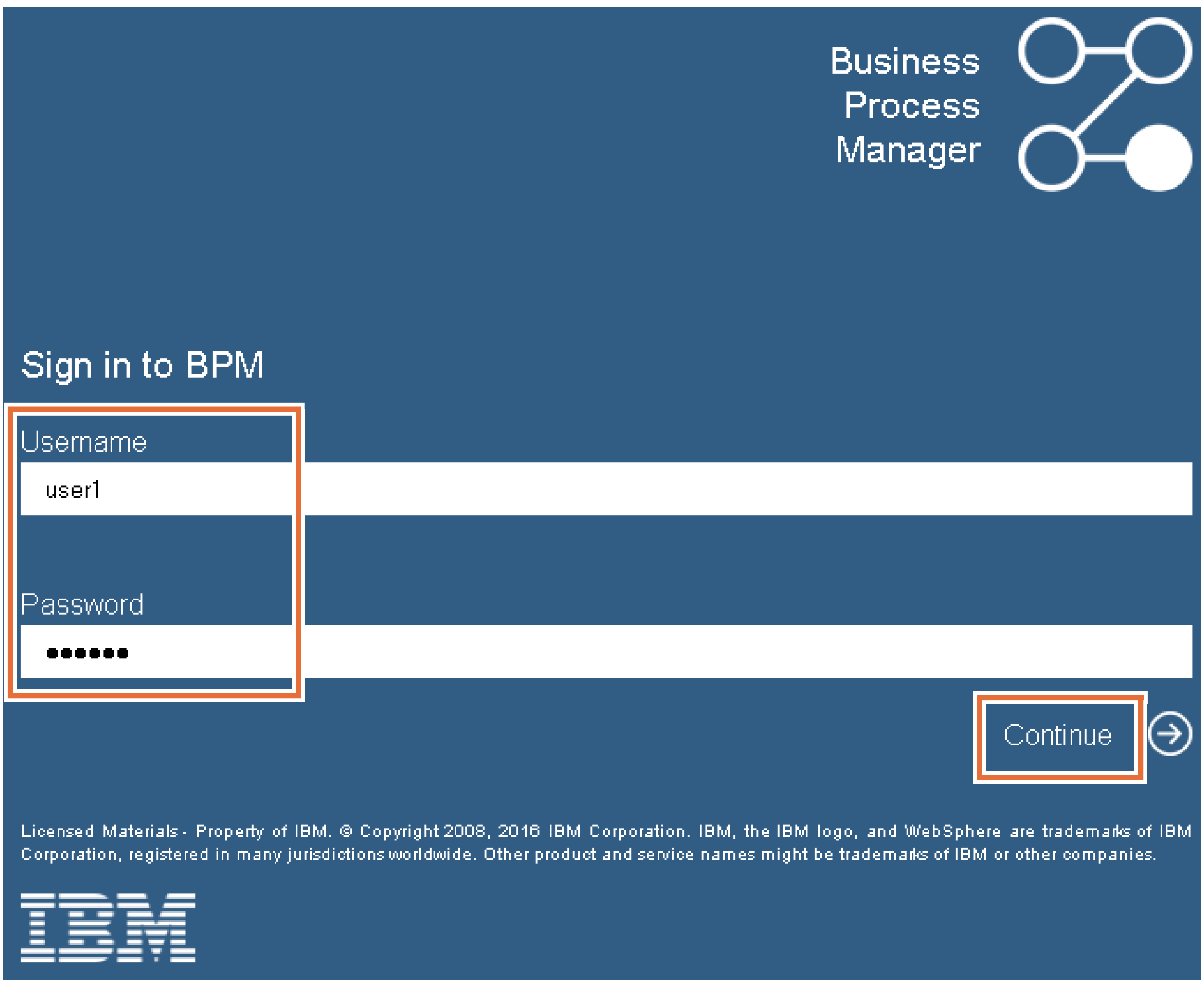


__ 4. Complete the Approve New Hire Request activity.

__ a. In the library, click **Log Out** to log out of the Process Portal.

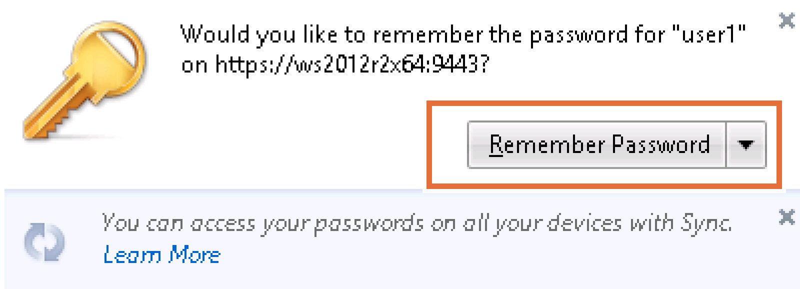


__ b. Log in to the Process Portal with the **Username** `user1` and **Password** `user01` and click **Continue**.

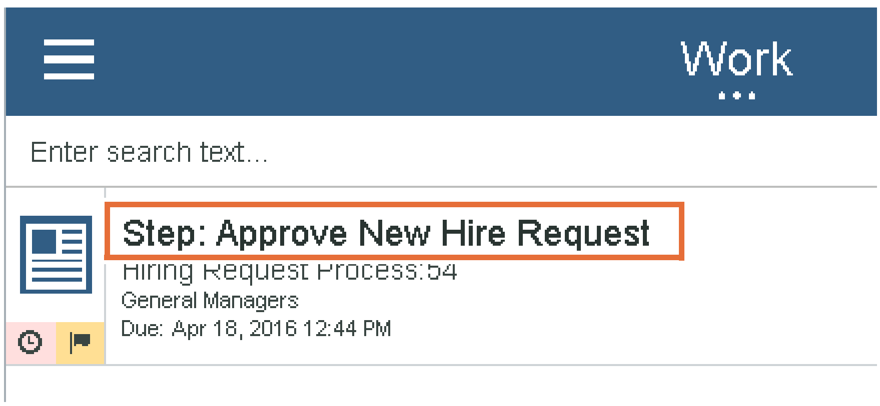


Note

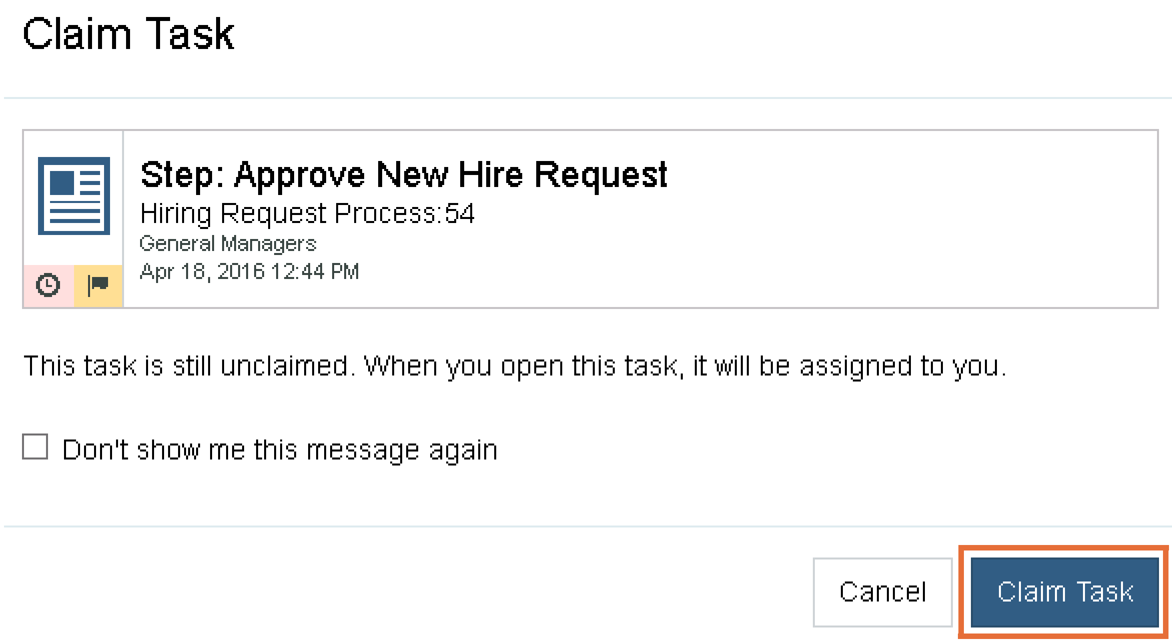
If prompted for Remember Password, click **Remember Password**.



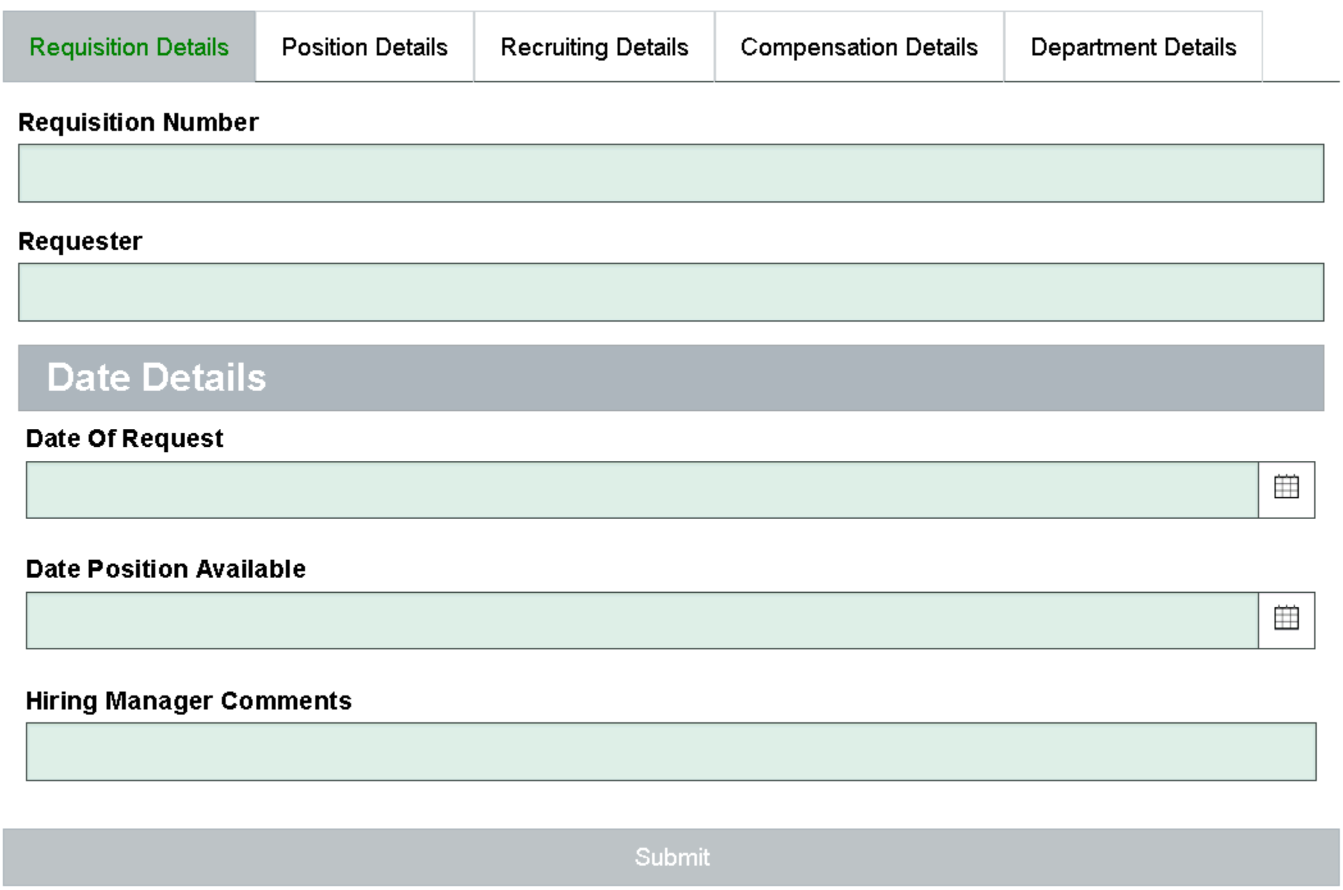
__ c. A task is in the inbox for the Hiring Request Process process. The task stopped at the Approve New Hire Request activity. Notice that the task is assigned to the General Managers team, of which user1 is a member. Click the **Step: Approve New Hire Request** task.



__ d. Click **Claim Task**, and the task is automatically assigned to you.



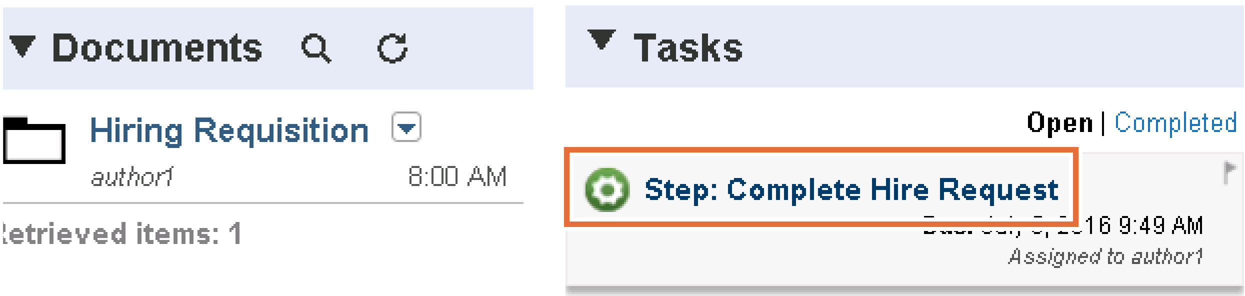
__ e. A human service coach is displayed for the Approve New Hire Request activity.



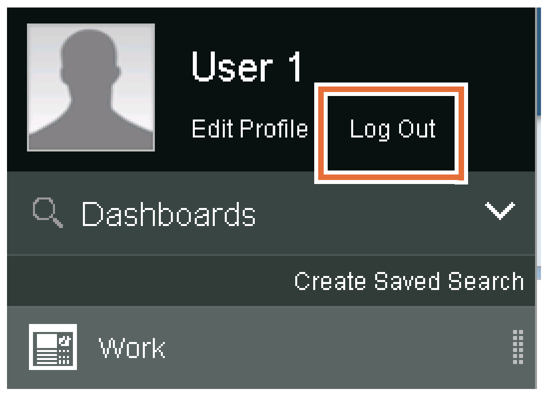
__ f. Enter the data in the fields and complete this task by clicking **Submit** in the human task service.

__ g. Click **Processes** in the library.

__ h. Click **Hiring Request Process**. The next task is assigned to `author1`.



__ i. **Log Out** from the Process Portal.



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.

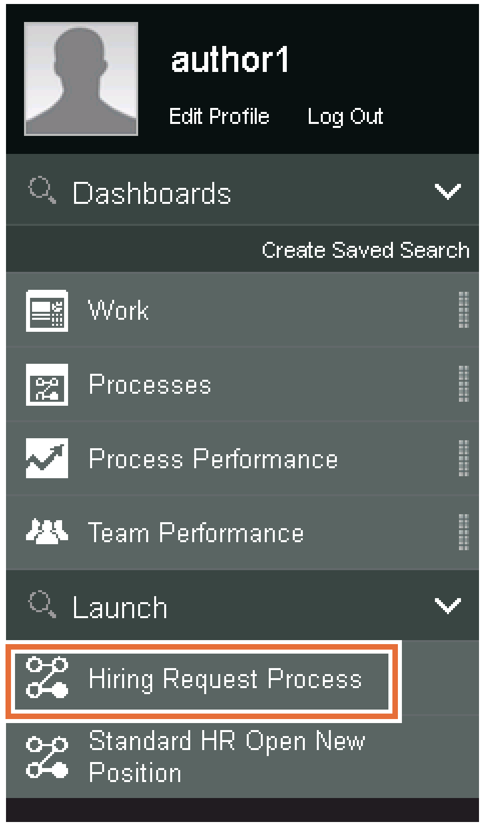
__ 1. Show the process as it follows the `Review Not Needed` path, and create an instance to test the "existing position" scenario.

__ a. Log in to the Process Portal with `author1` as the **Username** and `author01` as the **Password**.

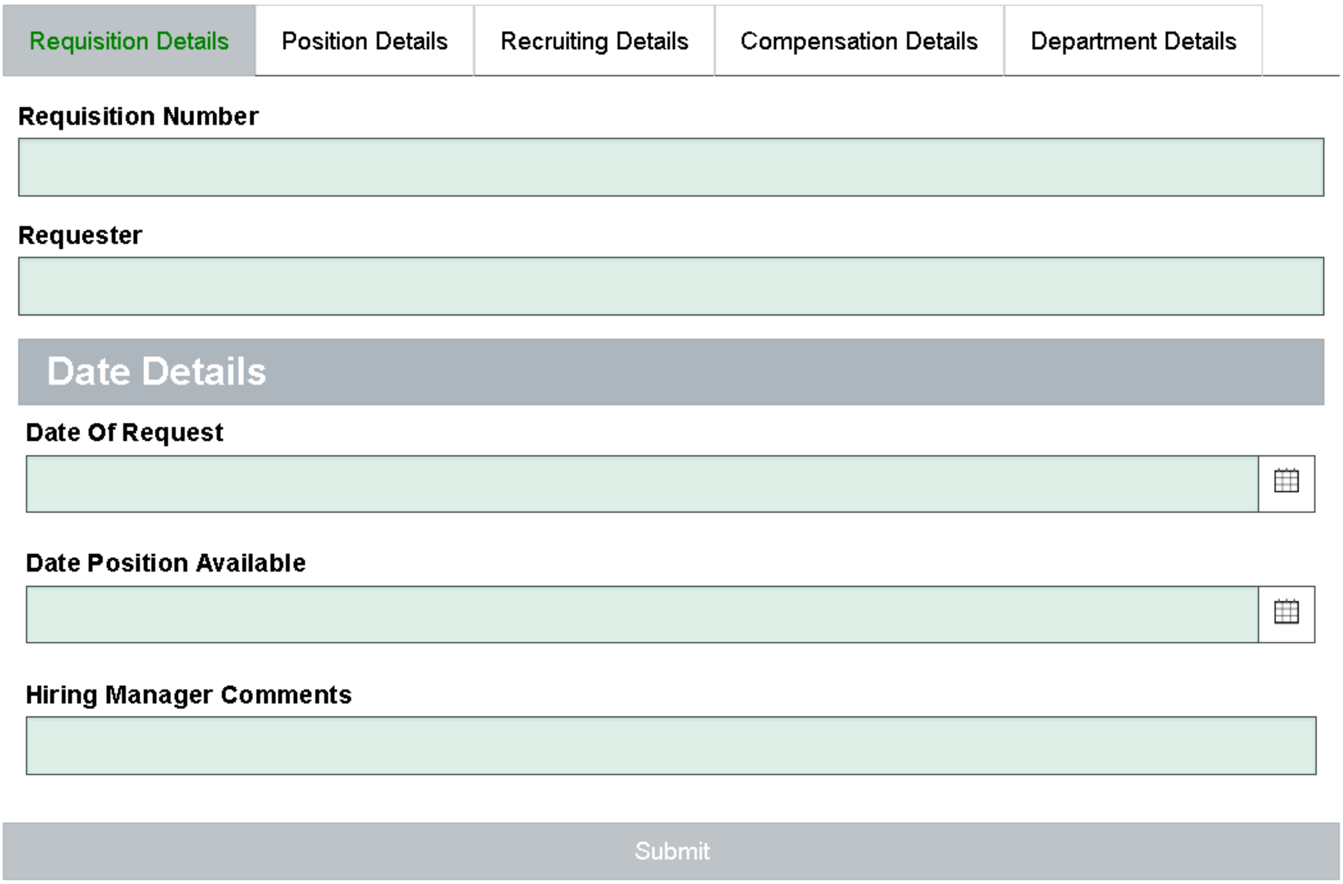
Hint

If you already closed the window, use the IBM Business Process Manager Quick Start browser window to start the Process Portal.

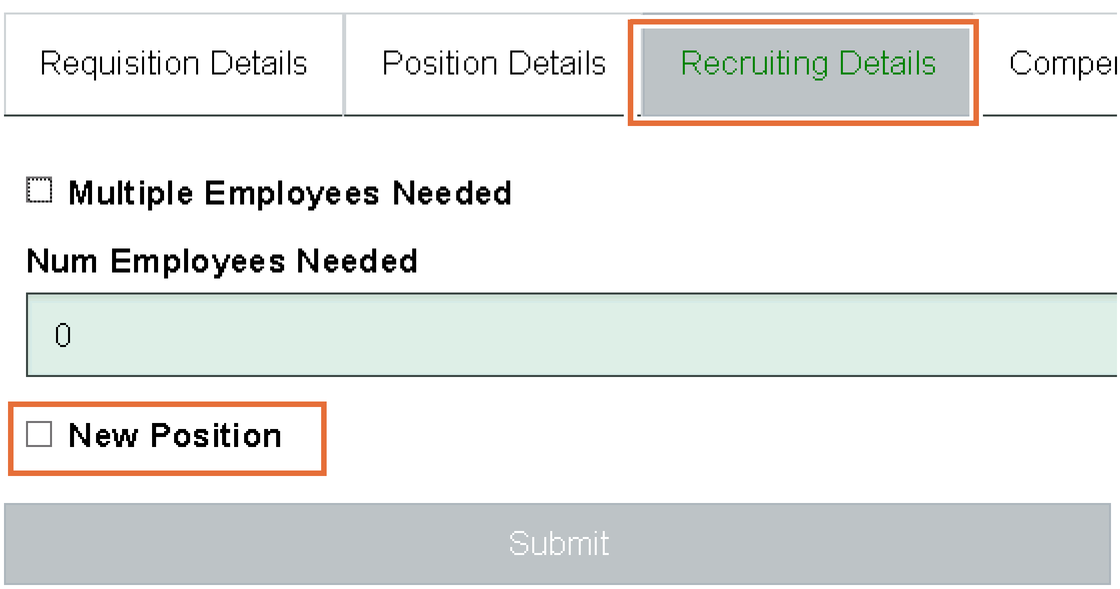
__ b. Create an instance of the **Hiring Request Process** by clicking the link in the Launch section.



__ c. You immediately see in a dialog box that you received a new task. A window is displayed with the human service coach.



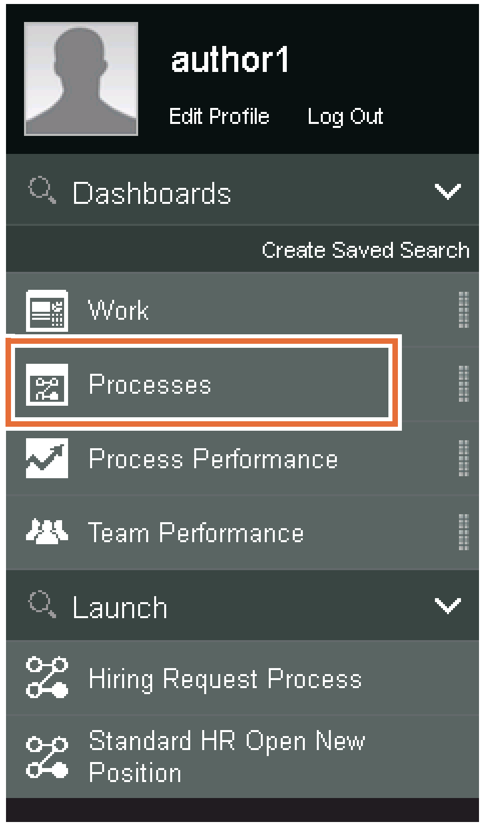
__ d. Select the **Recruiting Details** tab, and do ***not*** select the **New Position** check box so that it follows the Review Not Needed path.



__ e. Enter the data for other fields and click **Submit**.

__ 2. Complete the process.

__ a. In the library, click **Processes**.



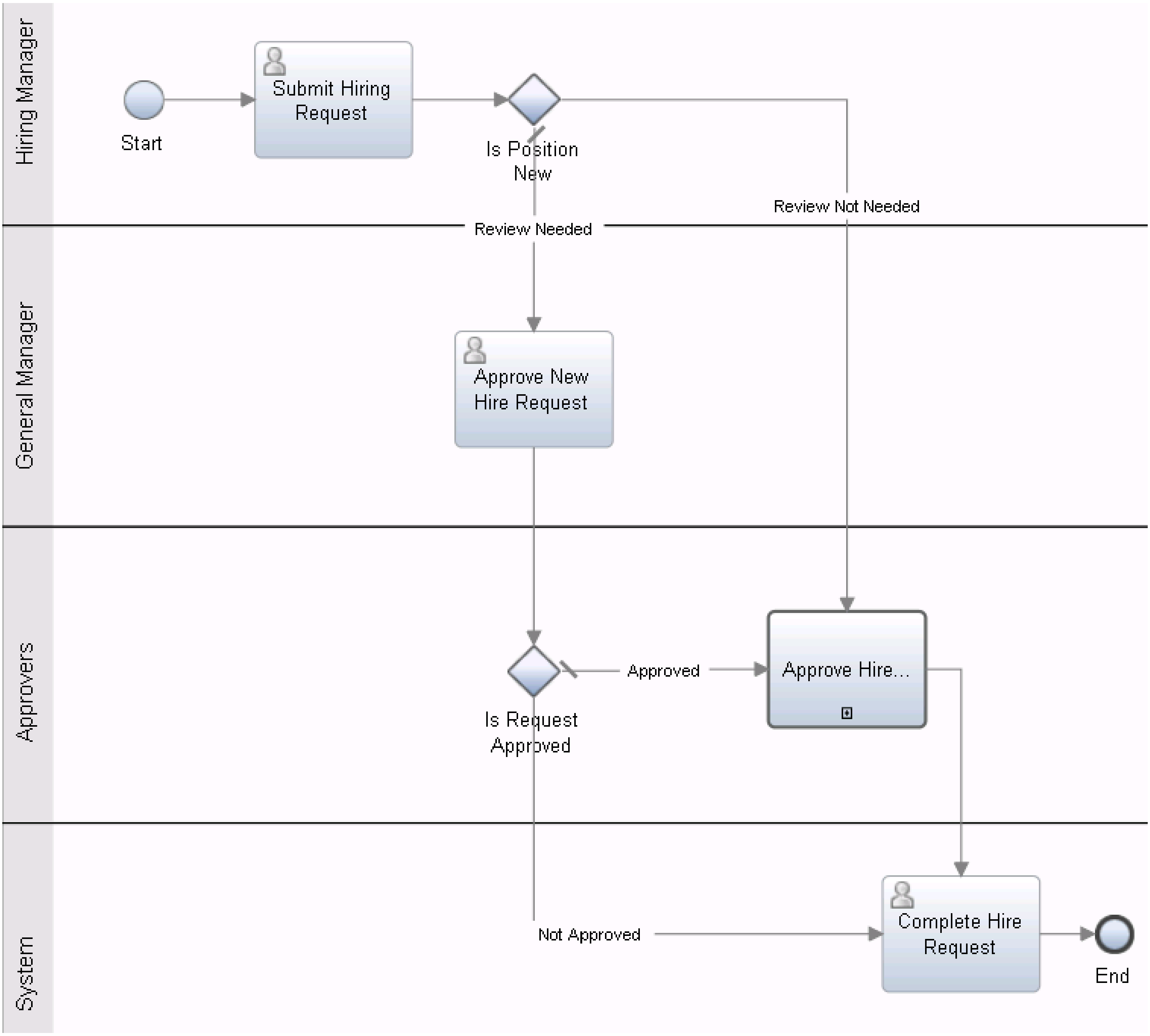
__ b. Click **Approve Hire Request**.

Note

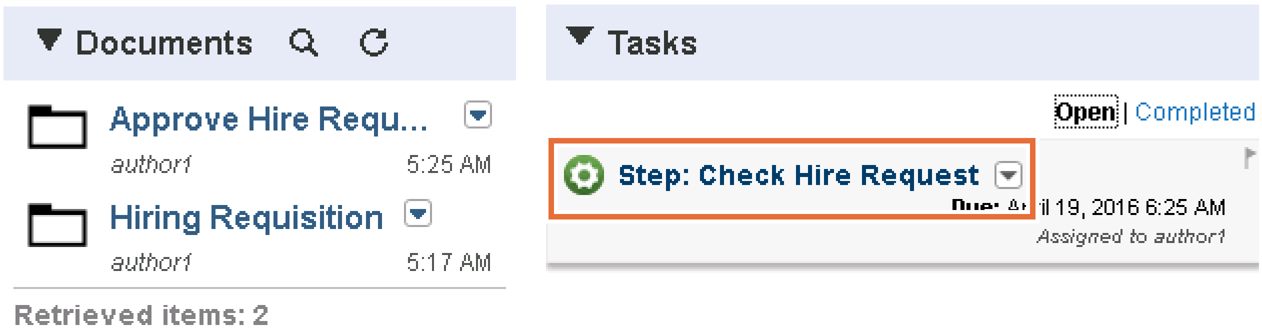
Click the task with the highest instance ID because that is the instance you created.

Information

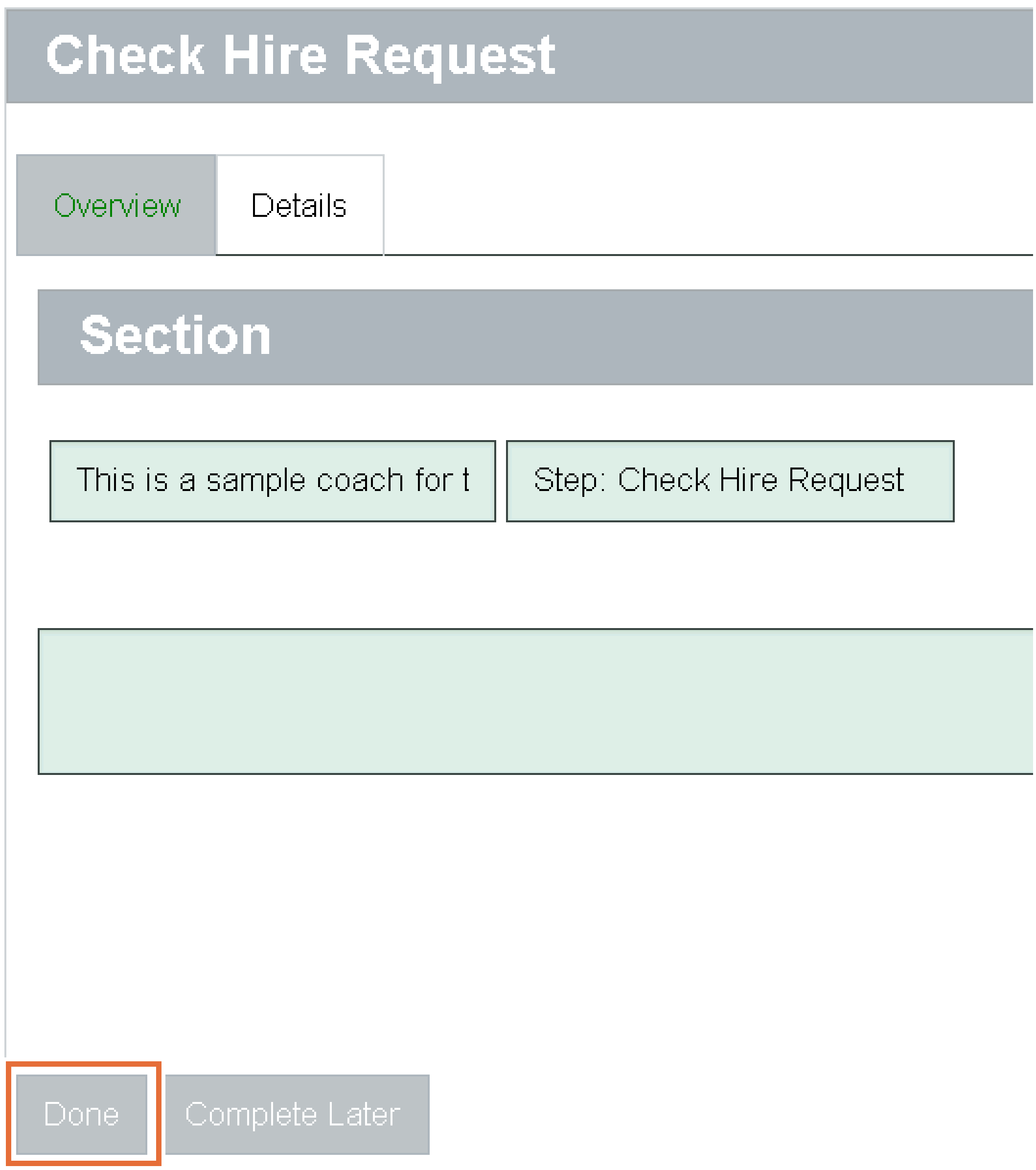
This time, the process follows the `Review Not Needed` path. Instead of passing to **Approve New Hire Request**, the process stepped through the **Approve Hire Request** linked process.



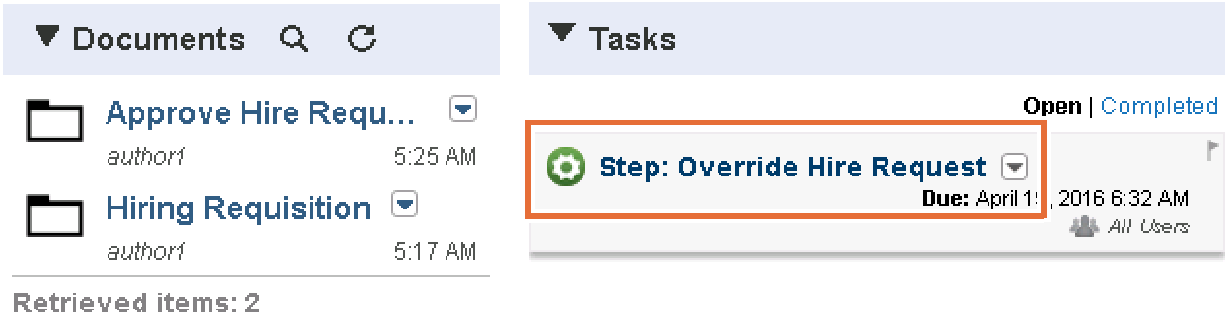
__ c. Click the **Step: Check Hire Request** task link.



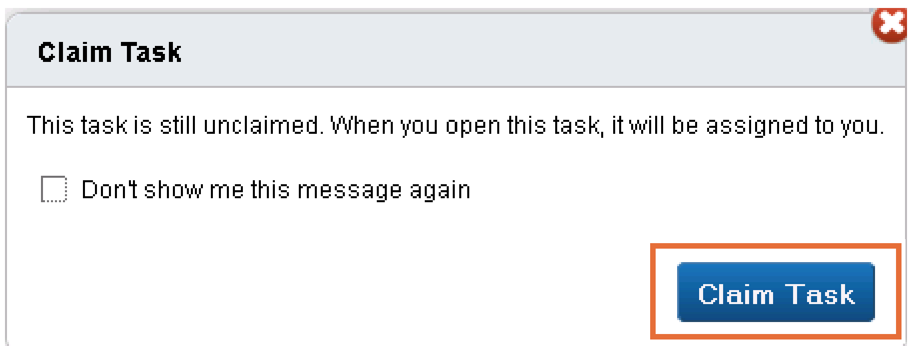
__ d. Click **Done** to complete the task.



__ e. Now the process moves to the next task. Click **Step: Override Hire Request**.



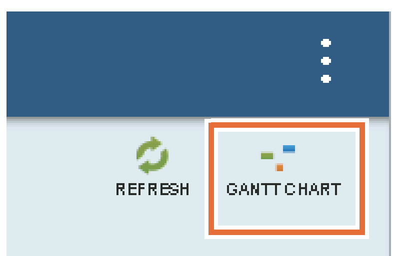
__ f. Click **Claim Task**.



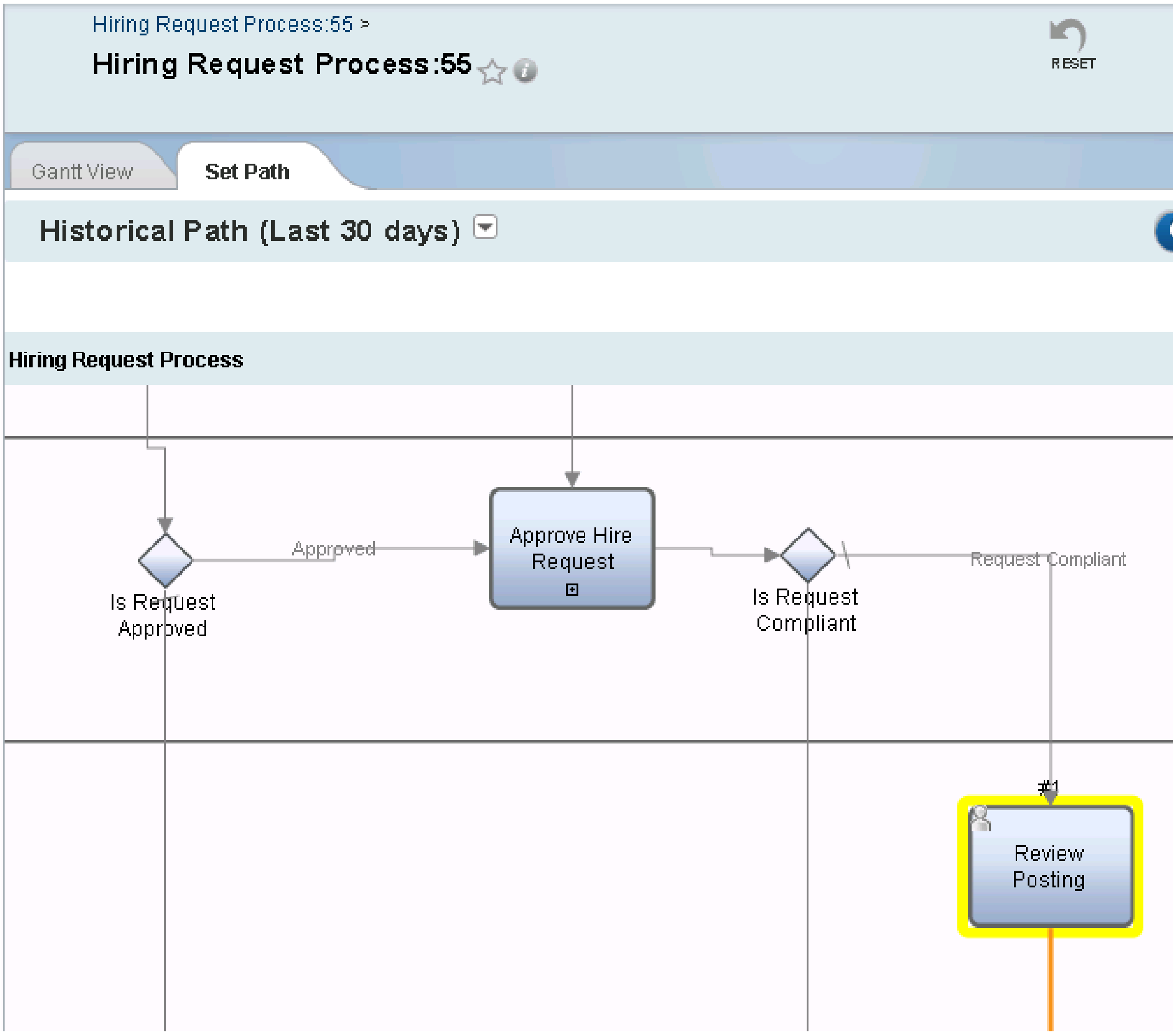
__ g. Click **Done** to complete the task.

__ h. The process is now moved to **Step: Review Posting**.

__ i. Click **Gantt Chart** icon on the upper-right corner of the window. Wait for few seconds to load the page.

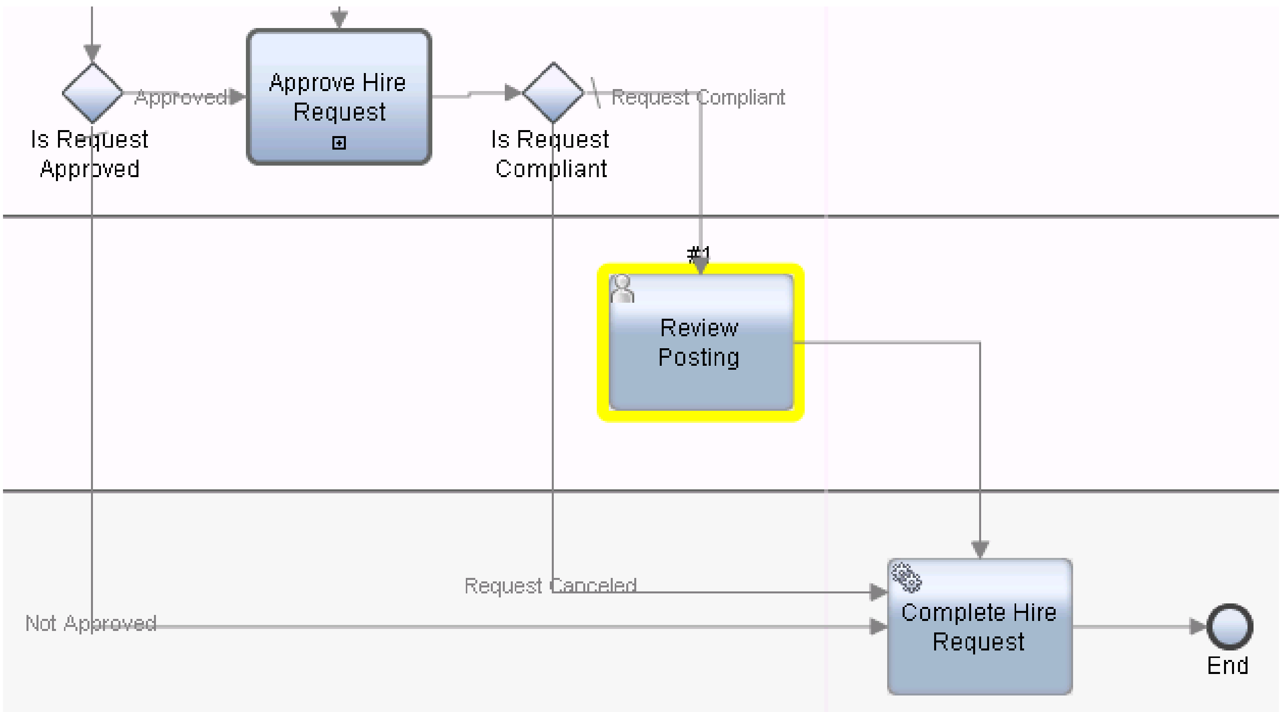


__ j. Click the **Set Path** tab for a model view of the same process instance.

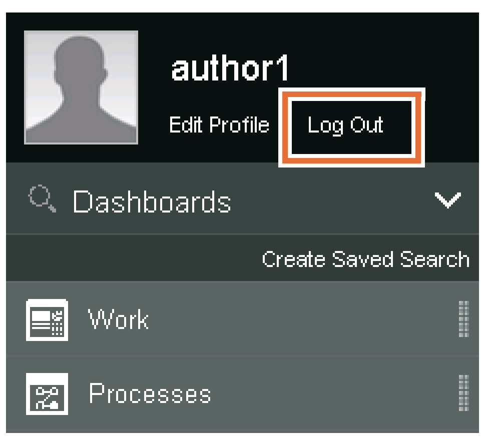


Note

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

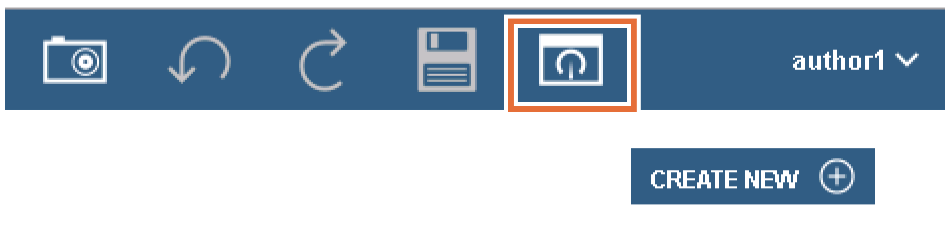
In this course, you did not implement all the coaches for the sake of brevity, but one of the requirements to move to Playback 3 is that all the coaches are implemented in the process and all of them have the finalized presentation elements in place (CSS and JS, for example).

Part 3: Create the Hiring Requisition Toolkit

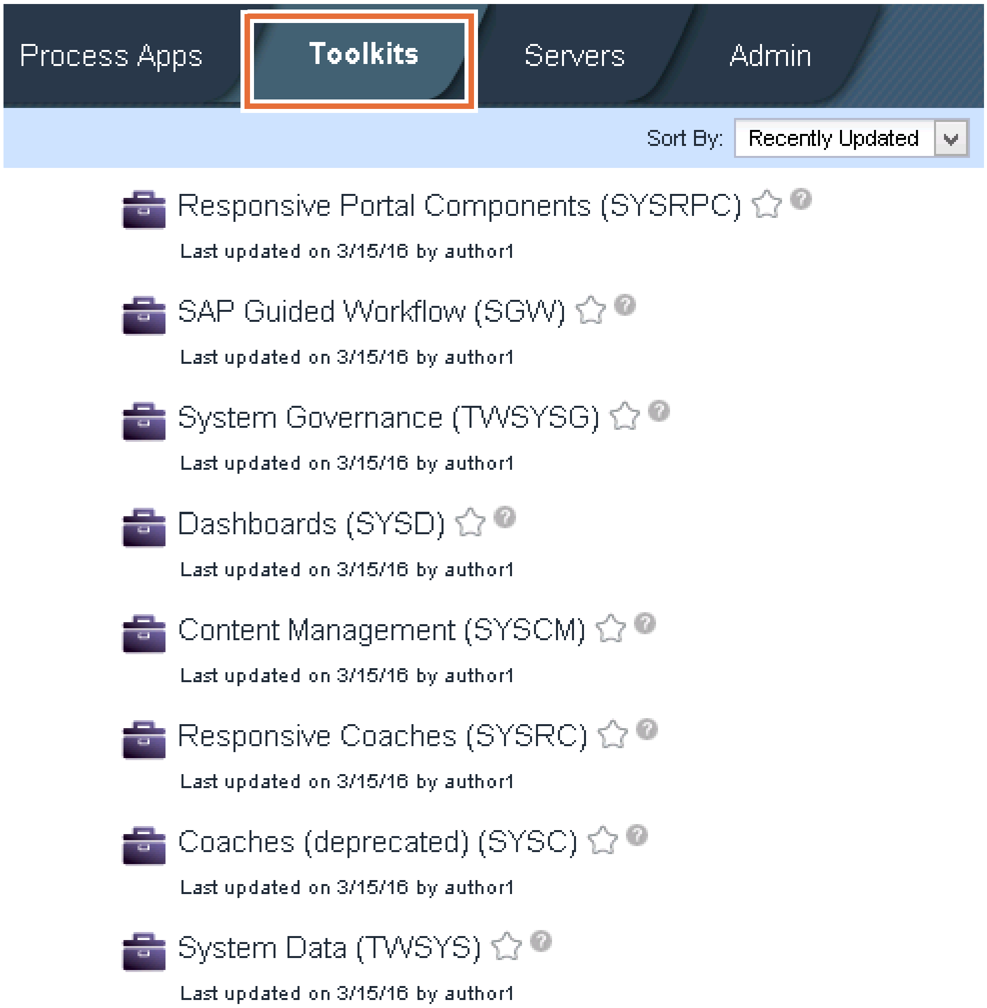
Now, you have numerous assets, some of which might be useful to other developers. Create a toolkit from the assets you created as part of the process application HR Recruitment Processes.

__ 1. Return to the IBM Web Process Designer.

__ 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).

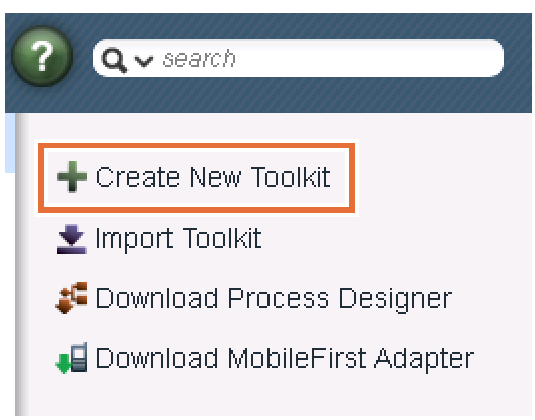


__ 2. Click the **Toolkits** tab.



__ 3. Create a toolkit.

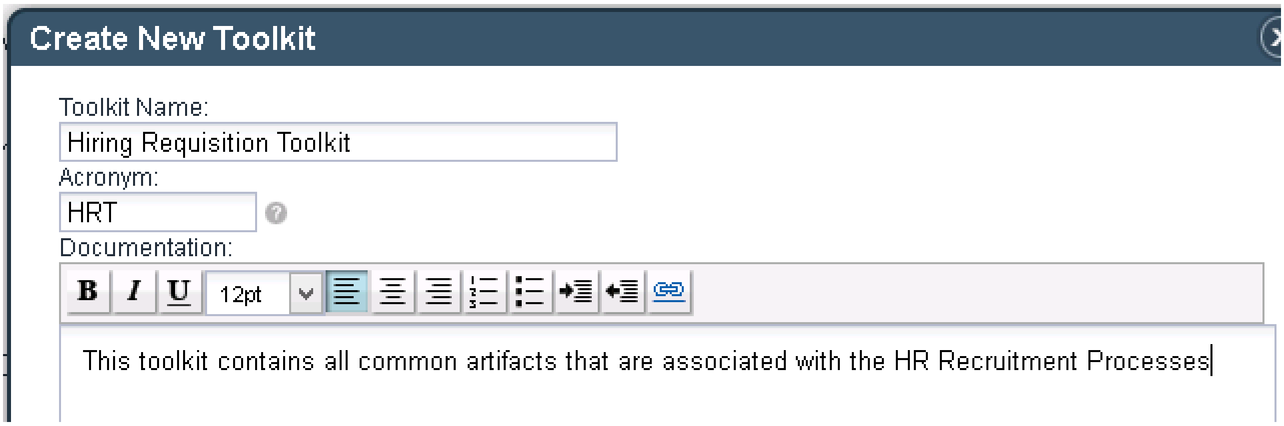
__ a. Click **Create New Toolkit** on the right side of the Process Center.



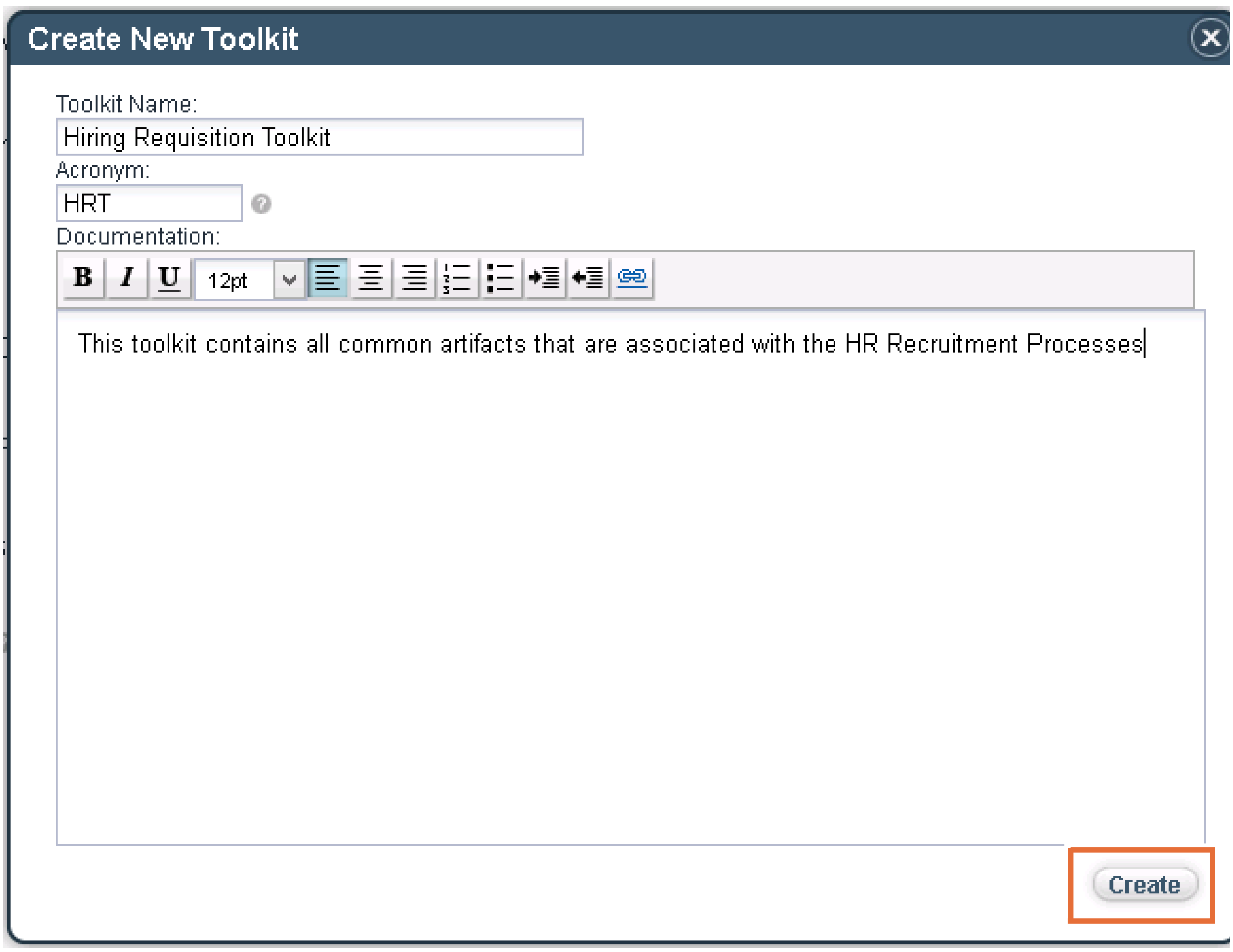
__ b. Enter the name for the toolkit: `Hiring Requisition Toolkit`

__ c. Enter the toolkit acronym: `HRT`

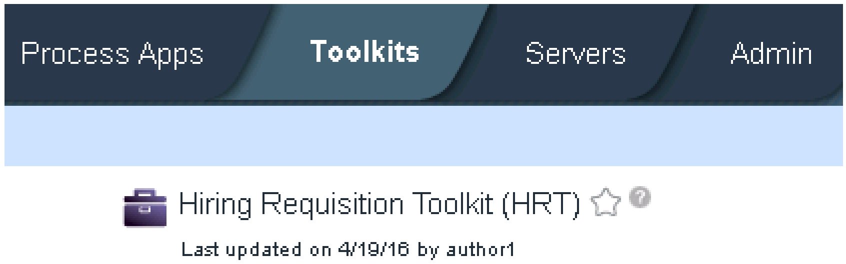
__ d. Enter a brief description: `This toolkit contains all common artifacts that are associated with the HR Recruitment Processes`.



__ e. Click **Create**.

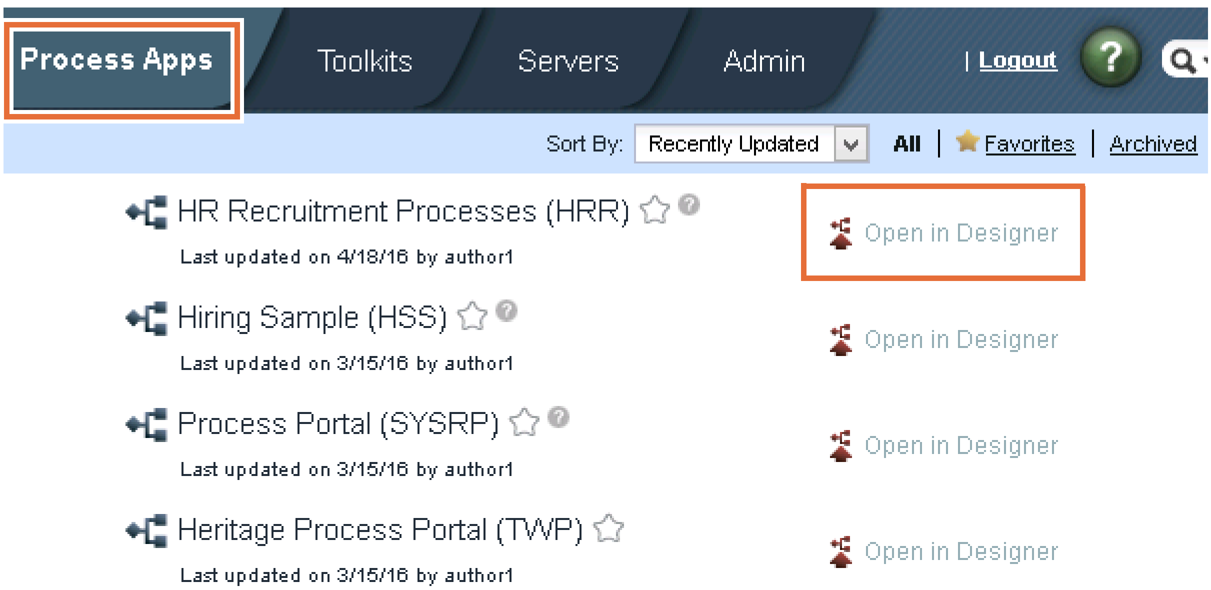


__ f. The Hiring Requisition Toolkit (HRT) shows up under the **Toolkits** tab.



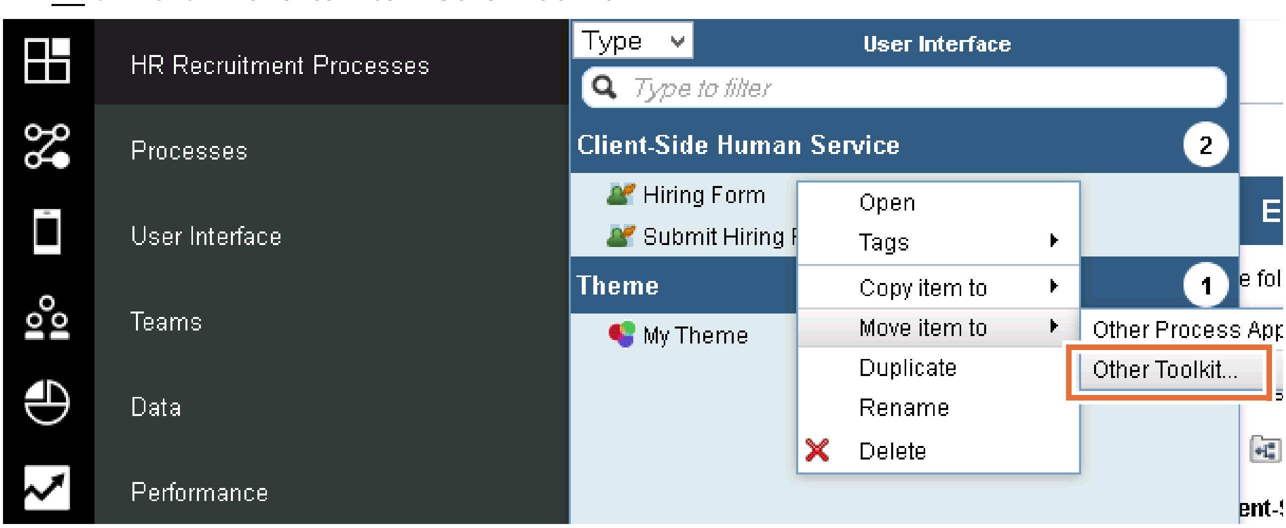
__ 4. Move the client-side human service into the toolkit and create a baseline snapshot for the test environment

__ a. Click the **Process Apps** tab. Open the **HR Recruitment Processes (HRR)** process application in the Process Designer by clicking **Open in Designer**.

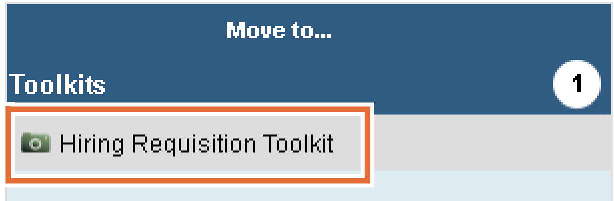


__ b. In the library, right-click the **User Interface > Hiring Form** client-side human service.

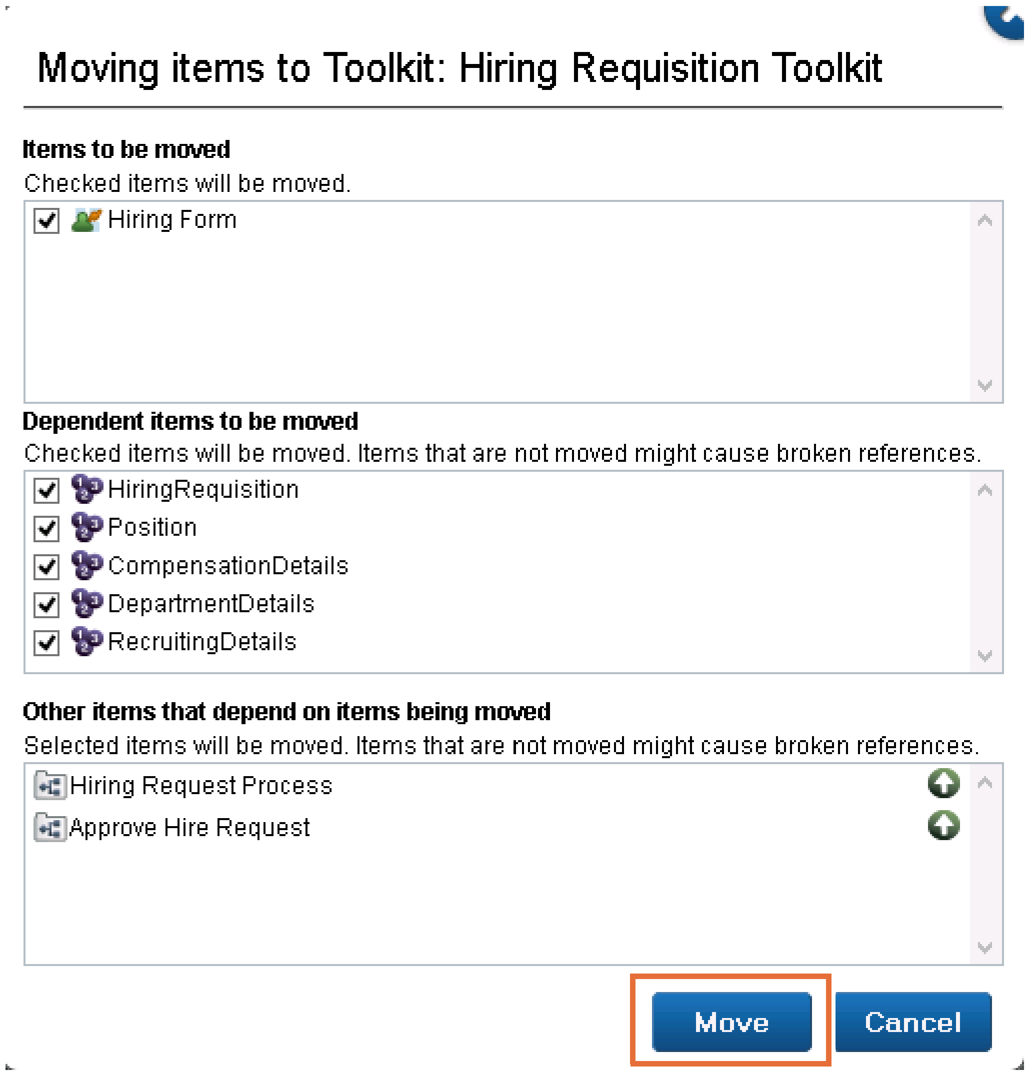
__ c. Click **Move Item to > Other Toolkit**.



__ d. Select **Hiring Requisition Toolkit**.

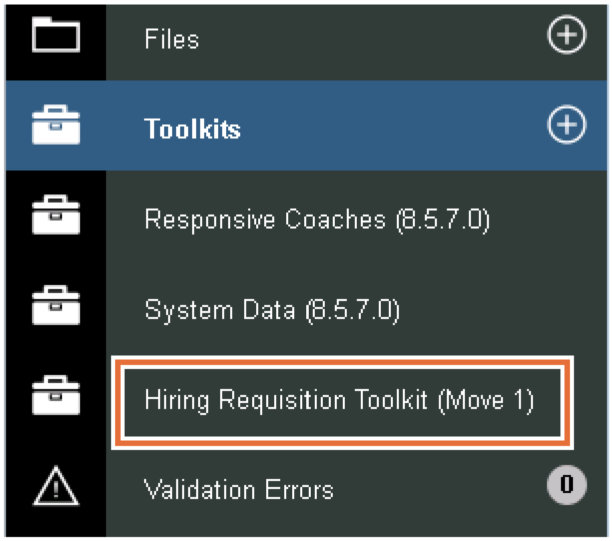


__ e. Keep the defaults and click **Move**.

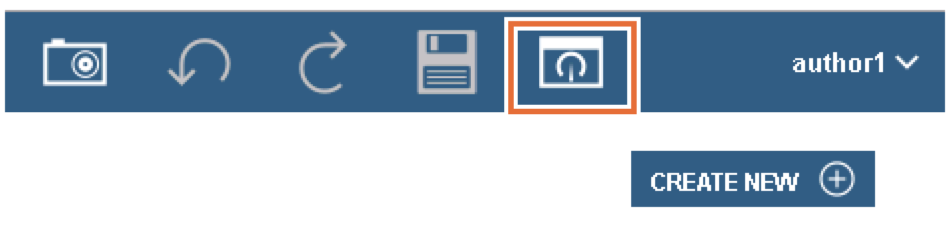


Note

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.

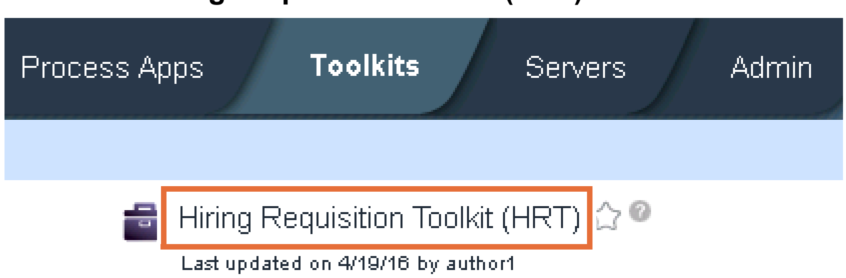


__ f. Return to the **Process Center**.

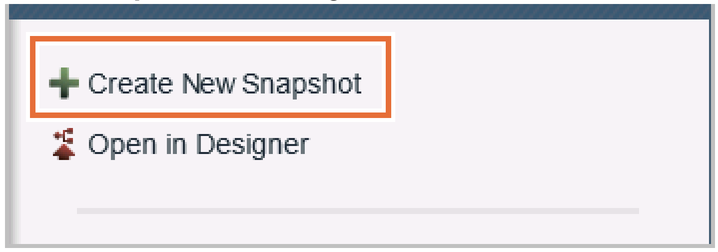


__ g. Click the **Toolkits** tab.

__ h. Click the toolkit **Hiring Requisition Toolkit (HRT)**.

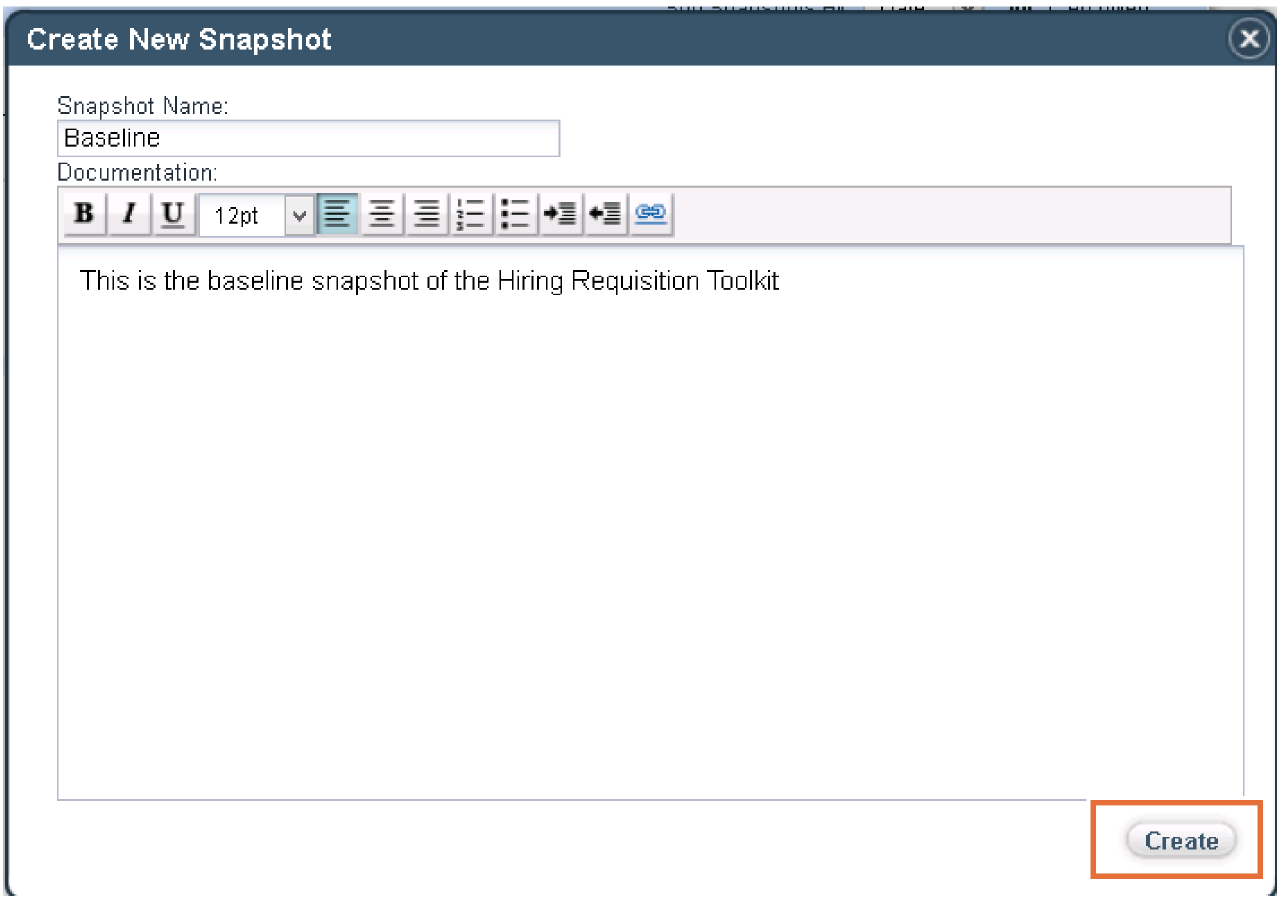


__ i. Click **Create New Snapshot** on the right side.

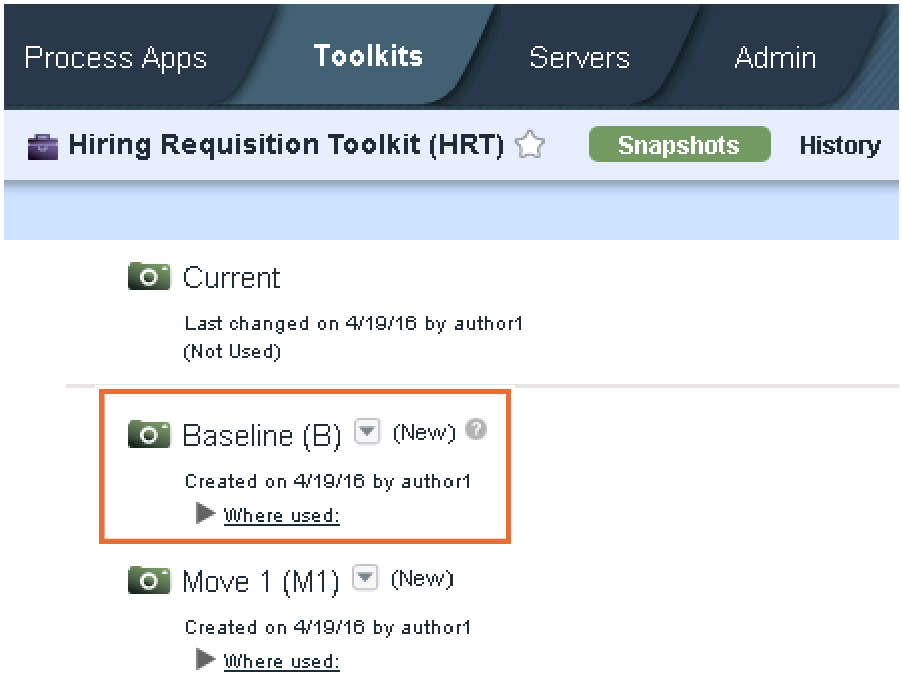


__ j. Name the snapshot `Baseline` and add a description: `This is the baseline snapshot of the Hiring Requisition Toolkit`

Then, click **Create**.



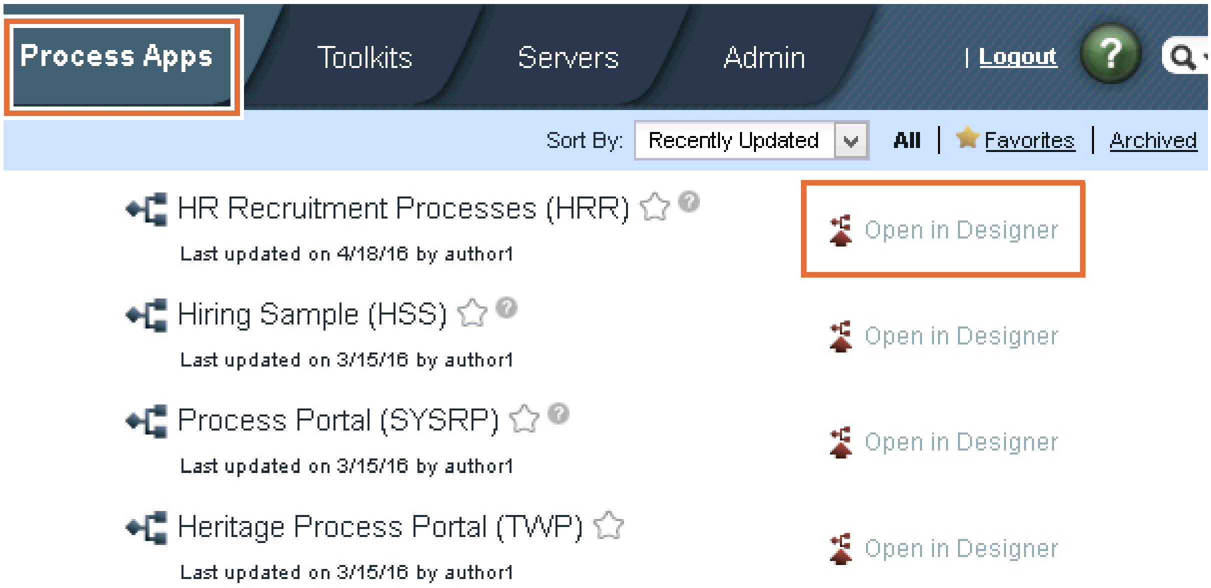
__ k. The snapshot **Baseline** is created.



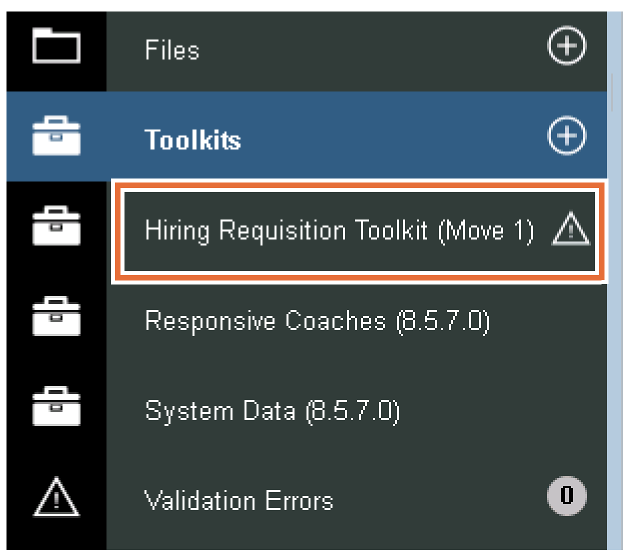
Part 4: Create or update dependency on a toolkit

__ 1. Update the toolkit version on a process application.

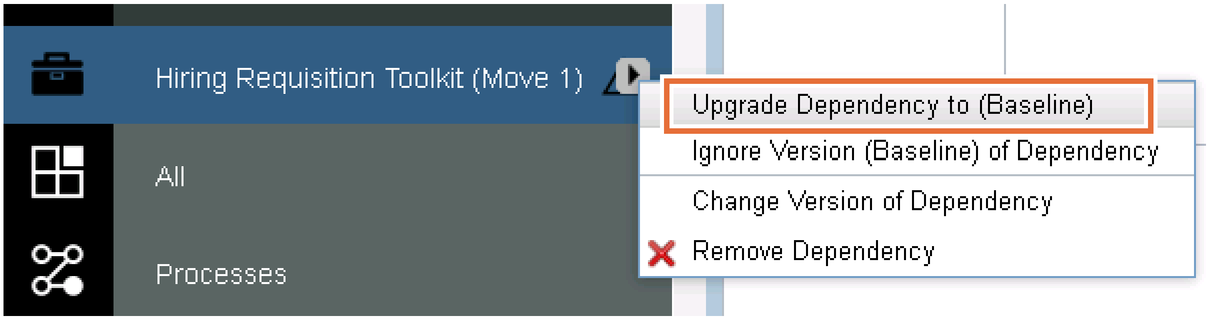
__ a. Click the **Process Apps** tab and open the `HR Recruitment Processes (HRR)` process application in the designer by clicking **Open in Designer**.



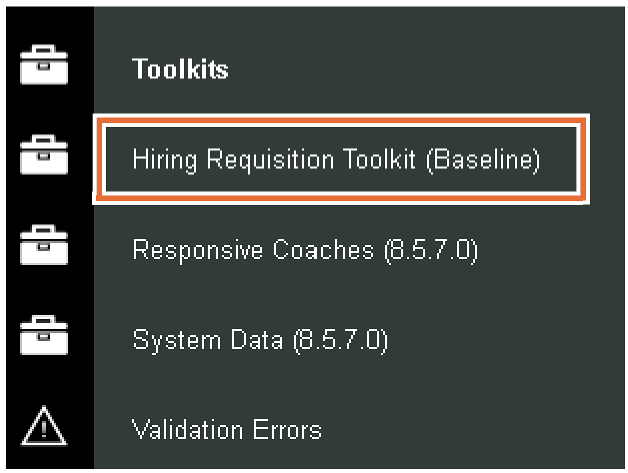
__ b. When you return to the process application, Expand the **Toolkits** category in the Library section. The `Hiring Request Toolkit` shows a warning.



__ c. Click the **arrow** next to the Toolkit and select **Upgrade Dependency to (Baseline)**.

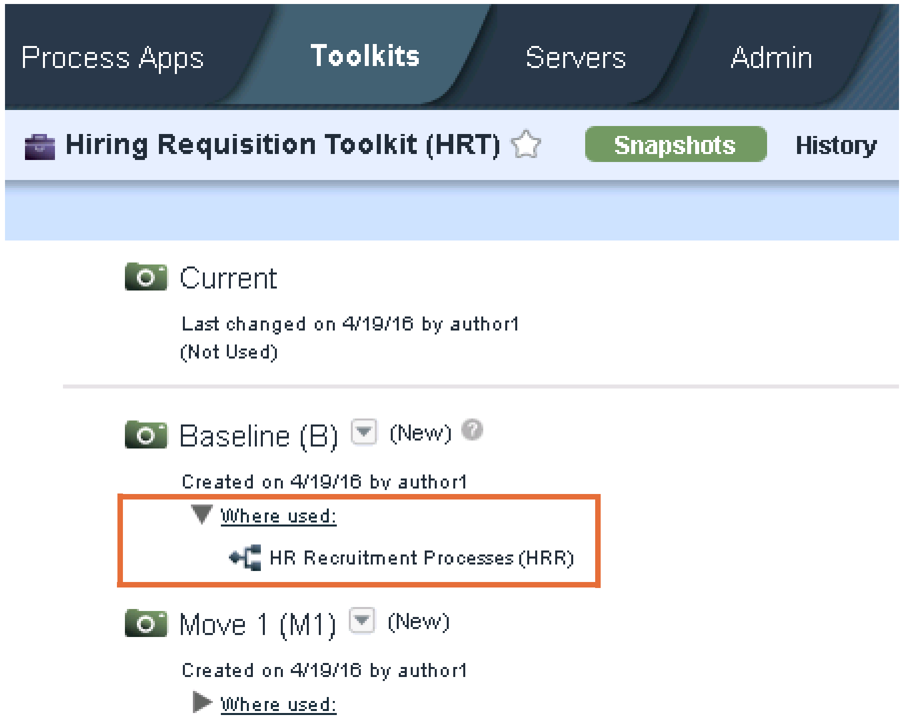


__ d. In the Toolkits category, verify that the Hiring Requisition Toolkit (Baseline) is now linked to the HR Recruitment Processes process application.



__ 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 and create a dependency on this Toolkit.



__ 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 makes the decision. In the to-be process, a user was going to make this decision, but now they want the system to make the decision 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 by using the IBM Process Designer client application.

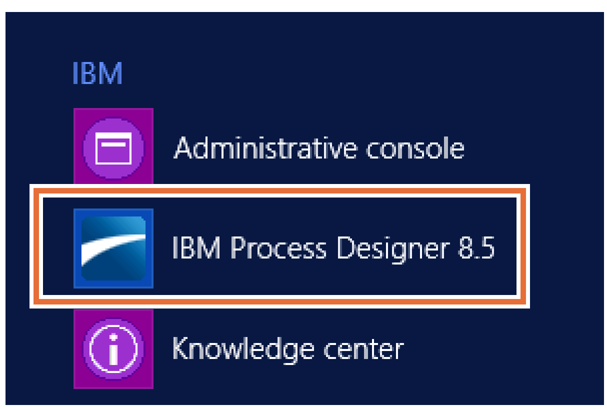
__ 1. Start the **IBM Process Designer** client application.

Note

For IBM BPM on Cloud users: To obtain and install the Process Designer client application, you must log in to your IBM BPM Cloud instance. At the bottom of the Process Center tile, you find a link to download Process Designer. The version that you download from IBM BPM on Cloud is already configured to communicate with your IBM BPM on Cloud Process Center.

__ a. Click the Windows **Start** menu.

__ b. Click the **Down Arrow > IBM > IBM Process Designer 8.5**.



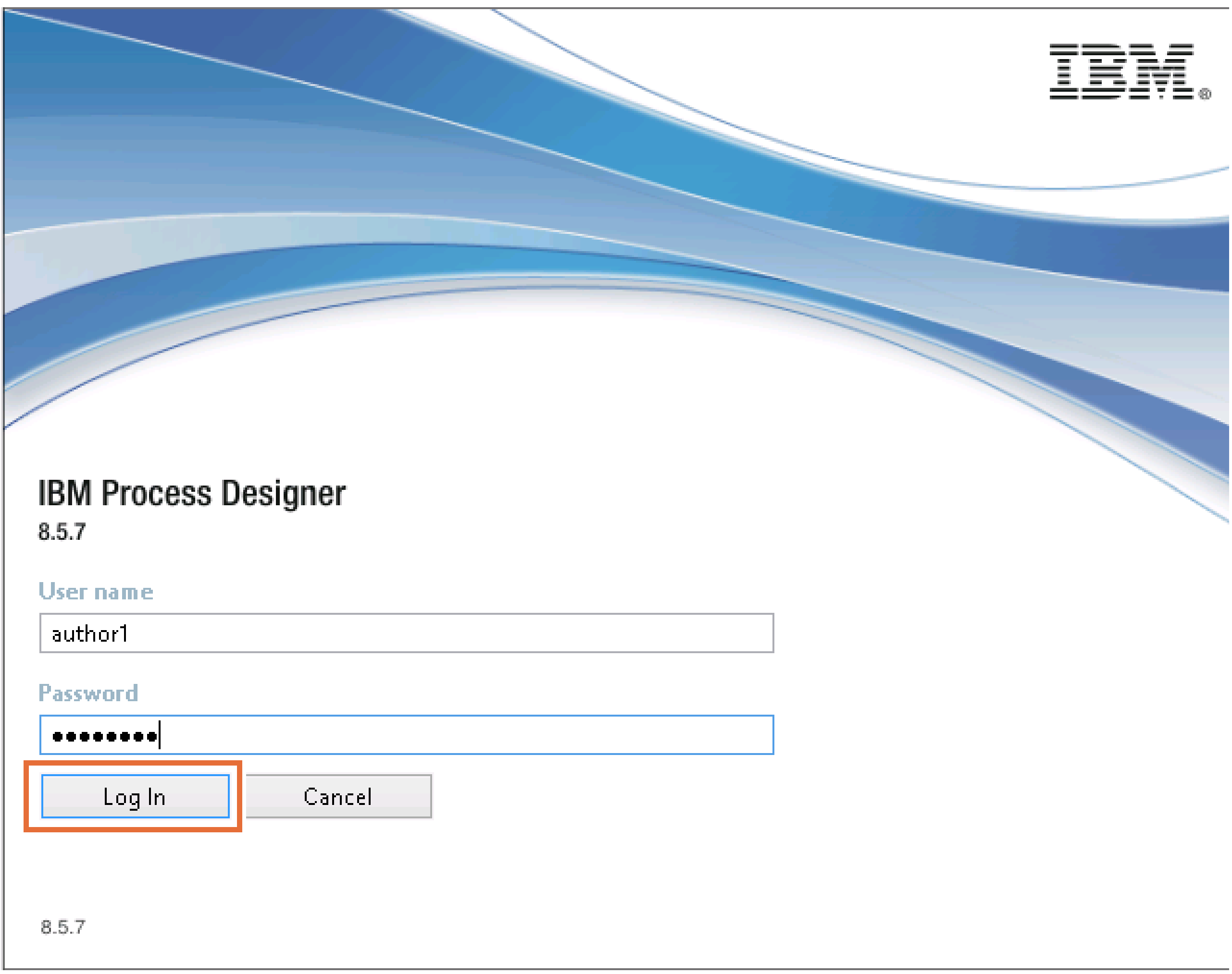
Note

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

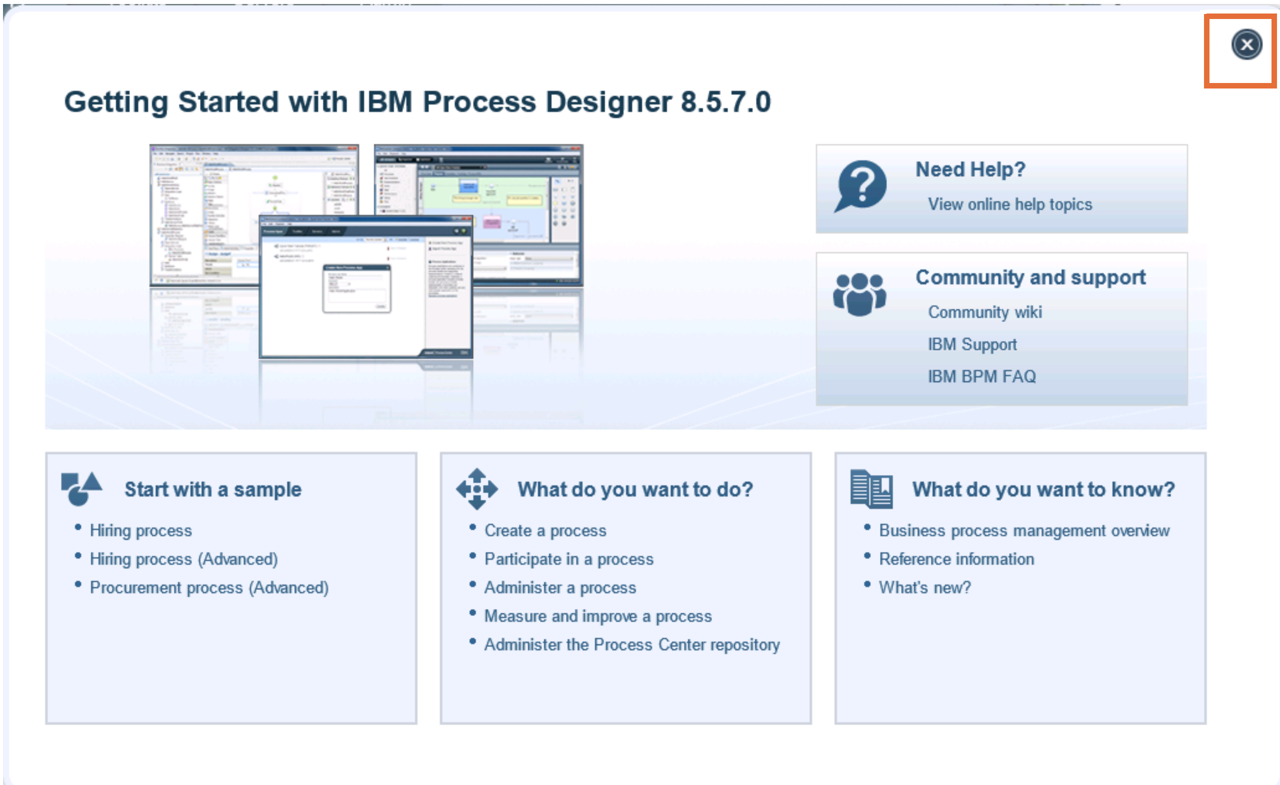
__ c. A loading screen and then a login screen are displayed for the IBM Process Designer client application.

__ d. Enter `author1` for the **User name** and `author01` for the **Password**.

__ e. Click **Log In**.

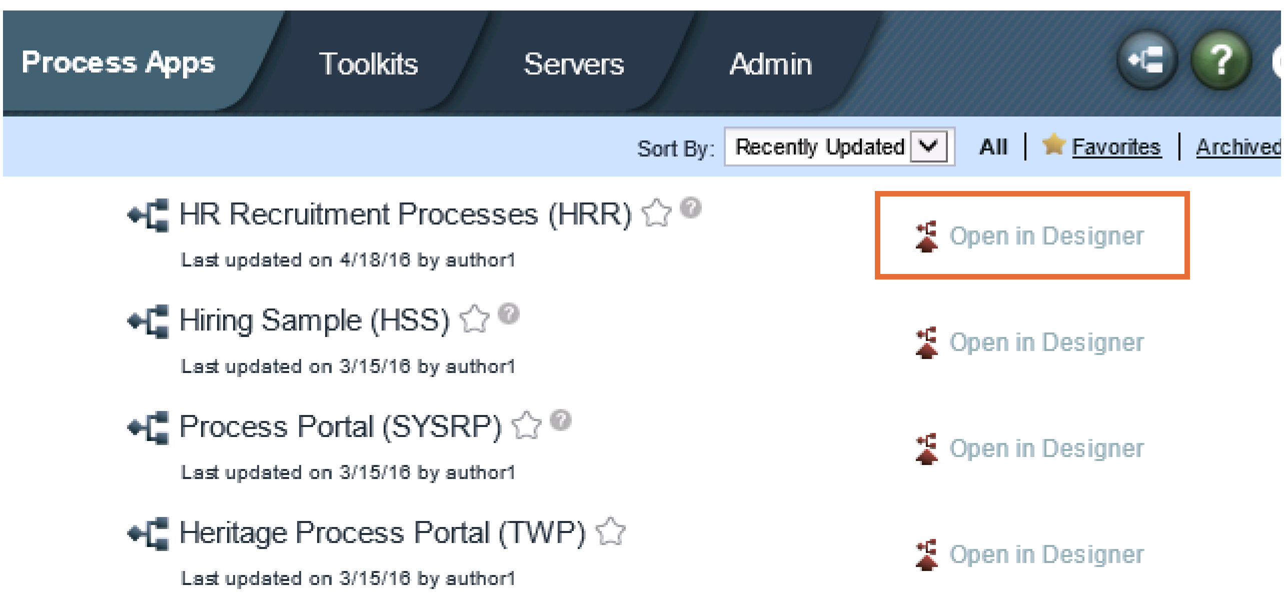


__ f. Close the **Getting Started** window that appears the first time you open the IBM Process Designer client application.

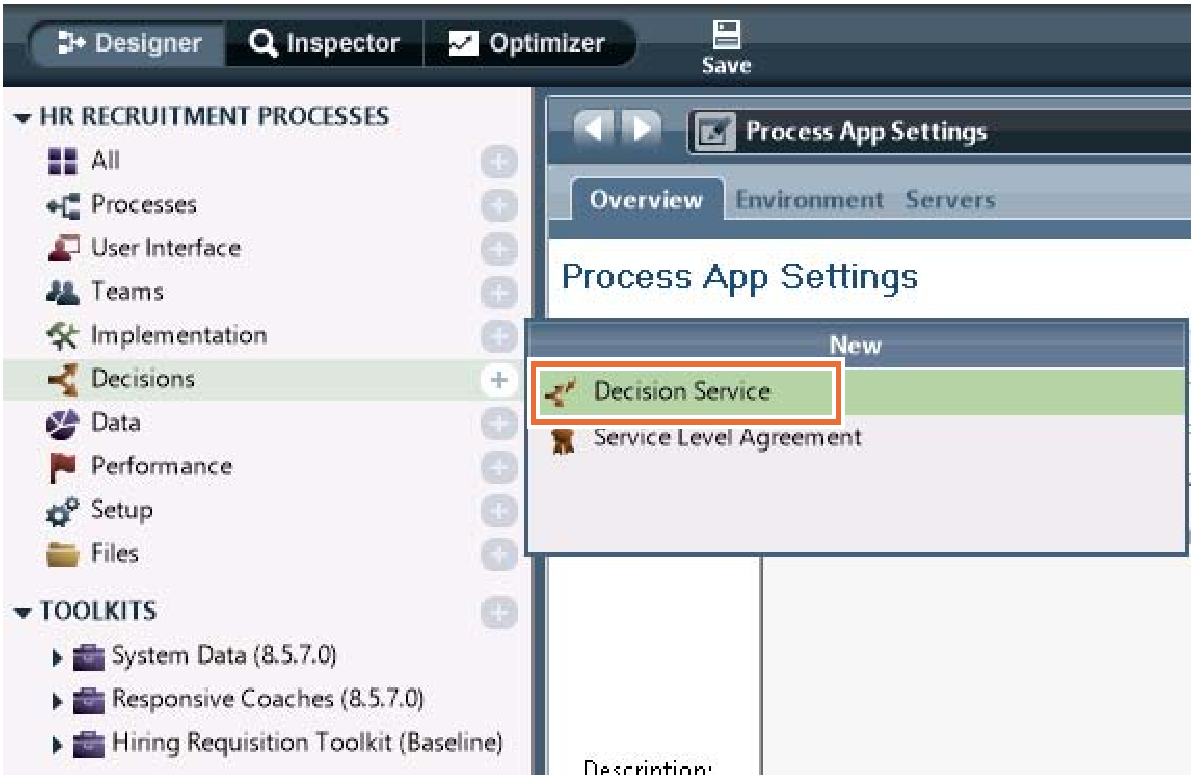


__ 2. Create a decision service.

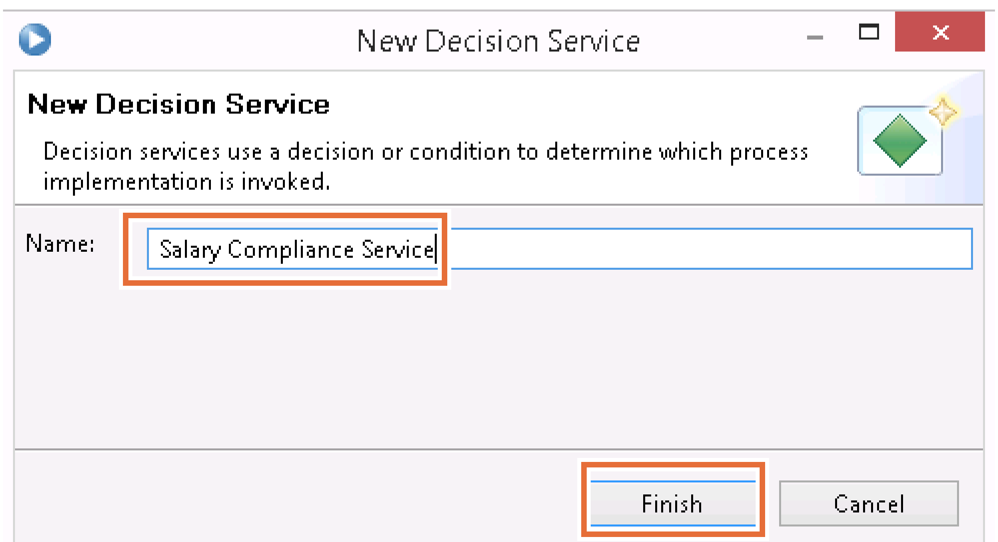
__ a. Open the HR Recruitment Process (HRR) inside the IBM Process Designer client application by clicking **Open in Designer**.



__ b. In the Process Designer client application library, click the (**+**) plus sign next to Decisions. Click **Decision Service** to create a service.

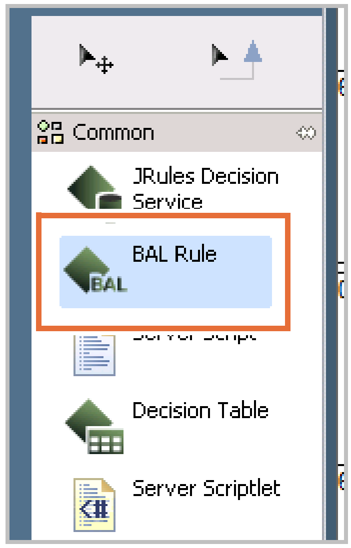


__ c. Name the decision service `Salary Compliance Service` and click **Finish**.

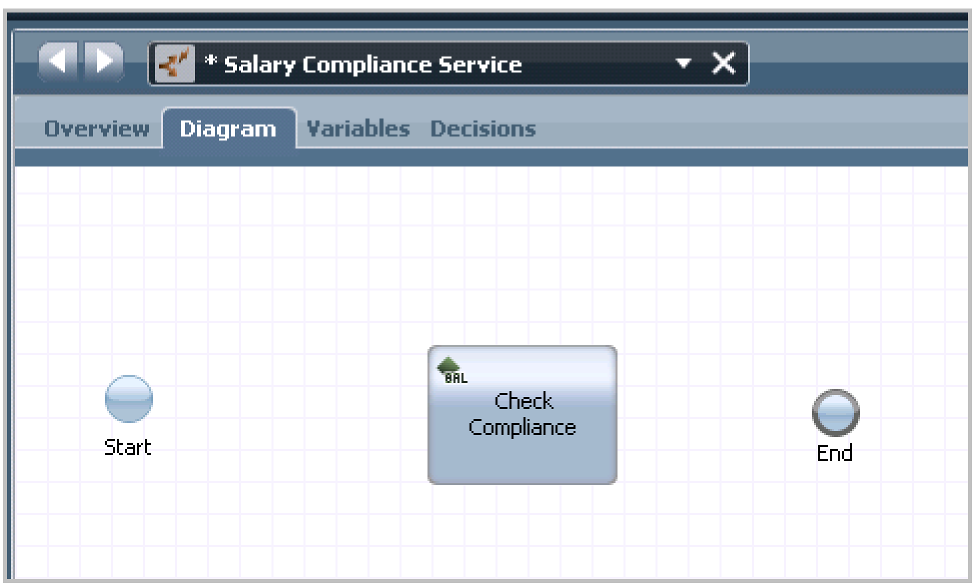


__ 3. Add a Business Action Language rule.

__ a. You are now in the new service. Drag a **BAL Rule** from the service palette onto the canvas between the Start and End events.



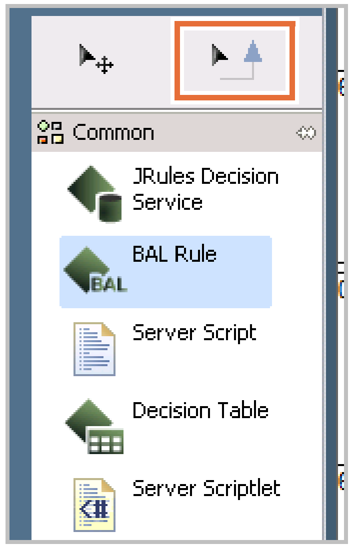
__ b. Name the **BAL Rule**: `Check Compliance`



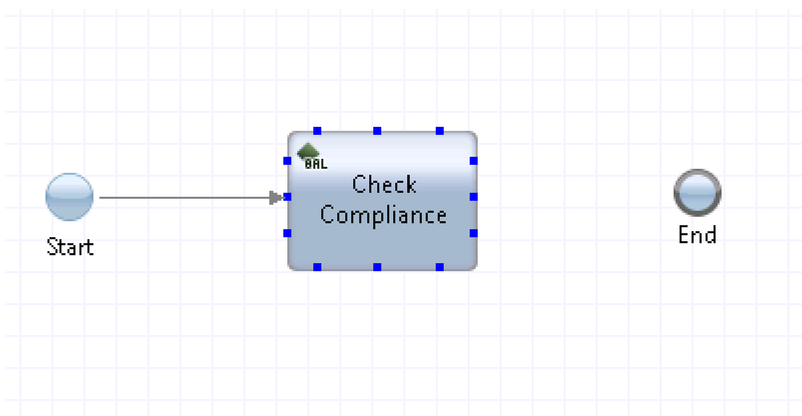
Reminder

You can rename artifacts in the Process Designer client application the same way as you rename the artifacts in the web Process Designer: either through the **Properties > General > Common** section, or by double-clicking the artifact and renaming it on the canvas.

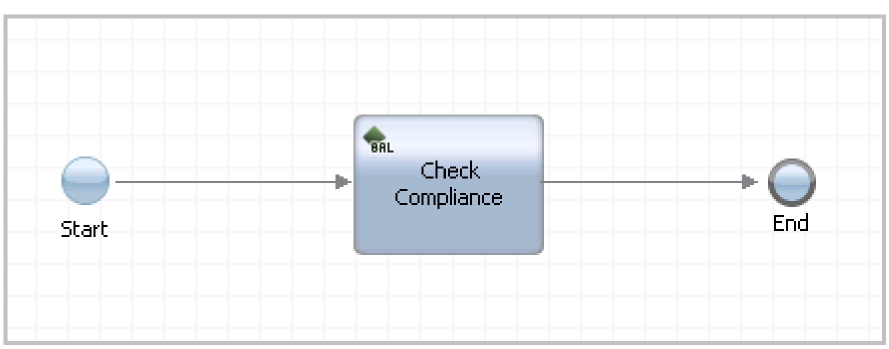
__ c. Use the sequence flow tool to connect the sequence flow from left to right. Click the **sequence flow icon** above the palette.



__ d. Beginning at the **Start** event, click an anchor **point** first and then click the **Check Compliance** activity connection point to connect the Start event to the Check Compliance activity.



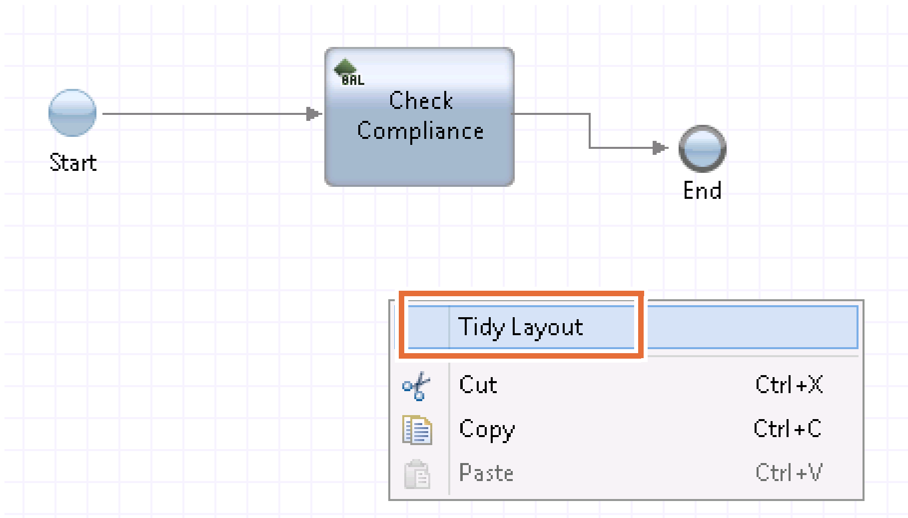
__ e. Connect the **Check Compliance** step to the **End** event. Press the Esc key on the keyboard to return to the sequence flow tool, or you can click the selection tool that is located next to the sequence flow icon.



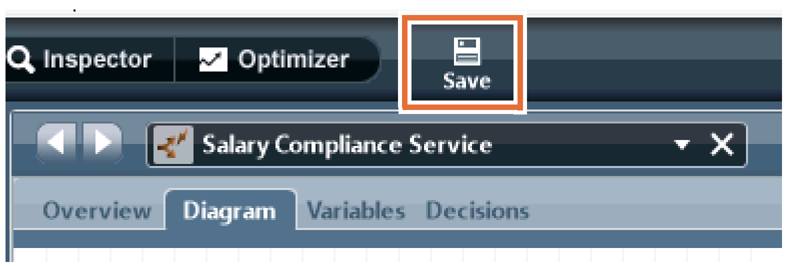
Note

To straighten the flow lines, select the item on the canvas and while holding the **Ctrl** key use the arrow key on the keyboard to align the nodes.

You can also align nodes by first ensuring no items are selected on the canvas. Then, right-click an area on the canvas that does not have any artifacts, and click **Tidy Layout** to align all the objects on the canvas.



__ f. Click **Save** to save the service.

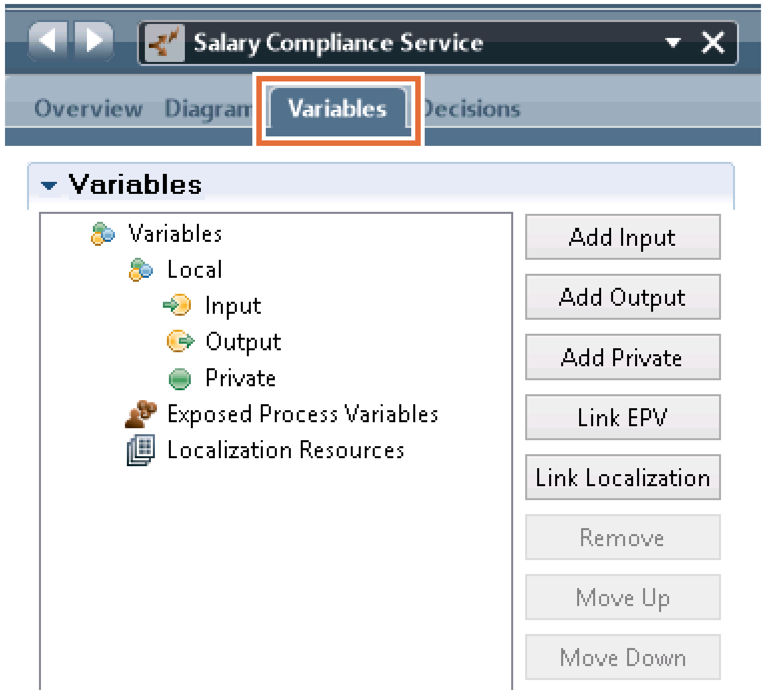


Hint

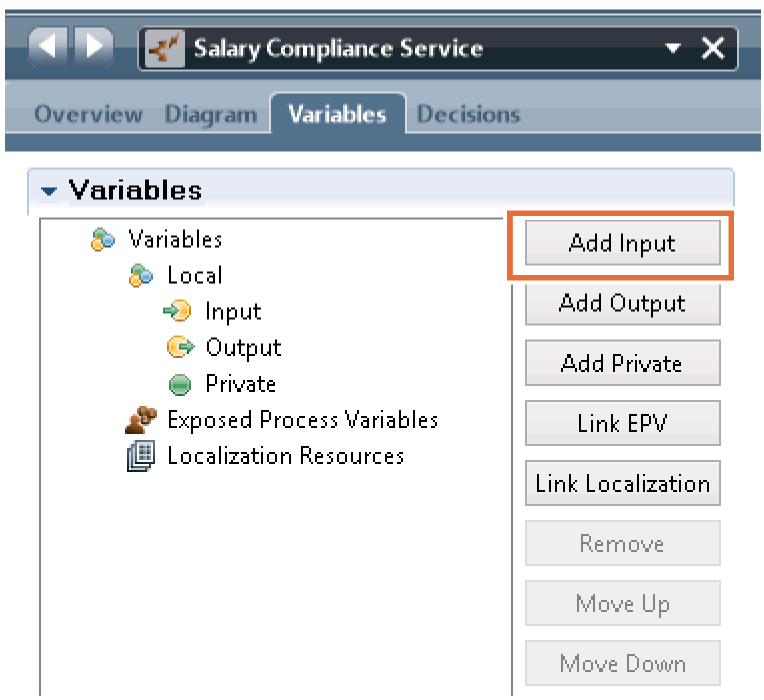
Similar to the web Process Designer, you can also use the Ctrl + S keyboard shortcut to save the Process Application changes.

__ 4. Add variables to the service.

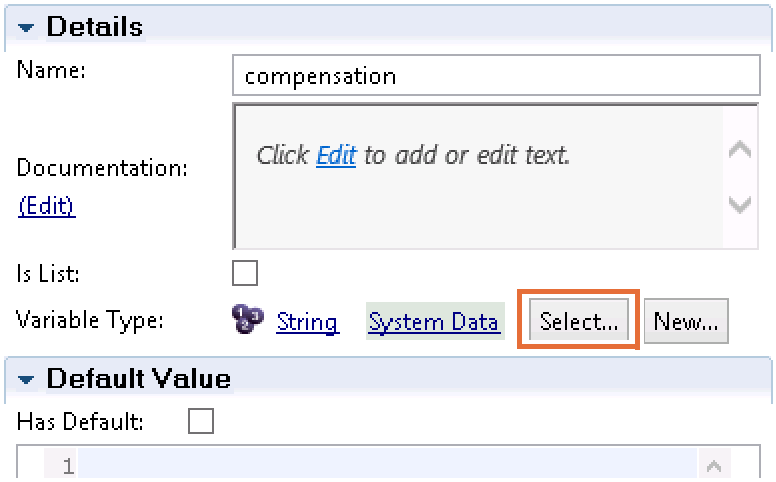
__ a. Click the **Variables** tab.



__ b. Click **Add Input** and name the variable: `compensation`

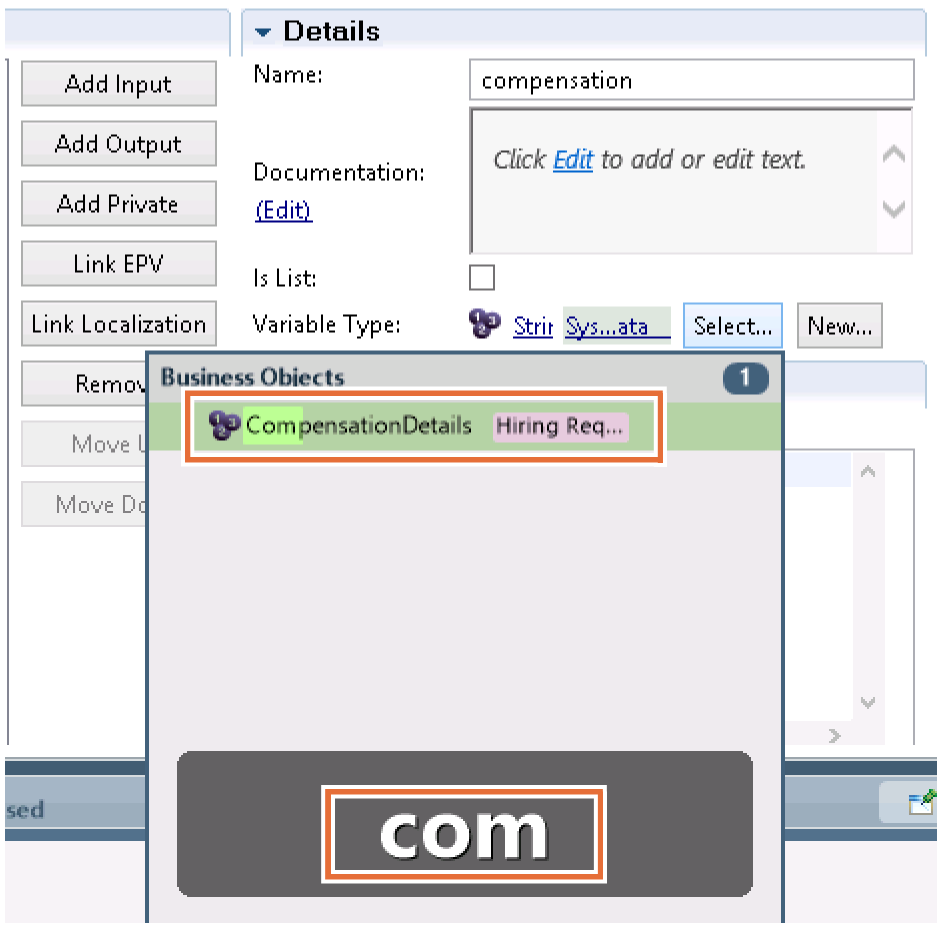


__ c. Click **Select** next to the Variable type.

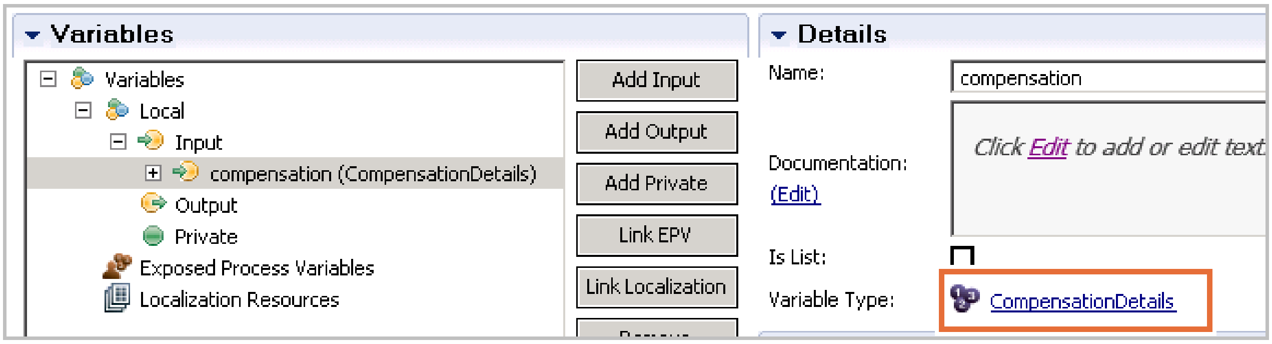


__ d. To filter the variable type list, type: `Com`

The variable type `CompensationDetails (Hiring Requisition)` is listed.

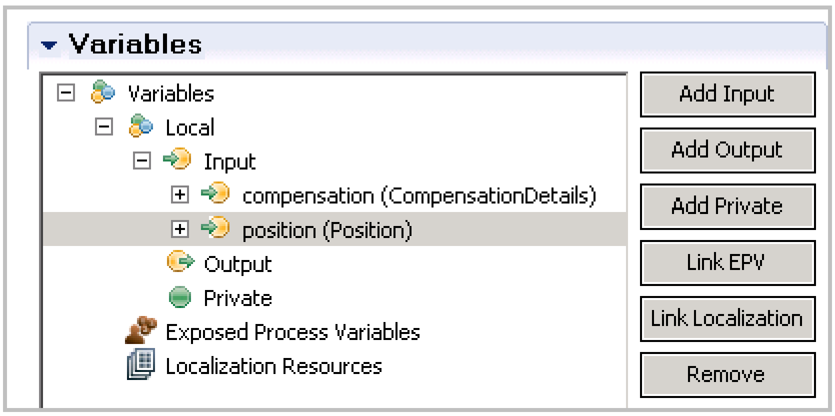


__ e. Select **CompensationDetails** as the variable type.



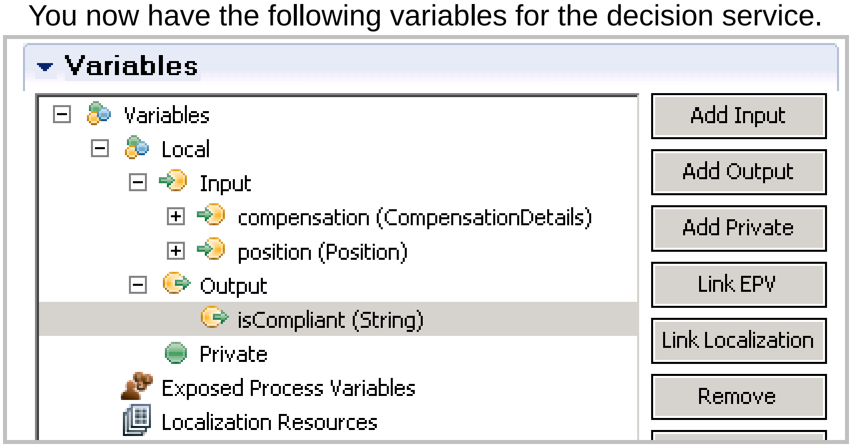
__ f. Add a second variable by clicking **Add Input** and name the variable: `position`

__ g. Click **Select** next to variable type and select **Position** as the variable type.



__ h. Click **Add Output** and create the variable: `isCompliant (String)`

You now have the following variables for the decision service.

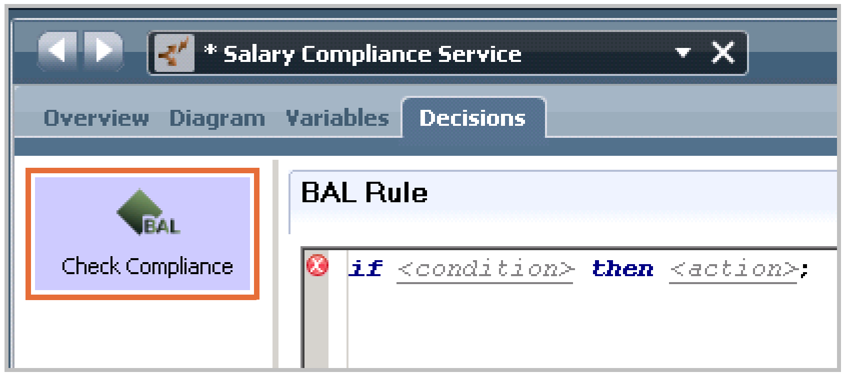


__ i. Save your changes.

__ 5. 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.

__ 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 ...

<string>

"Associate"

__ d. Select the **<action>** statement of your rule, and select the following rule pieces:

set <variable> to <variable value>

isCompliant

to

<string>

"1" ;

else <space>

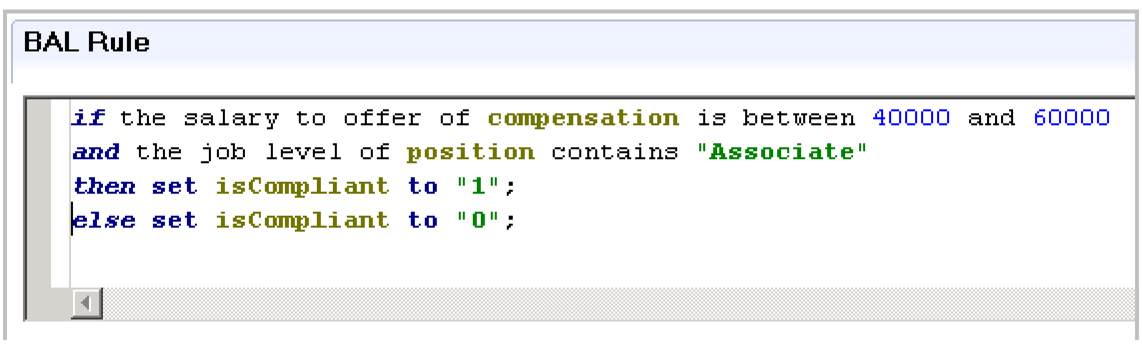
set <variable>

isCompliant

to <variable value>

<string>

"0" ;

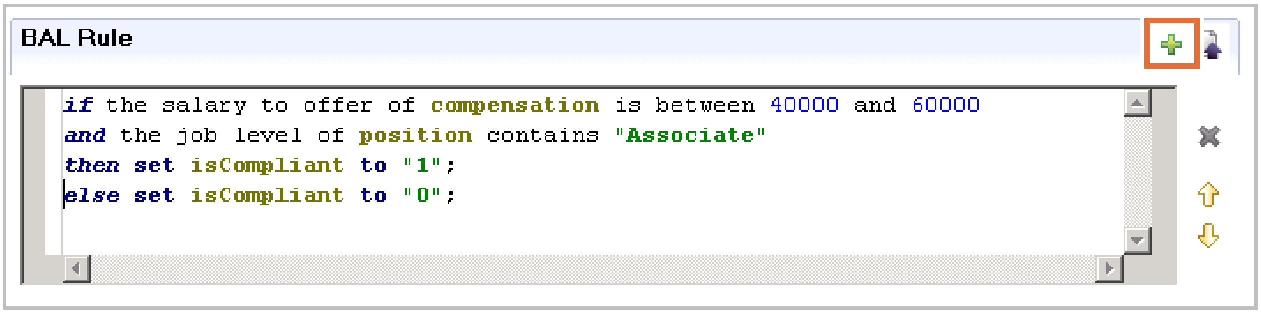


After the BAL rule is built, it reads much like a natural sentence. You assign a value in the first rule, and as the other rules below it evaluate, if one of them evaluates to true, then it overwrites the variable.

Note

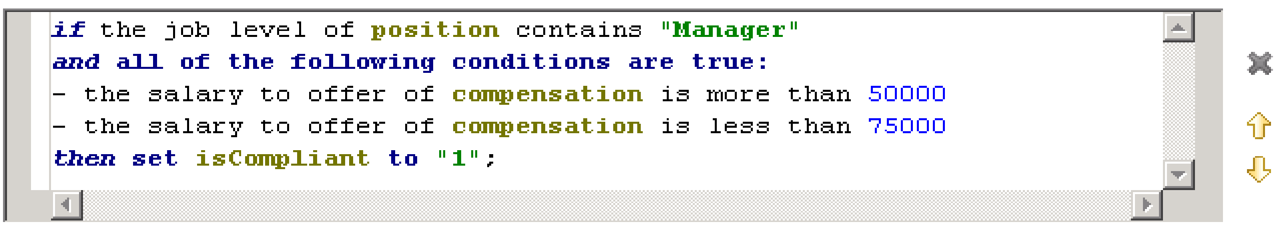
To avoid any typographical errors, the code for the rules is provided in the file BAL Rules.txt at the location: C:\labfiles\Exercise_Support_Files\Exercise09\

__ e. Click the (**+**) plus symbol to create a second rule.



__ f. Build the next rule by using the previous method to flag the salary as non-compliant if 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";
```



__ 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 code for the rules is provided in the file `BAL Rules.txt` at the location: `C:\labfiles\Exercise_Support_Files\Exercise09\`.

All the four rules are created.

BAL 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";
```

```
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 "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";
```

```
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";
```

__ 6. Save your work.

Information

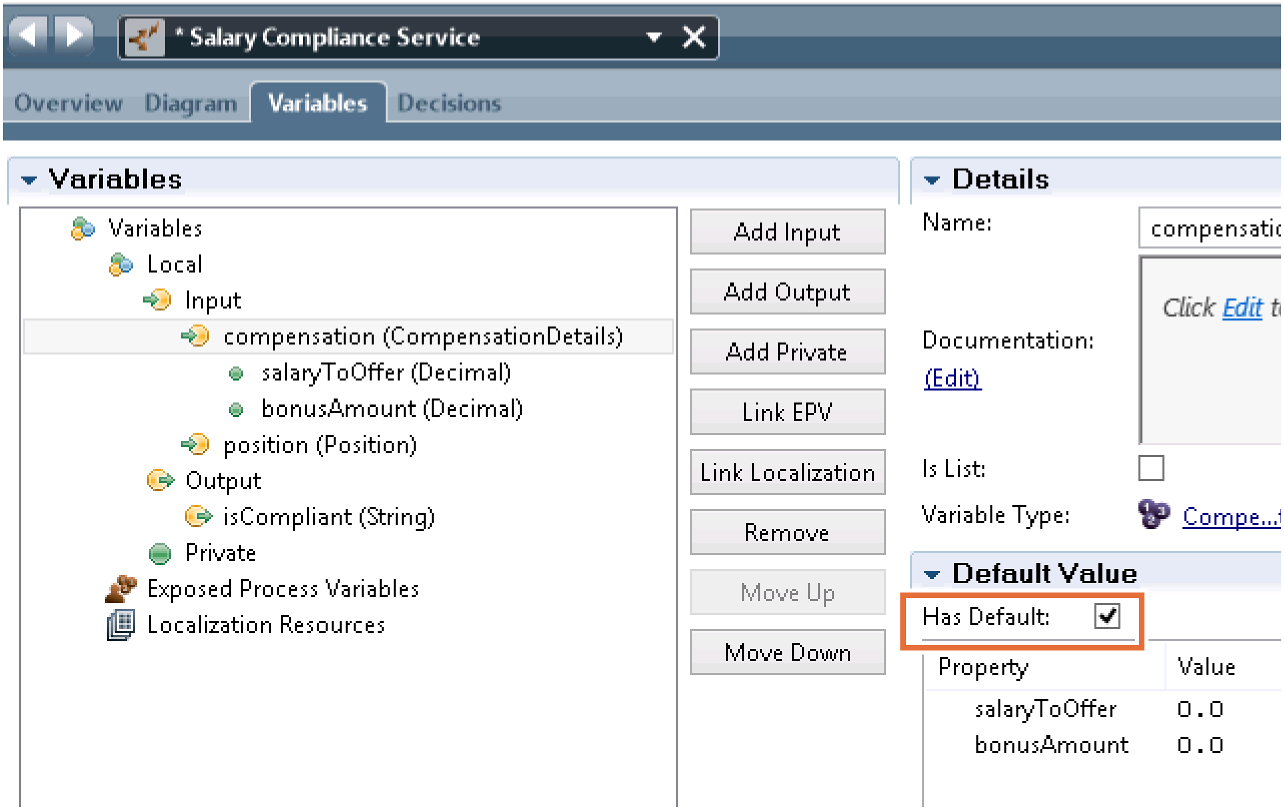
If this service was a correct implementation, you might consider a decision table to implement the rule requirements instead of “if” statements. BAL rules are more powerful and flexible than table rules, so after you master the concepts of BAL rule building, decision tables are an easier application of rules.

__ 7. 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.

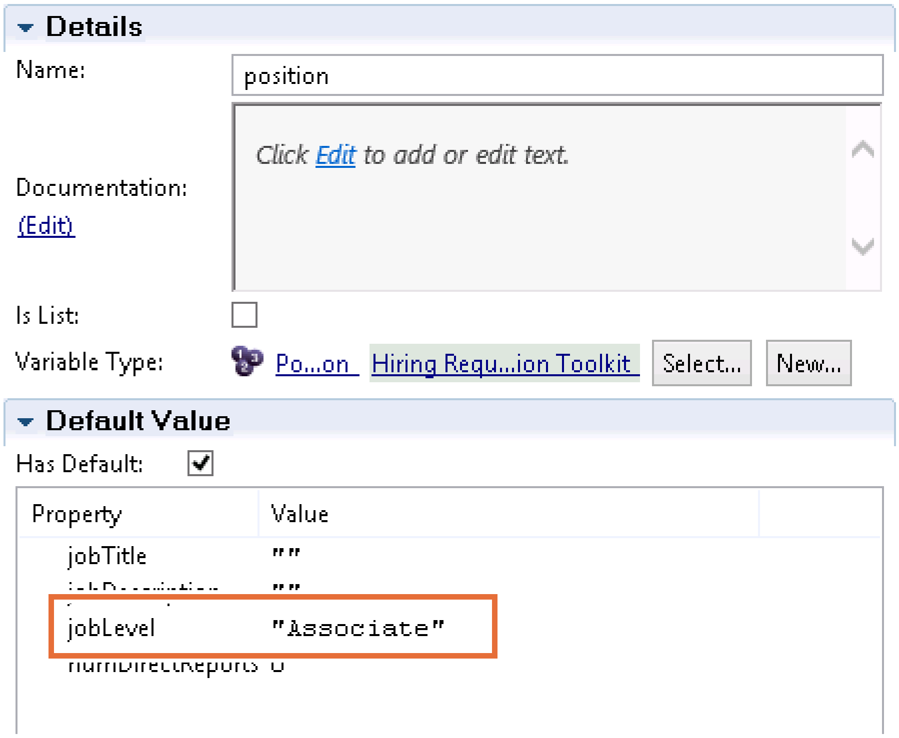
__ b. Click the input variable **compensation (CompensationDetails)**. Select **Has Default** under the Default Value section on the right.



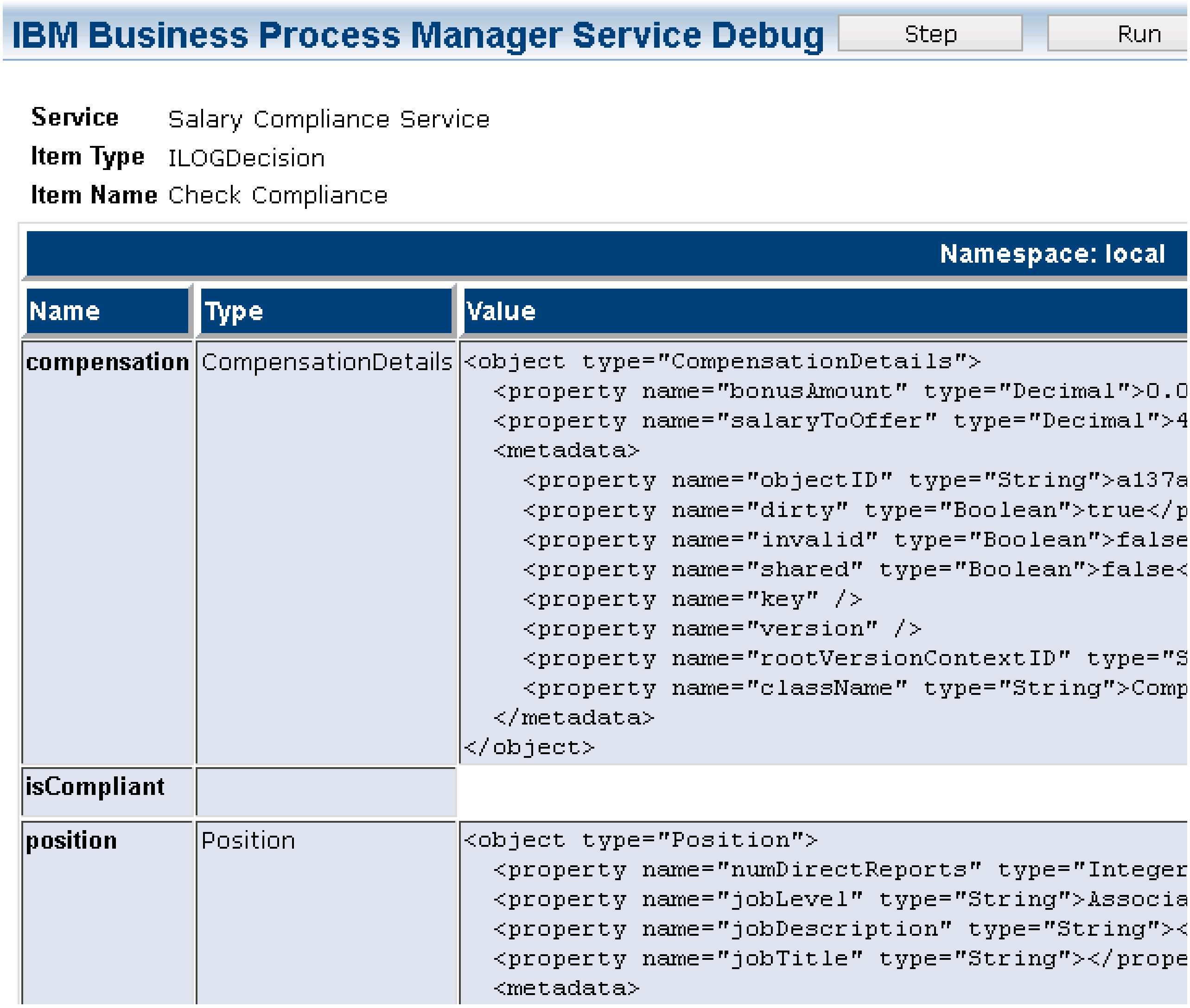
__ c. Enter the value for the variable **salaryToOffer** as: `45000.0`

__ d. Click the input variable **position (Position)**. Select **Has Default** under the Default Value section on the right.

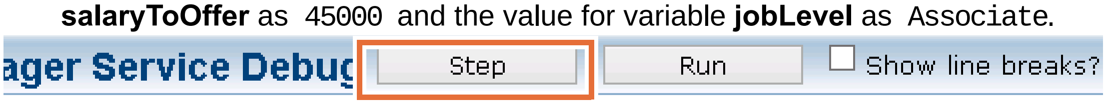
__ e. Enter the value for the variable **jobLevel** as: `"Associate"`



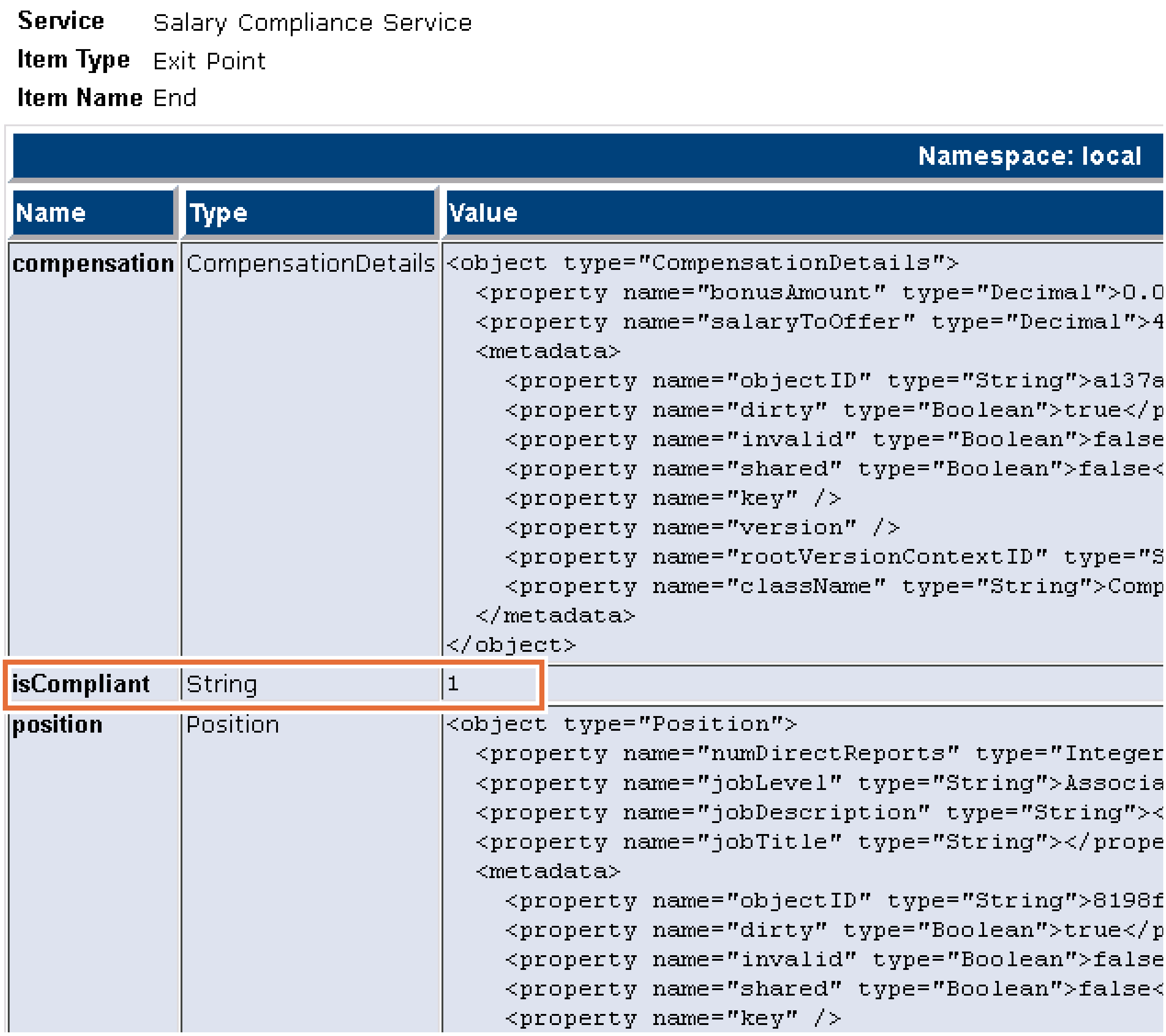
__ f. Save the service and click the **Debug** button on the top. The service opens in a browser window.



__ g. Click **Step** 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`.



__ h. 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.

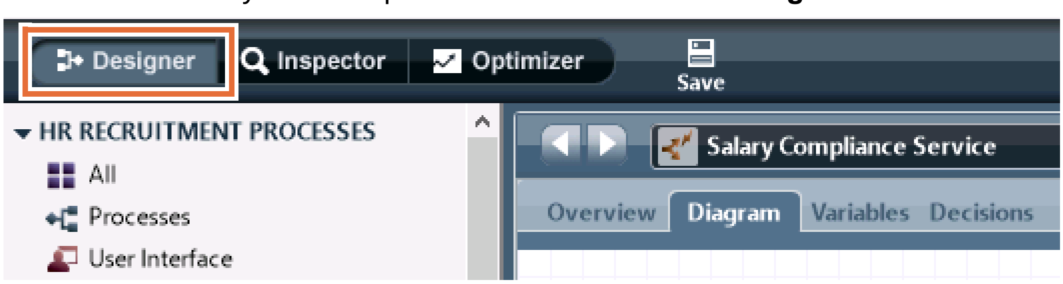


Service Salary Compliance Service
Item Type Exit Point
Item Name End

Namespace: local		
Name	**Type**	**Value**
compensation	CompensationDetails	<object type="CompensationDetails"> <property name="bonusAmount" type="Decimal">0.0 <property name="salaryToOffer" type="Decimal">4 <metadata> <property name="objectID" type="String">a137a <property name="dirty" type="Boolean">true</p <property name="invalid" type="Boolean">false <property name="shared" type="Boolean">false< <property name="key" /> <property name="version" /> <property name="rootVersionContextID" type="S <property name="className" type="String">Comp </metadata> </object>
isCompliant	String	1
position	Position	<object type="Position"> <property name="numDirectReports" type="Integer <property name="jobLevel" type="String">Associa <property name="jobDescription" type="String">< <property name="jobTitle" type="String"></prope <metadata> <property name="objectID" type="String">8198f <property name="dirty" type="Boolean">true</p <property name="invalid" type="Boolean">false <property name="shared" type="Boolean">false< <property name="key" />

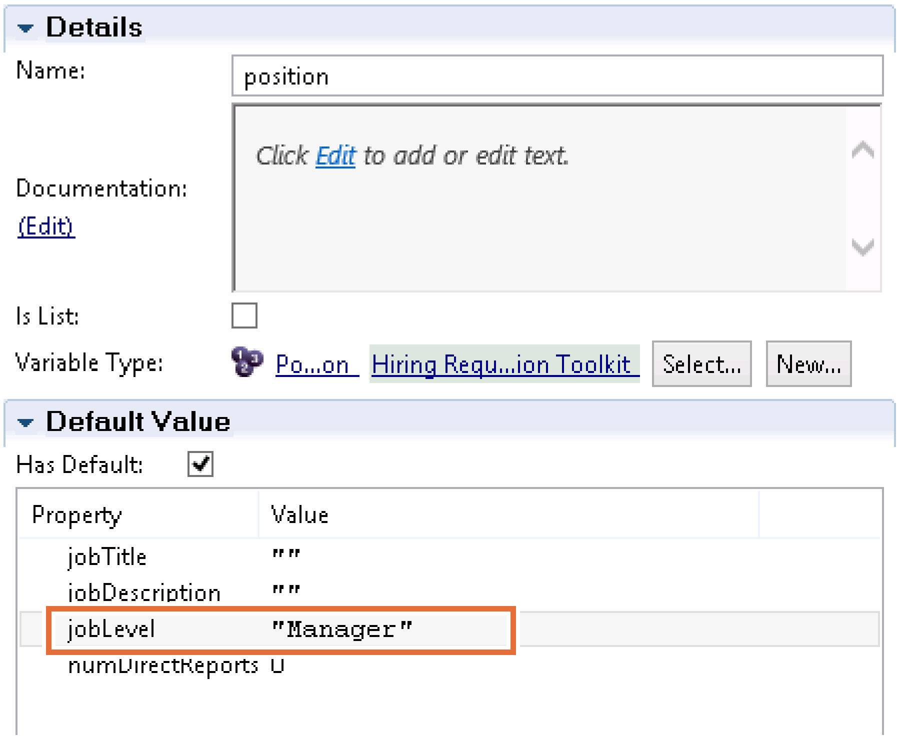
__ i. Close the browser window and return to the Process Designer client application.

__ j. You are currently in the Inspector view. Return to the **Designer** view.



__ k. Click the **Variables** tab and select the **position (Position)** input variable.

__ l. Change the value of the **jobLevel** variable to `"Manager"` and save the service. Click the **Debug** button on the top.



__ m. The service opens in the browser window. Click the **Step** button on the top.

__ n. 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

Service Salary Compliance Service
Item Type Exit Point
Item Name End

Namespace: local		
Name	**Type**	**Value**
compensation	CompensationDetails	<object type="CompensationDetails"> <property name="bonusAmount" type="Decimal">0.0 <property name="salaryToOffer" type="Decimal">4 <metadata> <property name="objectID" type="String">a137a <property name="dirty" type="Boolean">true</p <property name="invalid" type="Boolean">false <property name="shared" type="Boolean">false< <property name="key" /> <property name="version" /> <property name="rootVersionContextID" type="S <property name="className" type="String">Comp </metadata> </object>
isCompliant	String	1
position	Position	<object type="Position"> <property name="numDirectReports" type="Integer <property name="jobLevel" type="String">Associa <property name="jobDescription" type="String">< <property name="jobTitle" type="String"></prope <metadata> <property name="objectID" type="String">8198f <property name="dirty" type="Boolean">true</p <property name="invalid" type="Boolean">false <property name="shared" type="Boolean">false< <property name="key" />

Service Salary Compliance Service
Item Type Exit Point
Item Name End

Namespace: local		
Name	**Type**	**Value**
compensation	CompensationDetails	<object type="CompensationDetails"> <property name="bonusAmount" type="Decimal">0.0 <property name="salaryToOffer" type="Decimal">45 <metadata> <property name="objectID" type="String">61b8e <property name="dirty" type="Boolean">true</p <property name="invalid" type="Boolean">false <property name="shared" type="Boolean">false</ <property name="key" /> <property name="version" /> <property name="rootVersionContextID" type="St <property name="className" type="String">Compe </metadata> </object>
isCompliant	String	0
position	Position	<object type="Position"> <property name="numDirectReports" type="Integer" <property name="jobLevel" type="String">Manager <property name="jobDescription" type="String"></ <property name="jobTitle" type="String"></proper <metadata>

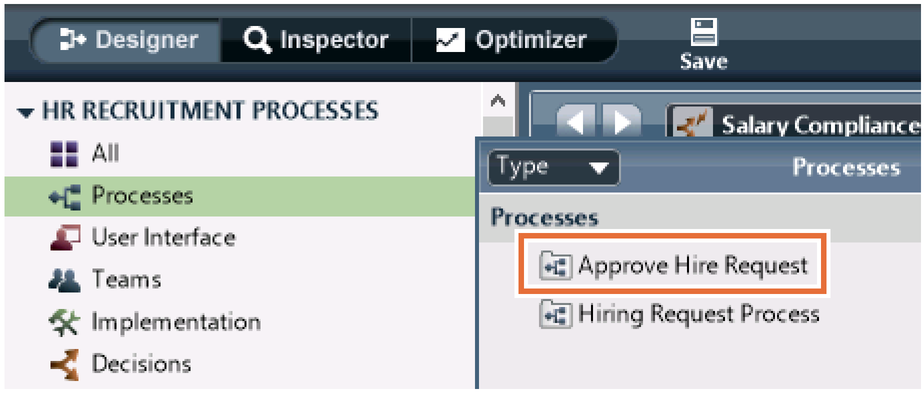
__ o. Close the browser window and return to the Designer view of the Process Designer client application.

__ p. Click the **Variables** tab and clear the **Has Default** field under the **Default Value** section for the input variables **compensation (CompensationDetails)** and **position (Position)**.

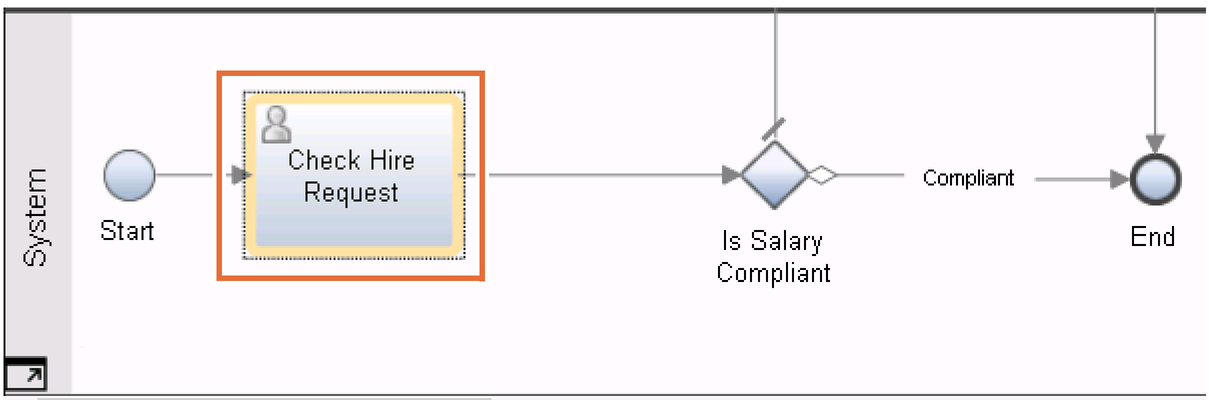
__ q. Save your changes.

__ 8. Add the rule to the process.

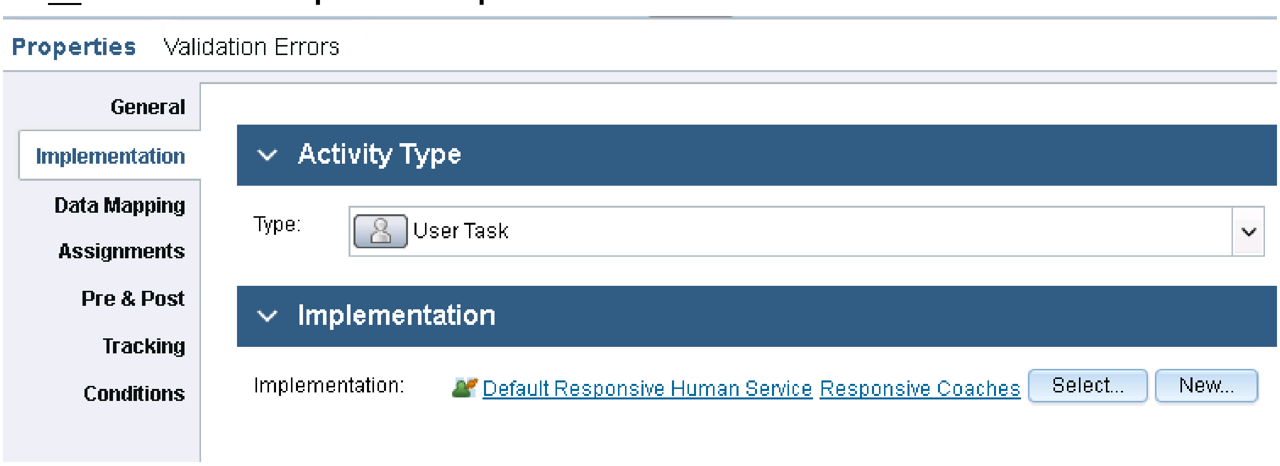
__ a. Open the **Approve Hire Request** process by clicking **Processes** in the Process Designer client application library and double-clicking **Approve Hire Request**.



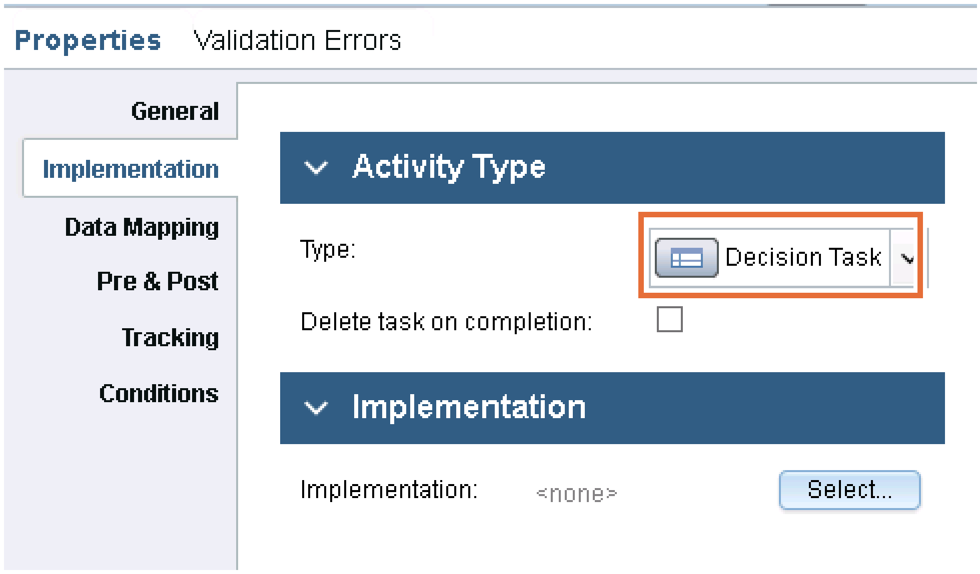
__ b. The process opens in the web Process Designer. Select the **Check Hire Request** activity in the `System` lane.



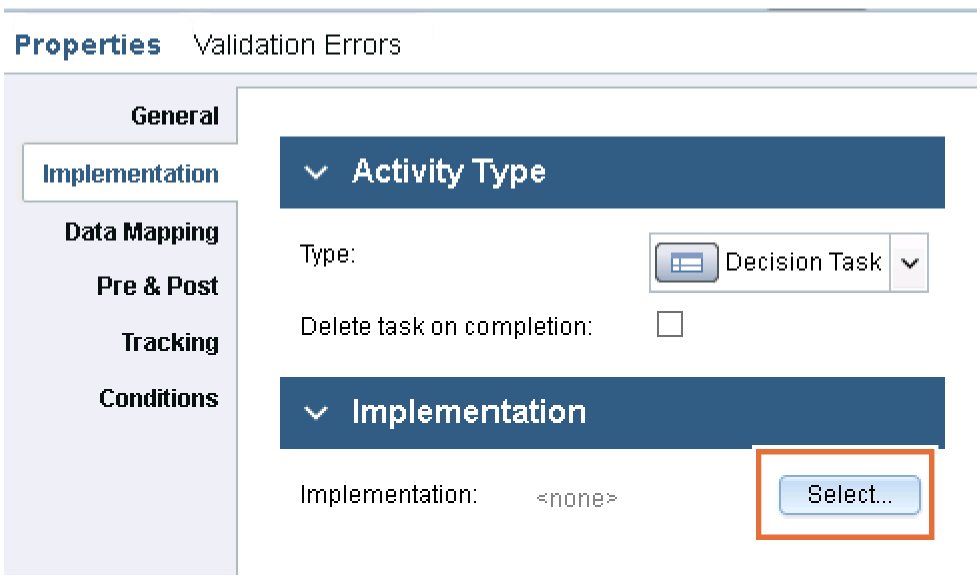
__ c. Click the **Properties > Implementation** menu.



__ d. Change the activity type to **Decision Task**.



__ e. Click **Select** in the **Implementation** section.

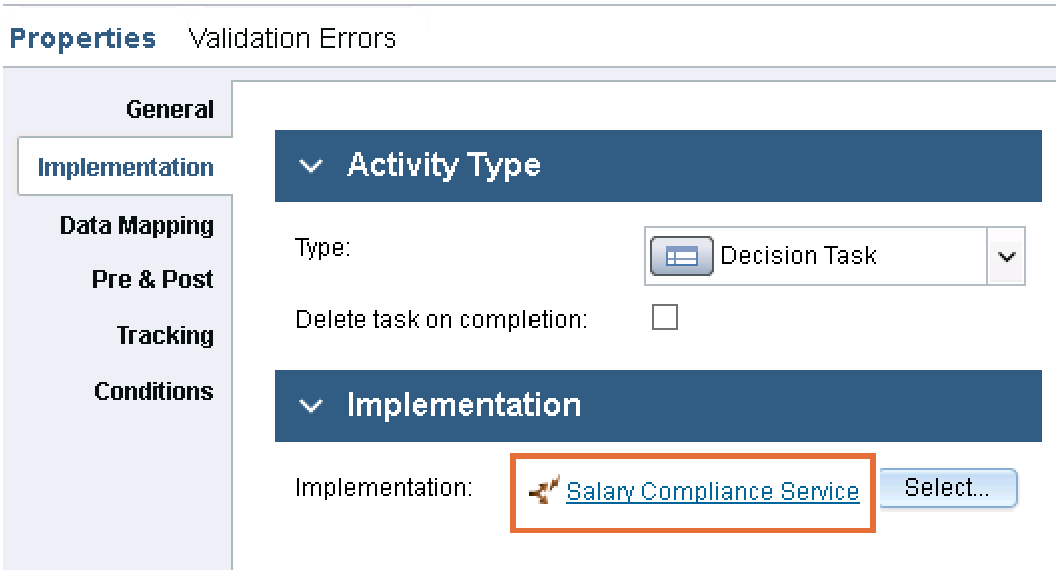


Information

You select **Delete task on completion** if you want to run an automated service that does not require routing and deletes the task from the system and the process inspector after it completes. When you select this check box, the Process Server does not retain audit data for the task. By default, this option is disabled.

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**.

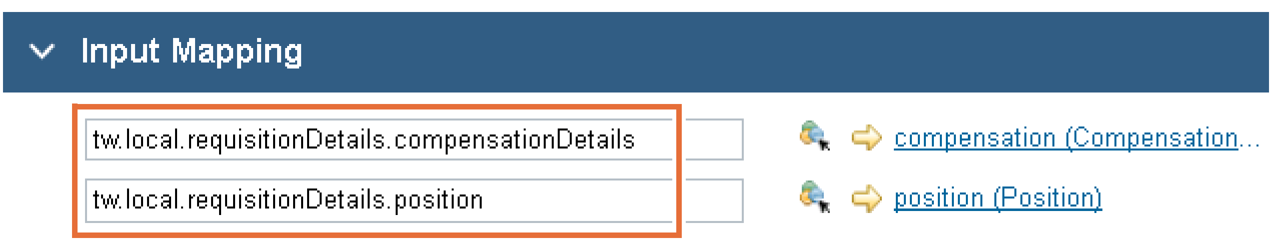


__ 9. Map the variables for the **Check Hire Request** activity.

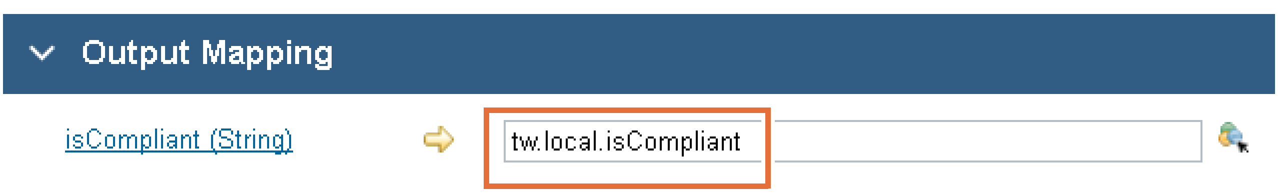
__ a. Open the **Properties > Data Mapping** menu.

__ 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**.



__ d. Enter `tw.local.isCompliant` for the output to map it with **isCompliant**.



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.

__ f. Minimize the browser window.

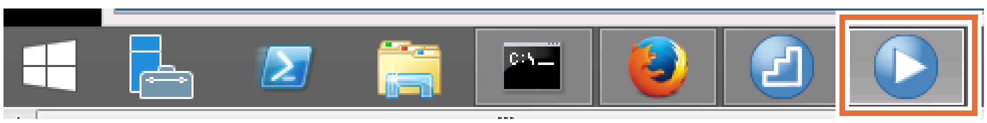
Part 2: Implement a message start event

In the **Hiring Request Process** modeling, you created a process that communicated how a process instance is started with an online data entry by either customers or customer representatives, or through 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.

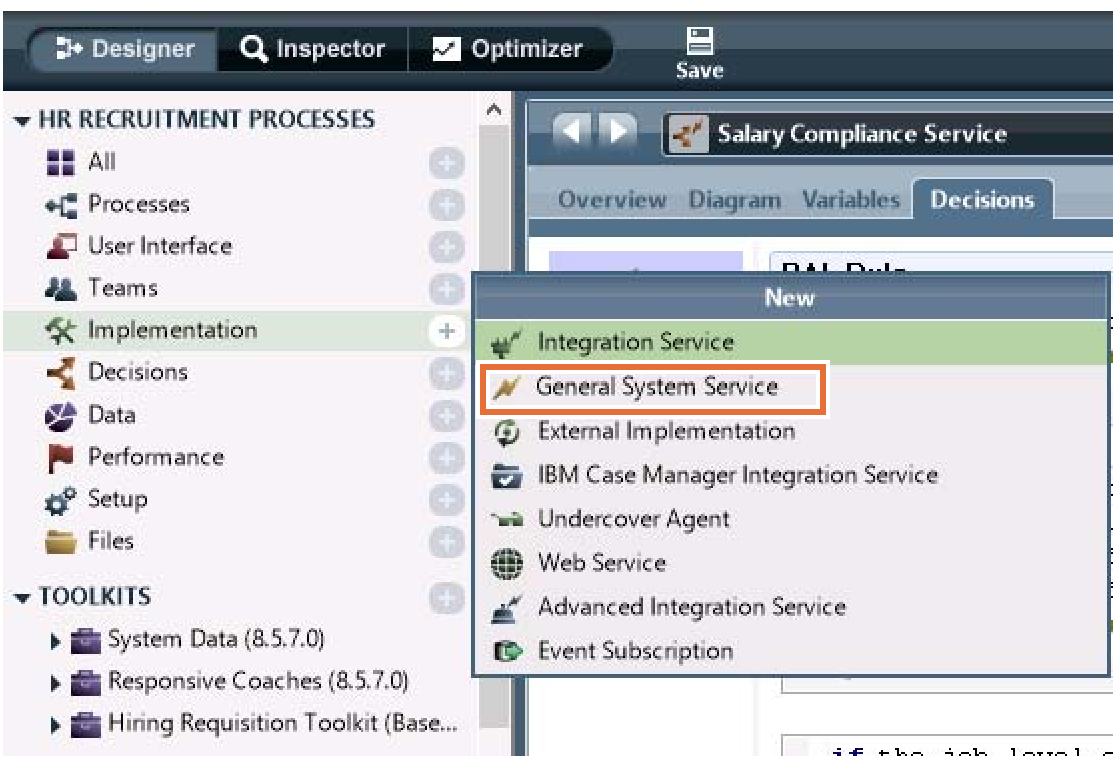
__ 1. Return to the IBM Process Designer client application.



__ 2. Create a **General System Service** to enable the undercover agent (UCA).

__ a. In the Process Designer client application library, click the (**+**) plus sign next to the **Implementation** menu option.

__ b. Click the **General System Service** option.

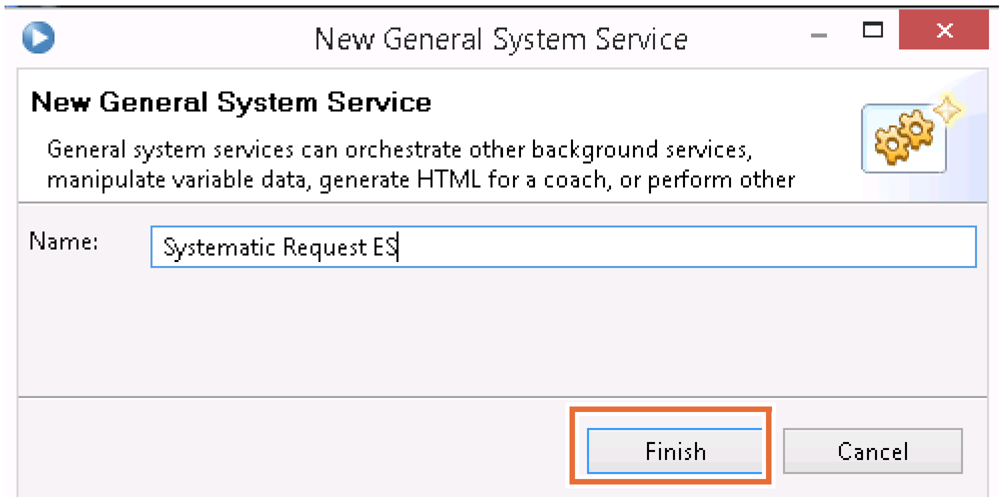


__ c. Name the service: `Systematic Request ES`

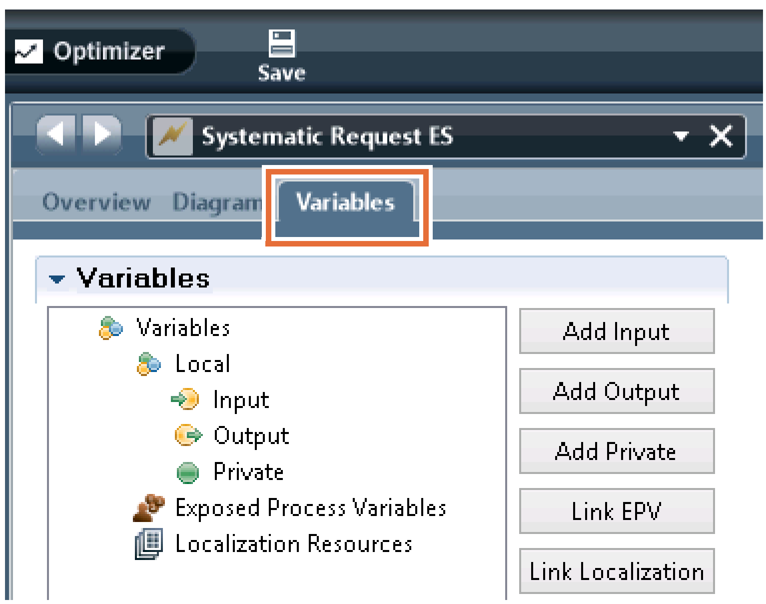
Information

Generally, the enabling service for a UCA is suffixed with `ES` for `Enabling Service`.

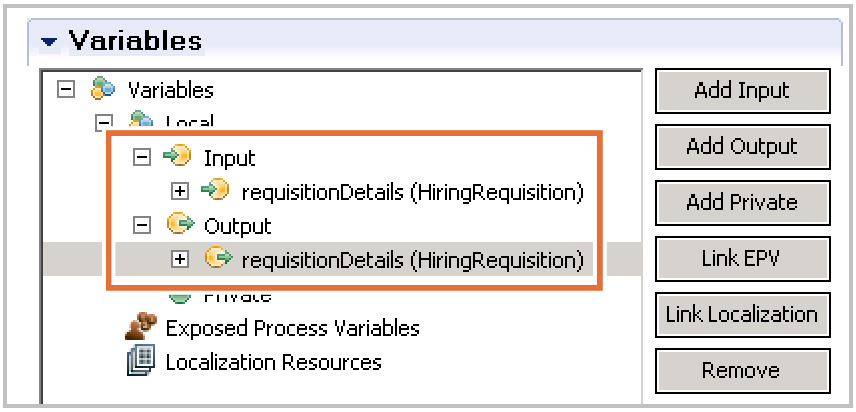
__ d. Click **Finish**.



__ e. Click the **Variables** tab for the general system service.



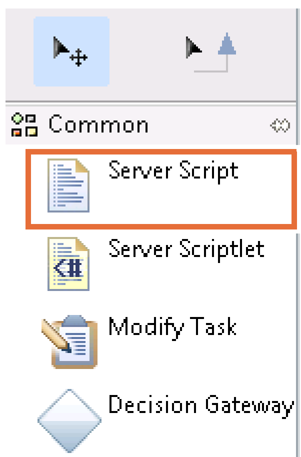
__ f. Following the step 4 from Part 1 of this exercise, create a `requisitionDetails (HiringRequisition)` input and output variable.



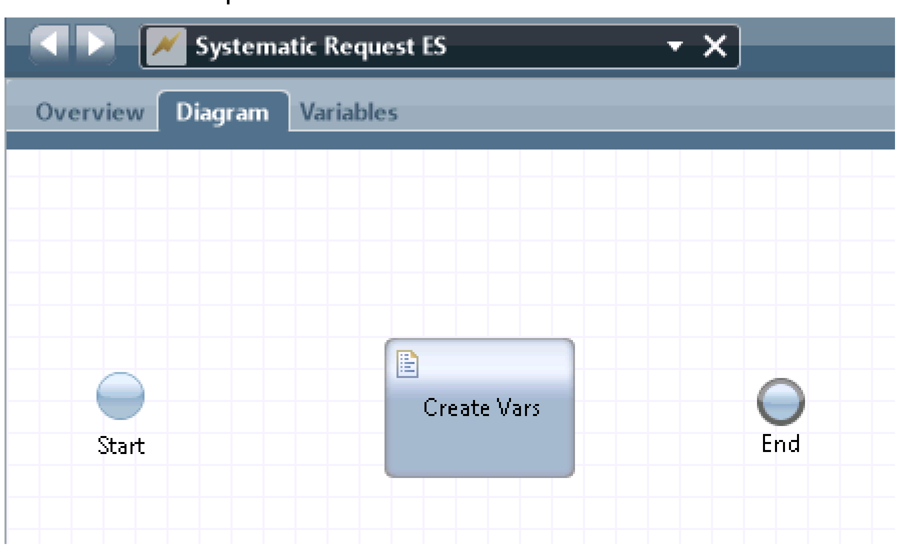
__ g. Save your work.

__ 3. Create a server script to initialize the variables and pass it out of the service.

__ a. Return to the **Diagram** tab and then drag a server script from the palette onto the canvas.

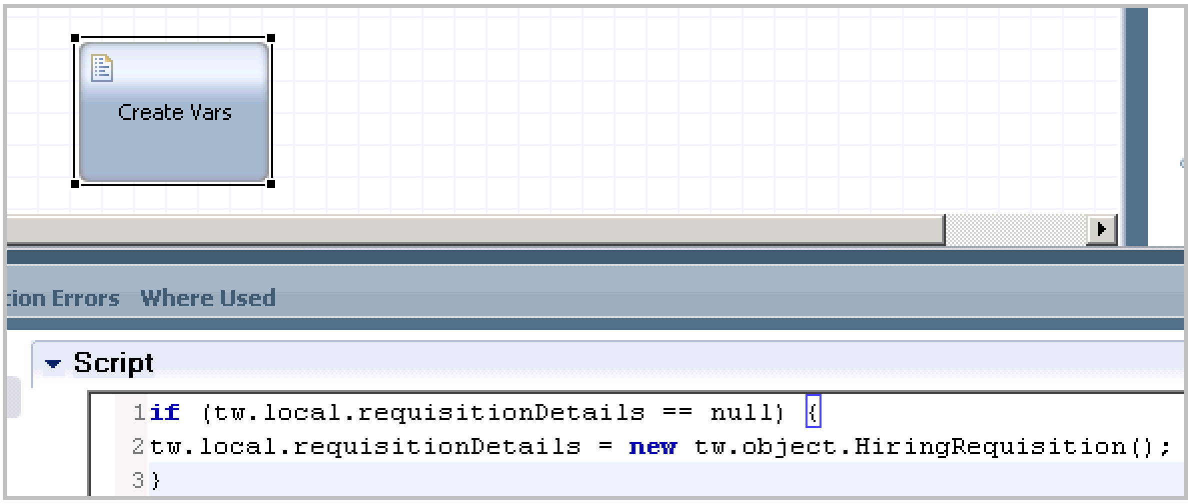


__ b. Name the server script: `Create Vars`



__ c. Create a server script to initialize the variables and pass it out of the service. In the **Properties > Implementation** menu, enter the following script in the box:

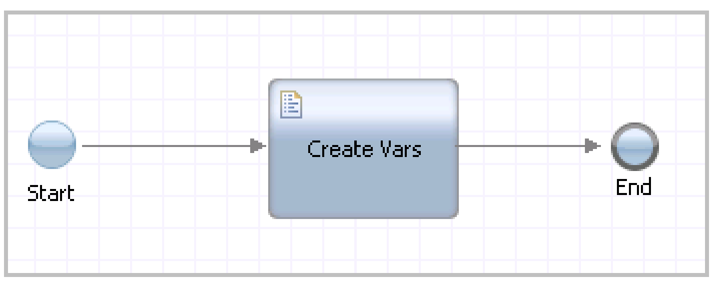
```
if (tw.local.requisitionDetails == null) {
tw.local.requisitionDetails = new tw.object.HiringRequisition();
}
```



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\`.

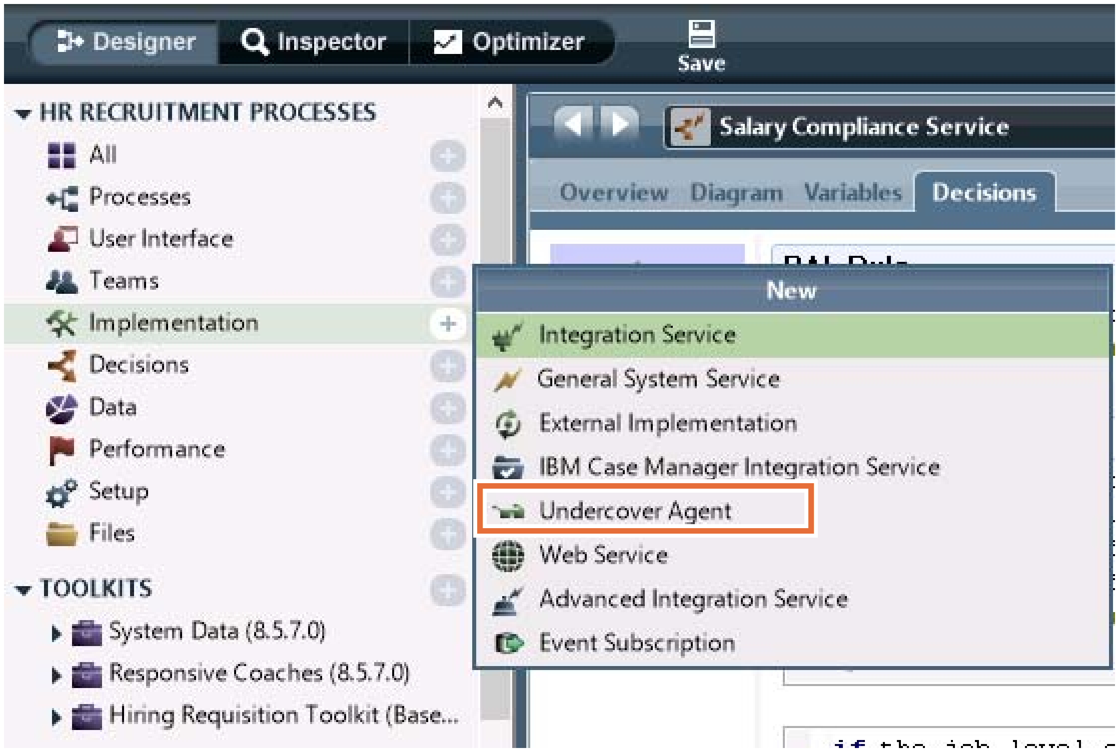
__ d. Use the sequence flow tool to connect the flow.



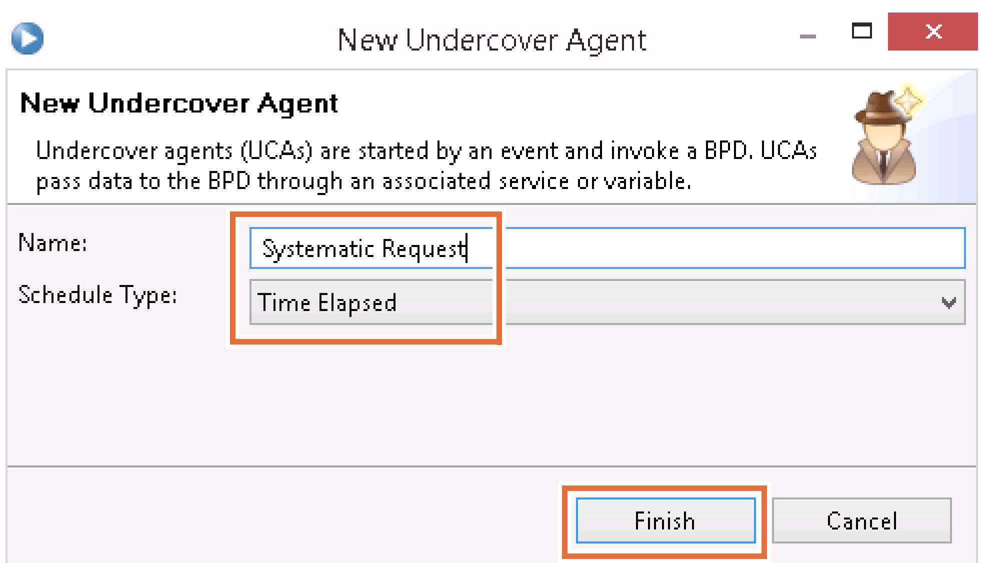
__ e. Save the service.

__ 4. Create and enable an Undercover Agent.

__ a. Click the (**+**) plus sign next to **Implementation**, 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.

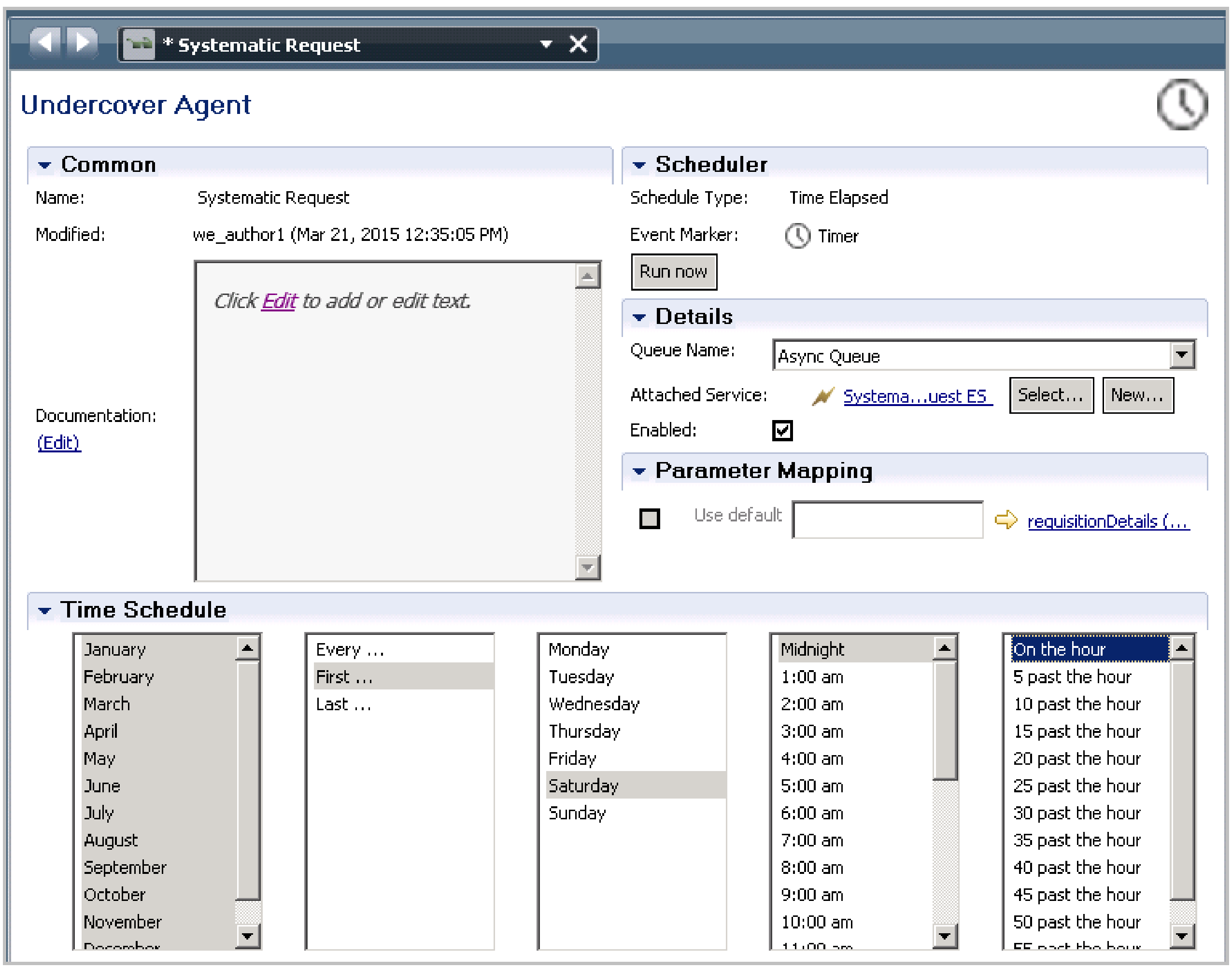
__ d. Select the **Systematic Request ES** general system service. Verify that the **Enabled** check box is selected for the new undercover agent.

__ e. 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.

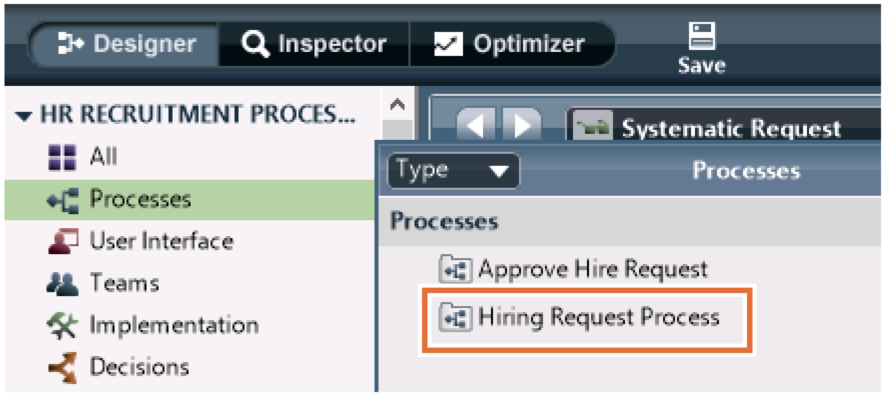
This setting generally provides time for the purging service to complete over the weekend.



__ f. Save your work.

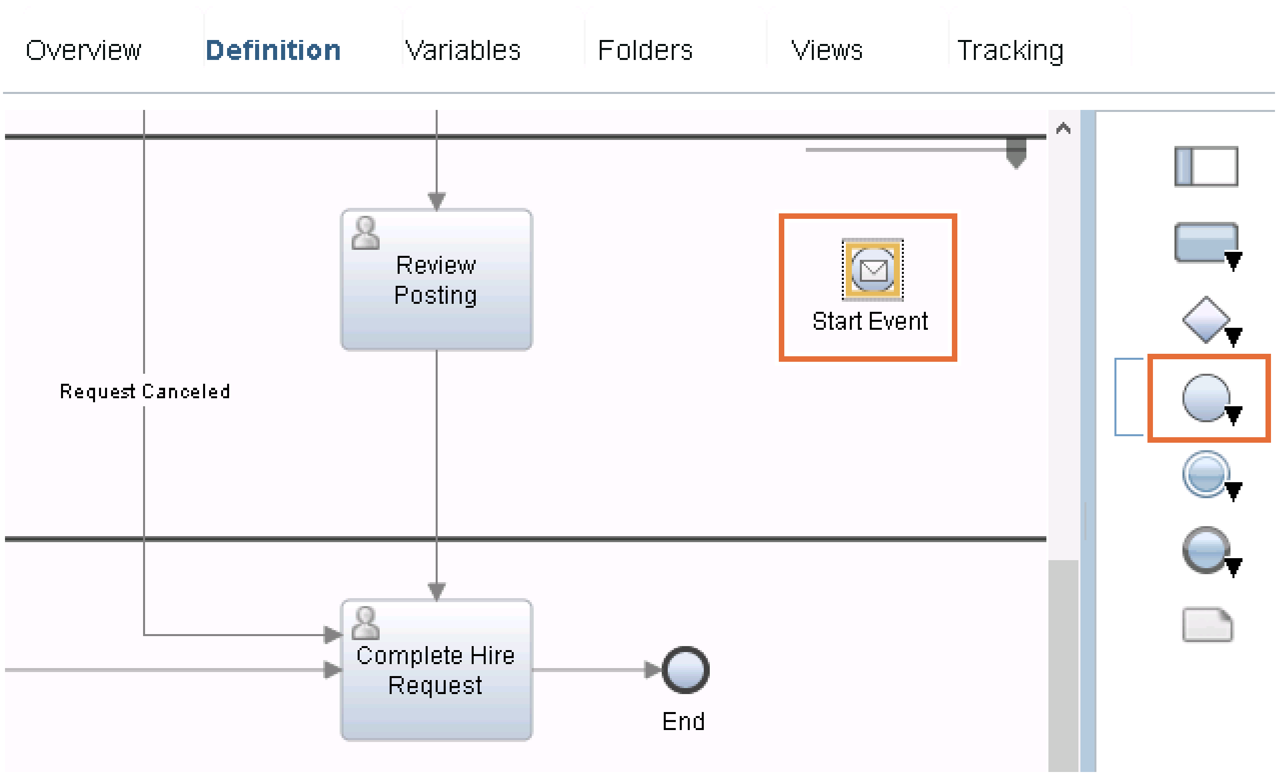
__ 5. Implement the **Systematic Request** start message event on the **Hiring Request Process** process.

__ a. Open the **Hiring Request Process** process by clicking **Processes** in the Process Designer client application library and double-clicking **Hiring Request Process**.

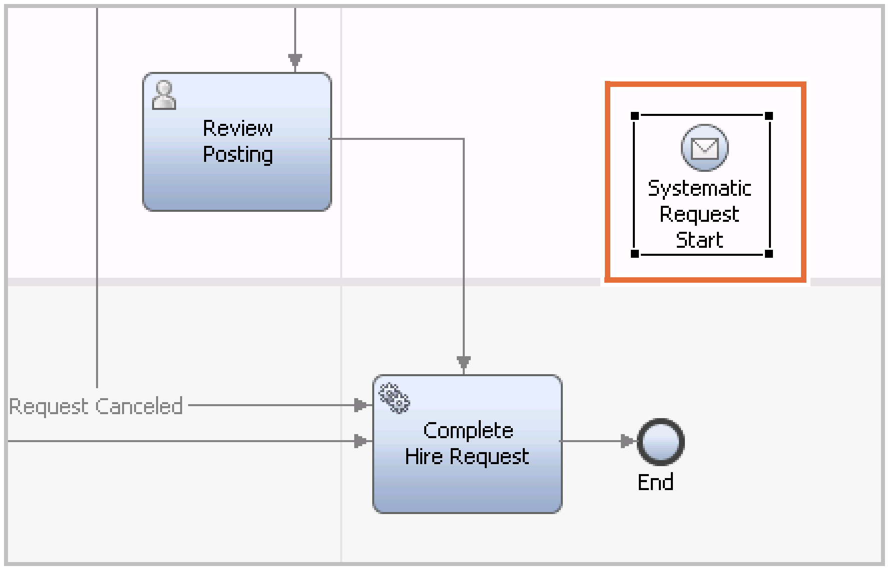


__ b. The process opens in web Process Designer.

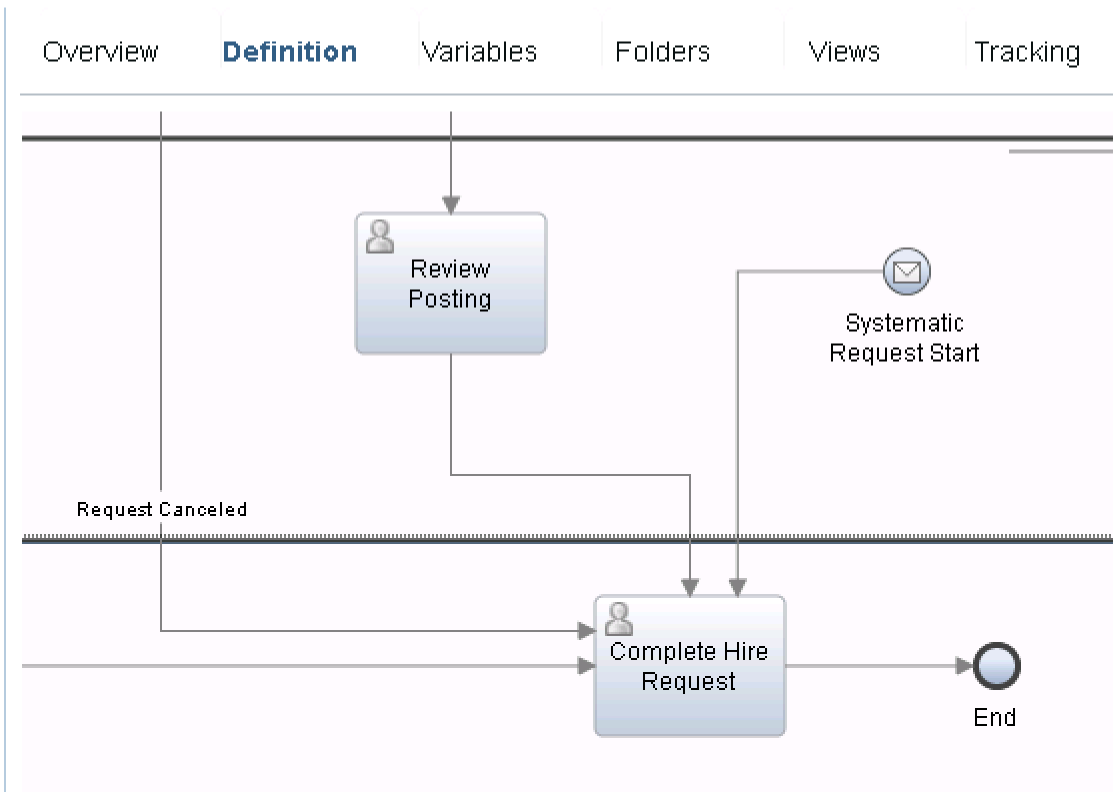
__ c. Drag a **Start** event from the palette to the process near the **End** event. The start event defaults to a Start message event.



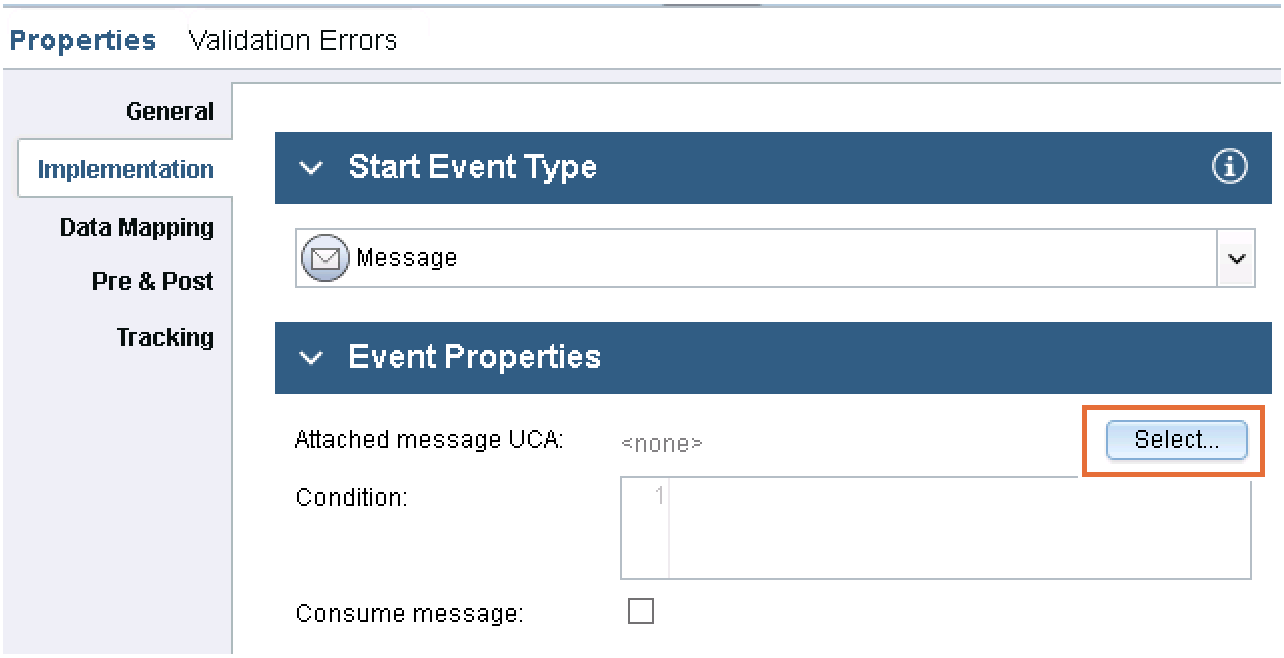
__ d. Rename the start message event to: `Systematic Request Start`



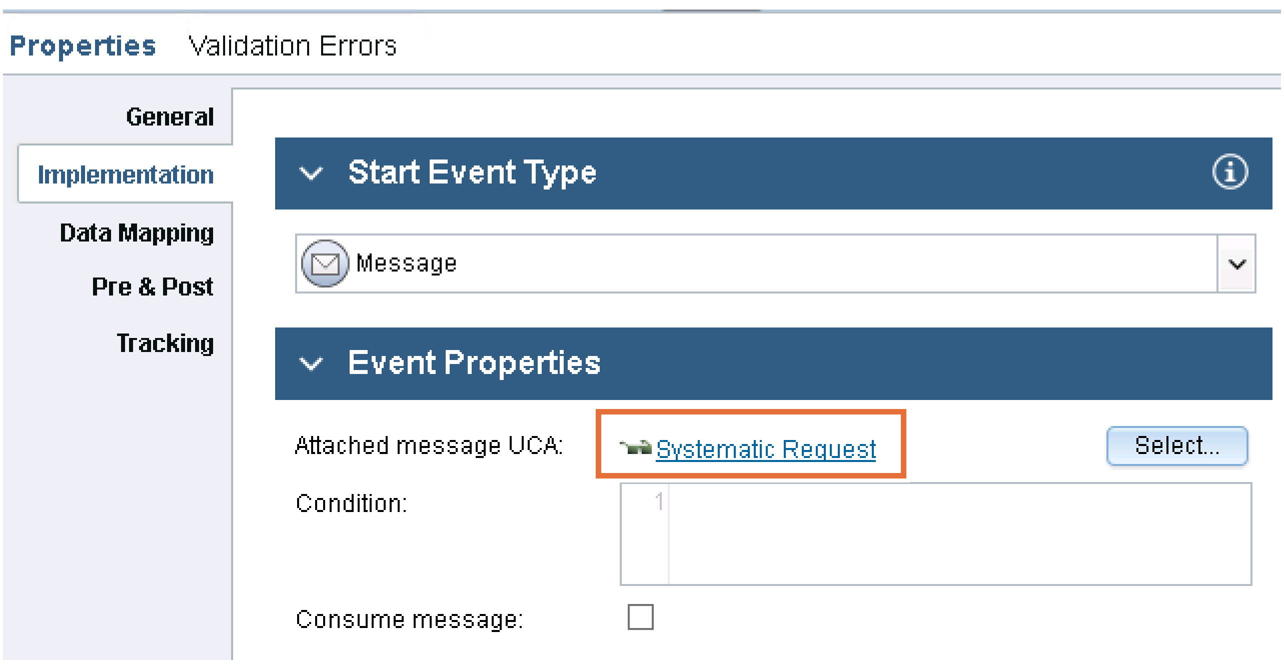
__ e. Connect `Systematic Request Start` to the **Complete Hire Request** activity.



__ f. Click the **Systematic Request Start** event, click the **Properties > Implementation** menu, and then click **Select** next to **Attached message UCA**.



__ g. Select the **Systematic Request** UCA that you created for the activity.

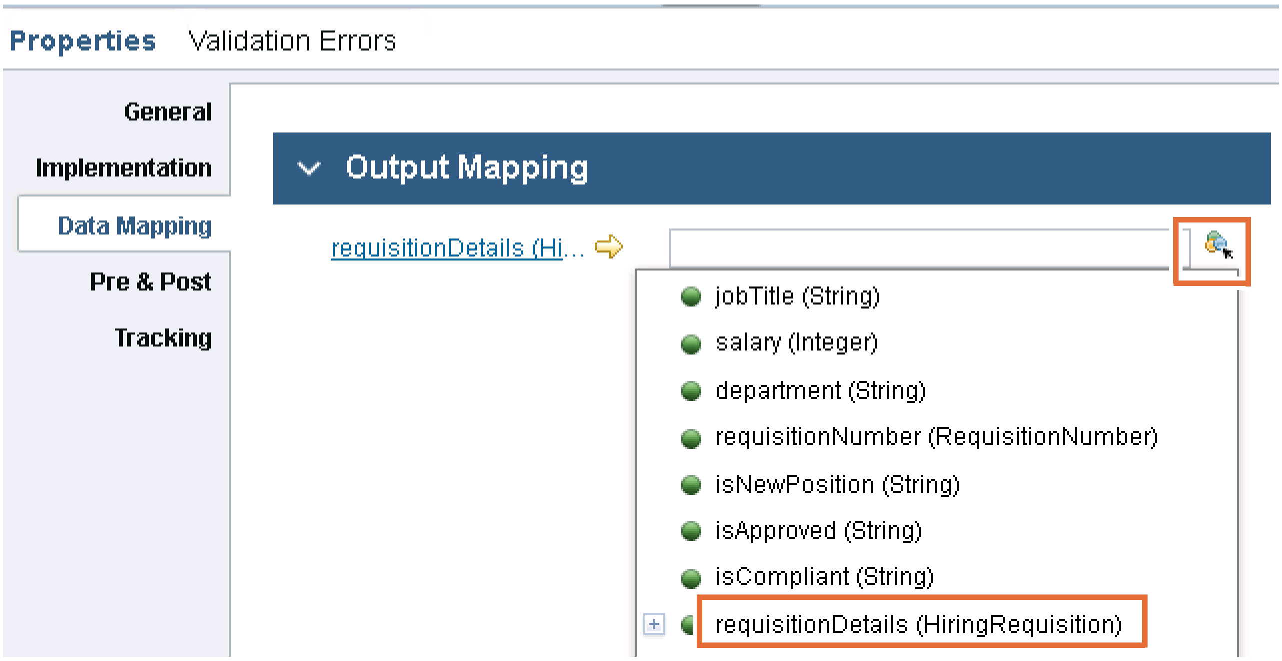


__ h. Save your work.

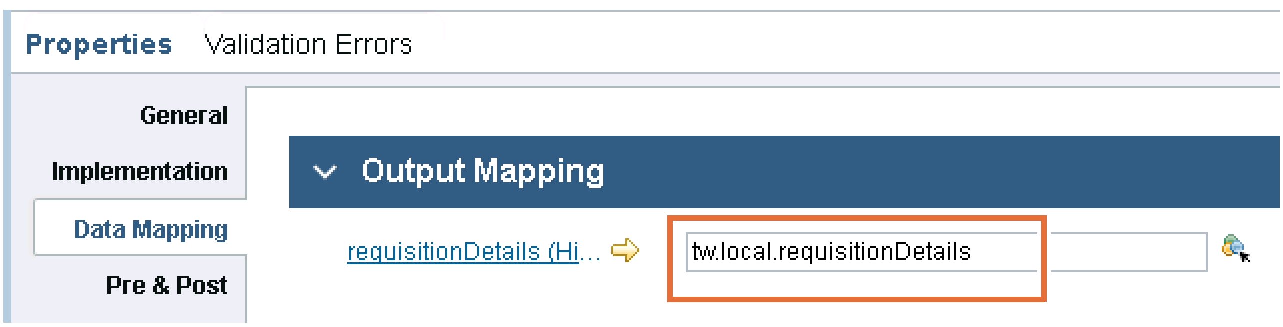
__ 6. Map the `Systematic Request` start event UCA outputs to the process variables.

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

__ b. In the **Output Mapping** section, map it to the `requisitionDetails (HiringRequisition)` variable.



__ c. Select the **`requisitionDetails`** variable. The output maps to the variable.

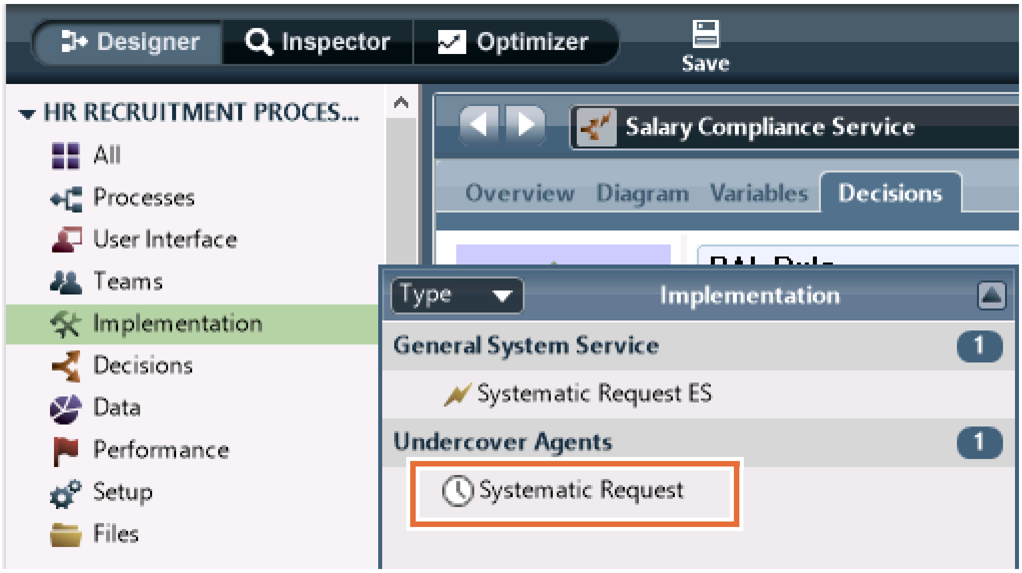


__ d. Save your work.

__ e. Minimize the browser window.

__ 7. Test the UCA.

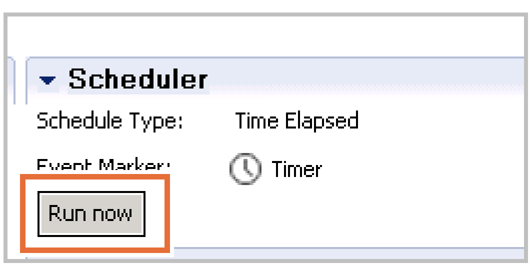
__ a. Return to the IBM Process Designer client application. Double-click **Implementation > Undercover Agents > Systematic Request** to open the **Systematic Request** UCA.



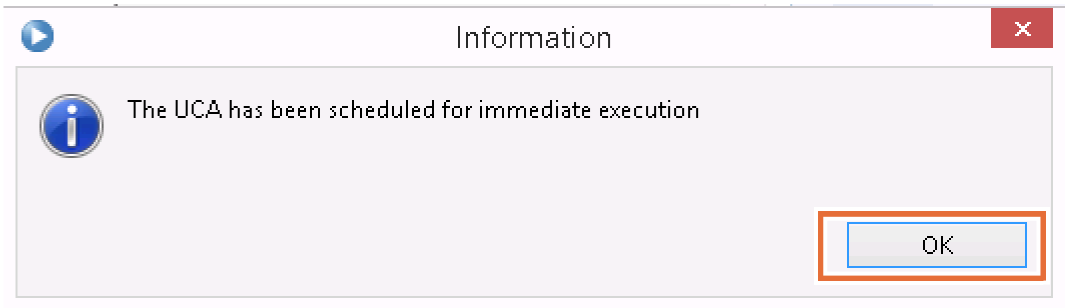
Important

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 Apps > HR Recruitment Processes (HRR) > Snapshots** menu. Select the current track, and then click **Make Default Track**.

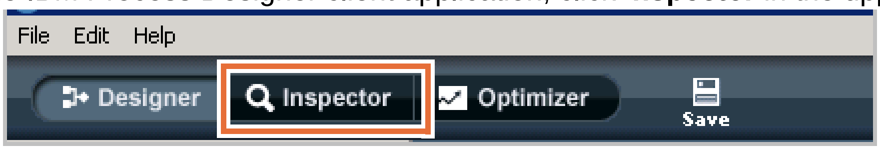
__ b. Click **Run now** in the Scheduler section.



__ c. A dialog box is displayed, which indicates that the message is scheduled for immediate execution. Click **OK**.

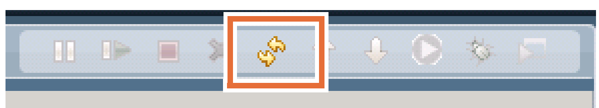


__ d. In the IBM Process Designer client application, click **Inspector** in the upper-left corner.

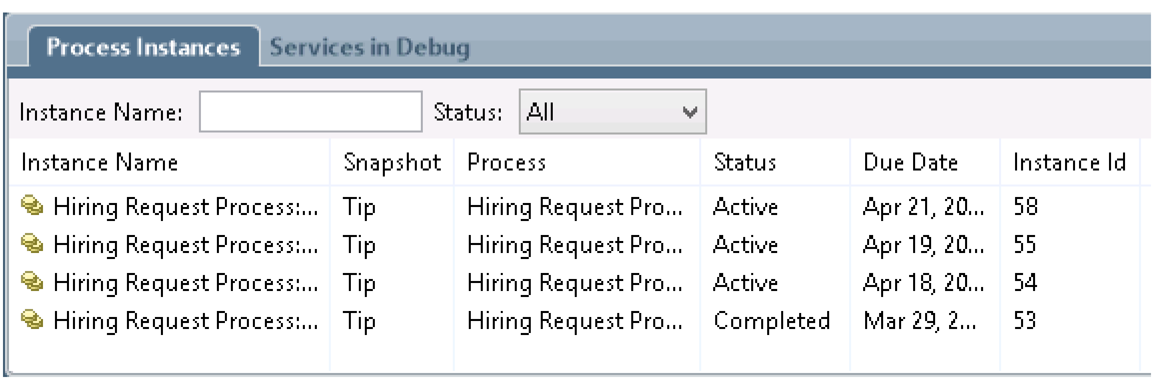


__ e. Click the **Process Instances** tab under the Inspector tab unless it is already selected.

__ f. Click **Refresh** on the right to see the instances that were created in the Process Center.



__ g. If everything was successful, you created a **Hiring Request Process** instance (the highest numbered instance) present in the left frame. This window shows all the instances that ran in the Process Designer and their status.



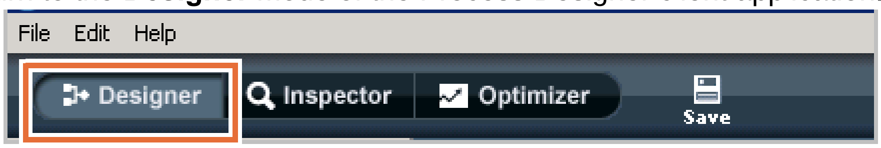
Instance Name	Snapshot	Process	Status	Due Date	Instance Id
Hiring Request Process:...	Tip	Hiring Request Pro...	Active	Apr 21, 20...	58
Hiring Request Process:...	Tip	Hiring Request Pro...	Active	Apr 19, 20...	55
Hiring Request Process:...	Tip	Hiring Request Pro...	Active	Apr 18, 20...	54
Hiring Request Process:...	Tip	Hiring Request Pro...	Completed	Mar 29, 2...	53

Part 3: *Apply asset tagging*

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.

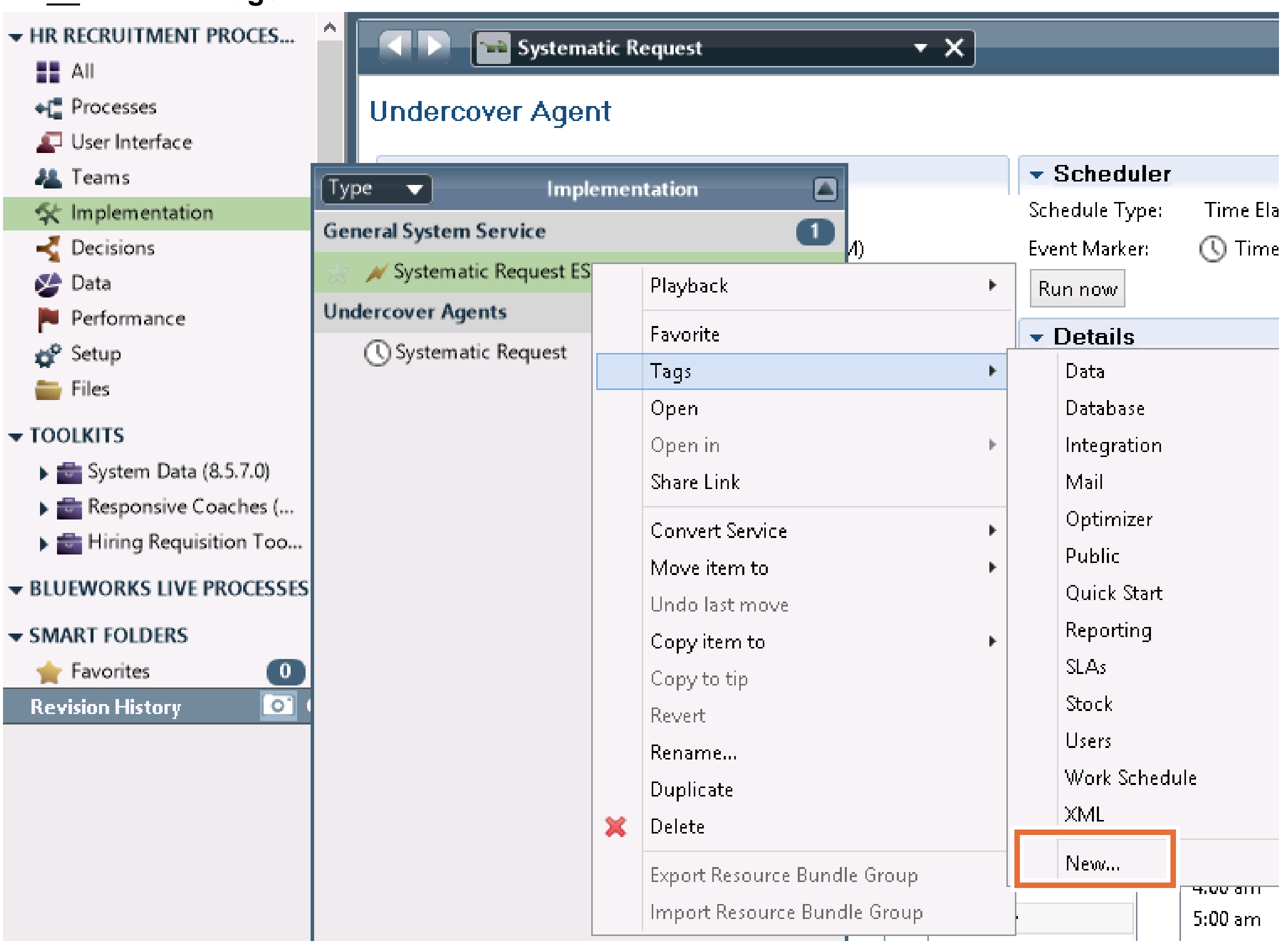
__ a. Return to the **Designer** mode of the Process Designer client application.



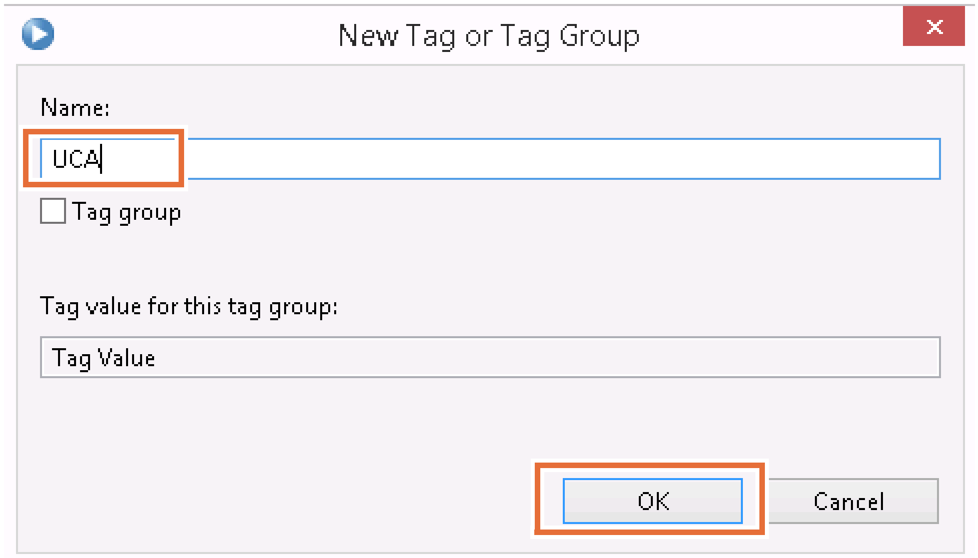
__ b. In the Designer library, click the **Implementation** category.

__ c. Right-click the **Systematic Request ES** general system service.

__ d. Click **Tags > New**.



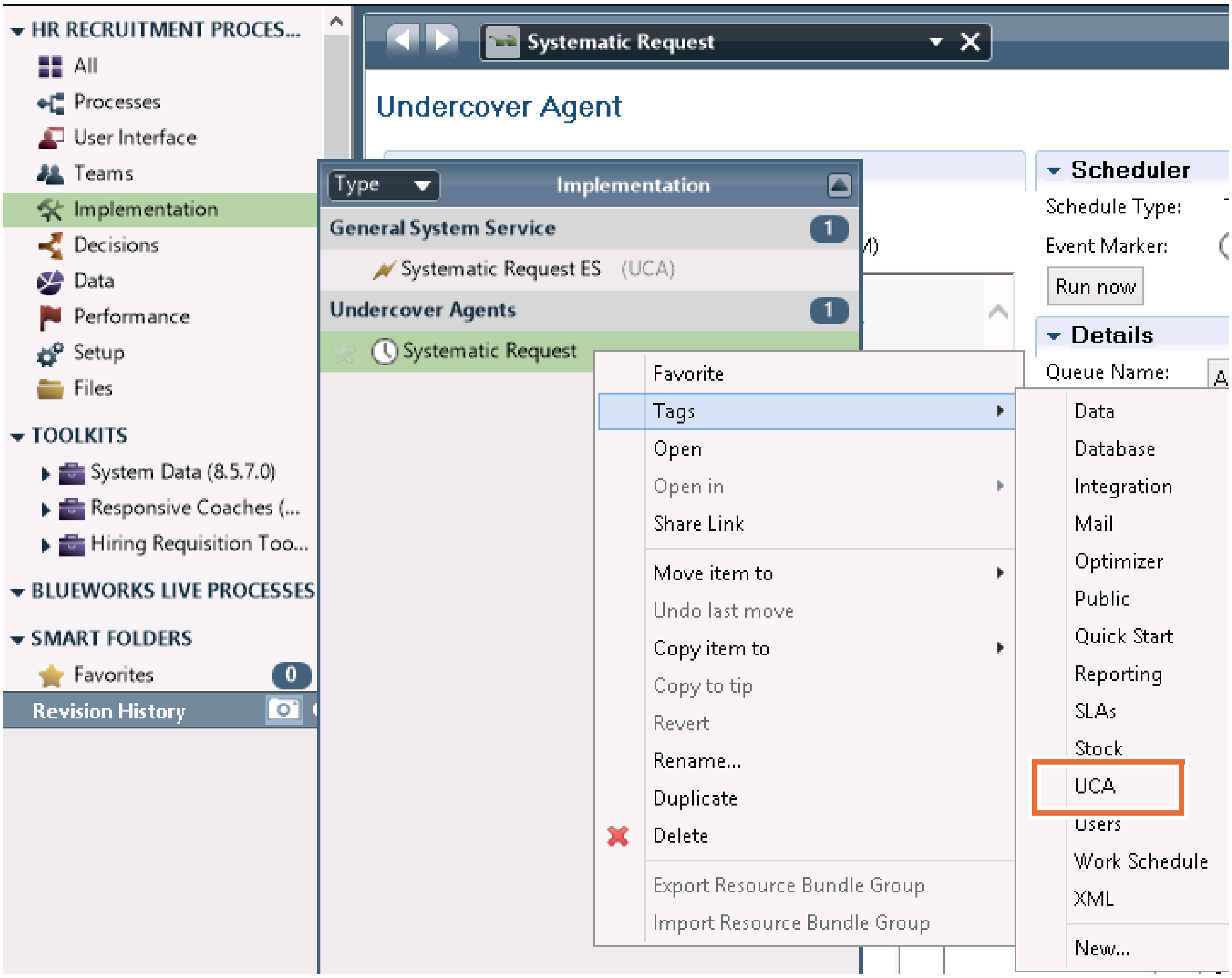
__ e. Name the tag `UCA` and then click **OK**.



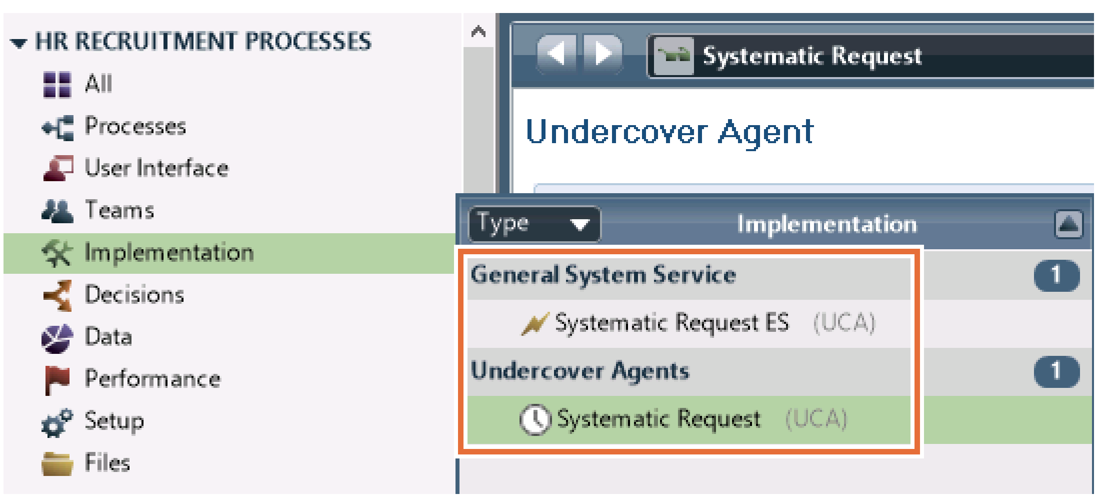
__ f. Go back to the Process Designer client application library and click the **Implementation** menu option.

__ g. Right-click the **Systematic Request** UCA.

__ h. Click **Tags > UCA**.

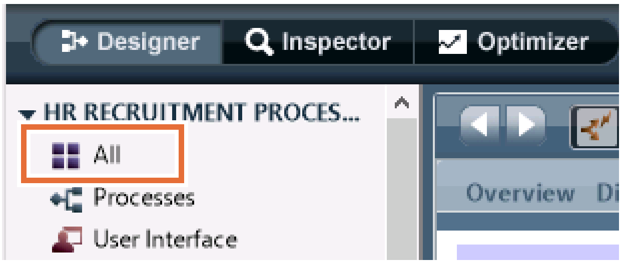


__ i. The two artifacts are now tagged UCA.



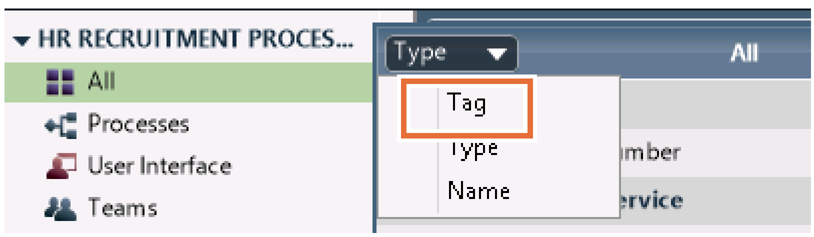
__ 2. Group by tag.

__ a. In the Process Designer client application library, click the **All** menu option. This category shows all the assets in the library, and the default grouping is by Type.

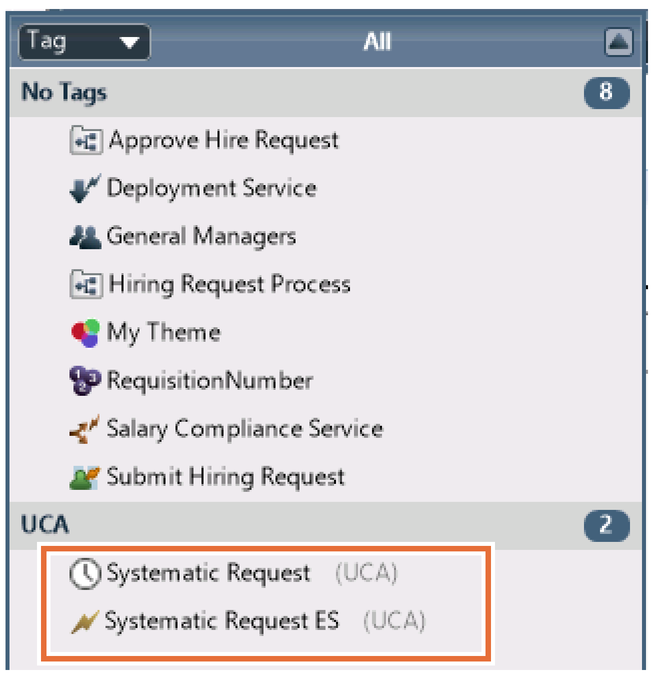


__ b. Click the arrow to the right of **Type**.

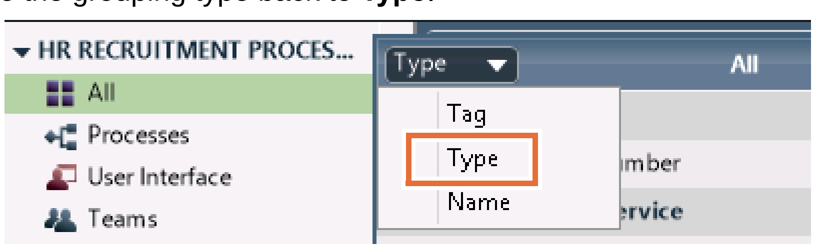
__ c. Select **Tag** from the options.



__ d. You see all the artifacts with the tag UCA grouped.



__ e. Change the grouping type back to **Type**.

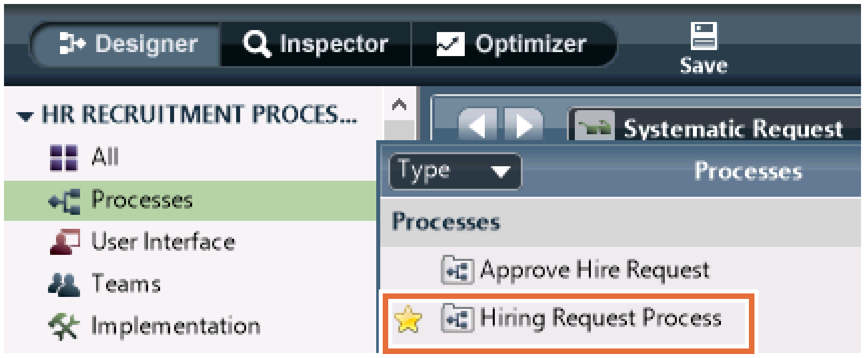


__ 3. Create a favorite.

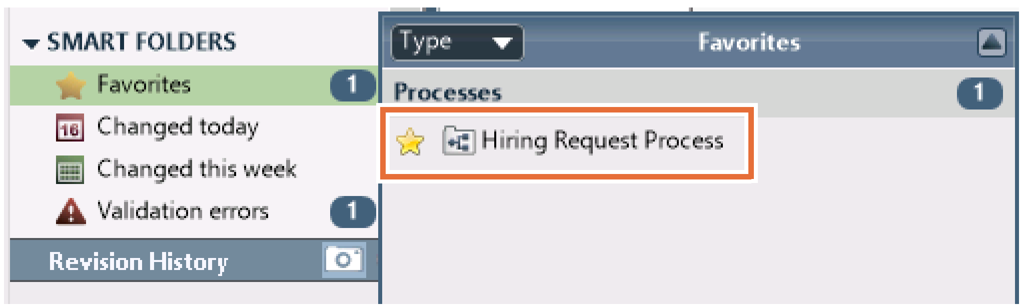
__ a. Click the **Processes** library category.

__ b. Hover to the left of the **Hiring Request Process**. You notice the outline of a star.

__ 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 in 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 client application 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 library items that are changed by all users who have access to the current process application.

Part 4: Create a service to query a database and populate a list

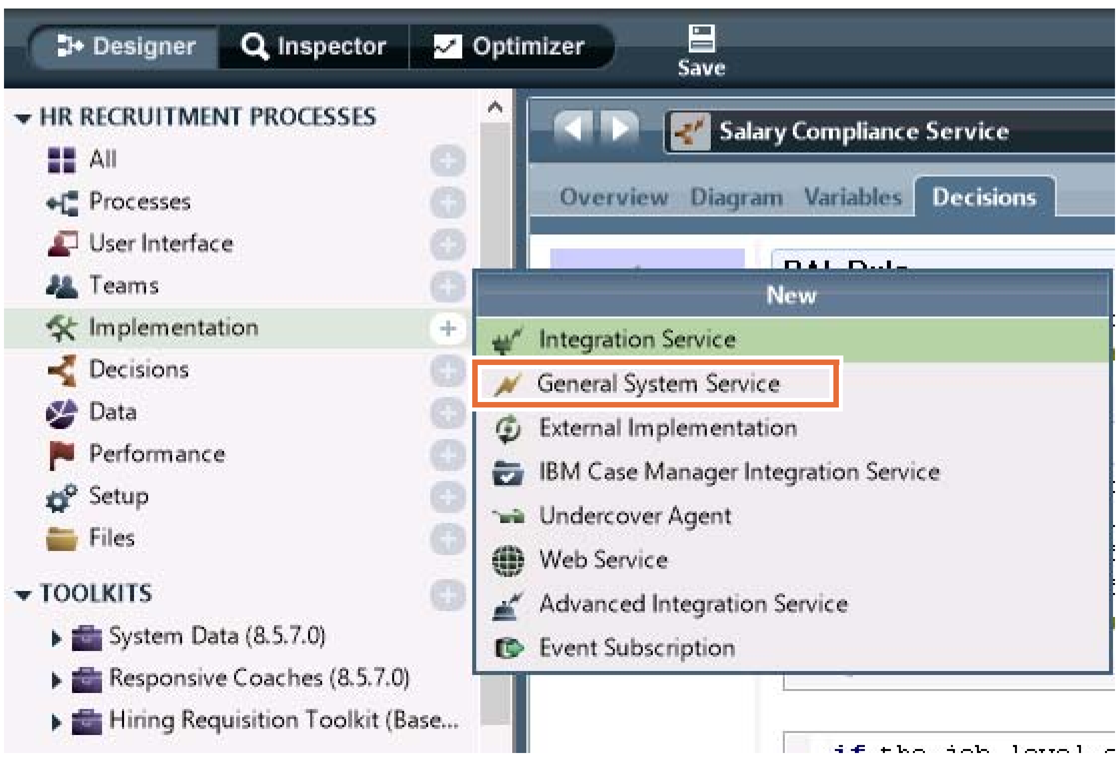
In this part of the exercise, you create a service to query a database and populate a list. The current systems require employees to look up the different codes and enter them into the system. Because customers use this system, a better approach is to offer a menu with readable options that correspond 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.

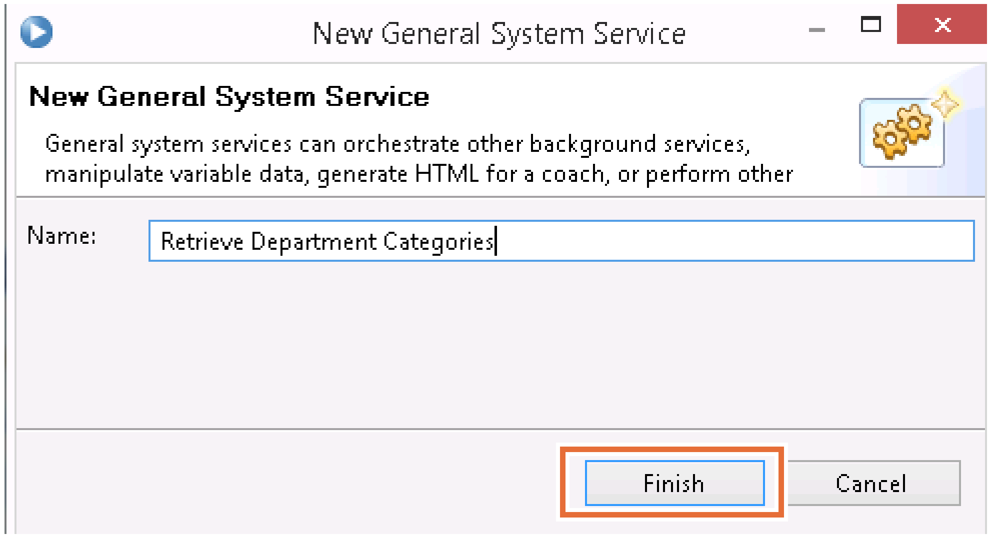
Use an environmental variable to hold database information for one query, and an exposed process variable (EPV) to hold the database information for the other.

__ 1. Create a **General System Service** to retrieve the incident categories.

__ a. In the Process Designer client application library, click the (**+**) plus sign next to the **Implementation** category and click the **General System Service** option.

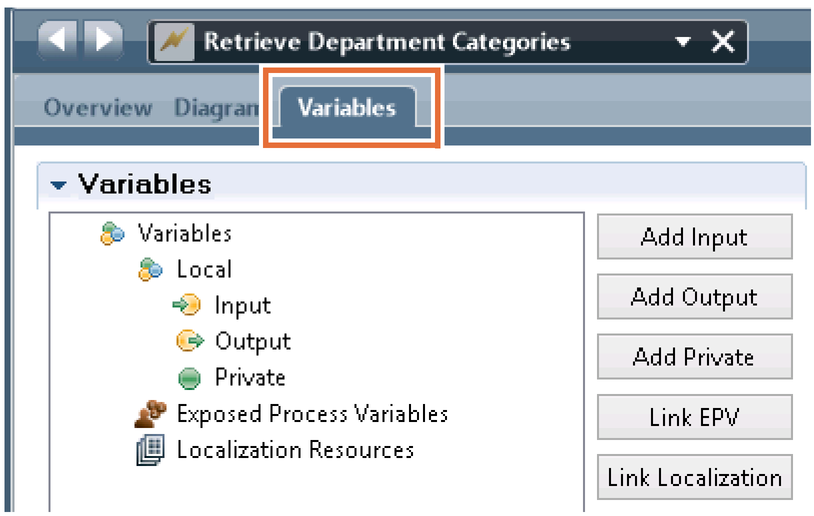


__ b. Name the service `Retrieve Department Categories` and click **Finish**.



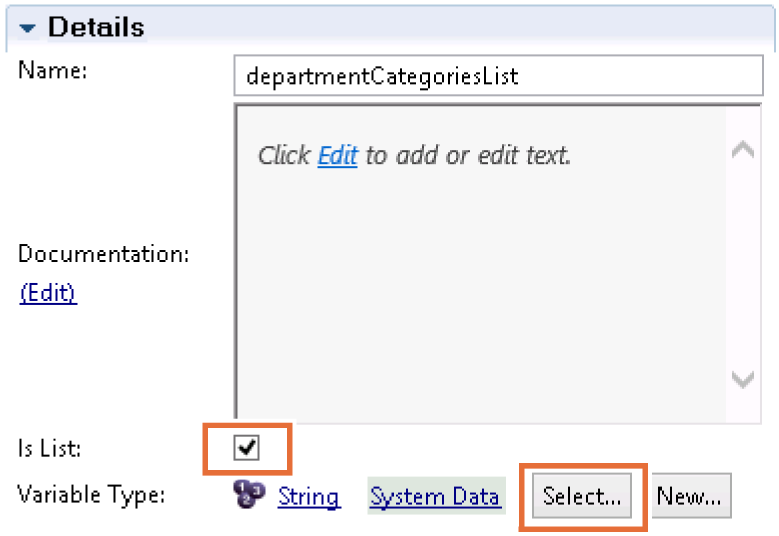
__ 2. Add variables to the service. These variables hold the list data that is retrieved from the database.

__ a. Click the **Variables** tab.

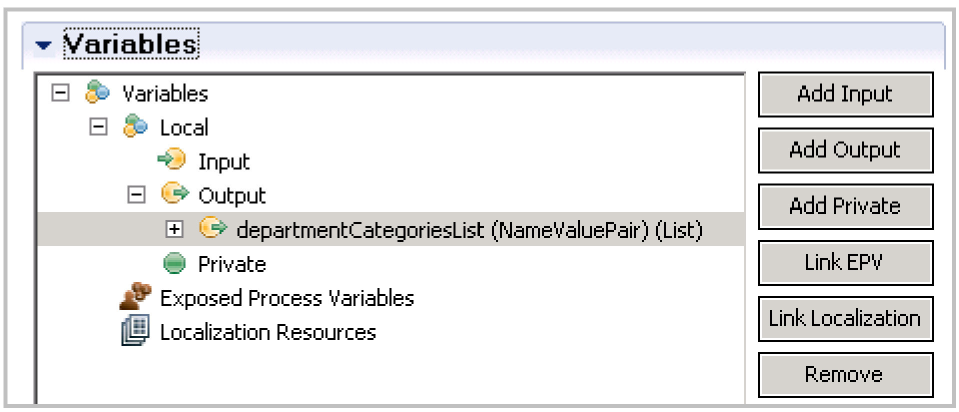


__ b. Click **Add Output** and name the variable: `departmentCategoriesList`

__ c. Select the **Is List** check box and then click **Select** for the variable type.



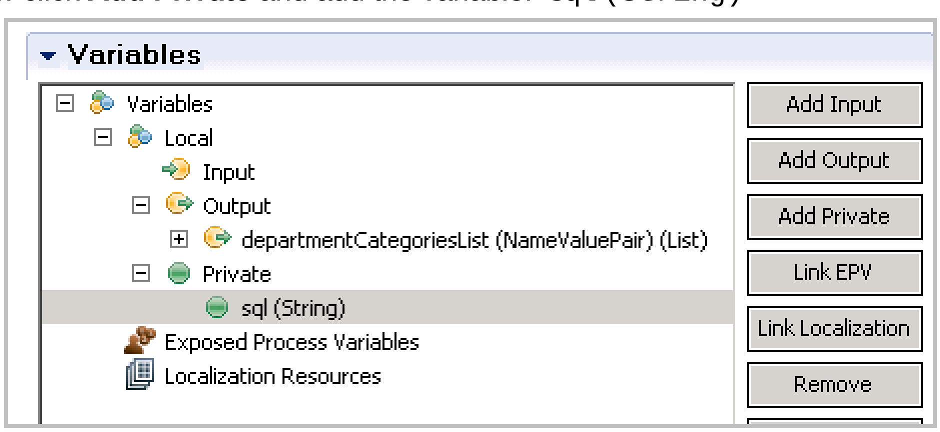
__ d. Select **`NameValuePair`** from the list.



Hint

This variable is used to populate the list of options in a select menu on the coach.

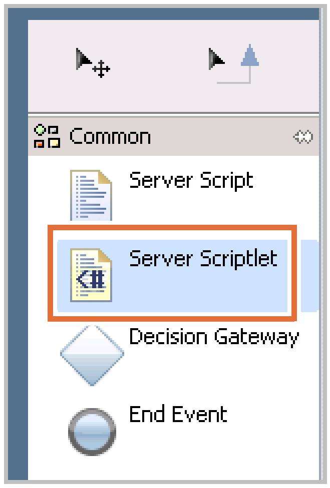
__ e. Now click **Add Private** and add the variable: `sql (String)`



__ 3. Add the elements onto the canvas.

__ a. Click the **Diagram** tab of the service.

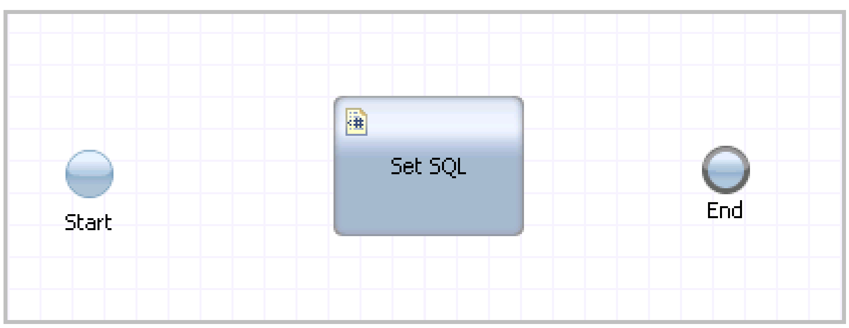
__ b. Drag a **Server Scriptlet** from the service palette to the canvas.



Hint

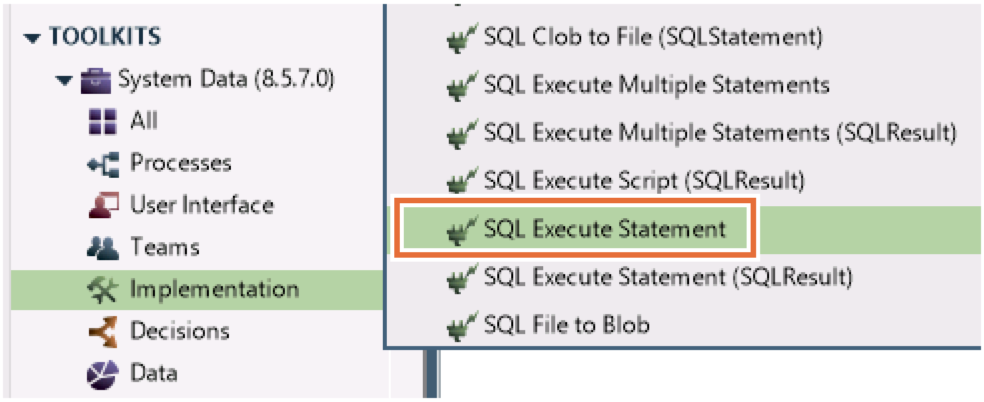
Use a server scriptlet when assigning a value to a single variable.

__ c. Rename the server scriptlet: `Set SQL`



__ d. Expand the **System Data** toolkit in the Process Designer client application library.

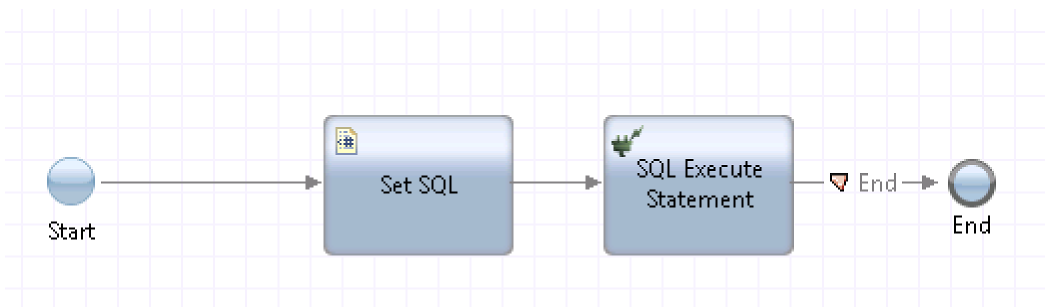
__ e. Select the **Implementation** category. Click the **SQL Execute Statement** and drag it onto the canvas to the right of the `Set SQL` server scriptlet.



Hint

Type `SQL Execute` to filter the list and find the service.

__ f. Connect the objects on the canvas from left to right: from the **Start** event to the `Set SQL` server scriptlet, to the `SQL Execute Statement`, and finally to the **End** event.

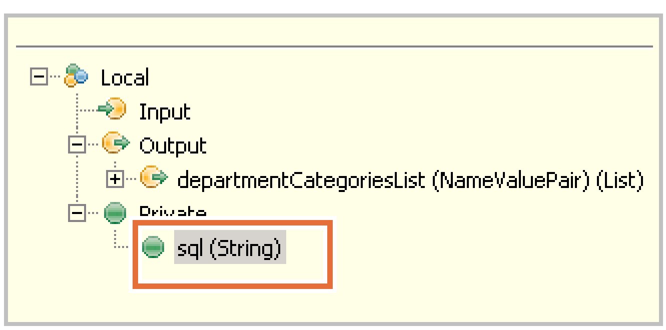


__ 4. Implement the `Set SQL` server scriptlet.

__ a. Click the `Set SQL` server scriptlet.

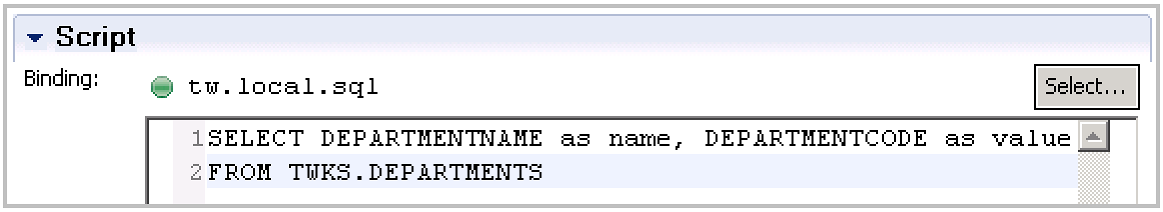
__ b. Click the **Properties > Implementation** menu option.

__ c. Click **Select** next to **Binding:** and click the **sql (String)** variable.



__ d. 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 DEPARTMENTNAME as name, DEPARTMENTCODE as value
FROM TWKS.DEPARTMENTS
```



Note

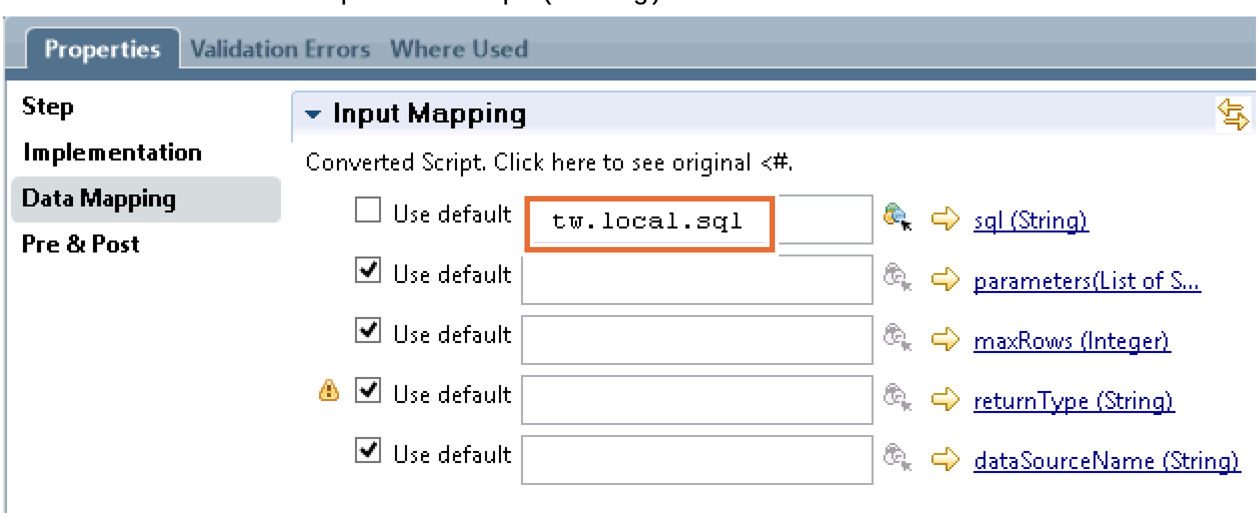
To avoid any typographical errors, the script code is provided in the file `Script2.txt` at the location: `C:\labfiles\Exercise_Support_Files\Exercise09\`.

__ 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, clear the **Use default** check box for the `sql (String)` variable and map it to the `sql (String)` variable.

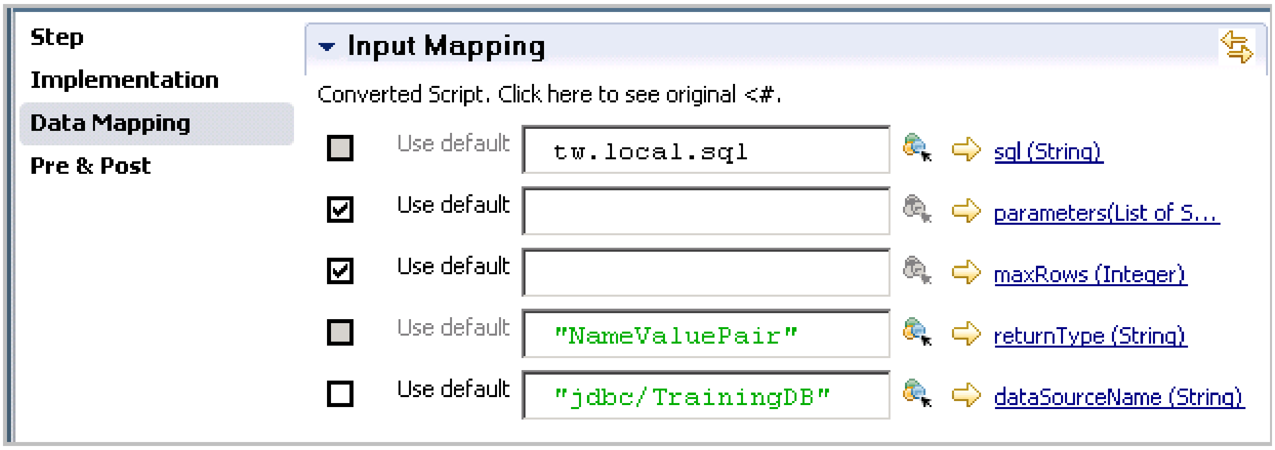


__ d. Clear the **Use default** check box for `returnType (String)`, and add the following text: `"NameValuePair"` (include the quotation marks).

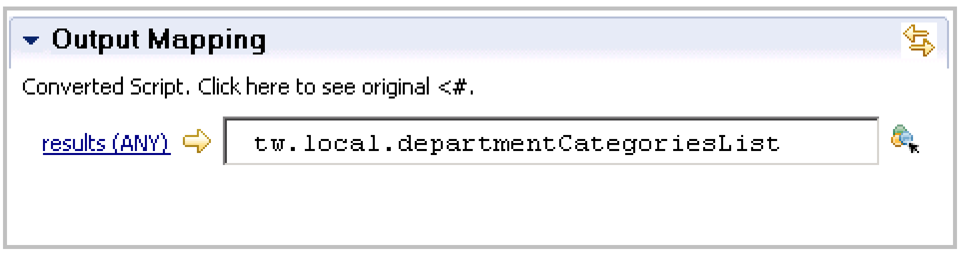
__ e. Ensure that the **Use default** check box is selected for the `parameters` and `maxRows` variables.

__ f. Clear the **Use default** check box for `dataSourceName (String)`.

__ g. A JDBC data source was already created for you. In the **dataSourceName** field, type `"jdbc/TrainingDB"` (include the quotation marks).

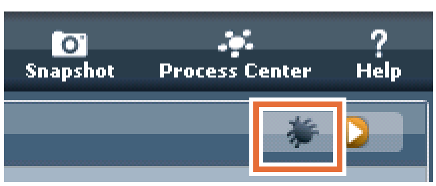


__ h. In the **Output Mapping**, map the `results (ANY)` variable to the `departmentCategoriesList` variable.

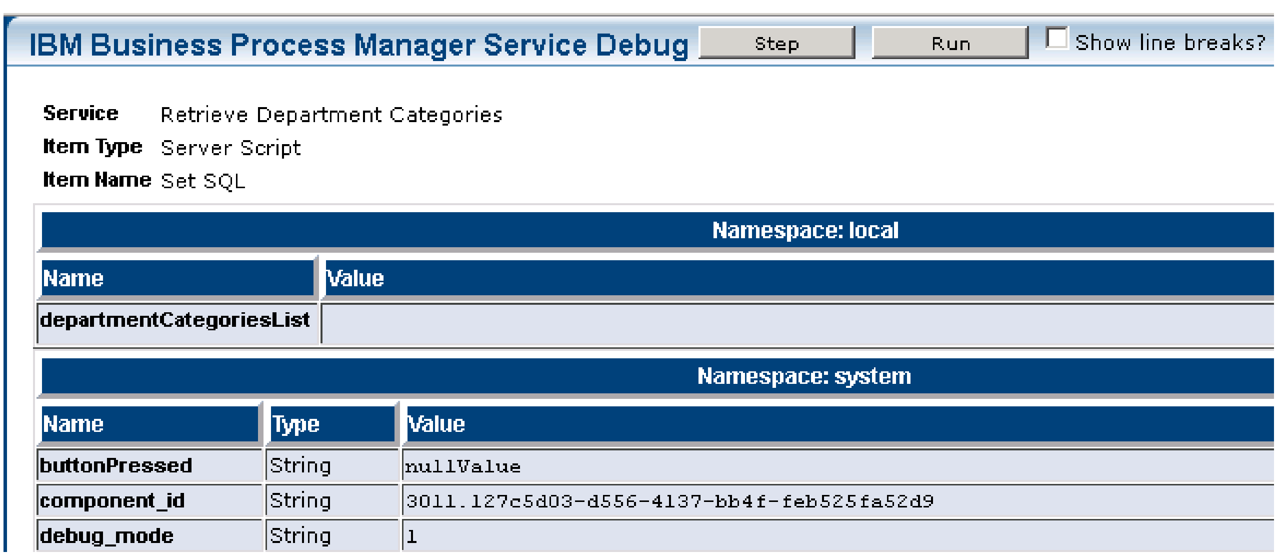


__ i. Save your work.

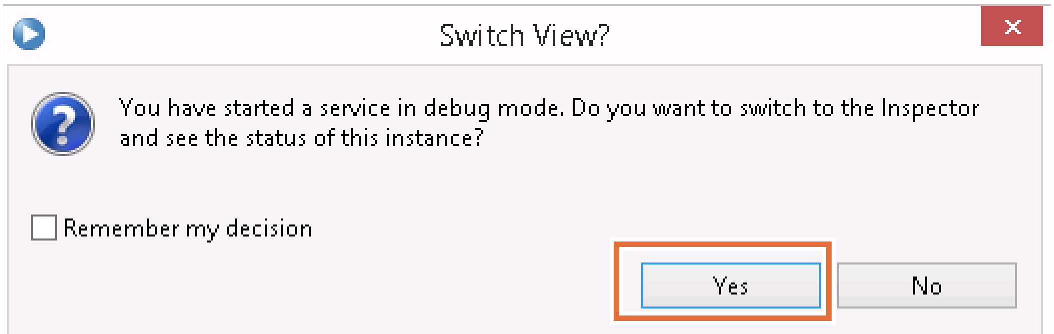
__ j. Click the **debug** icon to debug the service.



The default browser is shown with server-side information on the service and the variable values.



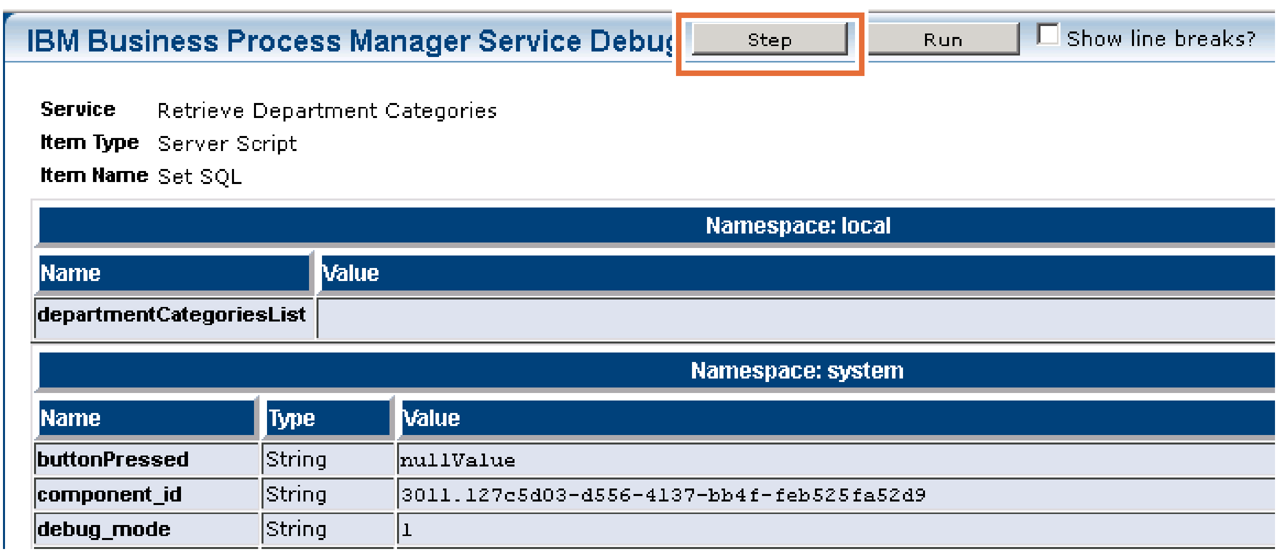
__ k. Return to the Process Designer client application. Click **Yes** to switch the view to the inspector.



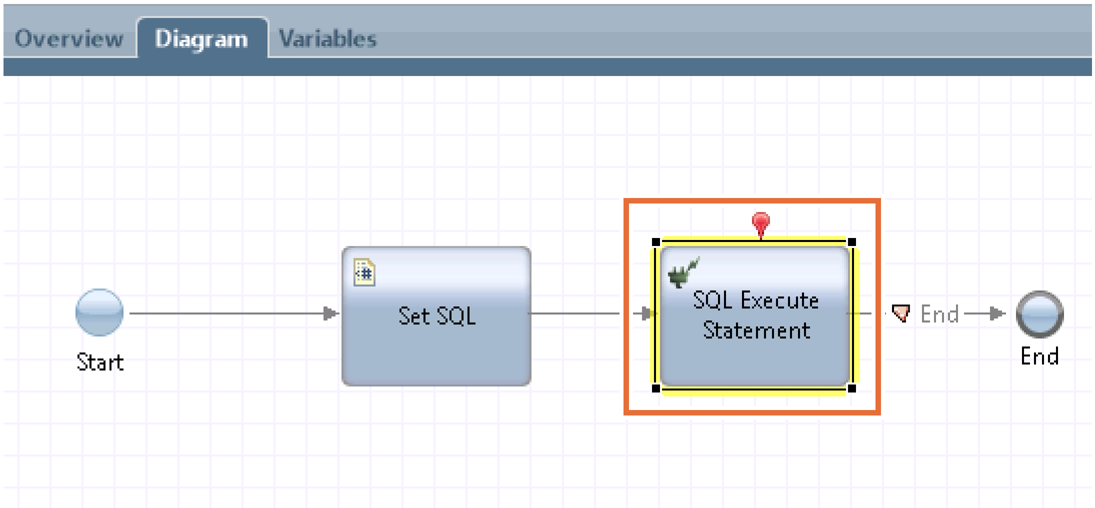
Important

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.

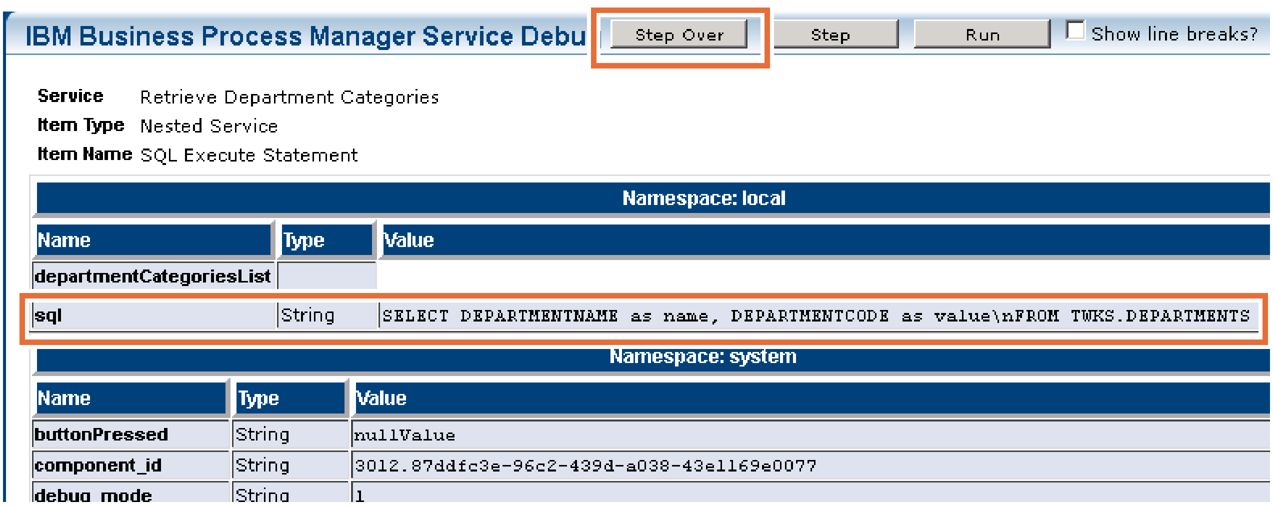
__ l. Return to the debugger inside the browser. Click **Step**.



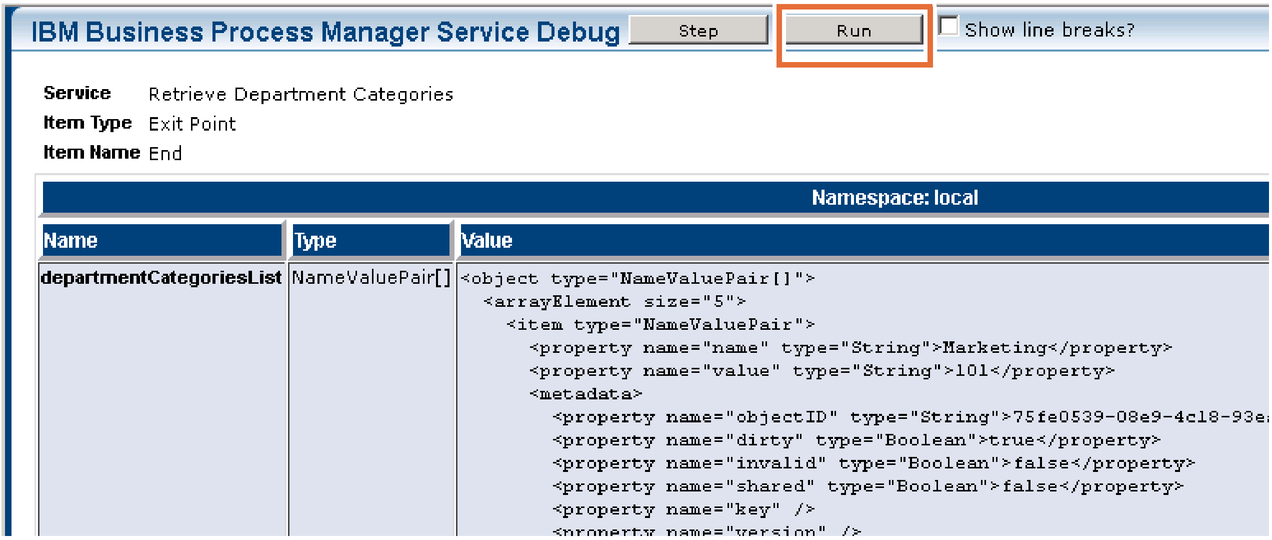
__ m. Return to the Process Designer client application and view the token that moved to the next step.



__ n. Return to the debugger inside the browser and verify that the `sql` variable is updated with the values that are assigned to the variable. Click **Step Over** to run the SQL Execute Statement nested service.

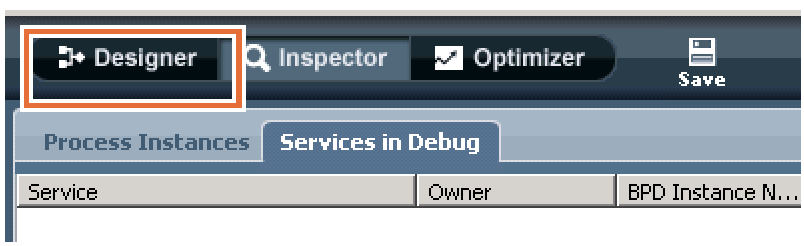


__ o. The `departmentCategoriesList` variable is populated with the values in the database. Notice that `arrayElement size="5"` means that the database returned five rows, which were mapped to the `NameValuePair` object per the configuration options of the **SQL Execute** service. Click **Run** to complete the service.



__ p. The service completes. Close the tab in the browser.

__ q. Return to the Process Designer client application and click **Designer** to return to the Designer view.



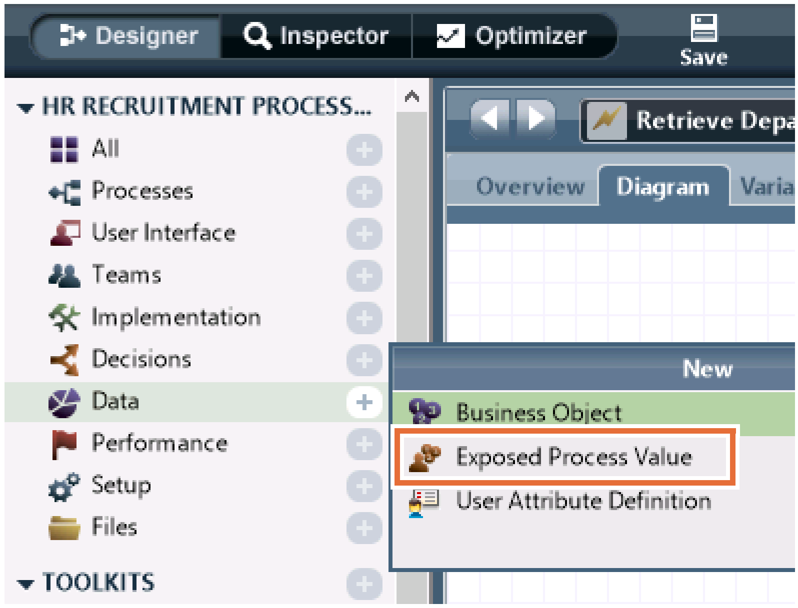
__ 6. Add an exposed process variable (EPV) to manage the data source name.

Note

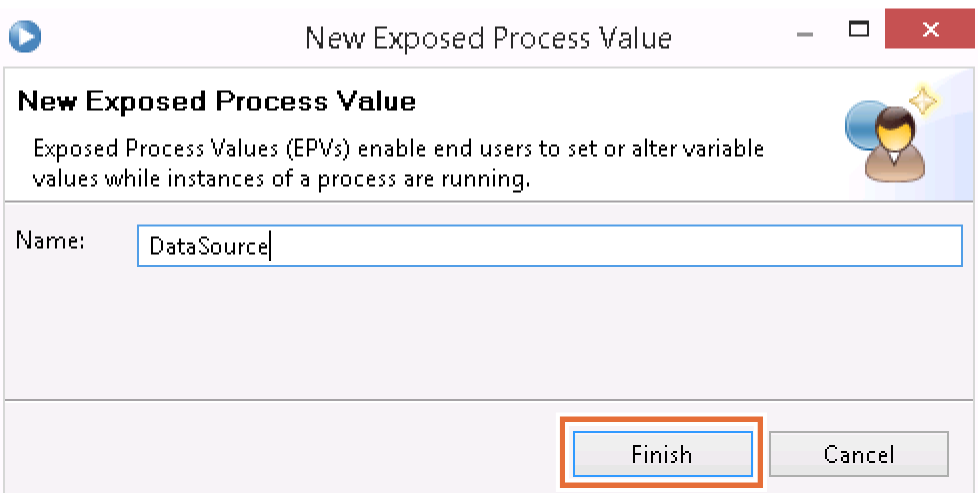
The data source name should normally be stored as an environment variable. EPVs are designed to store business data because they can be exposed to business users so they can set real-time process variables. In the next two steps, you practice the creation and use of both environment variables and EPVs.

__ a. Click the (**+**) plus sign in the **Data** menu option in the Process Designer client application library.

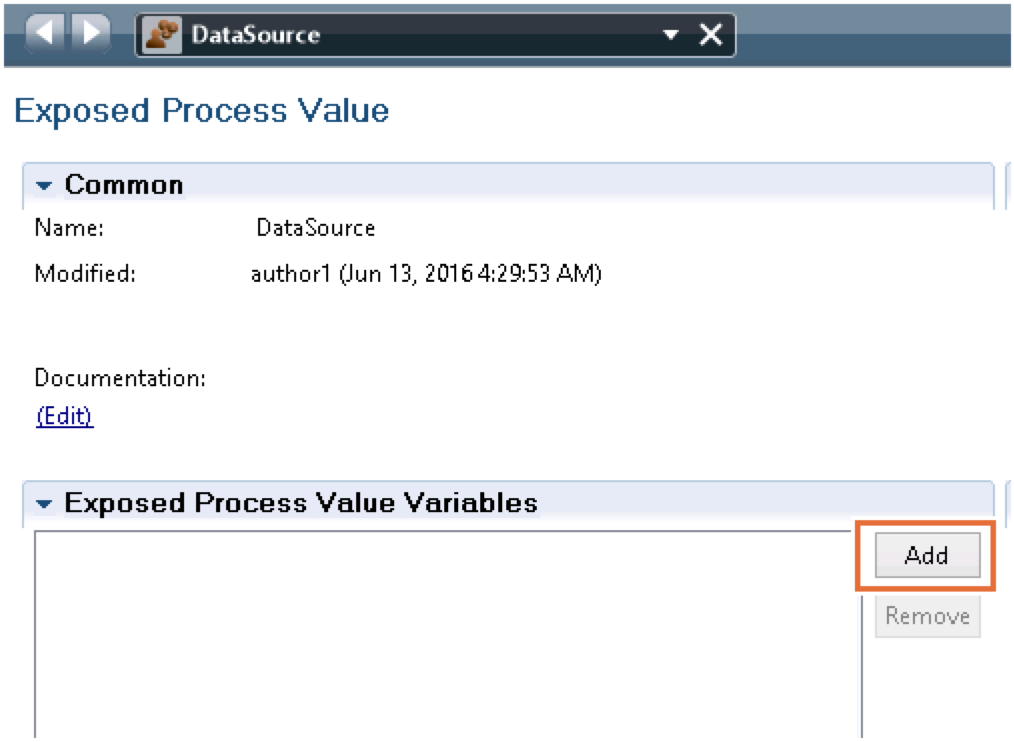
__ b. Select **Exposed Process Value**.



__ c. Name your EPV `DataSource` and click **Finish**.

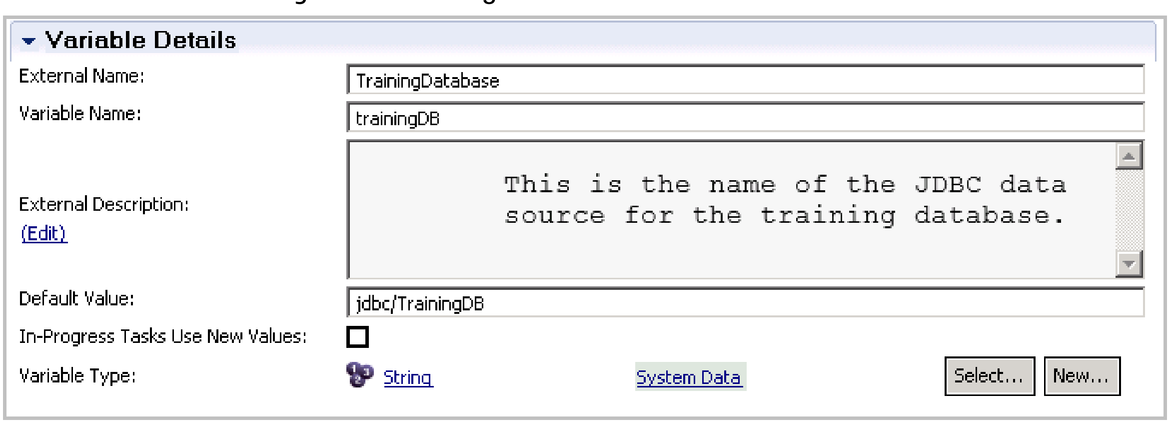


__ d. In the **Exposed Process Value** settings, click **Add** in the **Exposed Process Value Variables** section.



__ 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`

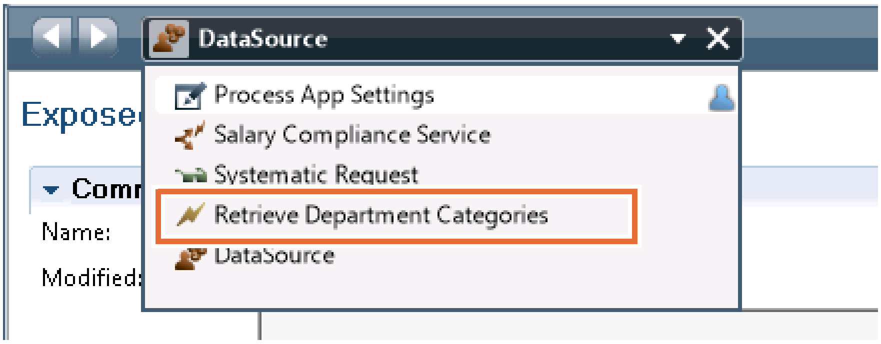


Note

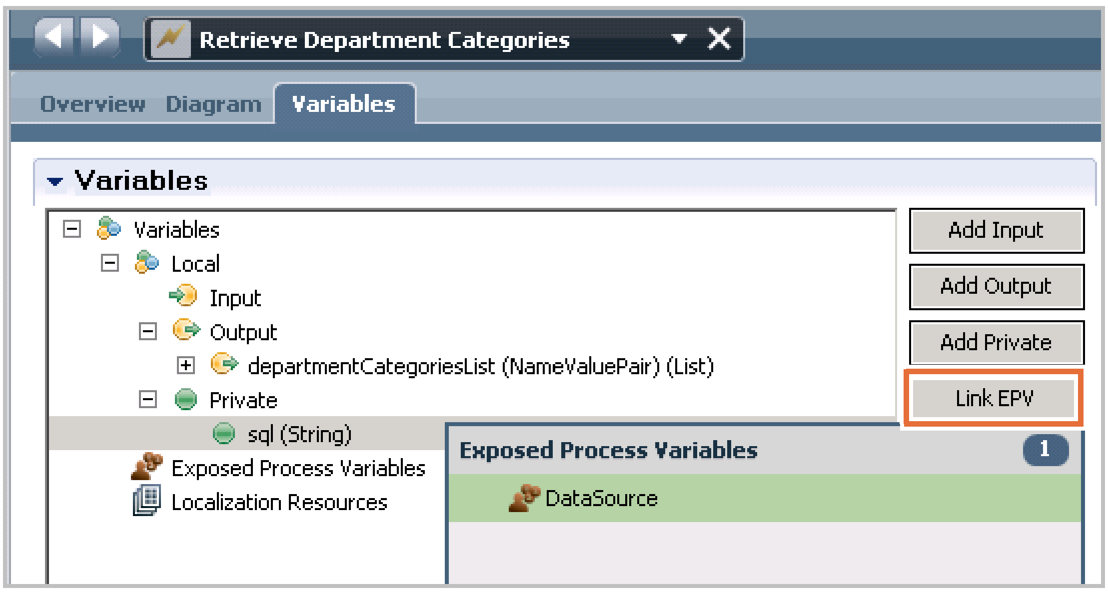
Because the variable type is String, you do not need to enclose the value in quotation marks.

__ f. Save your work.

__ 7. Click **Retrieve Department Categories** from the History menu to return to the service.



__ a. Click the **Variables** tab, click **Link EPV**, and then select the `DataSource` EPV.



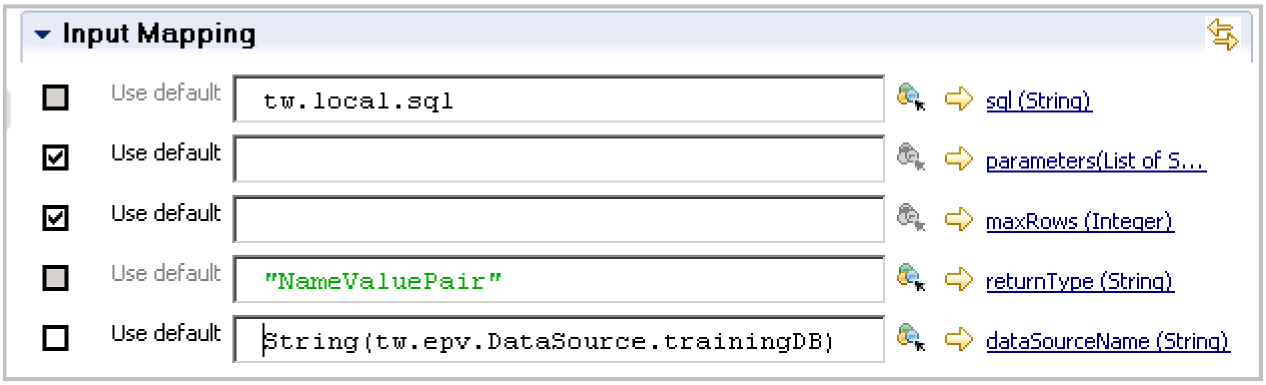
__ b. Click the **Diagram** tab to return to the service objects. Click the **SQL Execute Statement** object.

__ c. Click the **Properties > Data Mapping** menu option.

__ d. Map the `trainingDB` EPV to the dataSourceName variable. Enter `String(tw.epv.DataSource.trainingDB)` as the **dataSourceName**.

Important

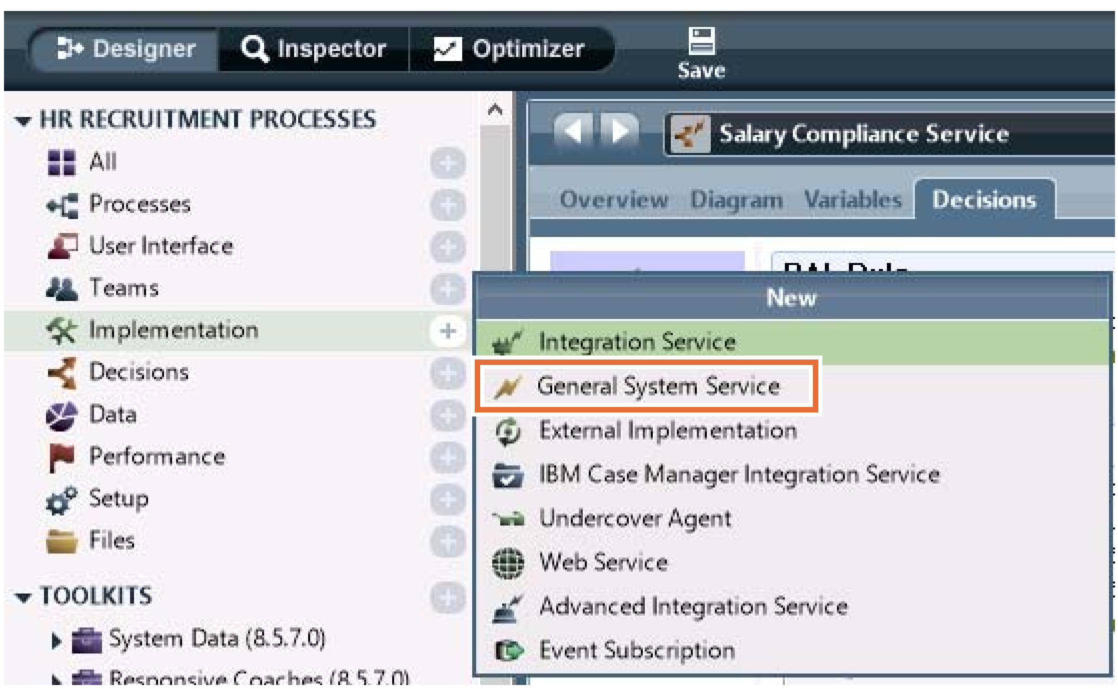
You must cast your EPV to a String. Replace the string that you previously mapped to the variable with `String(tw.epv.DataSource.trainingDB)`. Although you created the EPV as a String type, the system defines it as an EPV type when you map the variable. Therefore, you must cast the variable to String to use it.



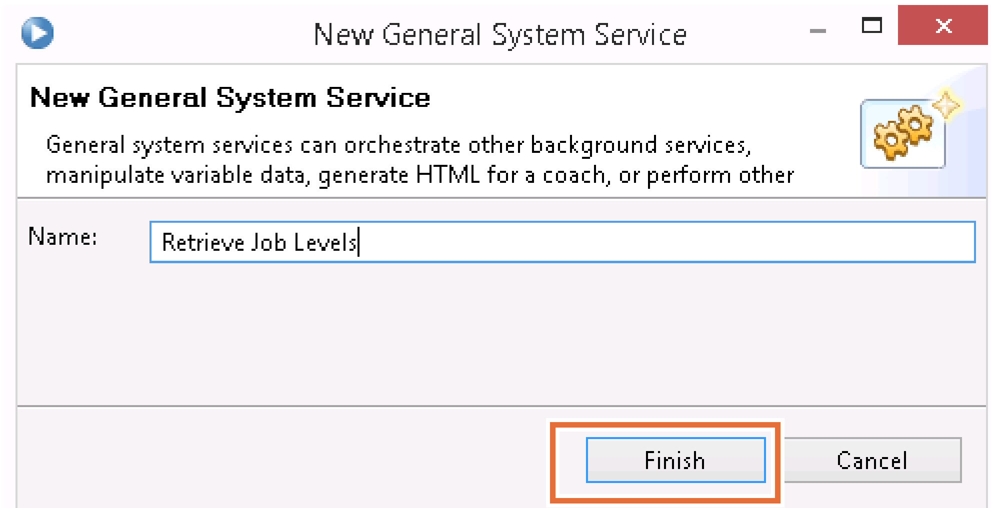
__ e. Save your work.

__ 8. Create a **General System Service** to retrieve job levels; however, use an environment variable (ENV) to accomplish this implementation.

__ a. In the Designer library, click the (**+**) plus sign for the **Implementation** menu option and click the **General System Service** option.



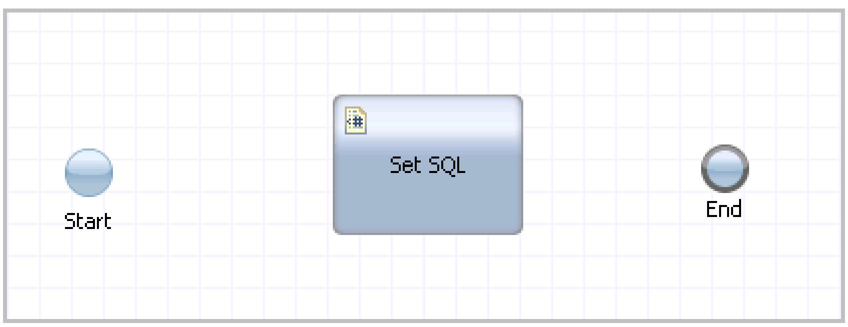
__ b. Name the service `Retrieve Job Levels` and click **Finish**.



__ 9. Add the service elements on the service modeling canvas.

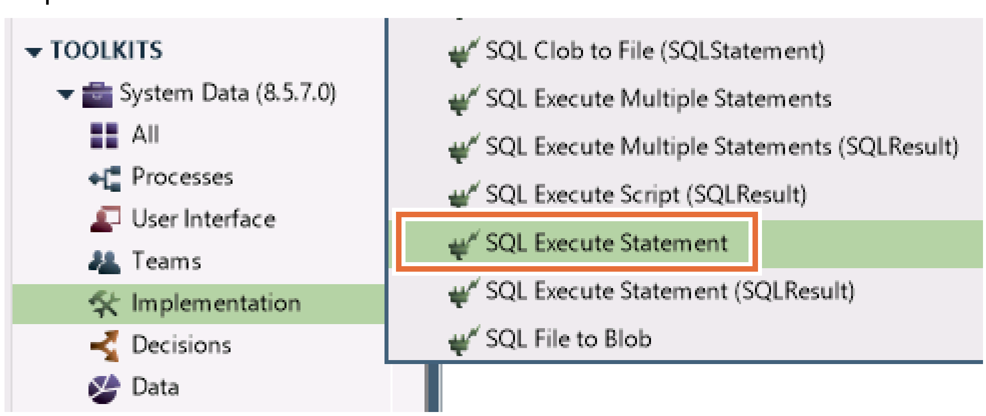
__ a. On the **Diagram** tab of the service, drag a **Server Scriptlet** from the service palette to the canvas.

__ b. Rename the server scriptlet: `Set SQL`

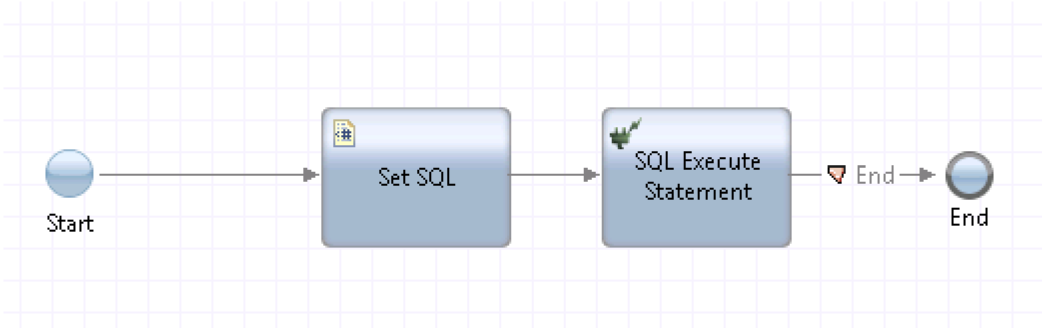


__ c. Expand the **System Data Toolkit** in the Process Designer client application library and then select the **Implementation** category.

__ d. Drag the **SQL Execute Statement** onto the canvas, next to the `Set SQL` server scriptlet.

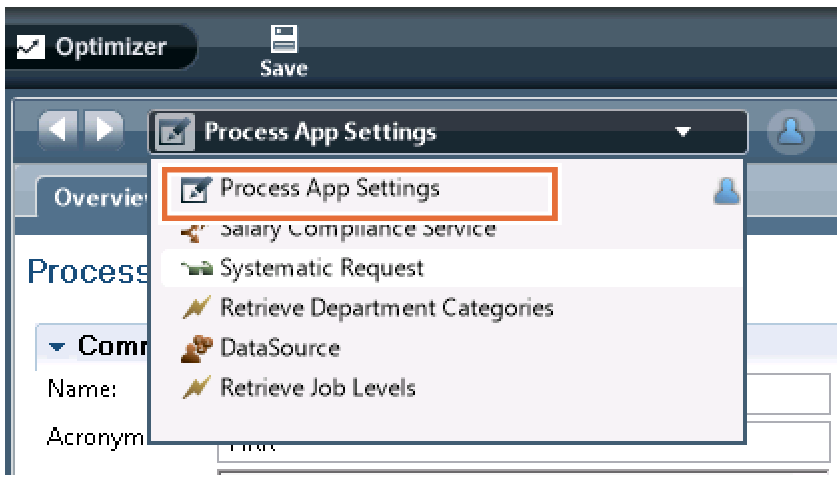


__ e. Finally, connect the objects on the canvas from the **Start** event to the server scriptlet **Set SQL** and then to the **SQL Execute Statement** and to the **End** event.

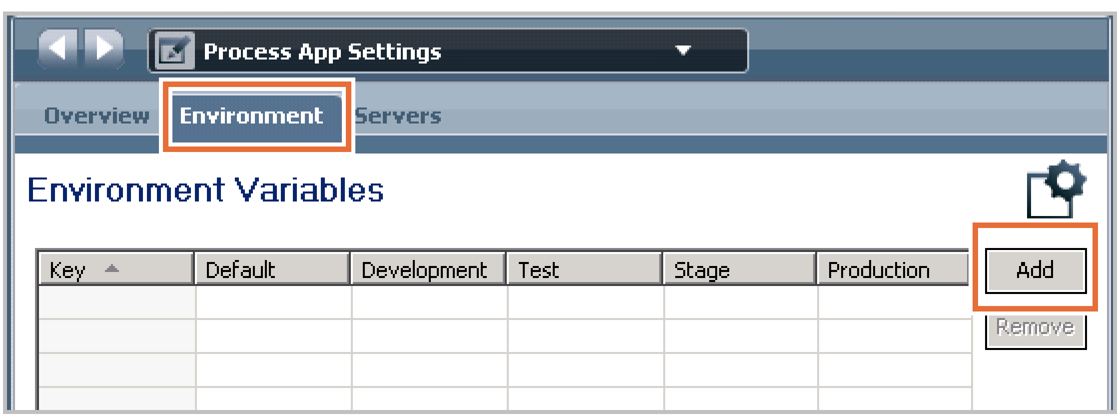


__ 10. Add an environment variable to your process application.

__ a. Select the **Process App Settings** from the History menu.

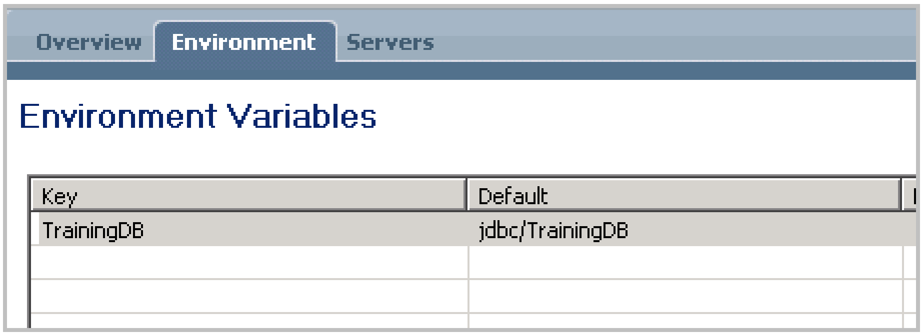


__ b. Select the **Environment** tab and then click **Add** on the right side of the **Environment Variables** section.



__ c. Complete the following values for the environment variable:

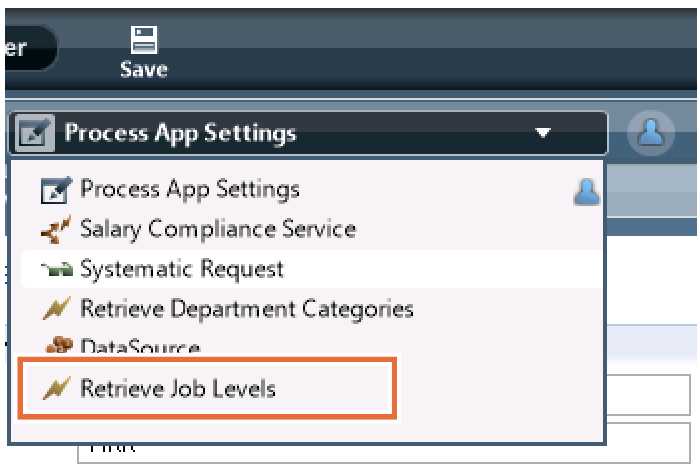
- **Key**: `TrainingDB`
- **Default**: `jdbc/TrainingDB`



__ d. Save your work.

__ 11. Add the variables to the **Retrieve Job Levels** service.

__ a. From the History menu, click the **Retrieve Job Levels** service.

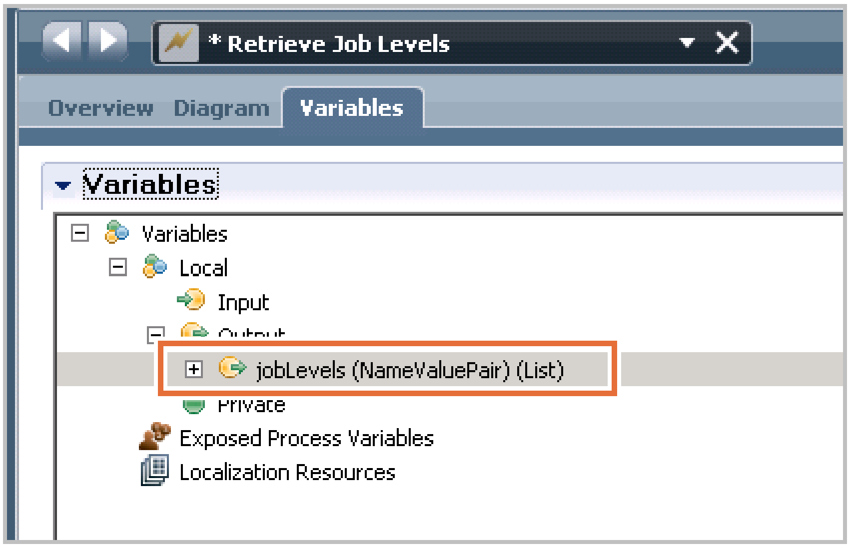


__ b. Click the **Variables** tab.

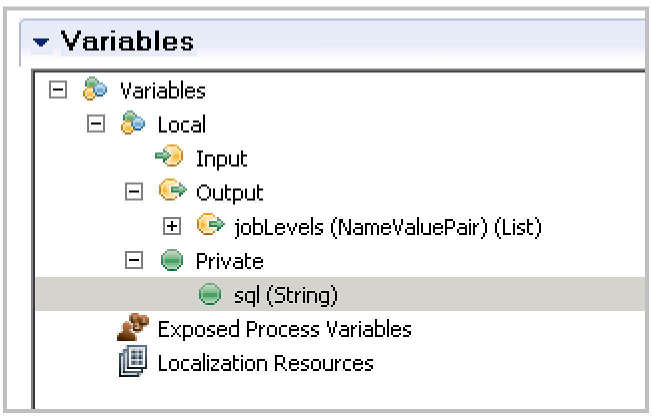
__ c. Click **Add Output** and name the variable: `jobLevels`

__ d. Select the **Is List** check box and then click **Select** for the Variable Type.

__ e. Select **NameValuePair** from the list.



__ f. Create a private variable: `sql (String)`



__ g. Save your work.

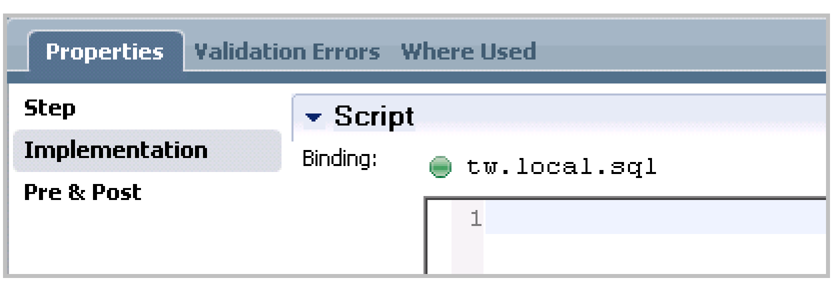
__ 12. Implement the `Set SQL` server scriptlet.

__ a. Click the **Diagram** tab.

__ b. Click the `Set SQL` server scriptlet.

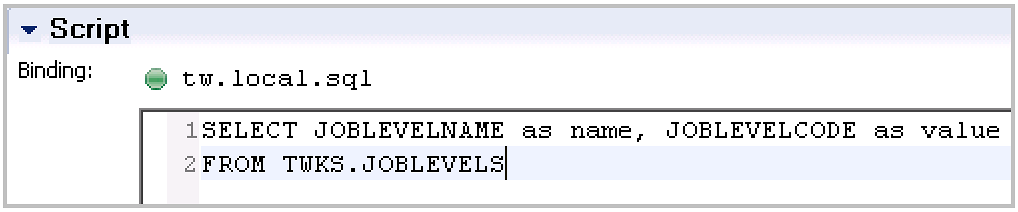
__ c. Click the **Properties > Implementation** menu option.

__ d. Click **Select** next to **Binding:** and click the **`sql`** `(String)` variable.



__ e. Type the following script in the **Implementation** script field:

```
SELECT JOBLEVELNAME as name, JOBLEVELCODE as value
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\`.

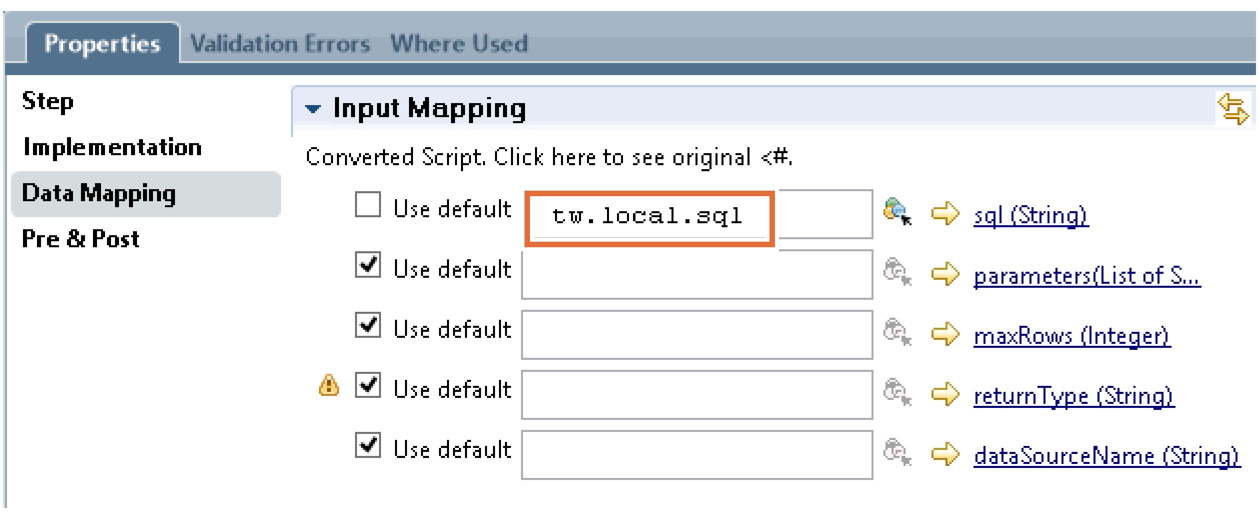
__ f. Save your work.

__ 13. Map the inputs and outputs of the `SQL Execute Statement` service.

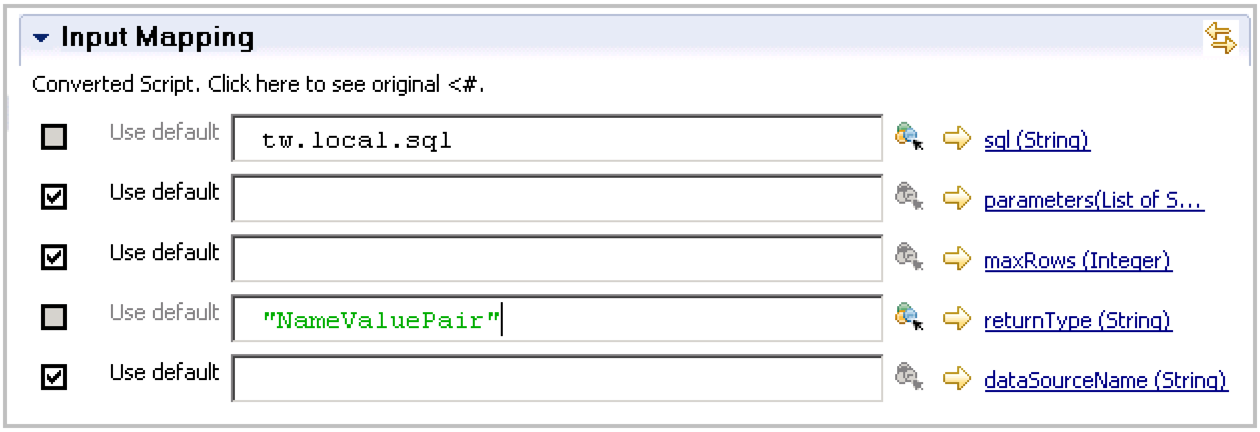
__ a. Click the **SQL Execute Statement** service.

__ b. Click the **Properties > Data Mapping** menu option.

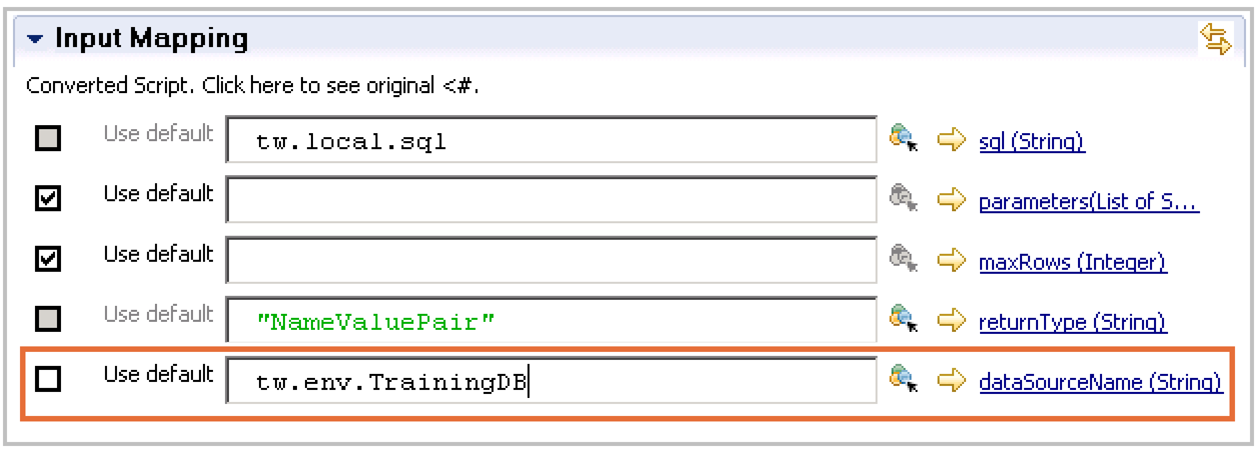
__ c. Under the **Input Mapping** section, clear the **Use default** check box for the `sql (String)` variable and map it to the **sql** `(String)` variable.



__ d. Clear the **Use default** check box for `returnType (String)` and add the text `"NameValuePair"` (include the quotation marks). Ensure that **Use default** is selected for the `parameters` and `maxRows` variables.



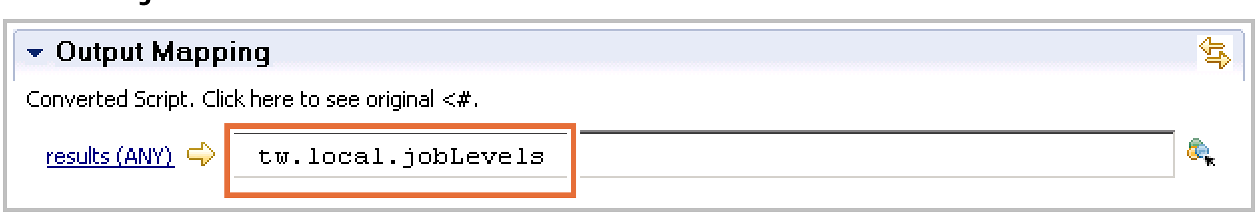
__ e. Clear **Use default** for `dataSourceName (String)` and enter: `tw.env.TrainingDB`



Note

You are not required to cast an ENV to a String.

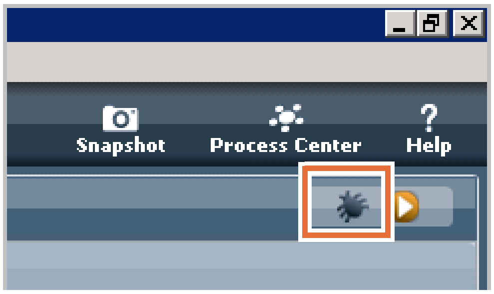
__ f. Under the **Output Mapping** section, map the `results (ANY)` variable to the **`jobLevels`** variable.



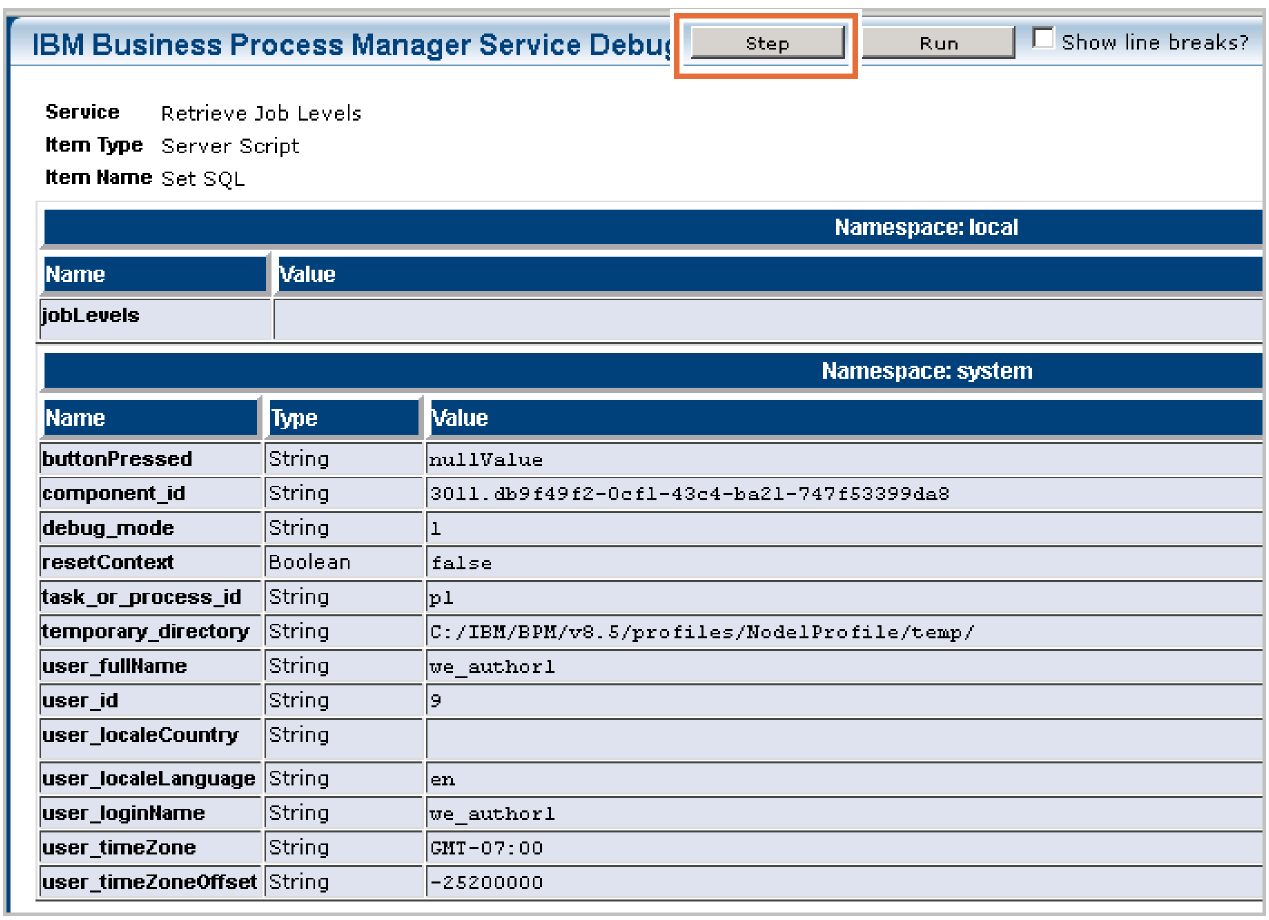
__ g. Save your work.

__ 14. Debug the service to determine whether the data is retrieved from the database correctly.

__ a. Similar to debugging a human service inside a browser, you can also use the debug services inside the Process Designer client application. Click the **debug icon** above the palette.



__ b. A browser window opens. This window provides metadata about the service that is debugged, and also displays all the variables and the values that are assigned to those variables as you step through the service. The top of the debugger shows the name of the service you are debugging and the item that is executed when you click **Step** (Set SQL). The `jobLevels` variable does not have an assigned value.



Service Retrieve Job Levels
Item Type Server Script
Item Name Set SQL

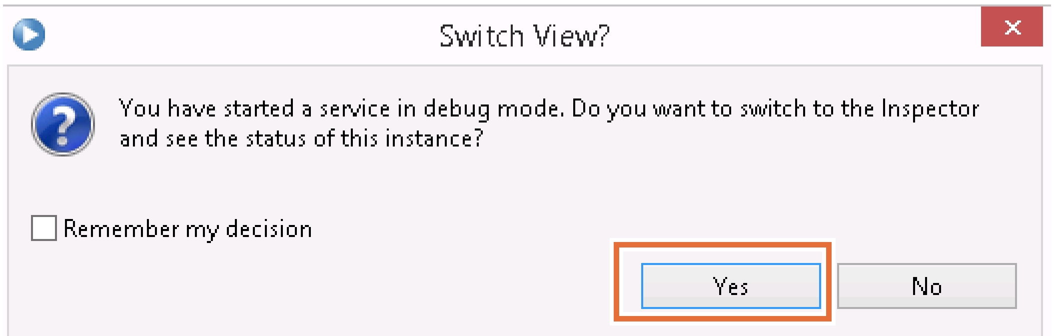
Namespace: local

Name	Value
jobLevels	

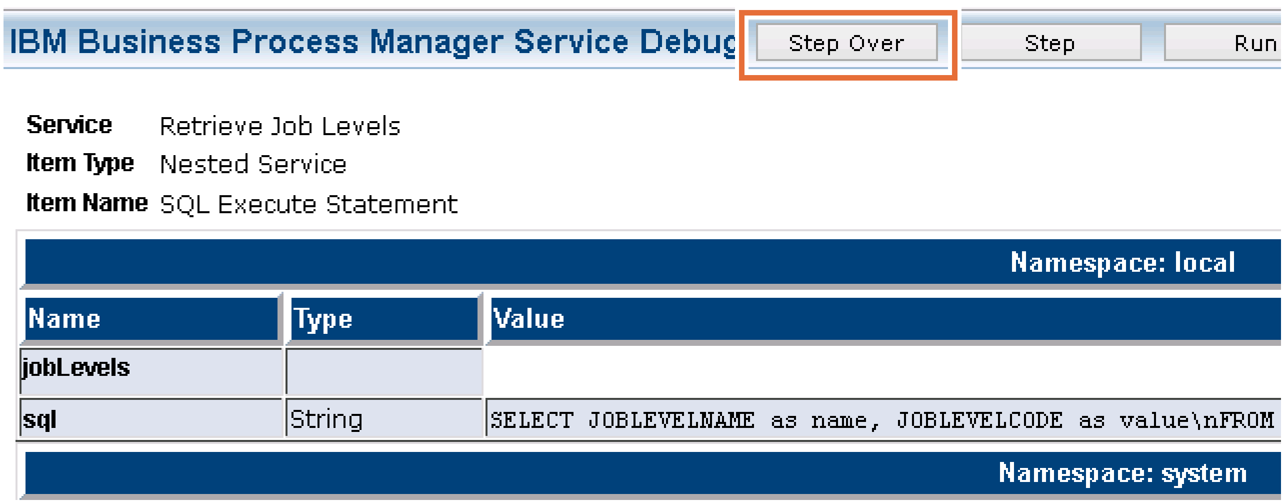
Namespace: system

Name	Type	Value
buttonPressed	String	nullValue
component_id	String	3011.db9f49f2-0cf1-43c4-ba21-747f53399da8
debug_mode	String	1
resetContext	Boolean	false
task_or_process_id	String	p1
temporary_directory	String	C:/IBM/BPM/v8.5/profiles/Node1Profile/temp/
user_fullName	String	we_author1
user_id	String	9
user_localeCountry	String	
user_localeLanguage	String	en
user_loginName	String	we_author1
user_timeZone	String	GMT-07:00
user_timeZoneOffset	String	-25200000

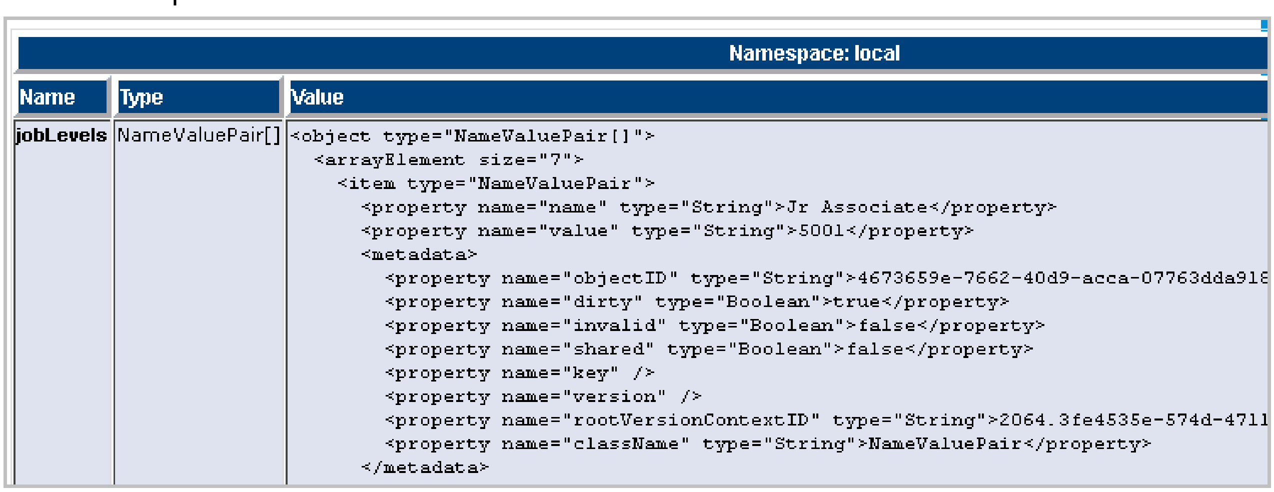
__ c. Minimize the browser window. An alert prompt asks whether you want to switch to the inspector view. Click **Yes**.



__ d. Return to the debugger inside the browser and verify that the `sql` variable is updated with the values that are assigned to the variable. Click **Step Over** to run the SQL Execute Statement nested service.



__ e. The database call executes and returns the data from the database. The variable is represented in XML format.

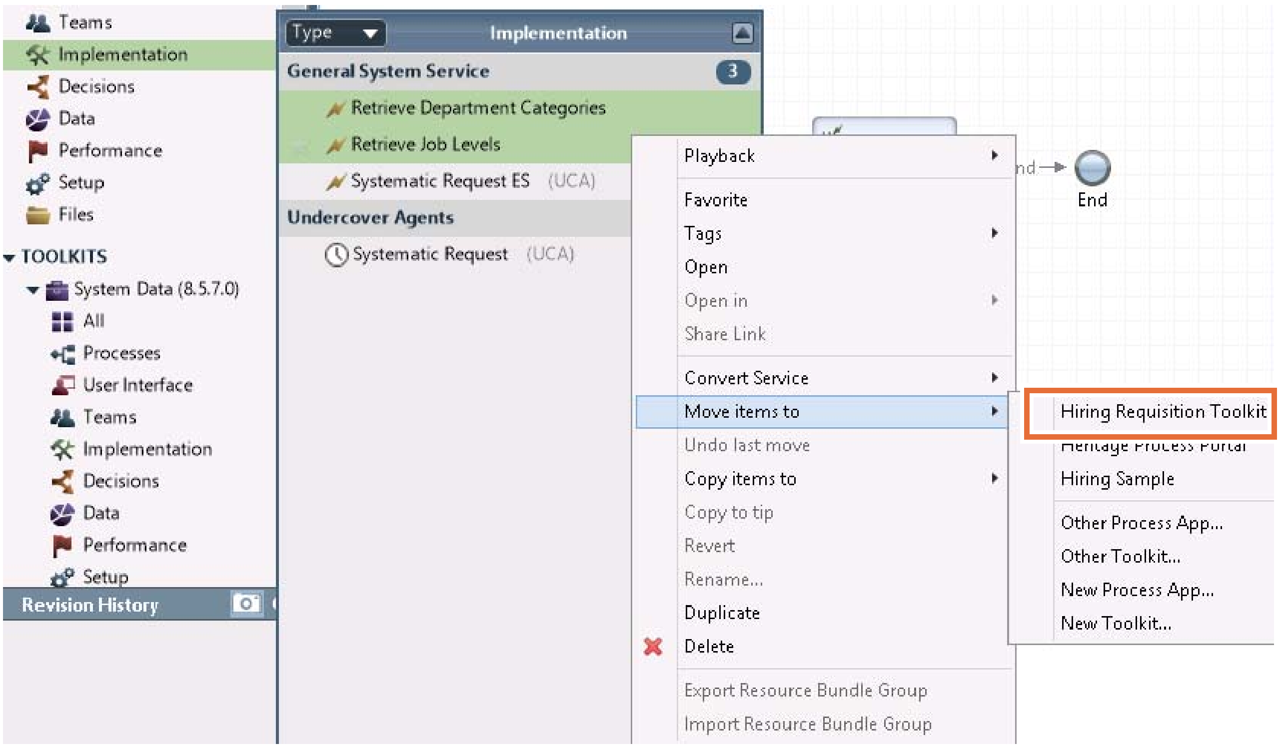


__ f. Click **Run** to complete the service, and close the browser.

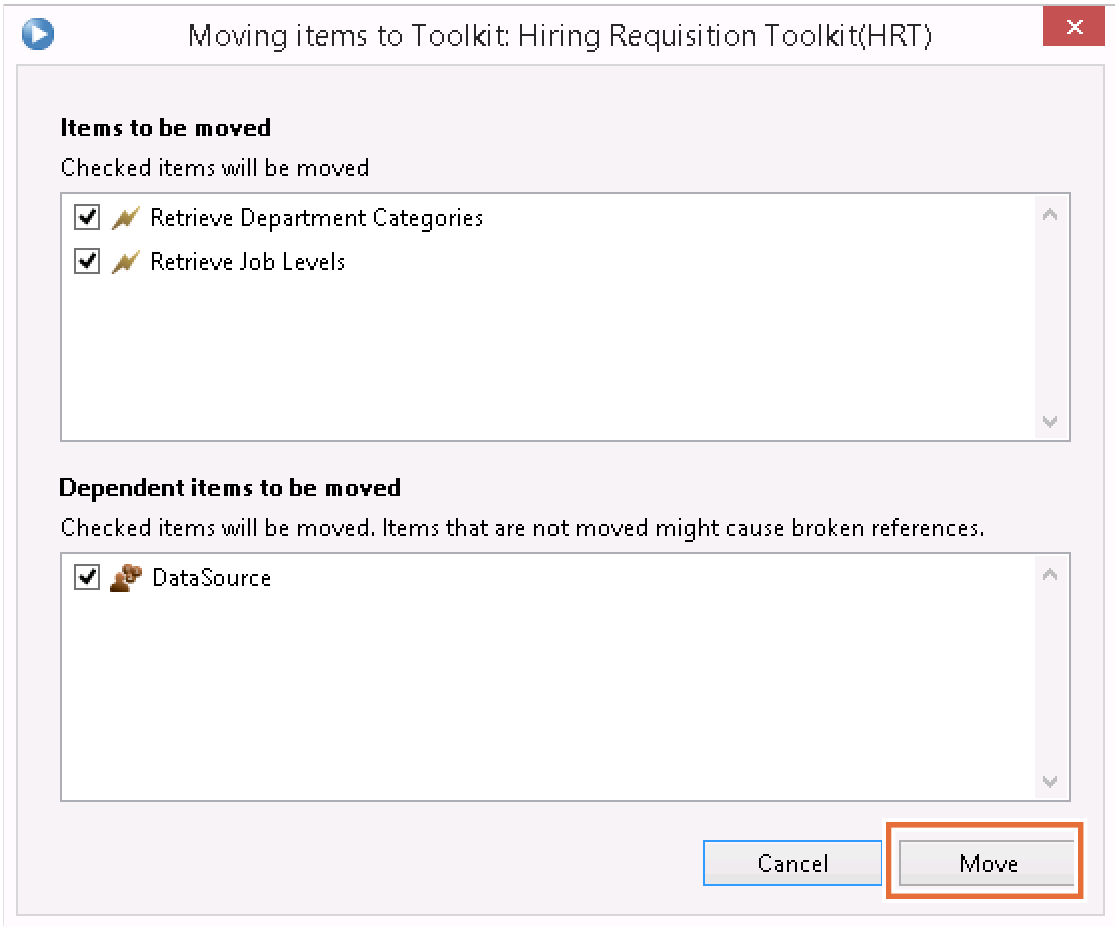
__ 15. To make these services available to other process applications, move the services to the toolkit.

__ a. Return to IBM Process Designer client application and switch to the **Designer** tab.

__ b. Open the **Implementation** category in the library. Hold down **Ctrl** and select both the **Retrieve Department Categories** and **Retrieve Job Levels** General System Services. Right-click and select **Move items to > Hiring Requisition Toolkit**.



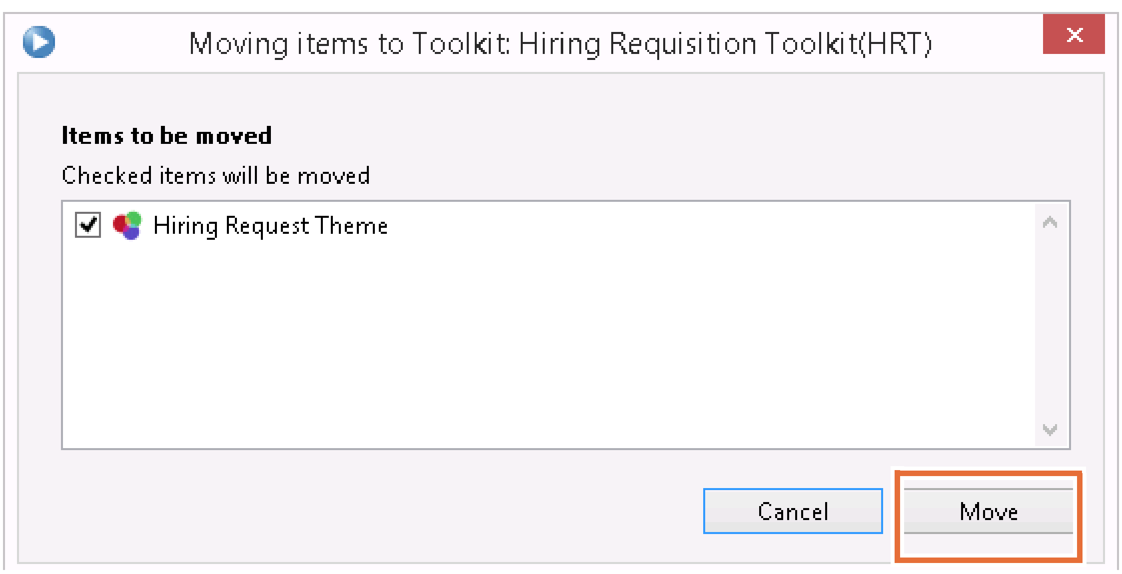
__ c. To move the DataSource EPV to the Toolkit, click **Move**.



__ 16. 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. Right-click **User Interface > Hiring Request Theme** in the library. Click **Move Item to > Hiring Requisition Toolkit**.

__ b. To move the Hiring Request Theme to the Toolkit, click **Move**.



Part 5: Change an input to a single select on a coach

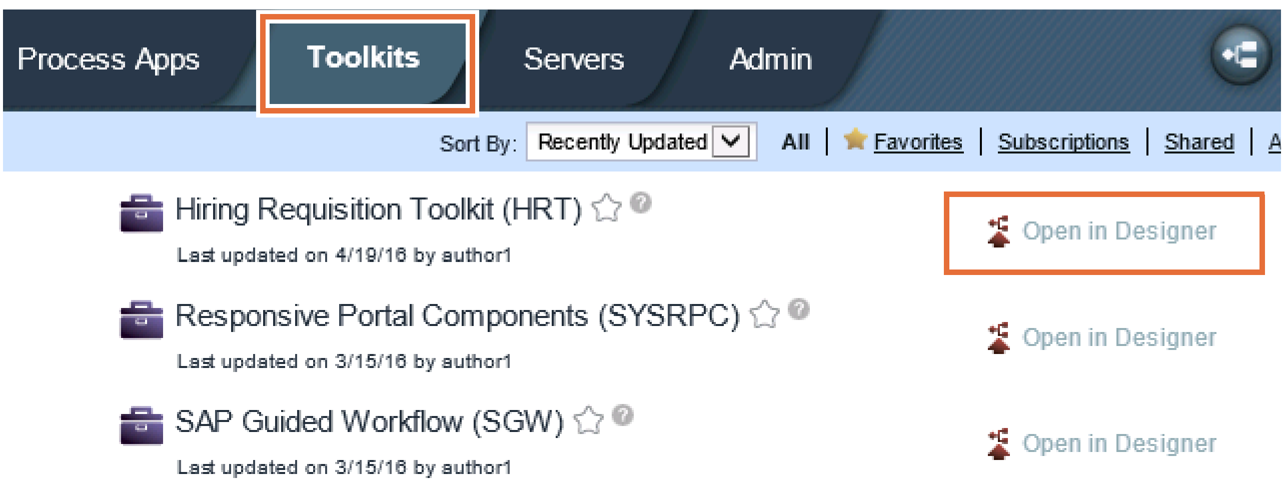
Until now, any input would be accepted for the **Department** input on the page. If only certain values are allowed, you must then approach the control in one of the following ways. The first way would be to limit the input to only acceptable values by using a selection like the single select on a coach. 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 Hiring Requisition Toolkit.

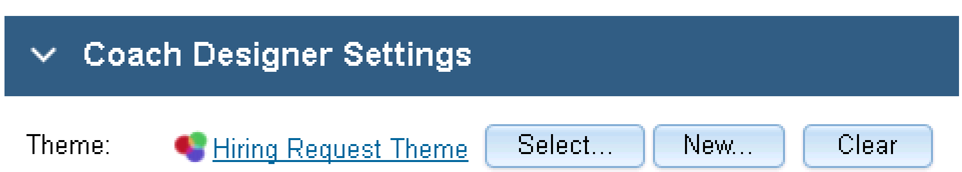
__ a. Return to the web Process Designer. Click the **Process Center** link.

__ b. Click the **Toolkits** tab.

__ c. Click **Open in Designer** for the Hiring Requisition Toolkit.



__ d. In the **Overview > Coach Designer Settings** section, click **Select** to select Hiring Request Theme.

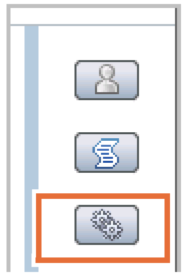


__ e. Save your changes.

__ 2. Add the **Retrieve Department Categories** service to the **Hiring Form** human service.

__ a. Select **User Interface** from the library and click **Hiring Form**.

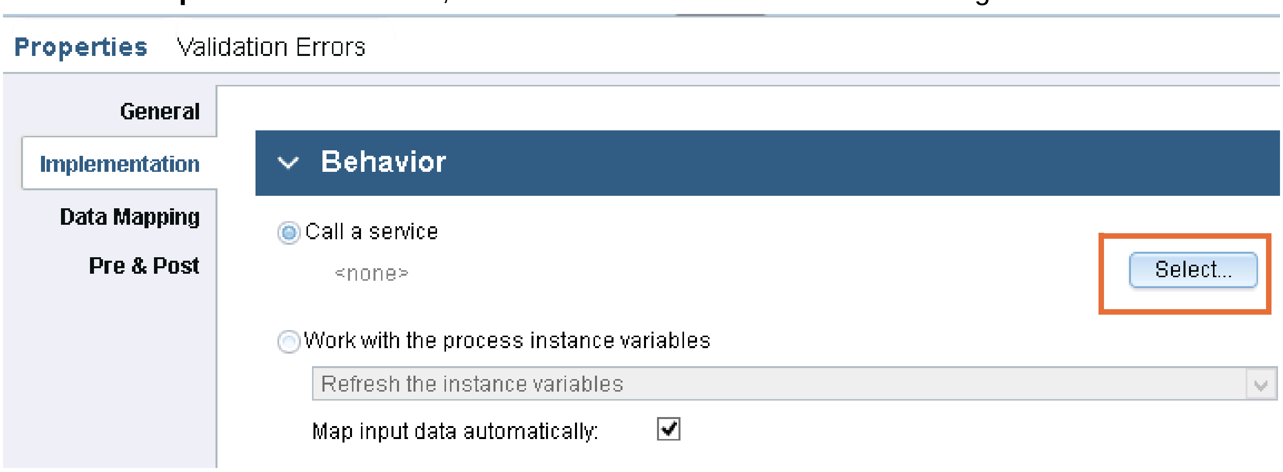
__ b. The **Hiring Form** service opens. In the **Diagram** tab, drag a **Service** from the palette onto the **Hiring Form** canvas between the **Start** event and the **Hiring Form** coach.



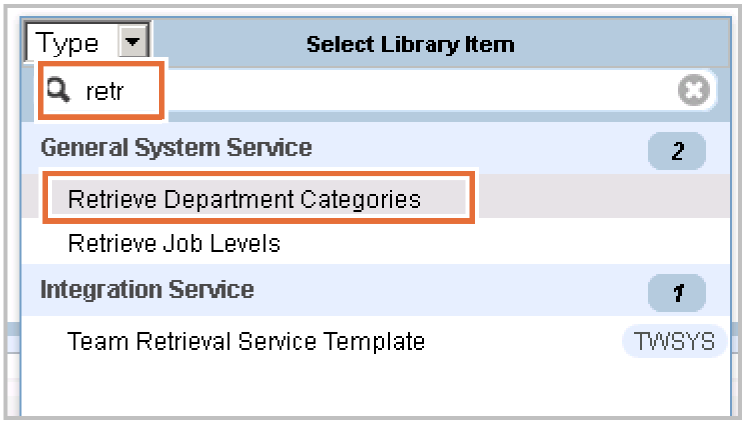
__ c. Reconnect the **Hiring Form** flow so that the new service is part of the flow.

__ d. Rename the service 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.

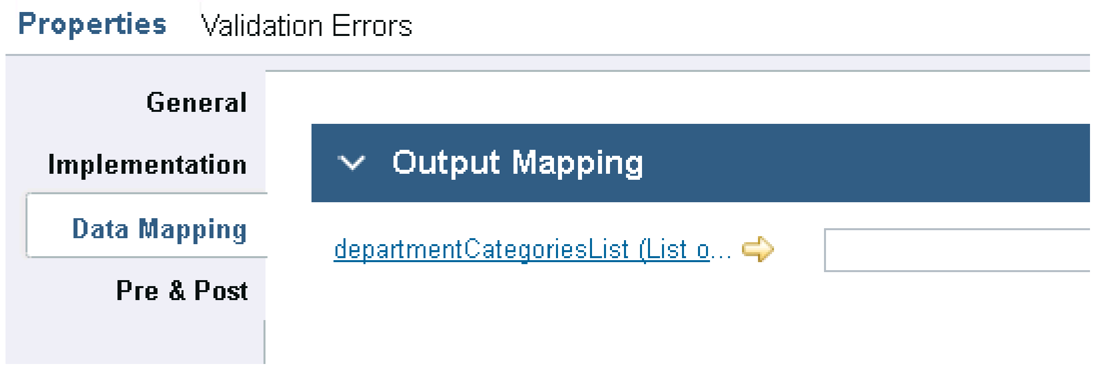


__ g. Save your work.

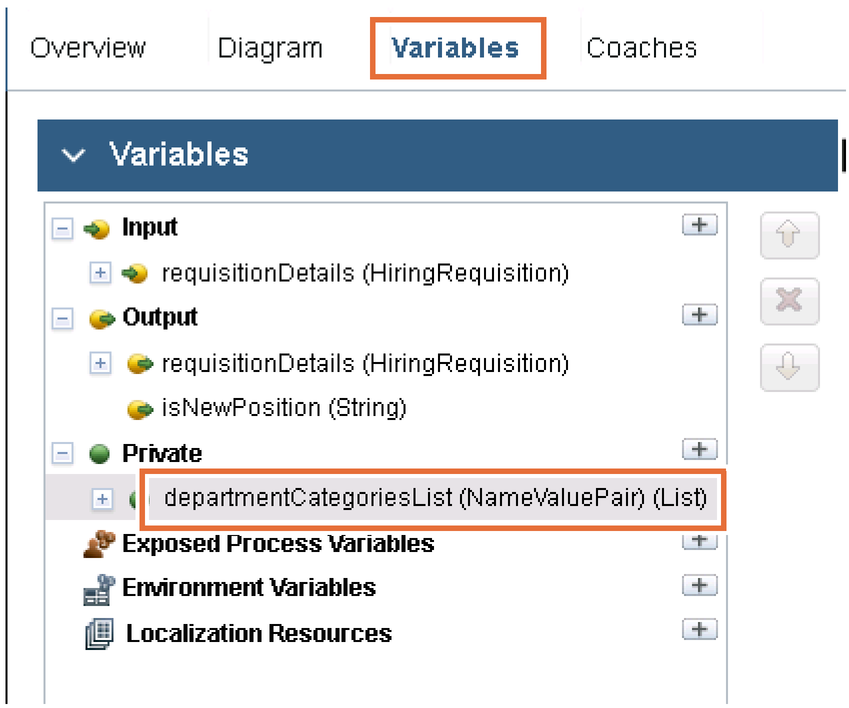
__ 3. Map the output variable of the integration service and allow the system to create the private variable.

Note

In the **Properties > Data Mapping > Output Mapping** section, you see that the service requires you to map a variable to the `departmentCategoriesList` variable.

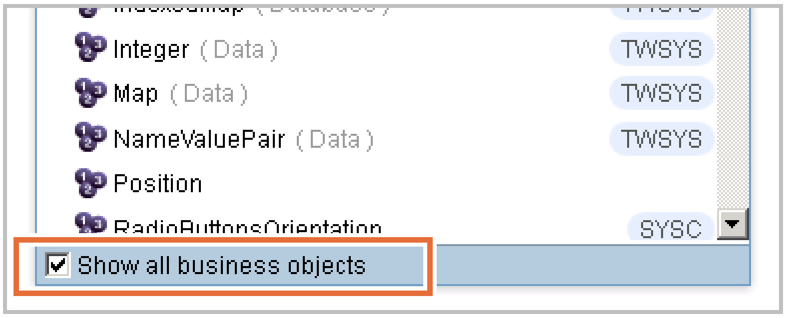


__ a. Click the **Variables** tab. Create a private variable: `departmentCategoriesList (NameValuePair) (List)`



Hint

Enable **Show all business objects** to choose the `NameValuePair` variable type.

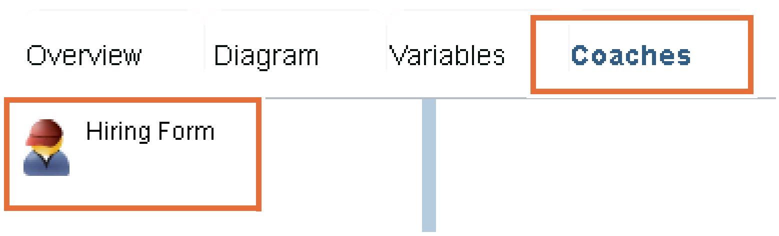


__ b. Return to the **Diagram** tab. In the **Properties > Data Mapping** section, map the `departmentCategoriesList` variable to the output.

__ c. Save your work.

__ 4. Change the control on the **Hiring Form** coach to a single select.

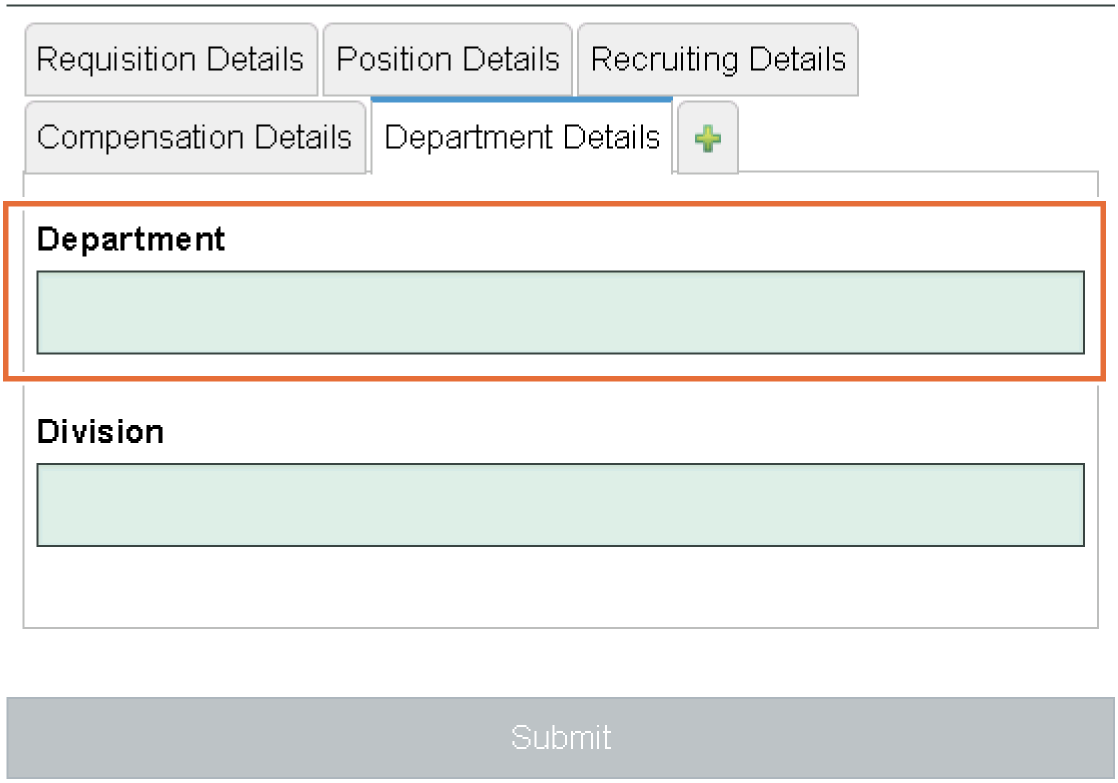
__ a. Click the **Coaches** tab and the **Hiring Form** coach.



Note

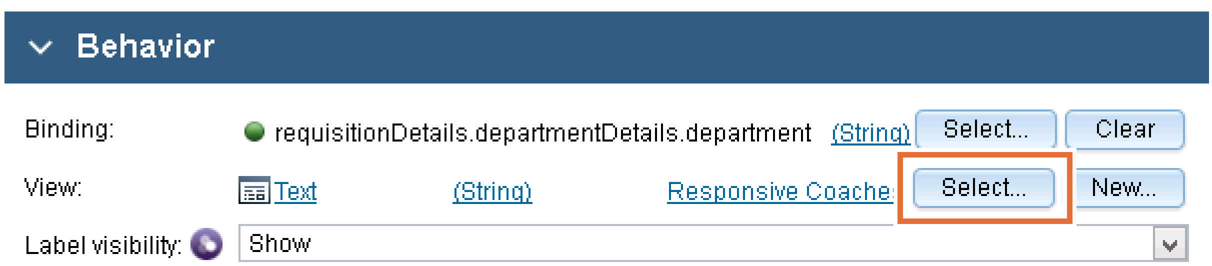
Initially when you view the coach, the server must generate the CSS and render the new coach on the screen, so a delay might occur before you see the effect.

__ b. From the **Department Details** tab, select the control **Department** that is bound to the department variable.

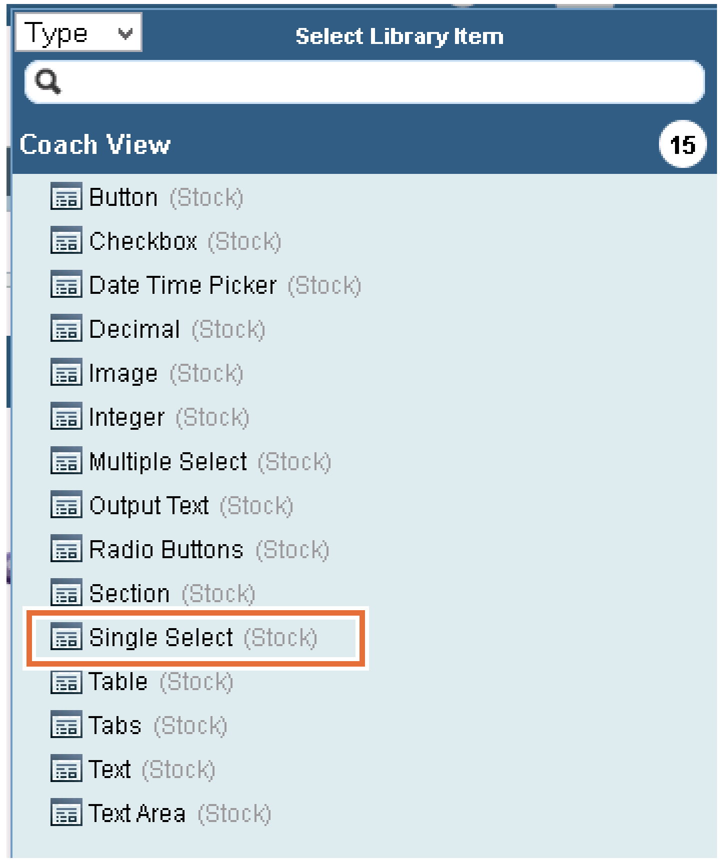


Change the view that the control is bound to. Right now, all the input boxes on the coach are text views. These controls are all variables that are bound to String data types.

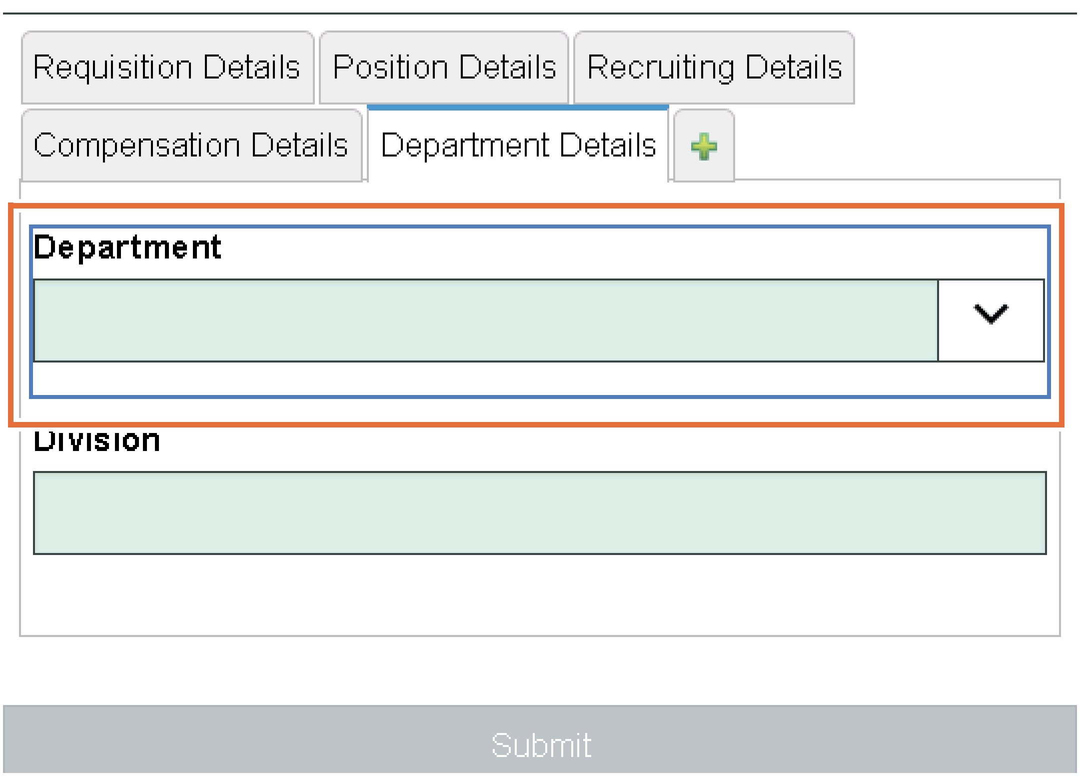
__ c. In the **Properties > General > Behavior** section, click **Select** next to View.



__ d. Select the option **Single Select (Stock)**.



__ 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.

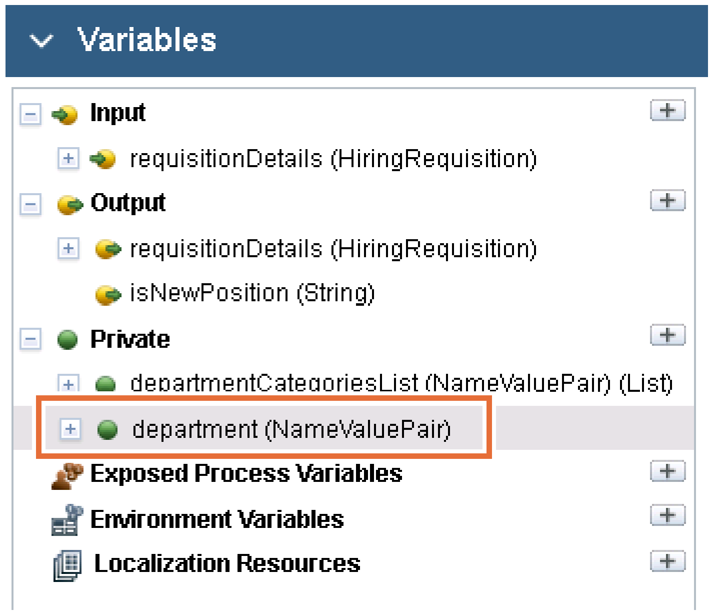


__ f. Save your work.

__ 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 on the **Hiring Form**.

__ b. Create a private variable: `department (NameValuePair)`



__ c. Save your work.

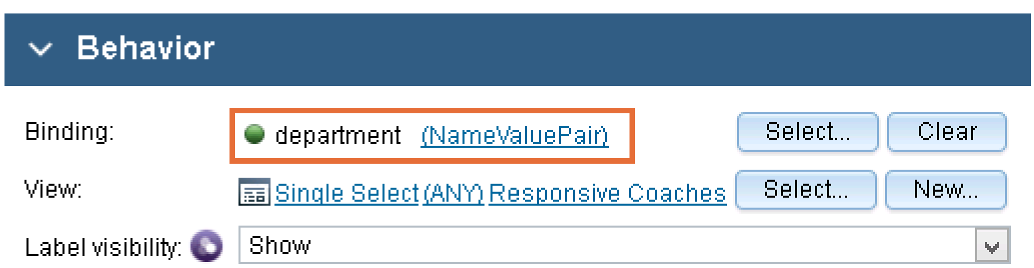
__ 6. Store the selected department to the department variable.

__ 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 **department (NameValuePair)** from the variable list.

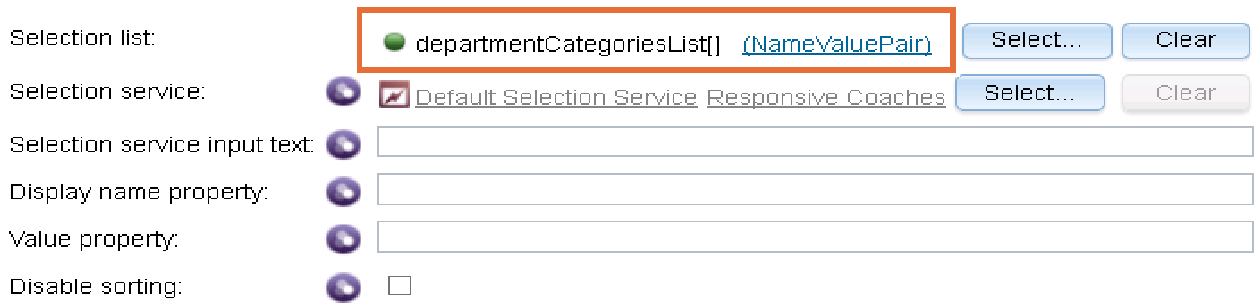


__ 7. Configure the setting that determines the list of selections available to the user on the `Department` select control.

__ a. Click the **Properties > Configuration** menu.

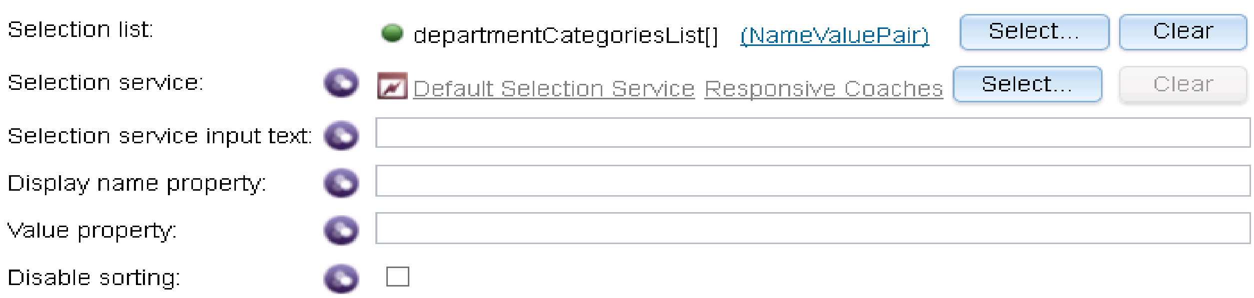
__ b. Click **Select** next to the **Selection list**.

__ c. Click the **departmentCategoryList (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. Your settings now resemble the following diagram:



__ d. Save your work.

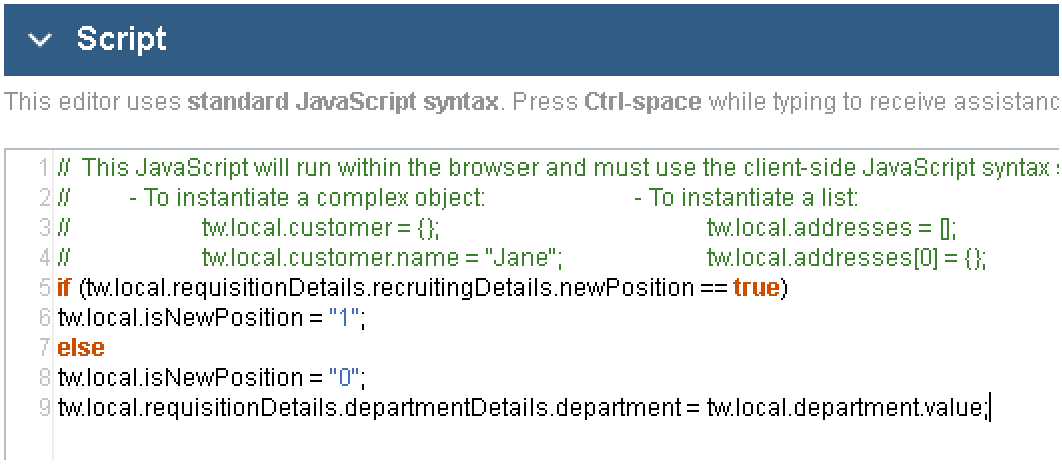
__ 8. The Department selection that is stored in the private `department (NameValuePair)` variable must be mapped to the pipeline variable business object. Bind the selection value back to the variable.

__ 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.value;
```



Note

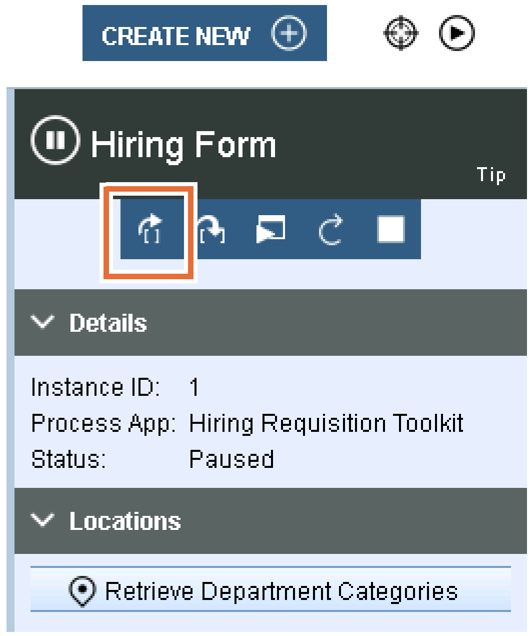
The script code is provided in the file `Script4.txt` at the location: `C:\labfiles\Exercise_Support_Files\Exercise09\`.

__ d. Save your work.

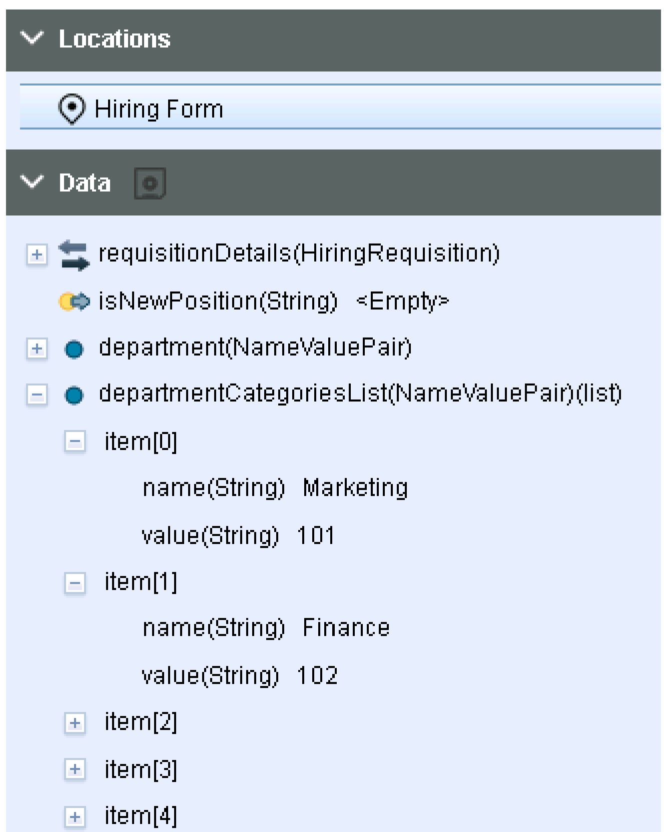
__ 9. Debug the coach, and verify that the **Department** single select works and binds the data to the variable.

__ a. Click the **Debug** icon.

__ b. Click **Step over**.

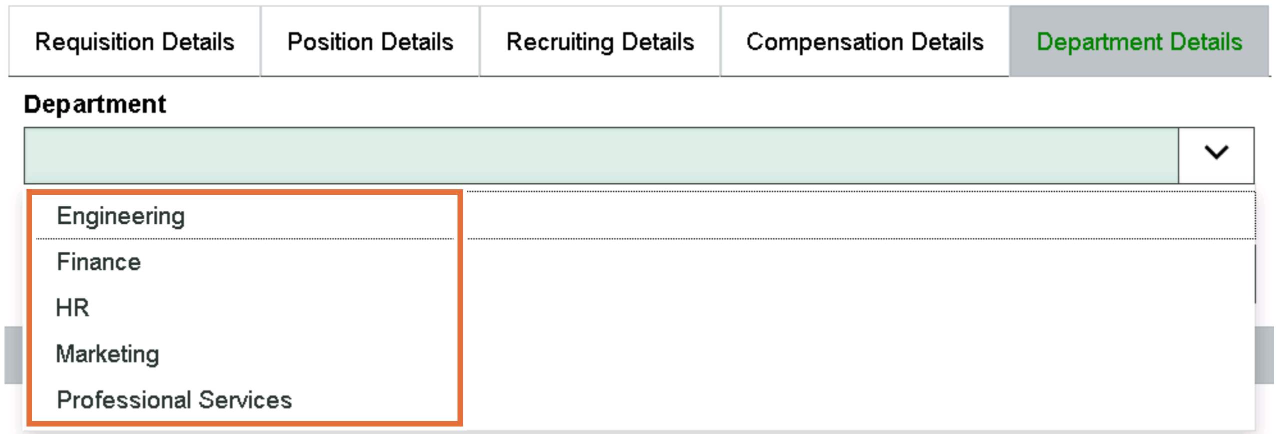


__ c. The `departmentCategoriesList` variable is populated with data from the integration that you created in the last exercise.



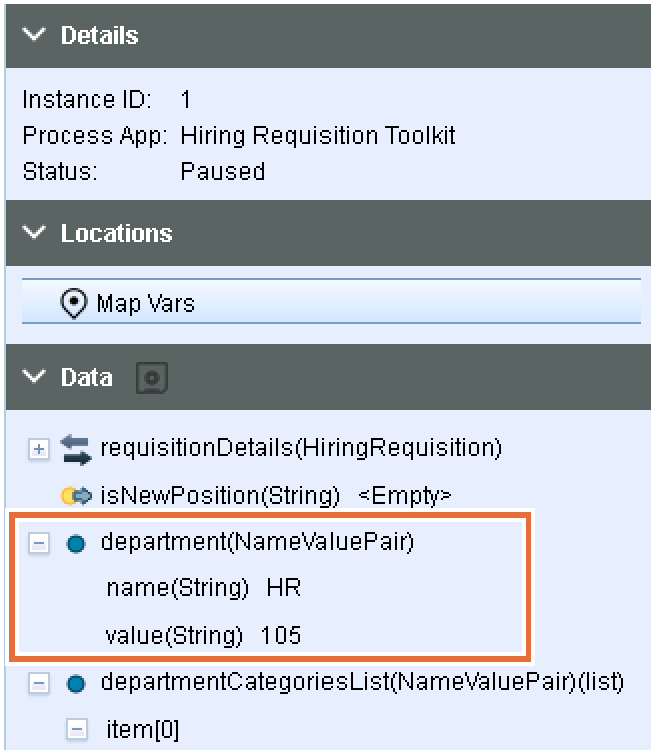
__ d. Click **Step over** to show the coach. The coach is displayed in the Playback window.

__ e. Click the **Department Details** tab. The database retrieval service now drives the selections that are presented to the user.



__ f. In the **Department** field, select **HR** and click **Submit**.

__ g. Verify that the data is bound to the variable inside the debugger.



__ h. 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.

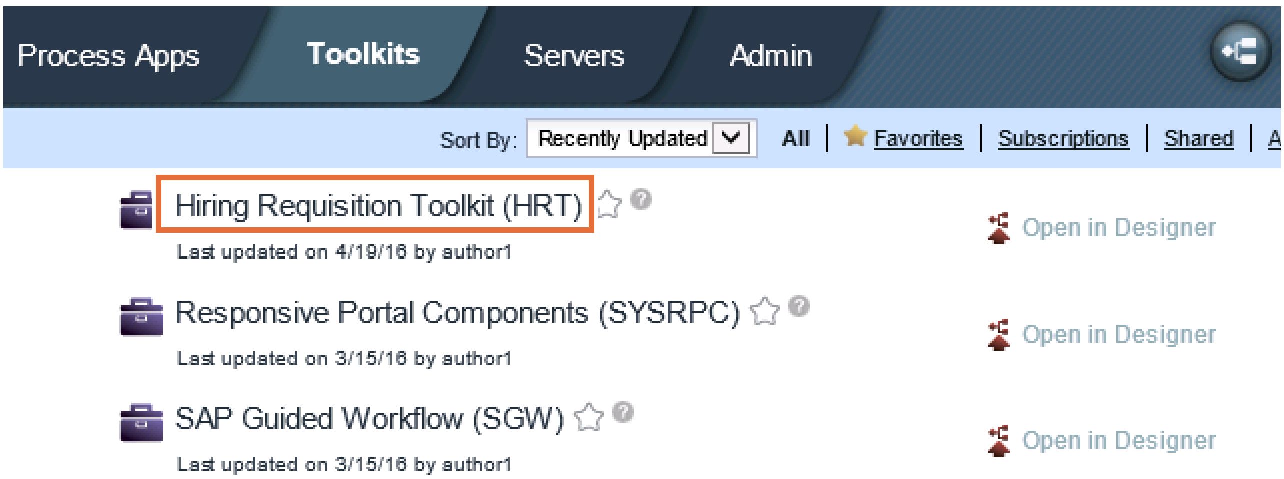
__ i. Close the **Debug Hiring Form** browser window.

__ j. Minimize the web Process Designer browser window.

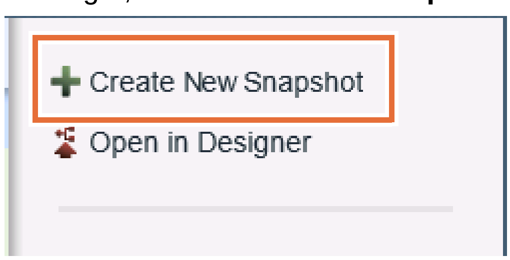
__ 10. Update the process application HR Recruitment Processes dependency to the most recent version of the Hiring Requisition Toolkit.

__ a. Return to the web Process Designer and click **Process Center** icon.

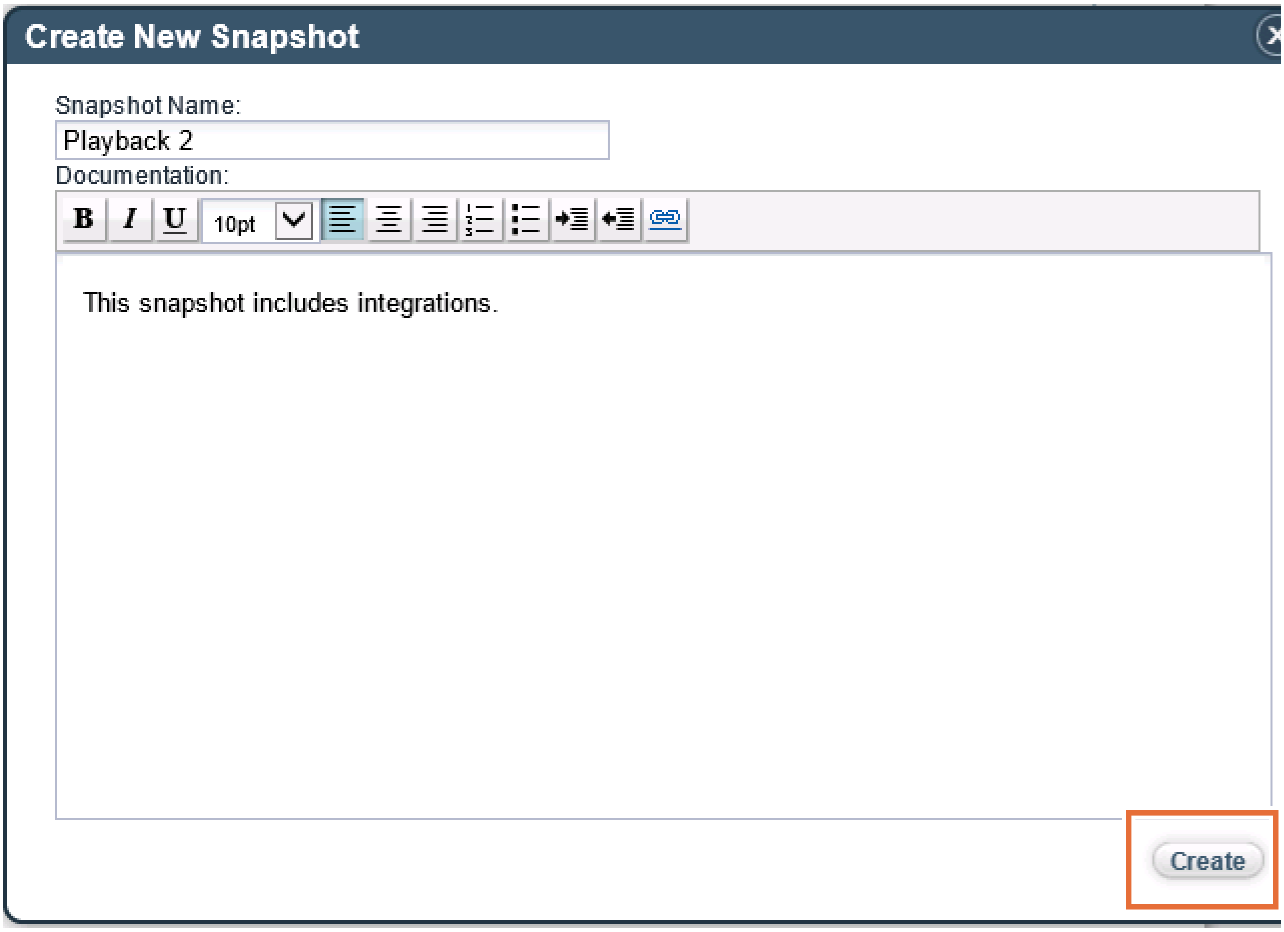
__ b. Click the **Toolkits > Hiring Requisition Toolkit (HRT)** link.



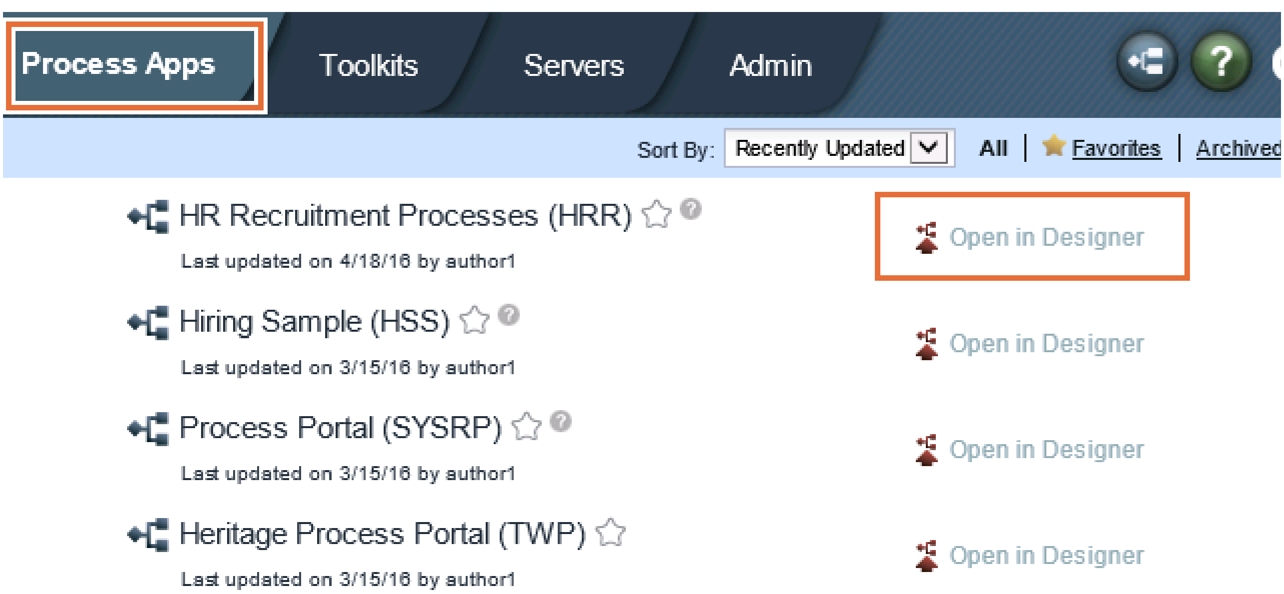
__ c. On the right, click **Create New Snapshot**.



__ d. Name the snapshot `Playback 2` and provide the following documentation: `This snapshot includes integrations.` Click **Create**.

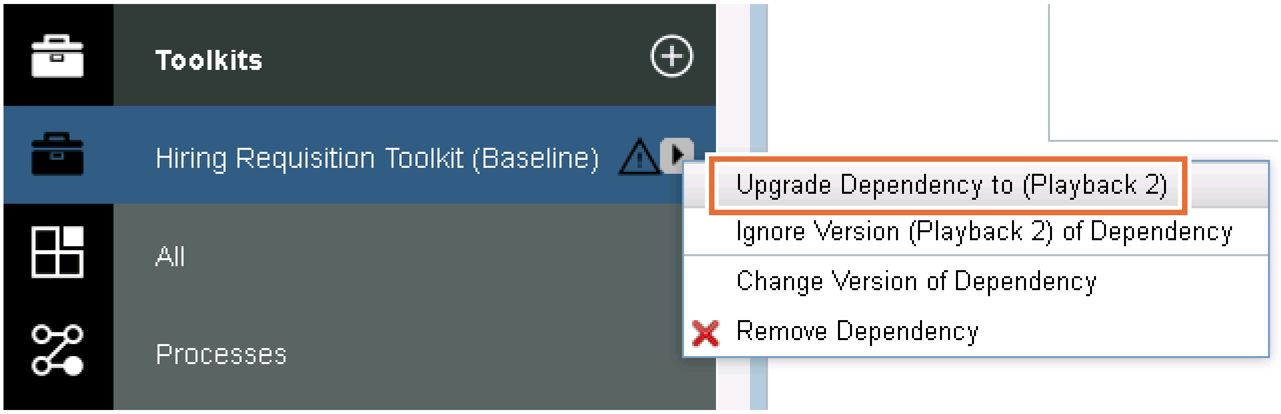


__ e. Click the **Process Apps** tab. Open the **HR Recruitment Processes** process application.

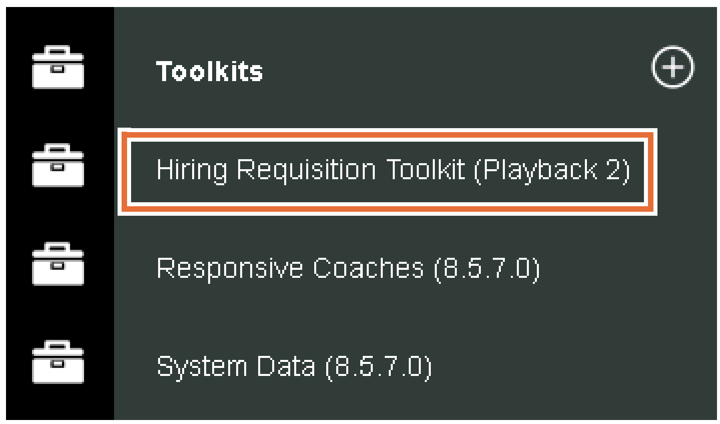


__ f. Click **Toolkits** in the library.

__ 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.

If you need more information about conducting the playback, consult 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 web 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

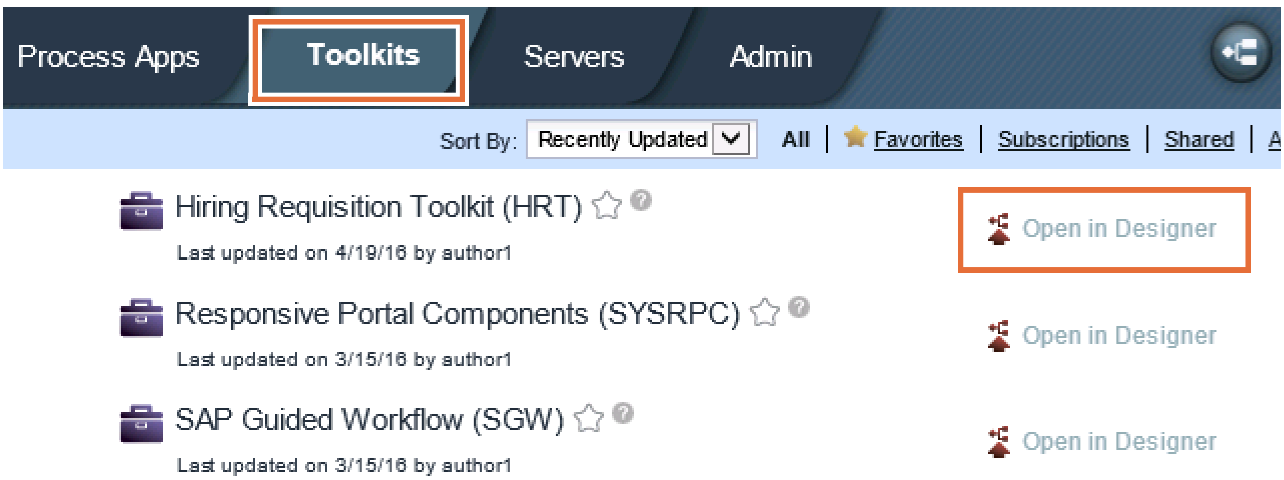
Part 1: Implement exception handling in a service

__ 1. Add a Catch Exception event to the Hiring Form service.

__ a. Return to the web Process Designer and click the **Process Center** icon.

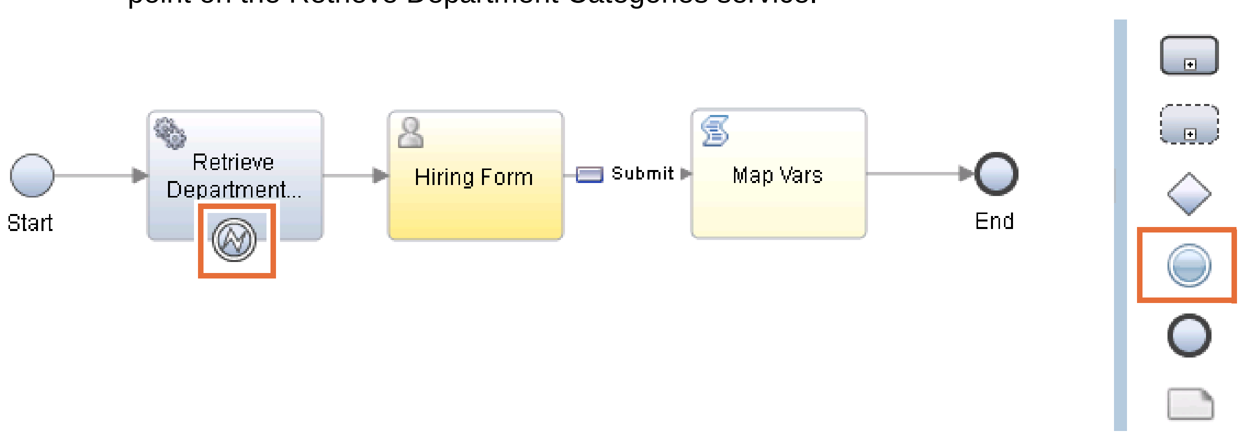
__ b. Click the **Toolkits** tab.

__ c. Click **Open in Designer** to open the **Hiring Requisition Toolkit (HRT)**.



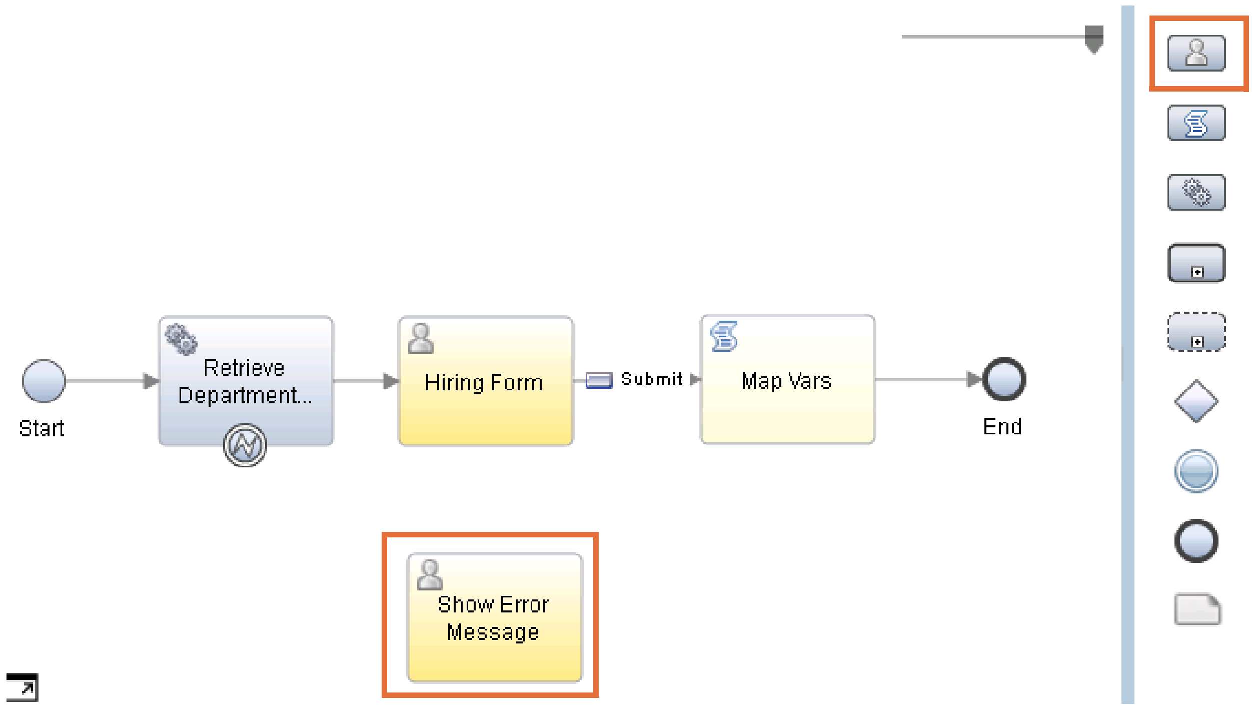
__ d. In the Hiring Requisition Toolkit, open the **User Interface > Hiring Form** human service.

__ e. In the **Diagram** tab, drag an **Intermediate Event** from the service palette on to a control point on the Retrieve Department Categories service.



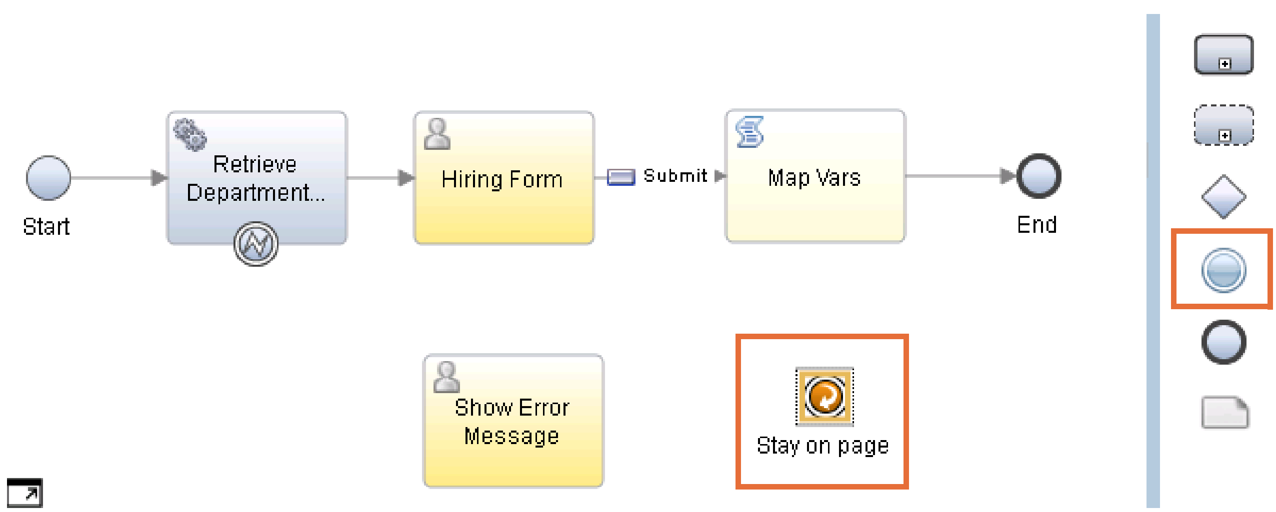
__ 2. Add a coach to display an error message.

__ a. Drag a **Coach** from the service palette onto the canvas. Rename the coach: `Show Error Message`

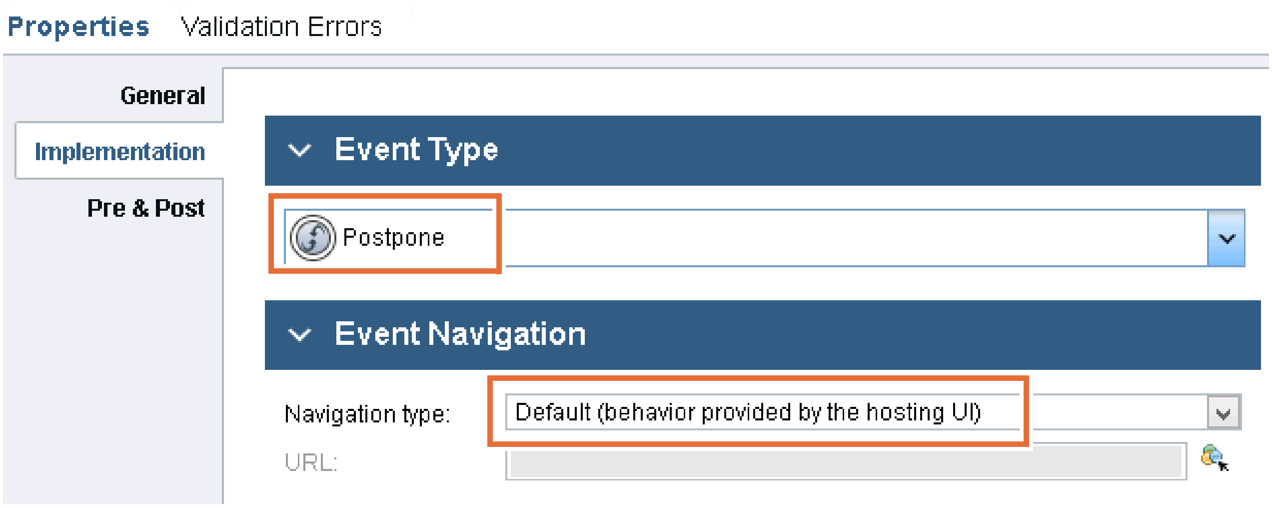


__ 3. Add a postpone task service.

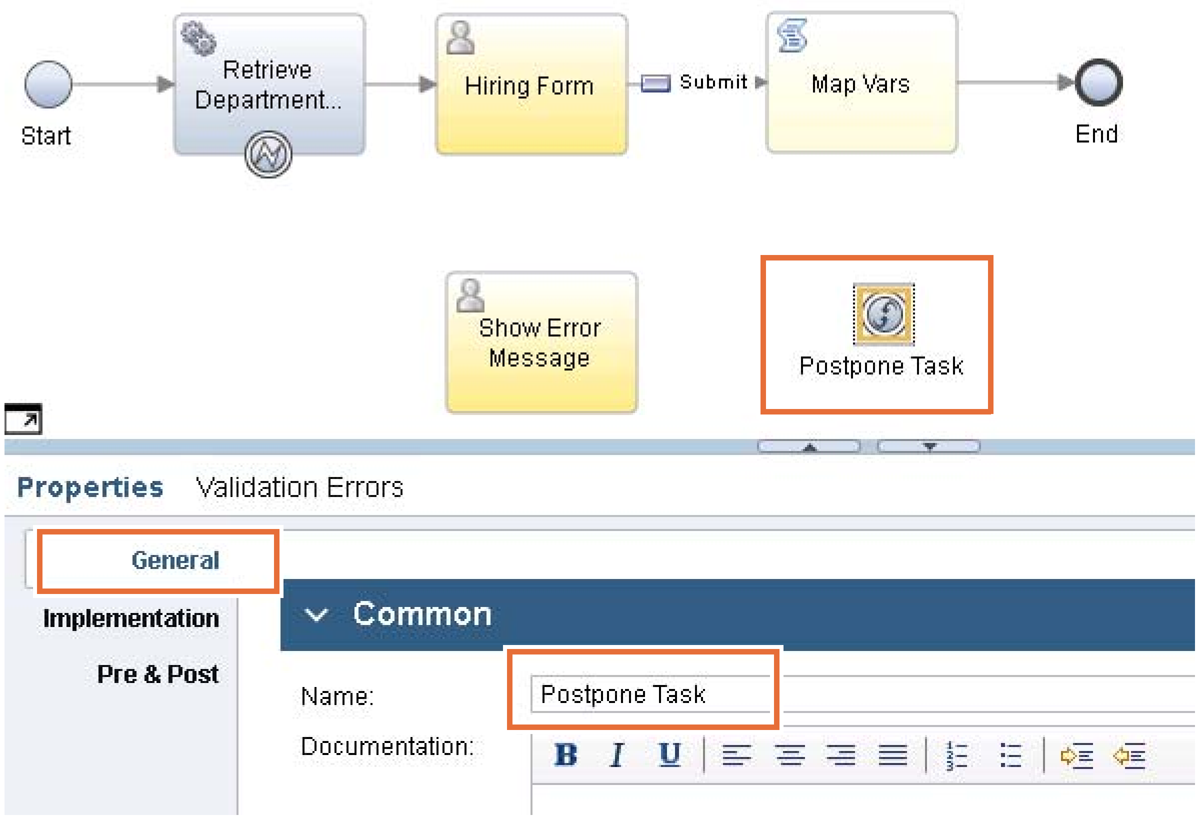
__ a. Drag an **Intermediate Event** from the palette to the right of the **Show Error Message** coach.



__ b. Open the **Properties > Implementation** menu and change the Event Type to **Postpone**. Leave the Event Navigation as **Default (behavior provided by the hosting UI)**.



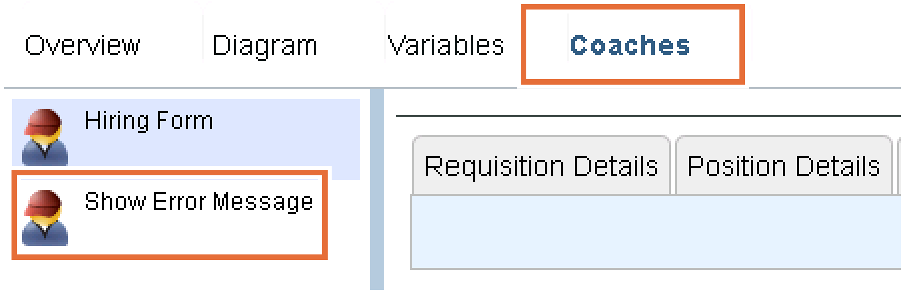
__ c. Open the **Properties > General > Common** section and rename the new event: `Postpone Task`



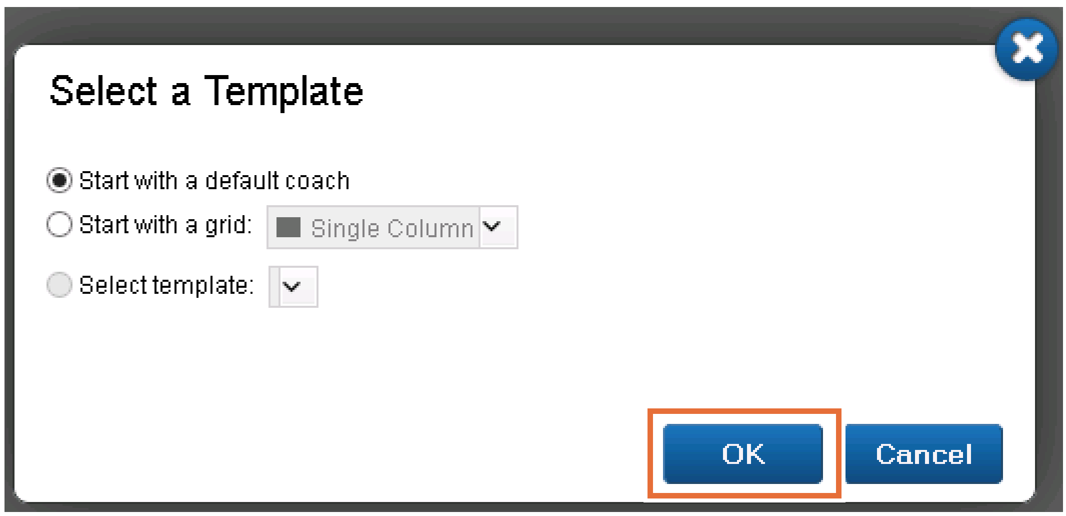
__ d. Save your work.

__ 4. Create the error message coach.

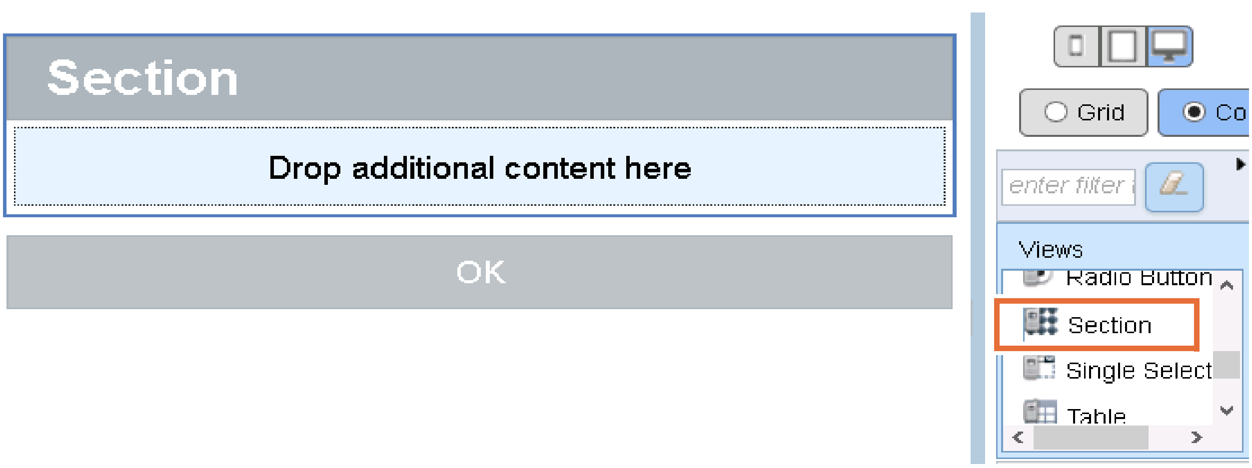
__ a. Click the **Coaches** tab and select the **Show Error Message** coach.



__ b. In the **Select a Template** window, select the default **Start with a default coach** and click **OK**.



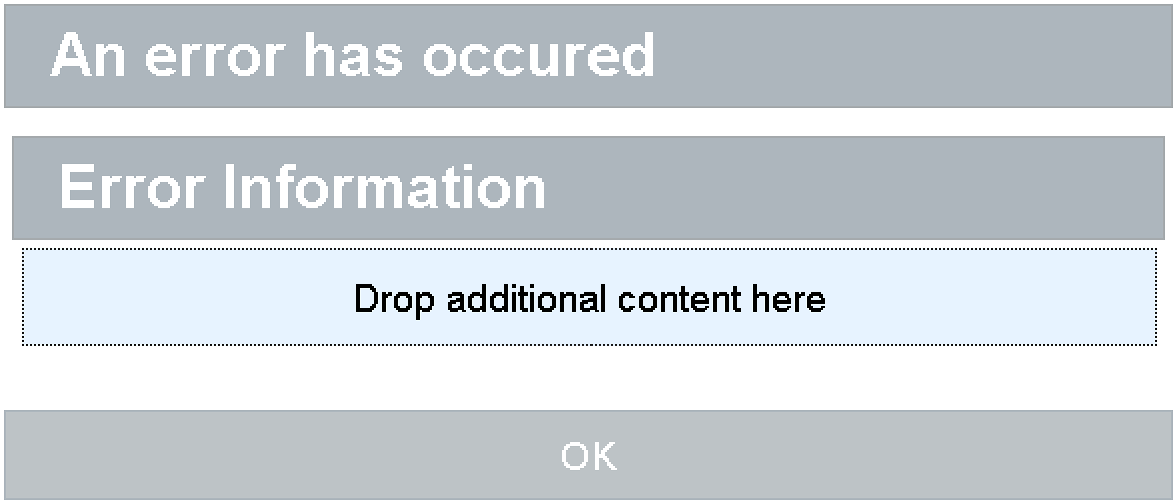
__ c. Drag a **Section** from the Stock section in the palette to the top of the coach designer canvas.



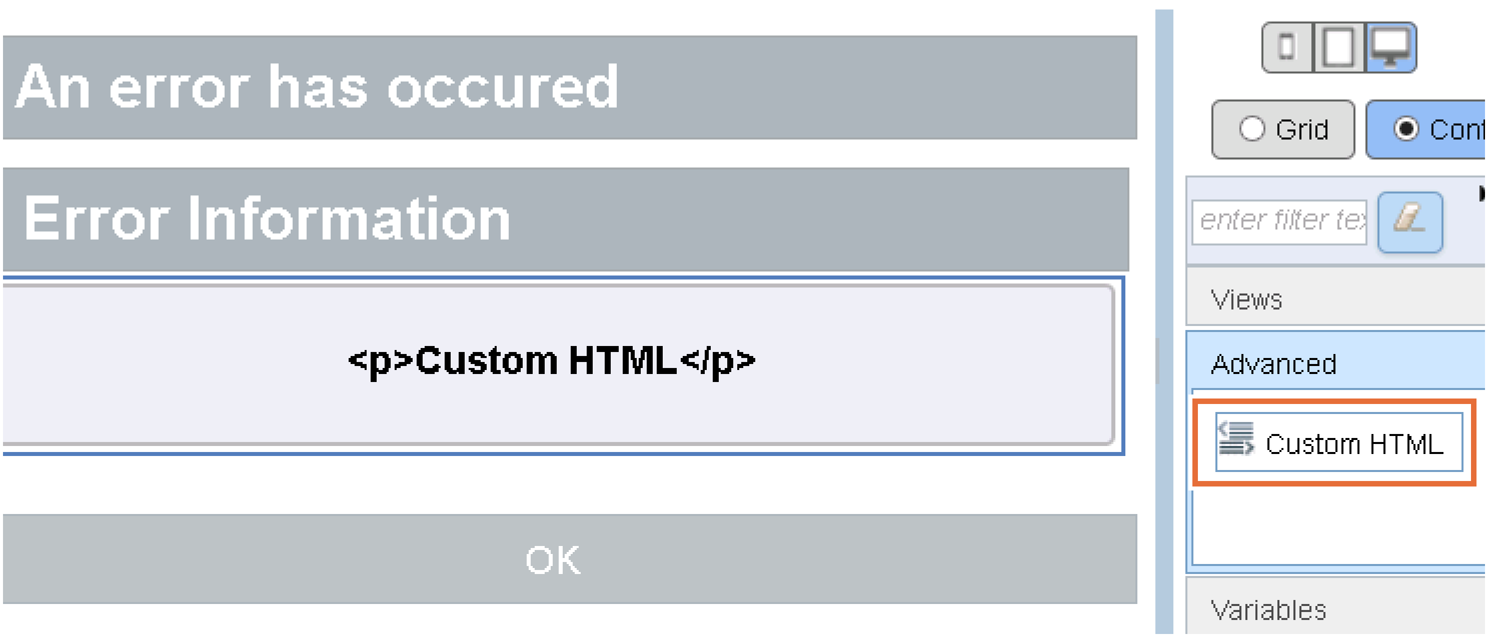
__ d. Change the display label on the section to: `An error has occurred`

__ e. Drag another section inside the first one.

__ f. Change the display label on the internal section to: `Error Information`

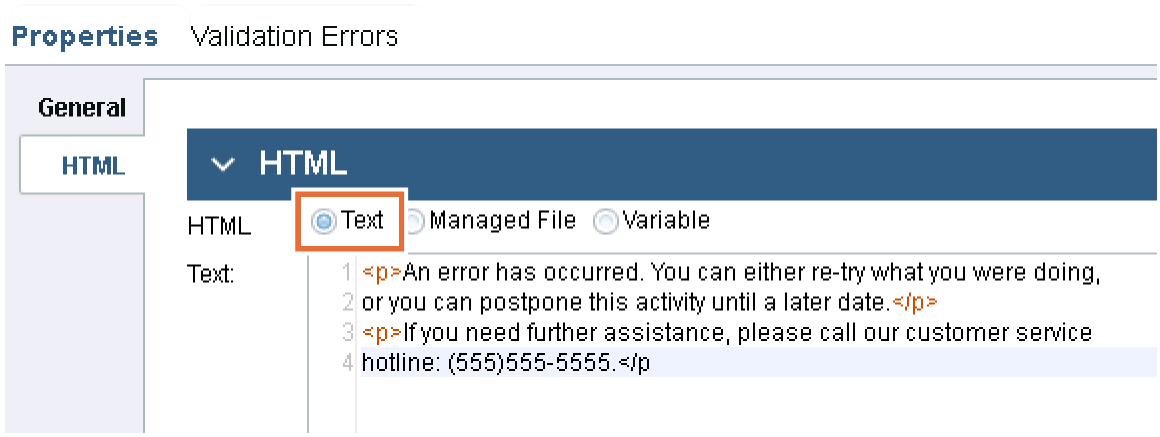


__ g. Drag a **Custom HTML** control from the **Advanced** palette tray into the Error Information section.

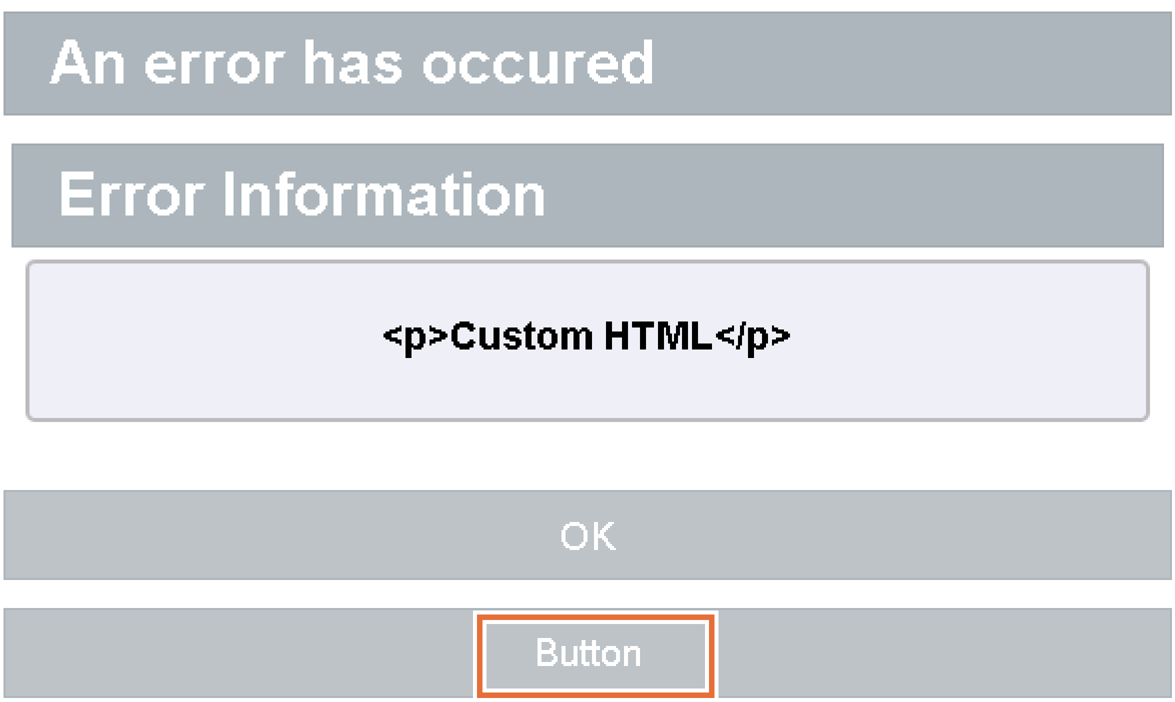


__ h. Under the **Properties > HTML** menu, enable **Text** and add error information to the HTML text block that is provided to the user:

```
<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>
```

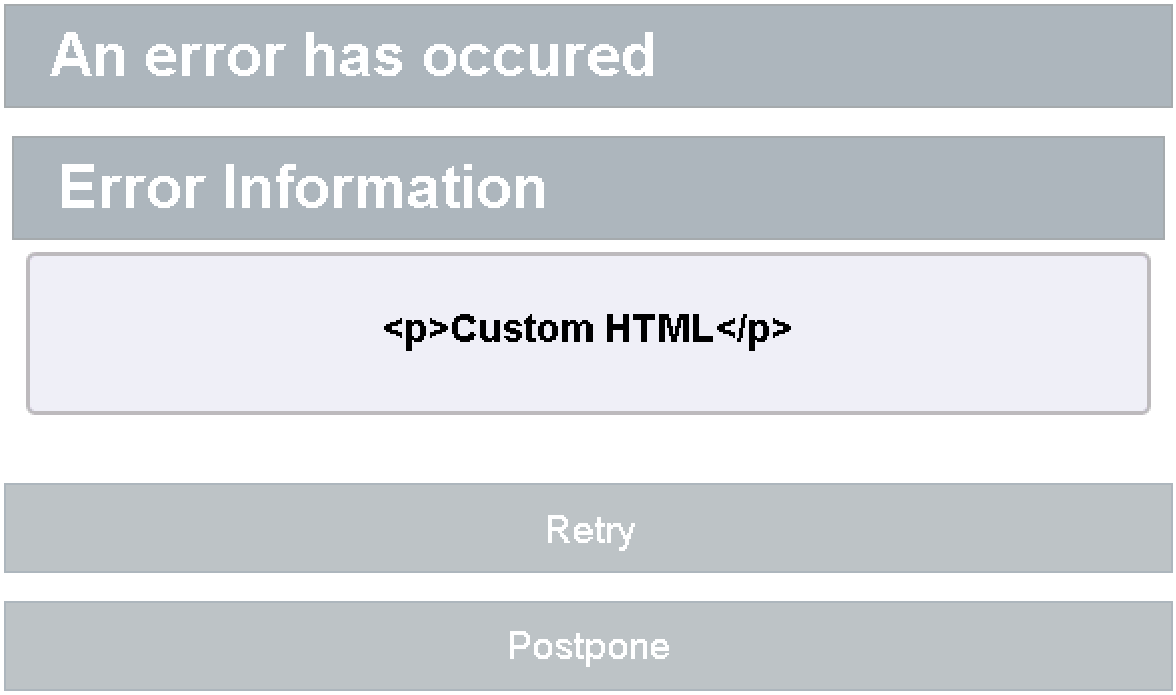


__ i. From the Stock palette, drag a **button** below the **OK** button.



__ j. Change the top button label to: `Retry`

__ k. Change the second button label to: `Postpone`



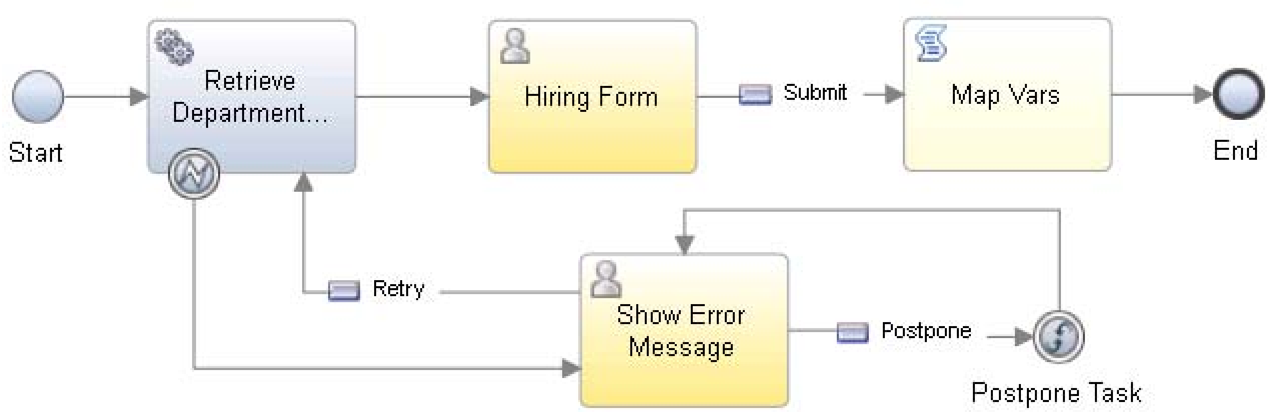
__ l. Save your work.

__ 5. Reconnect the flow on the **Hiring Form** canvas.

__ a. Click the **Diagram** tab.

__ b. Connect the new elements on the canvas:

- Connect the **Error Intermediate Event** event to the **Show Error Message** coach.
- Connect the **Show Error Message** coach to the **Postpone Task** event.
- Connect **Postpone Task** back to **Show Error Message**.
- Connect **Show Error Message** to **Retrieve Department Categories** by the **Retry** control.

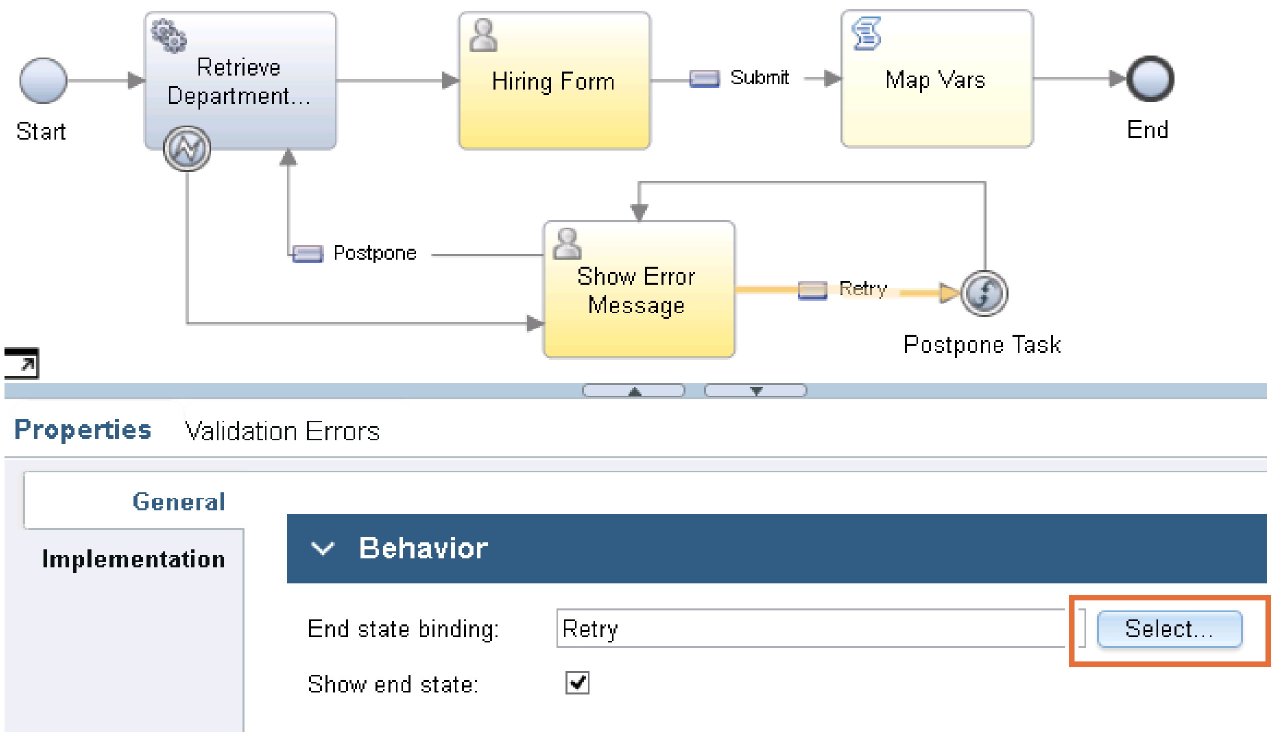


Hint

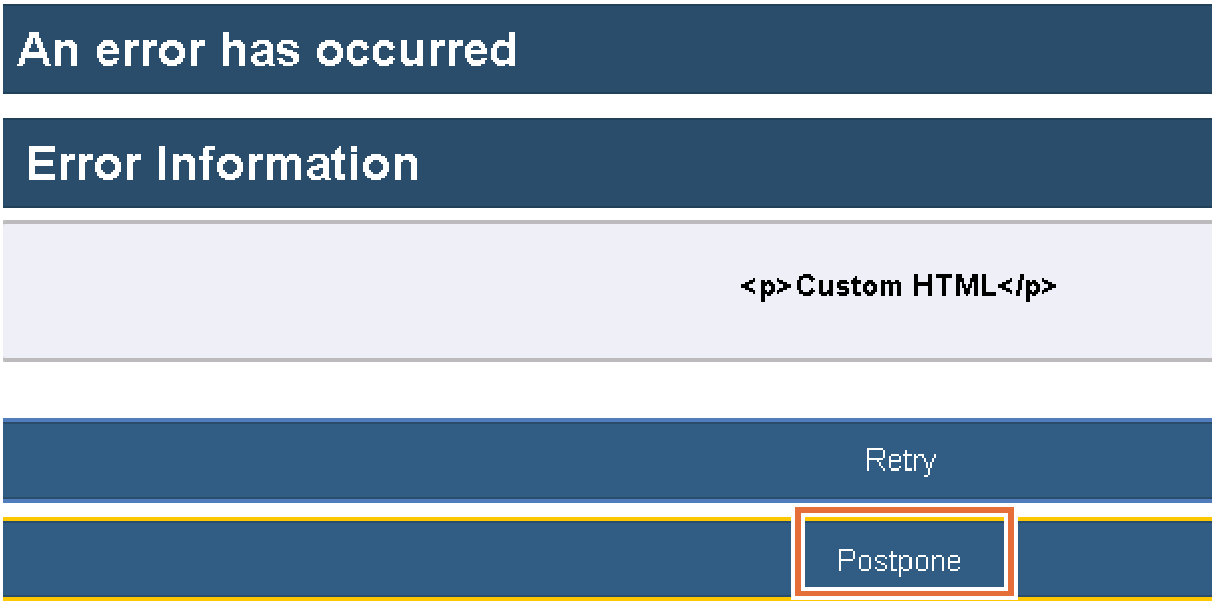
If the label on the flow is incorrect (for example, you wired the flows up to the wrong event), select a flow, and from the **Properties > General > Behavior** section, click **Select** next to **End state binding**. Select the correct button that corresponds with the flow you want.

After you connect the **Show Error Message** coach to other activities, the flow editor automatically uses the button controls from the coach. If the controls do not match the flow, you can either reroute the connection arrows or change the end state binding.

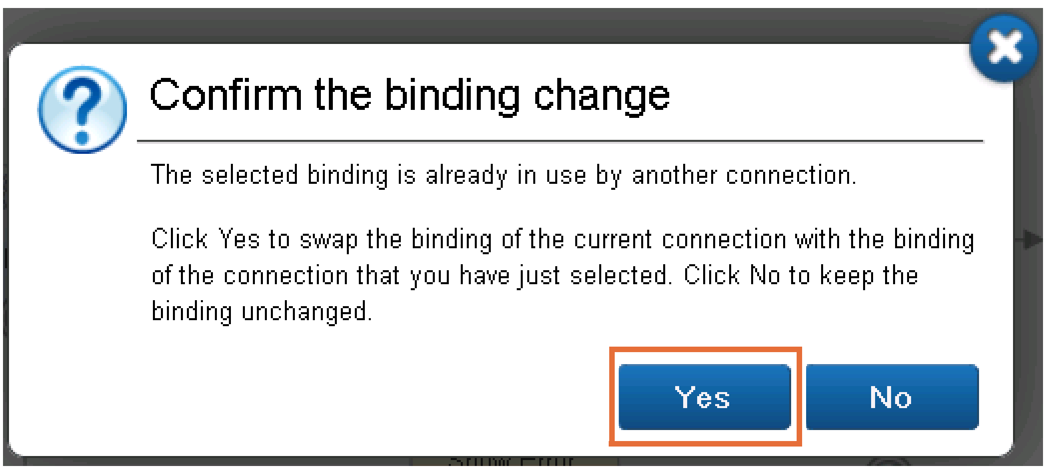
- To change the end state binding, select the flow.
- In the **Properties > General > Behavior** section, click **Select**.



- From the coach dialog box, select the button control to which you want to bind the flow.



- Click **Yes** to confirm the binding change.

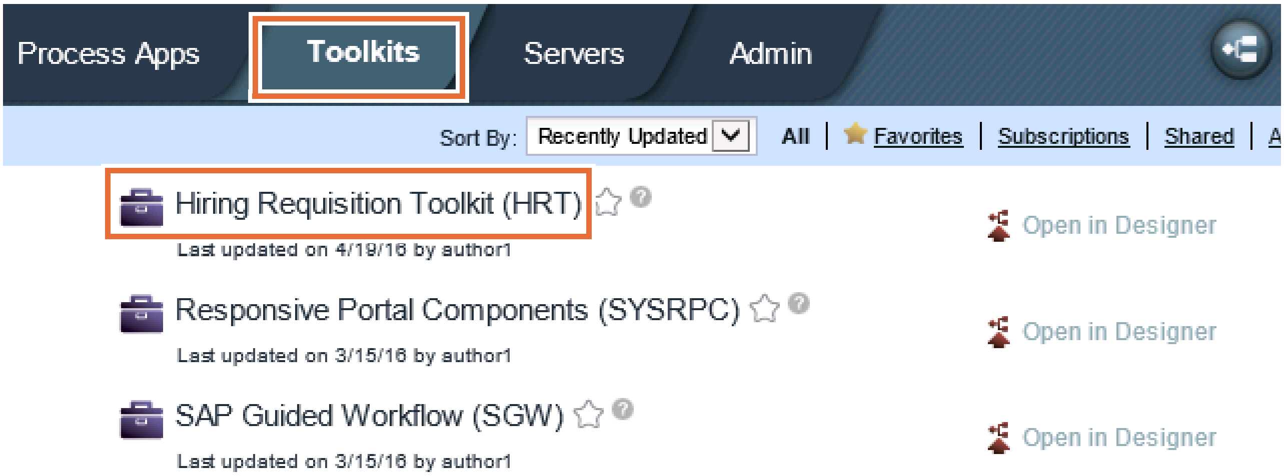


__ 6. Save your work.

__ 7. Create a snapshot of the new toolkit and Upgrade the dependency to the new snapshot.

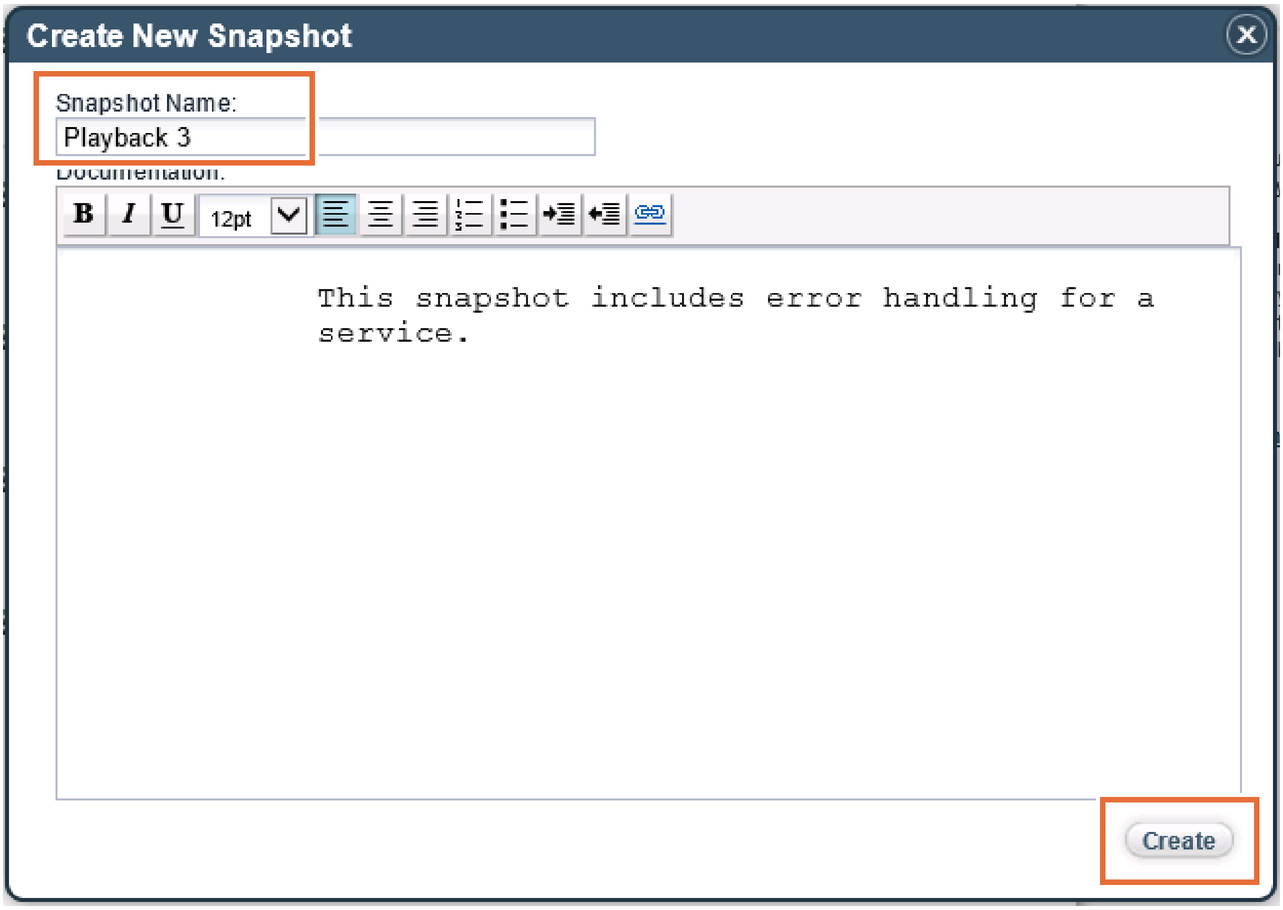
__ a. Return to Process Center.

__ b. Click the **Toolkits > Hiring Requisition Toolkit (HRT)** link.



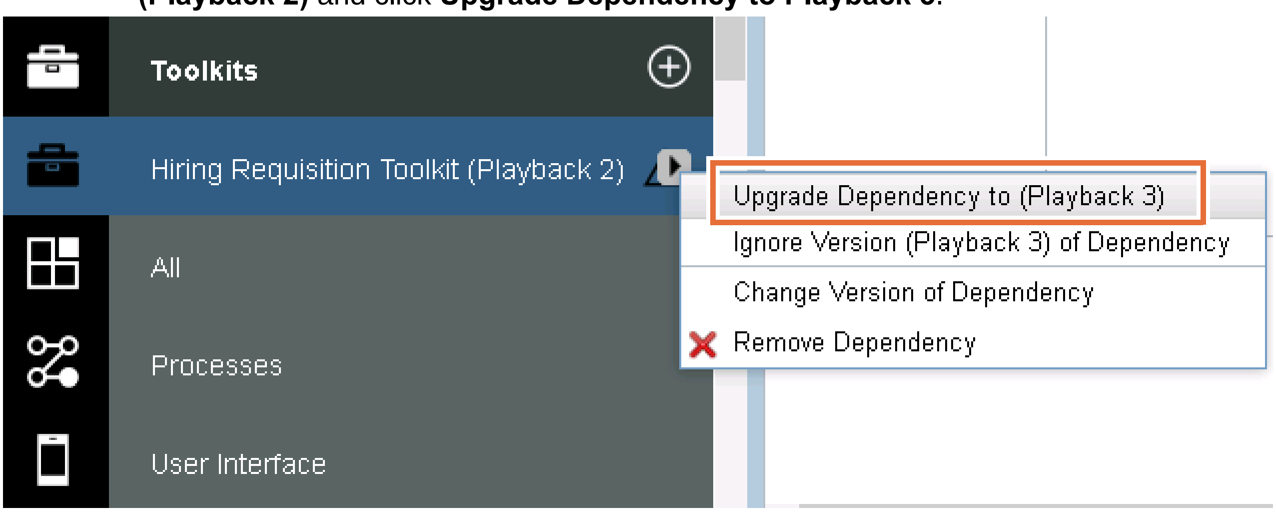
__ c. On the right, click **Create New Snapshot**.

__ d. Name the snapshot `Playback 3` and provide the following documentation: `This snapshot includes error handling for a service.` Click **Create**.



__ e. Open the **HR Recruitment Processes** process application.

__ f. Click **Toolkits** in the library. Click the arrow next to **Hiring Requisition Toolkit (Playback 2)** and click **Upgrade Dependency to Playback 3**.



__ g. Save your work.

Important

Playback 3: Creating error handling for a service is complete.

To test the error handling capability of the service, refer to Appendix A: Conducting playbacks, Part 6.

As part of the development process, you 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 Web Process Designer:

1. Inside of IBM Web Process Designer, click **Snapshot**.
2. Give the snapshot a descriptive name.

Requirements

Successful completion of the previous exercise is required.

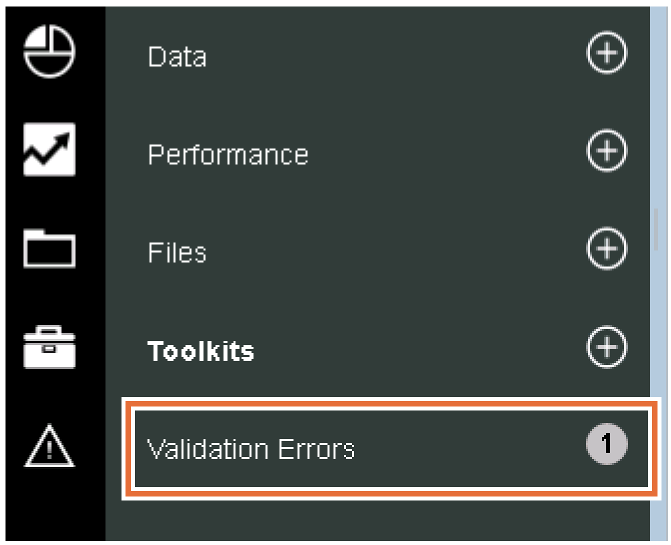
Exercise instructions

Part 1: Prepare for final snapshot

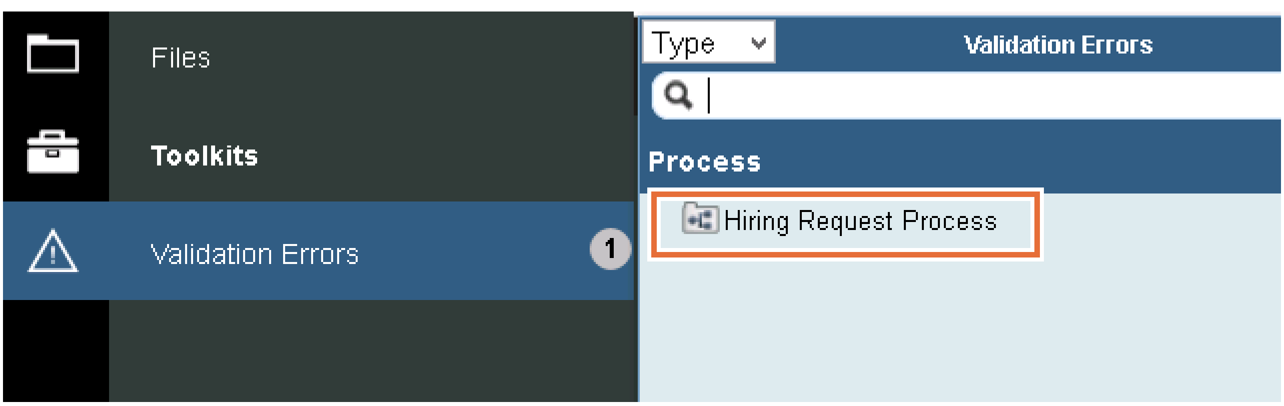
__ 1. Check for any validation errors in the process application.

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.

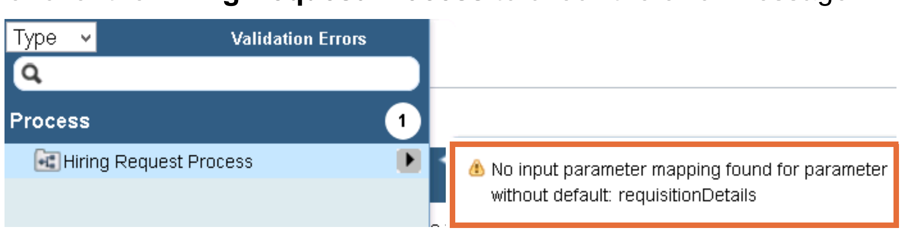
__ a. In the IBM Web Process Designer library, check for any errors in the **Validation Errors** category.



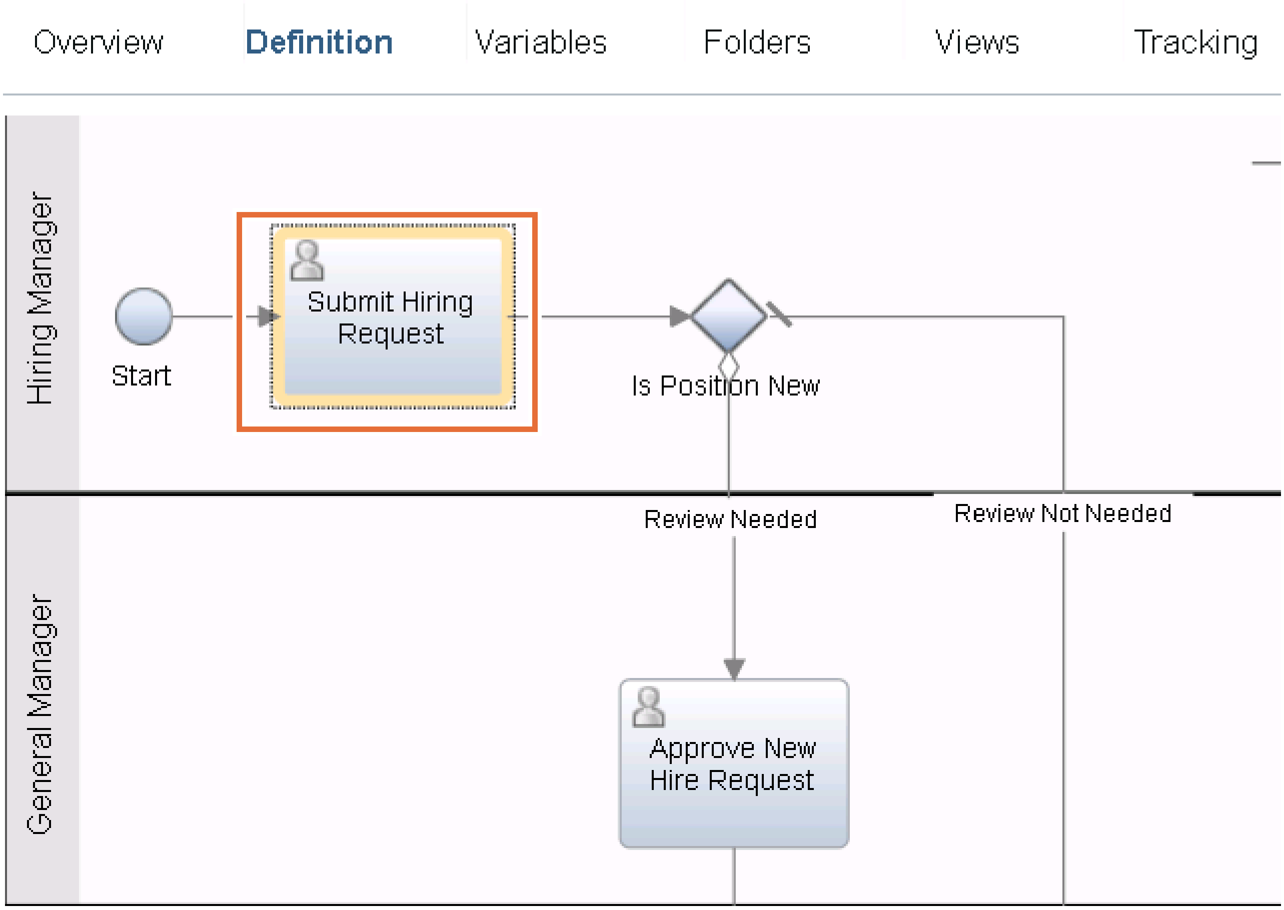
__ b. Click the **Validation Errors** category to check the name of the process that contains the error.



__ c. Hover over the **Hiring Request Process** to check the error message.



__ d. To resolve the parameter mapping errors, open the **Hiring Request Process**. Click the **Submit Hiring Request** activity.

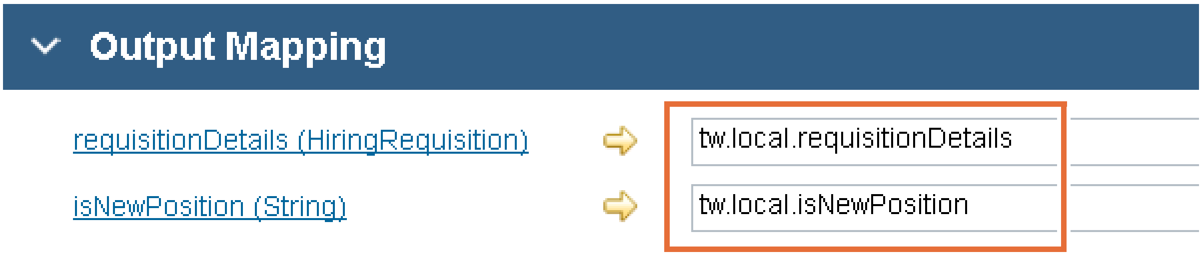


__ e. Open the **Properties > Data Mapping** menu.

__ f. In the **Input Mapping** section, map the input to the `requisitionDetails (HiringRequisition)` variable.

__ g. In the **Output Mapping** section, map the first output to the `requisitionDetails (HiringRequisition)` variable.

__ h. In the **Output Mapping** section, map the second output to the `isNewPosition (String)` variable.



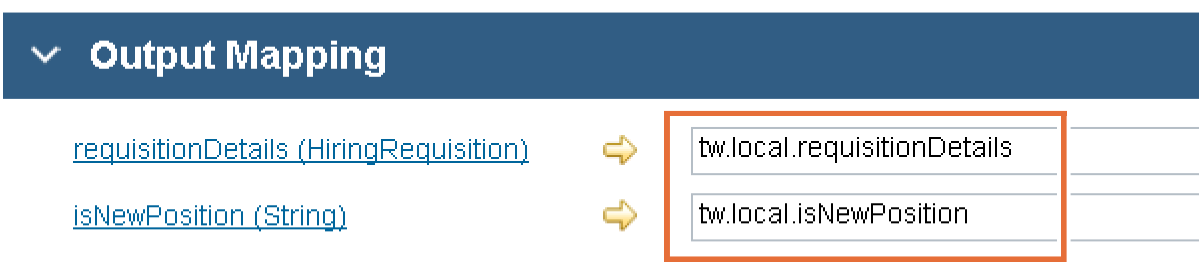
__ i. Select the **Approve New Hire Request** activity.

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

__ k. In the **Input Mapping** section, map the input to the `requisitionDetails (HiringRequisition)` variable.

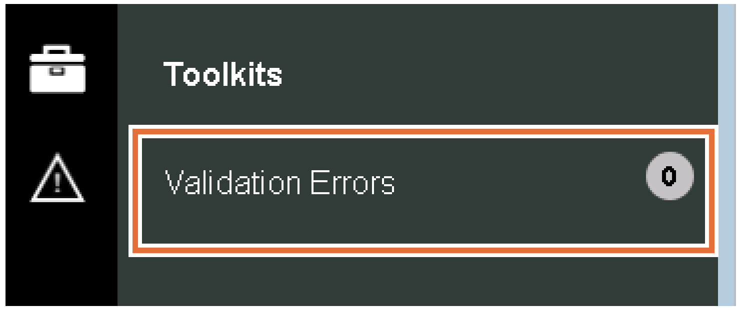
__ l. In the **Output Mapping** section, map the first output to the `requisitionDetails (HiringRequisition)` variable.

__ m. In the **Output Mapping** section, map the second output to the `isNewPosition (String)` variable.



__ n. Save your changes.

__ o. Verify that no more errors are in the **Validation Errors** category in the IBM Web Process Designer library.

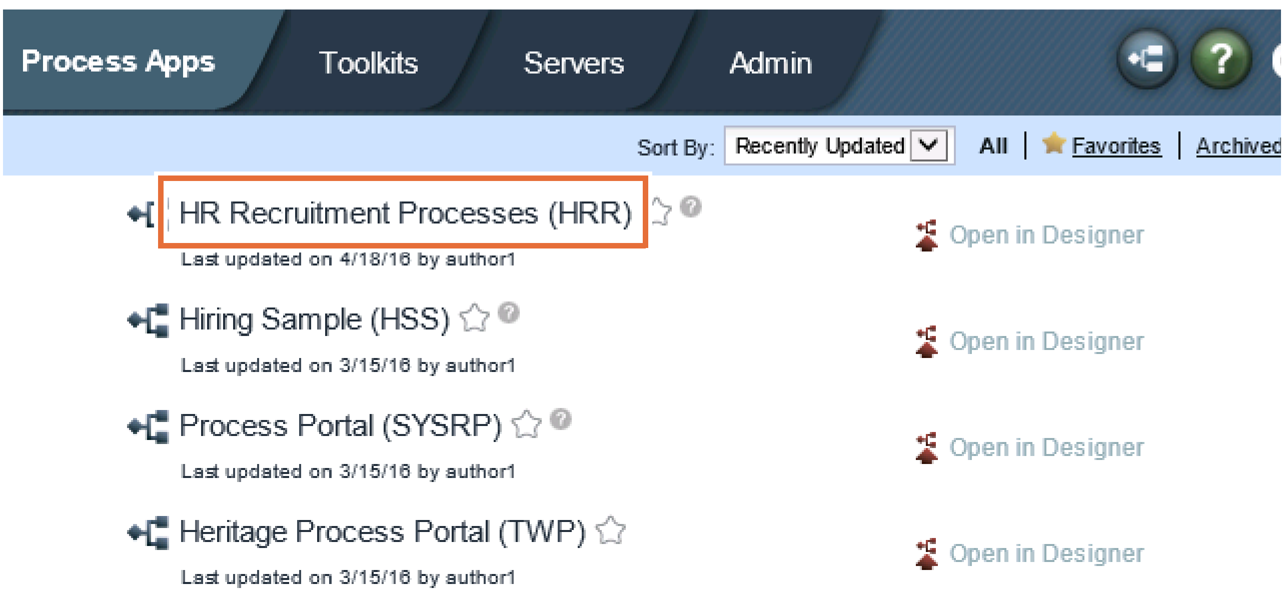


Part 2: Check the history of the process application

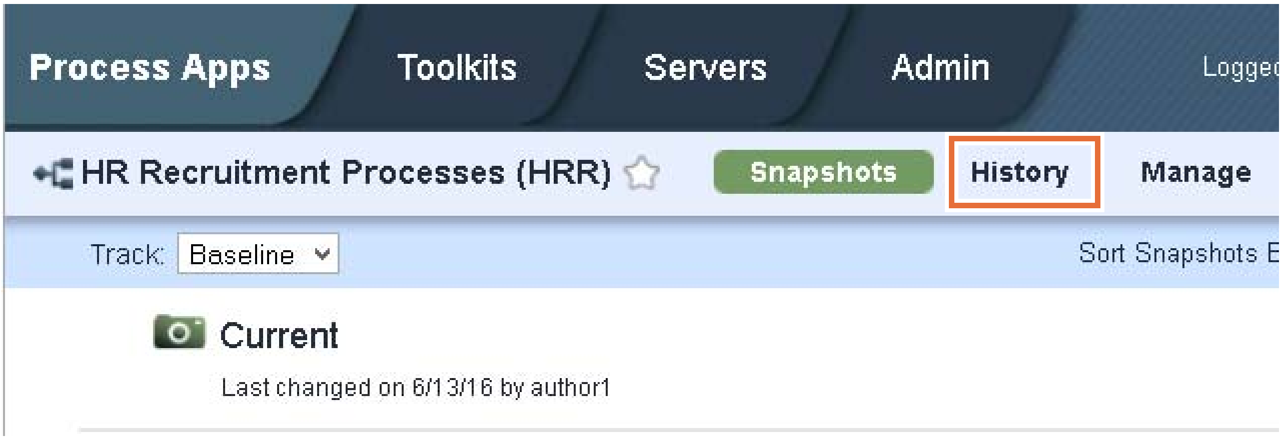
__ 1. Before taking the final snapshot of the process application for deployment, you can check the history of the actions that you performed on the process application.

__ a. Return to the Process Center.

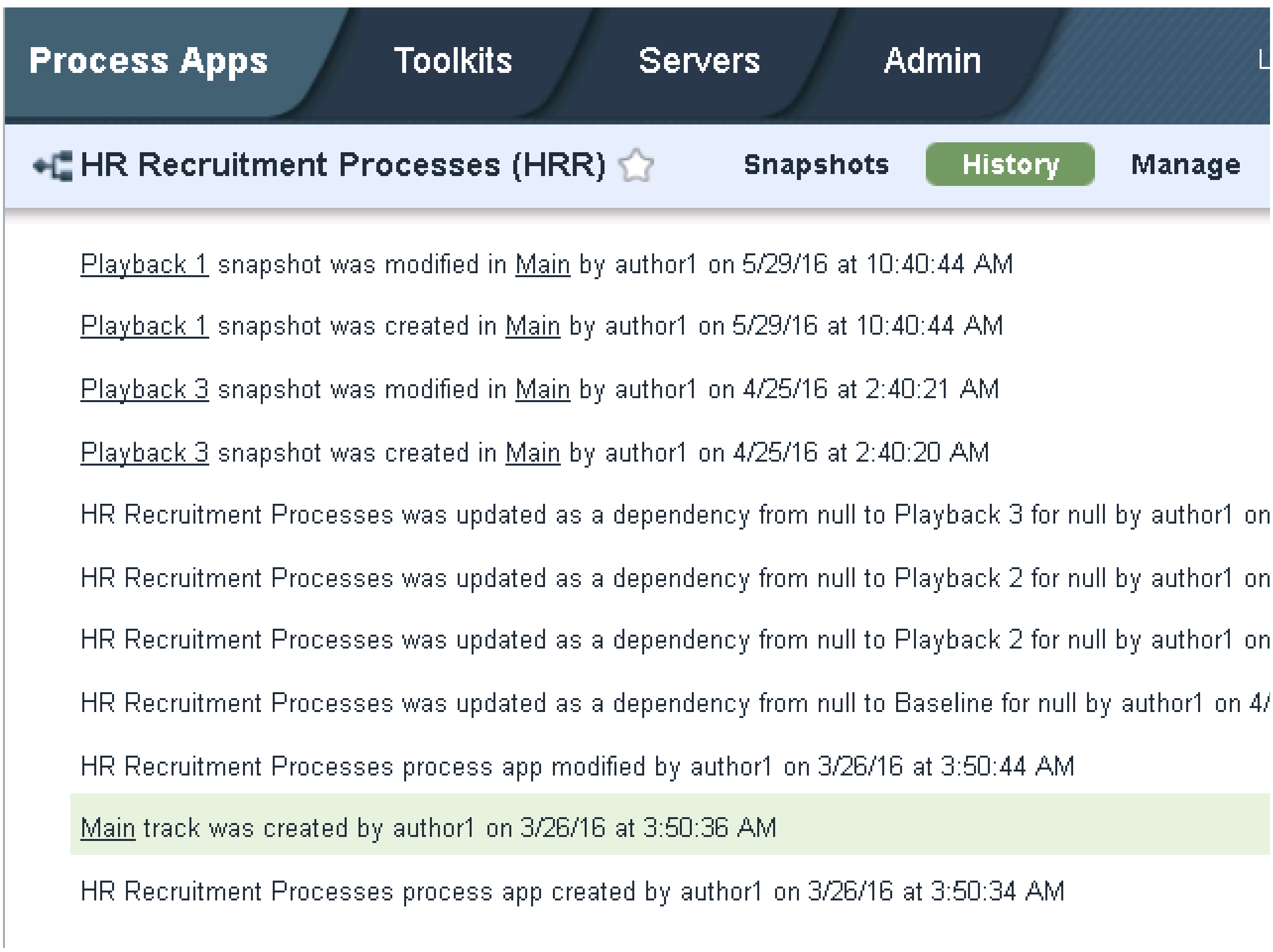
__ b. Click the **HR Recruitment Processes** process application.



__ c. Click **History**.



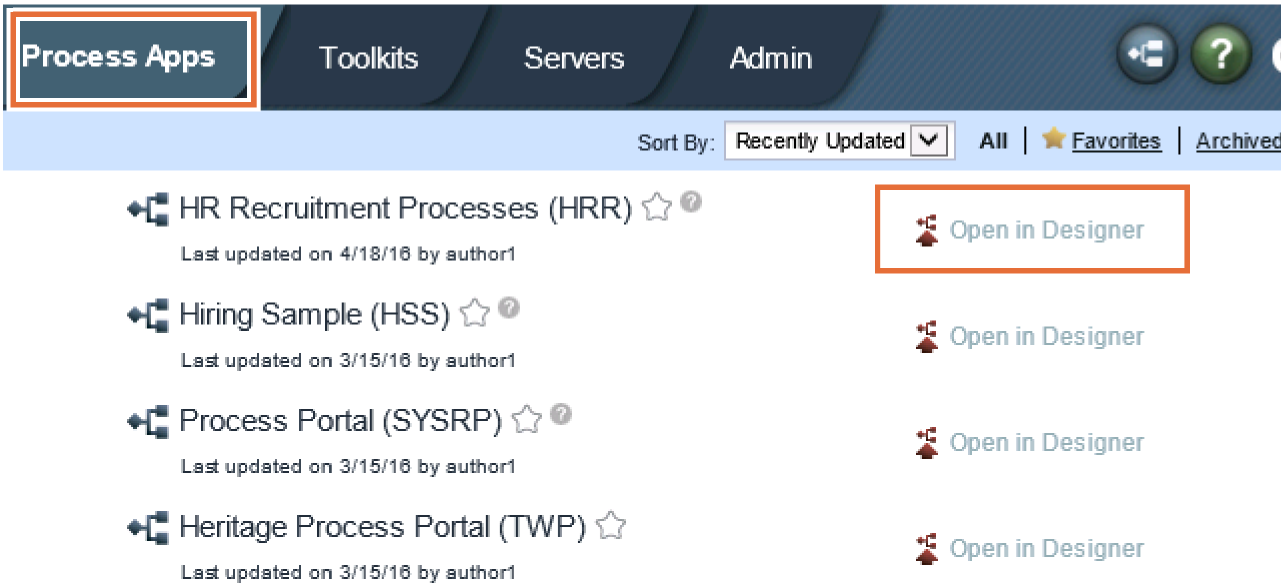
__ d. Verify the set of actions that you performed on the process application from the beginning.



Part 3: Create a snapshot of the process application

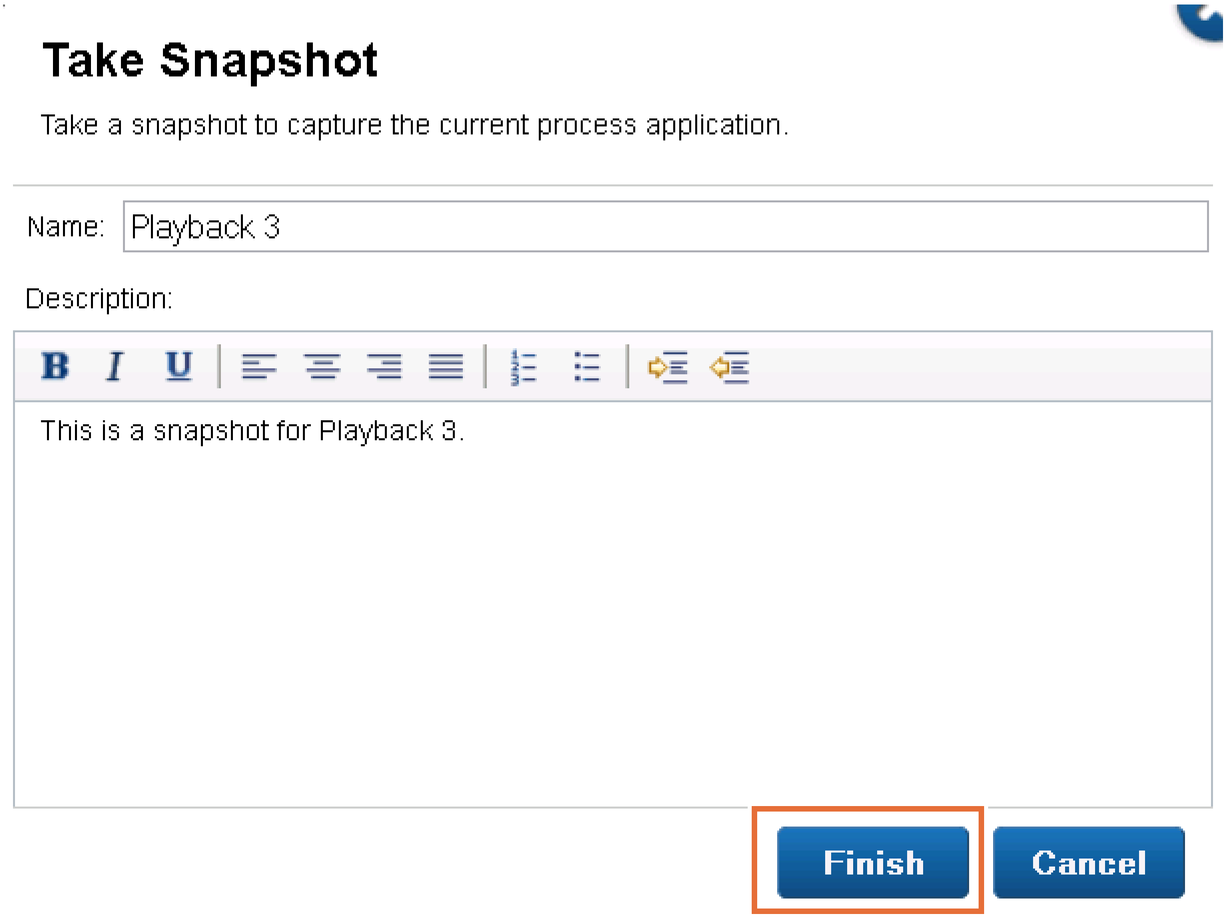
__ 1. You created snapshots throughout development from the previous exercises. This exercise is for the final deployment to the next environment. Create a snapshot of the **Hiring Request Process** in IBM Web Process Designer.

__ a. Click the **Process Apps** tab. Open the **HR Recruitment Processes** process application in the Designer.



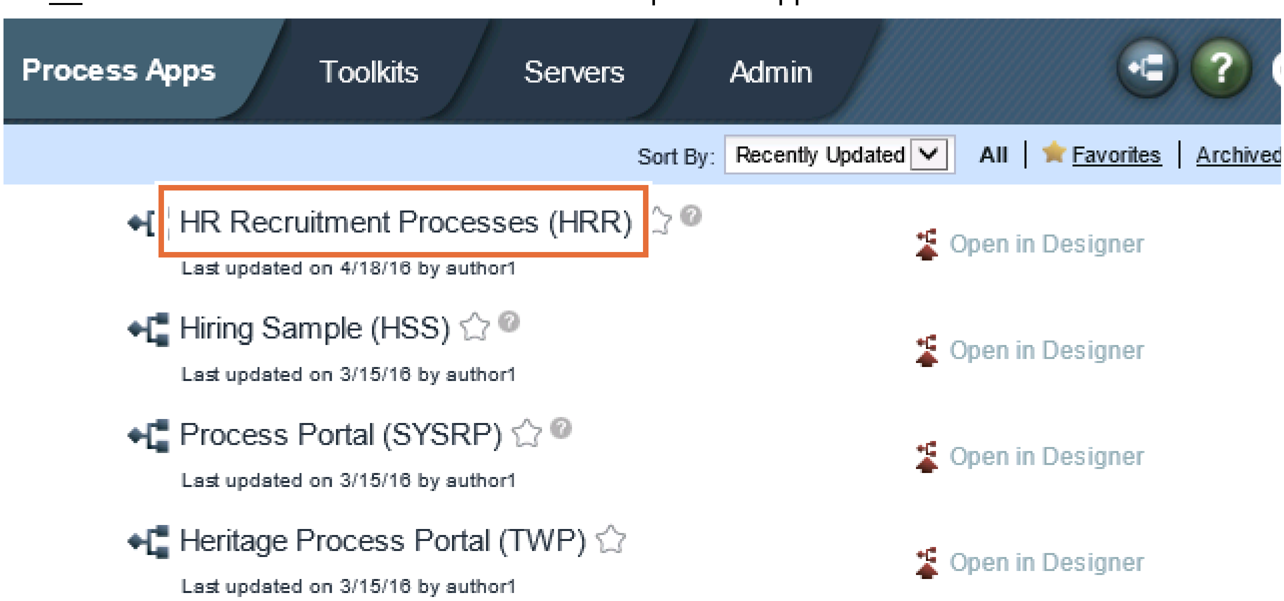
__ b. Click the **Snapshot** icon on the top of the web Process Designer window.

__ c. Give your snapshot the name `Playback 3` and description: `This is a snapshot for Playback 3.` Click **Finish**.

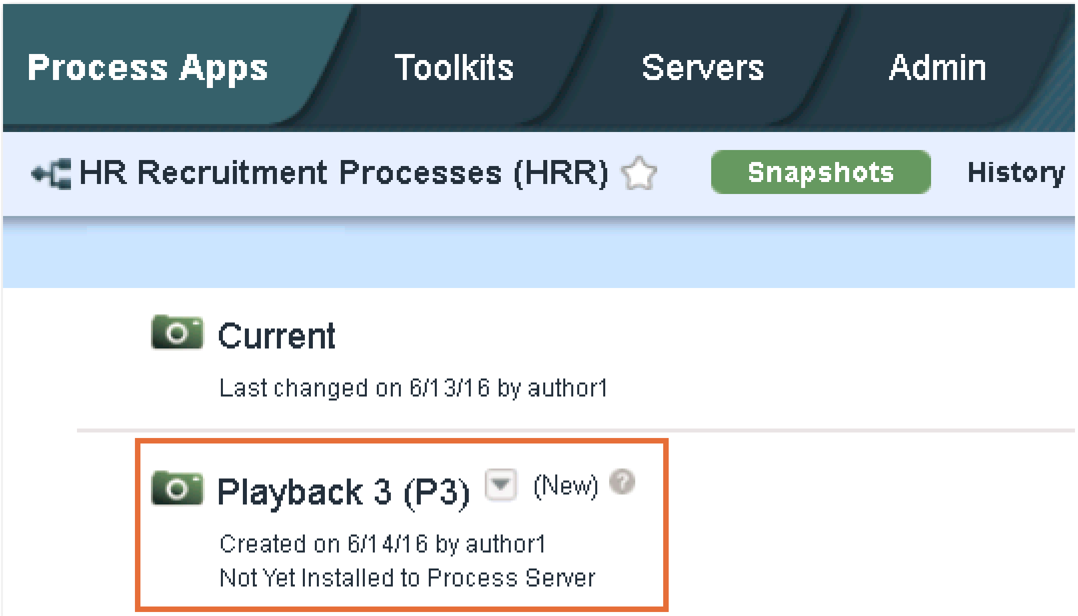


__ d. Return to the **Process Center**.

__ e. Click the **HR Recruitment Processes** process application.



__ f. Your snapshot is displayed in the list of snapshots.



Information

Your process application is ready to deploy to the test environment. You can use the IBM Web Process Designer to deploy the snapshot if your account is authorized to deploy this process application. You can also use the Process Designer client application or the Process Admin console to do the deployment.

End of exercise

Exercise review and wrap-up

In this exercise, you got ready for the final snapshot by looking at the validation errors if any were present in the process application. Next, you looked at the history of the actions that were performed on the process application and creating 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.

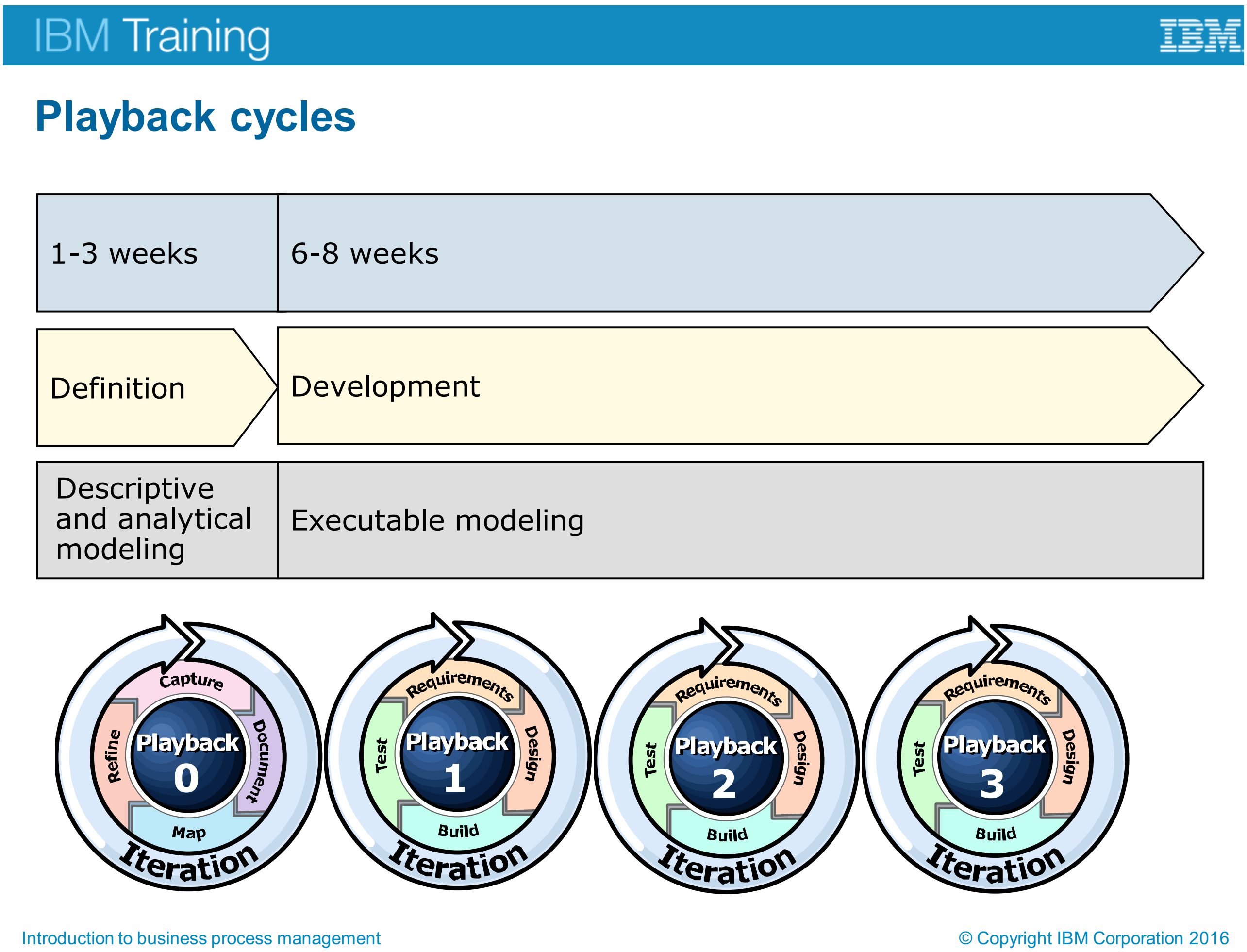
This exercise relies upon the `HR Recruitment Processes` process application.

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.

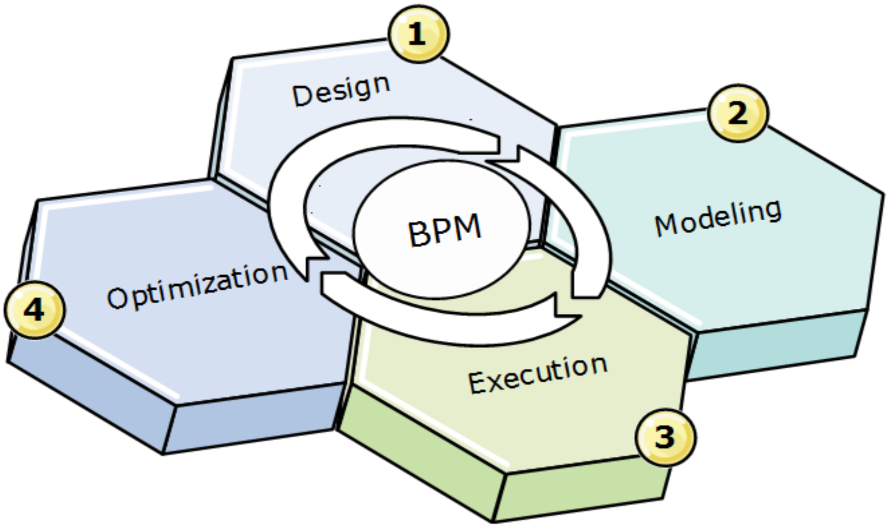
Themed playbacks

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.



Project management and the BPM lifecycle

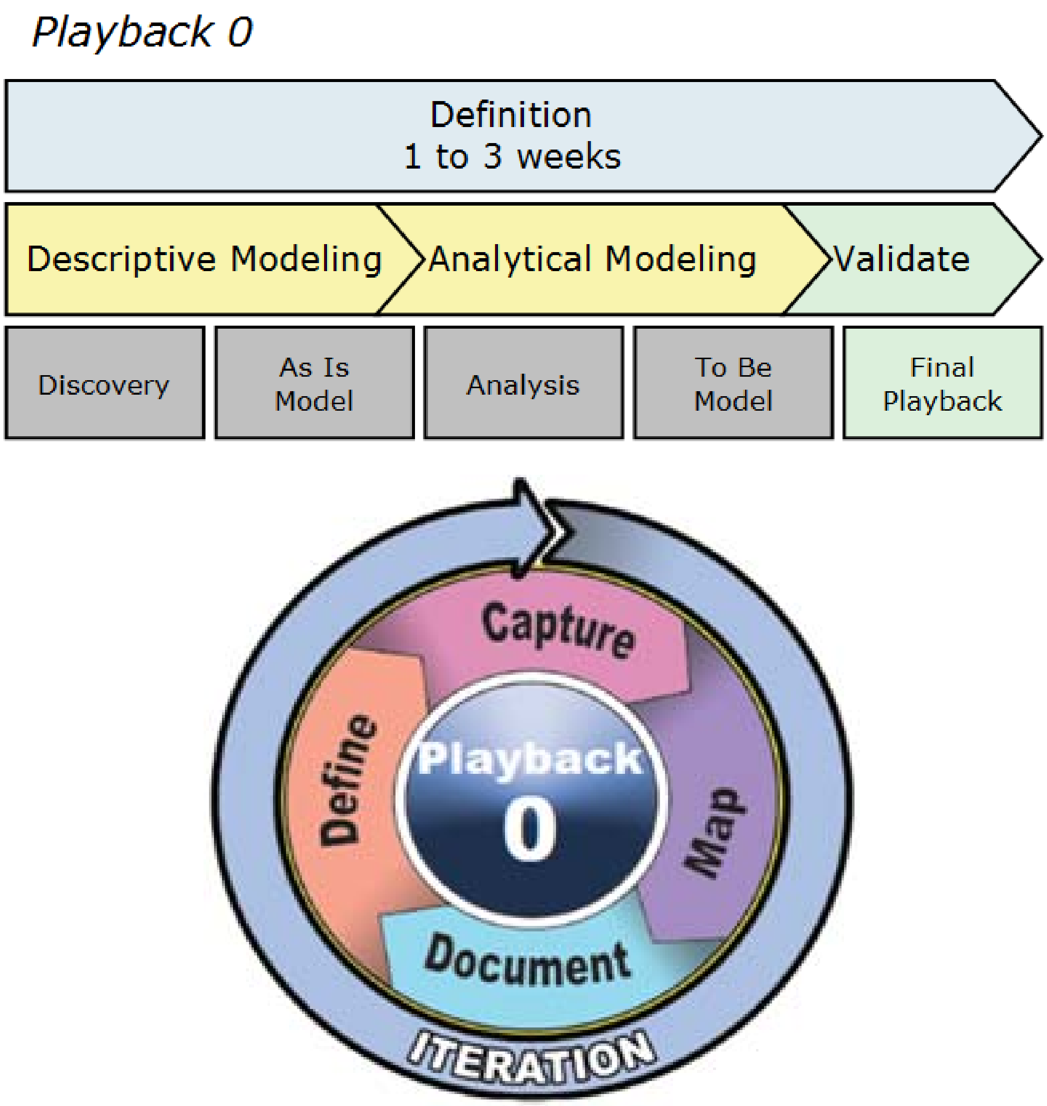
Completion of the project management means that a large portion of the BPM lifecycle is complete. After the project development is complete, which is 9 – 14 weeks, the process application is monitored during production. The process performance data is then evaluated and analyzed for more efficiency. 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

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.

Part 3: Conduct Playback 1: Business data and services

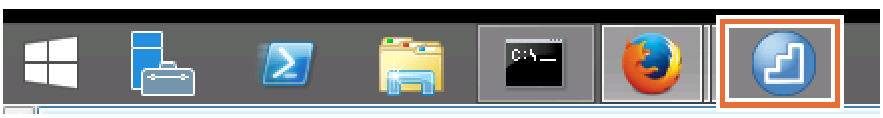
In Playback 1: Business data and services, human services are added with the Service Modeler. Human services allow users to access task assignments in an activity-centered web form. At this stage of development, it is important that the functions of a human service, which is known as a “coach,” are implemented. Enhancements can be added later. This playback is about making sure that users have what they need in terms of business data and task assignment information to complete the process activity.

This playback is also where developers provide stakeholders with the ability to manually control process flow. You do not introduce automation and computation until Playback 2: Integration, so that the users can do extra testing with temporary process flow control inputs to ensure that the flow is correct based on the user requirements. After Playback 2: Integration, deploy a snapshot to a test environment for the initial testing phase.

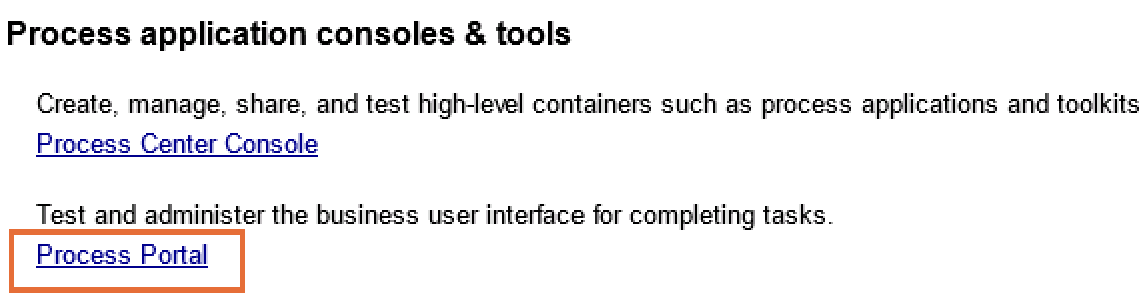
The default human services that were used in Playback 1 are replaced with human services that allow for entry and capture of business data. The human services with coaches are implemented on two activities in the **Hiring Request Process** process. This session validates the Playback 0 process application.

__ 1. Log on to the Process Portal.

__ a. Maximize the **IBM Business Process Manager Quick Start** browser window from the Windows taskbar.



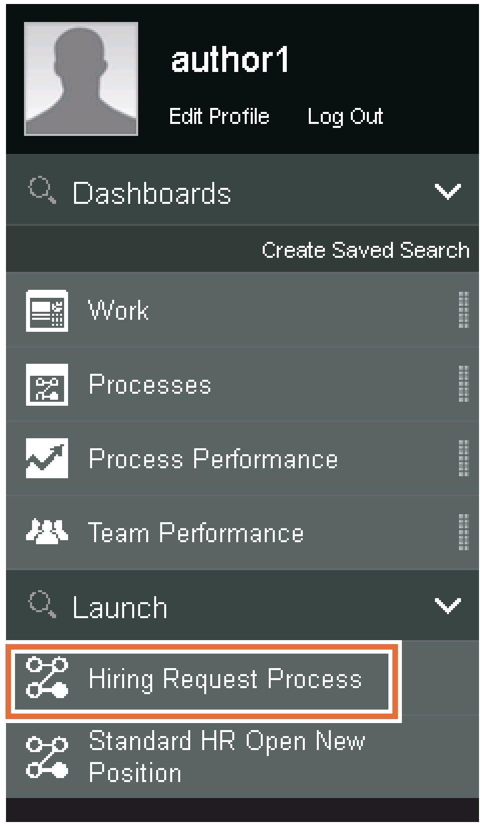
__ b. Click the **Process Portal** link.



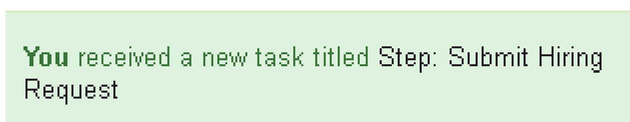
__ c. If prompted with the Login screen, login with `author1` as the **Username** and `author01` as the **Password**.

__ 2. Create an instance of the **Hiring Request Process** in the Process Portal.

__ a. In the left frame, click **Hiring Request Process** to start an instance of the process.



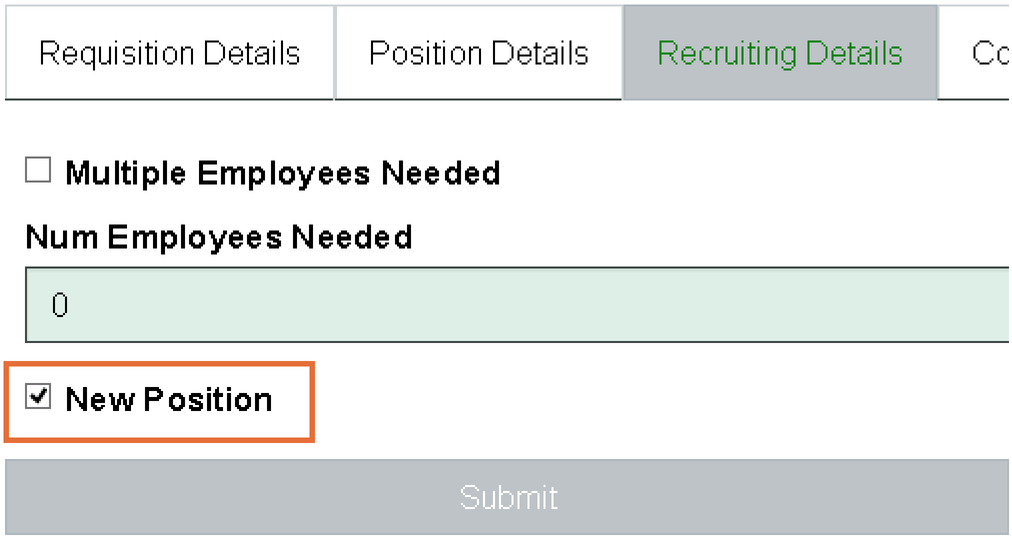
__ b. A notice is displayed to indicate that the process was started.



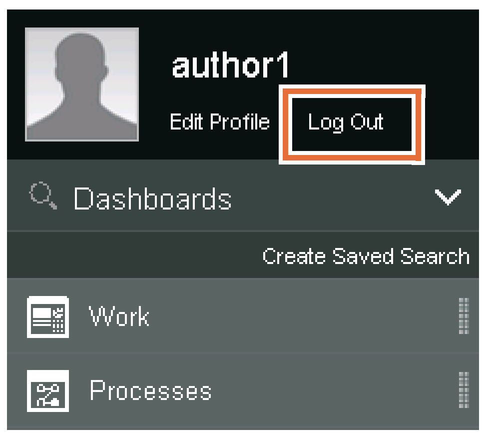
__ 3. A window is displayed with the human service coach.

__ 4. Complete the coach with any data.

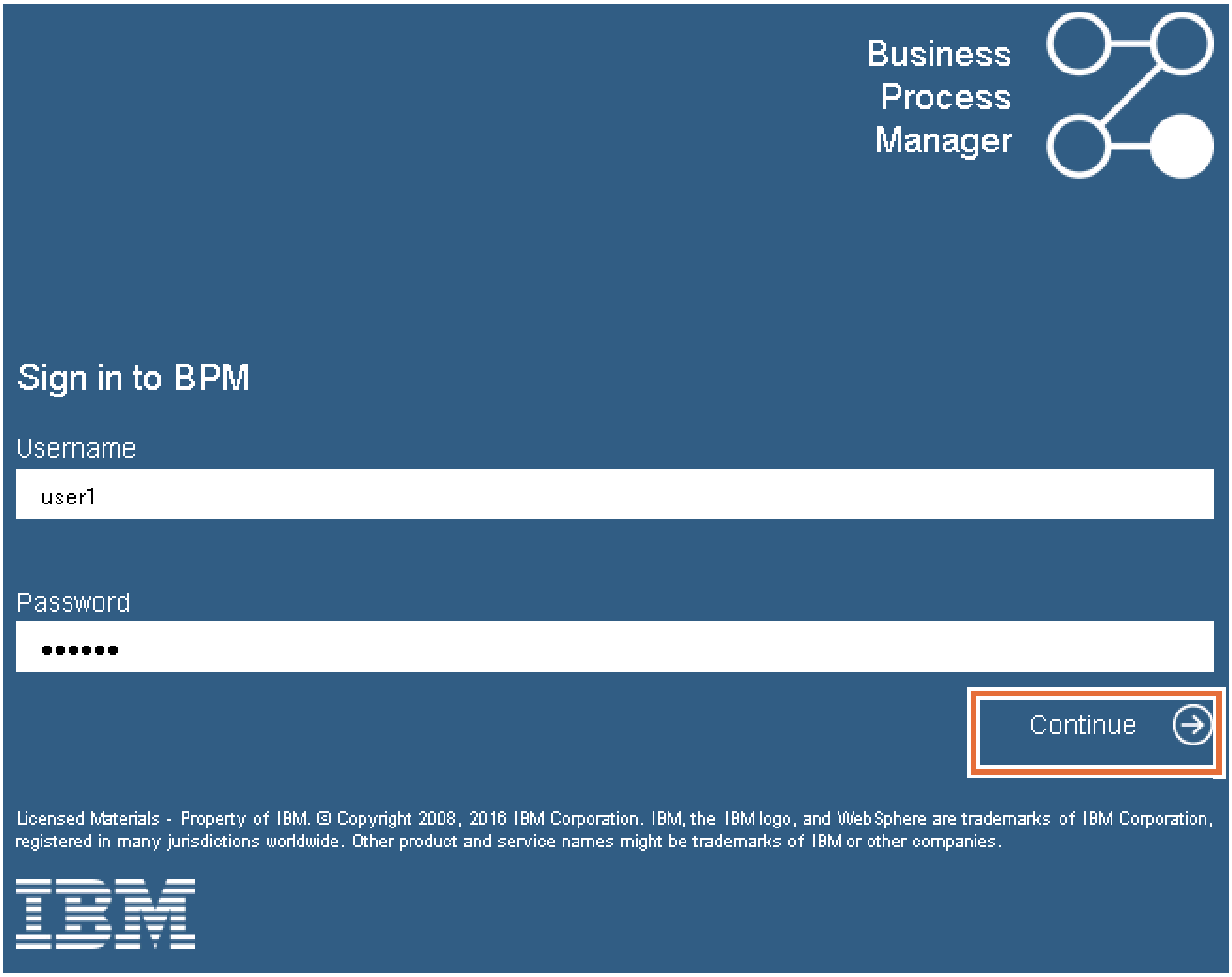
__ 5. Enable the **New Position** selection under **Recruiting Details** tab and click **Submit**.



__ 6. Click **Log Out** to log out of Process Portal.

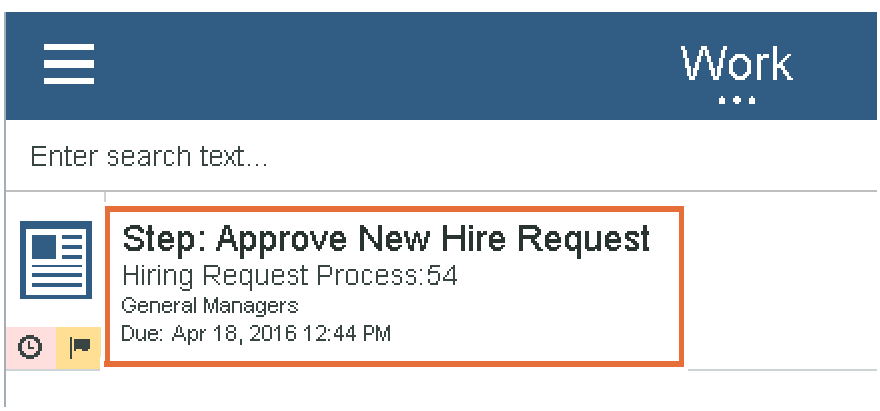


__ 7. Log back in with `user1` as the **Username** and `user01` as the **Password**. Click **Continue**.



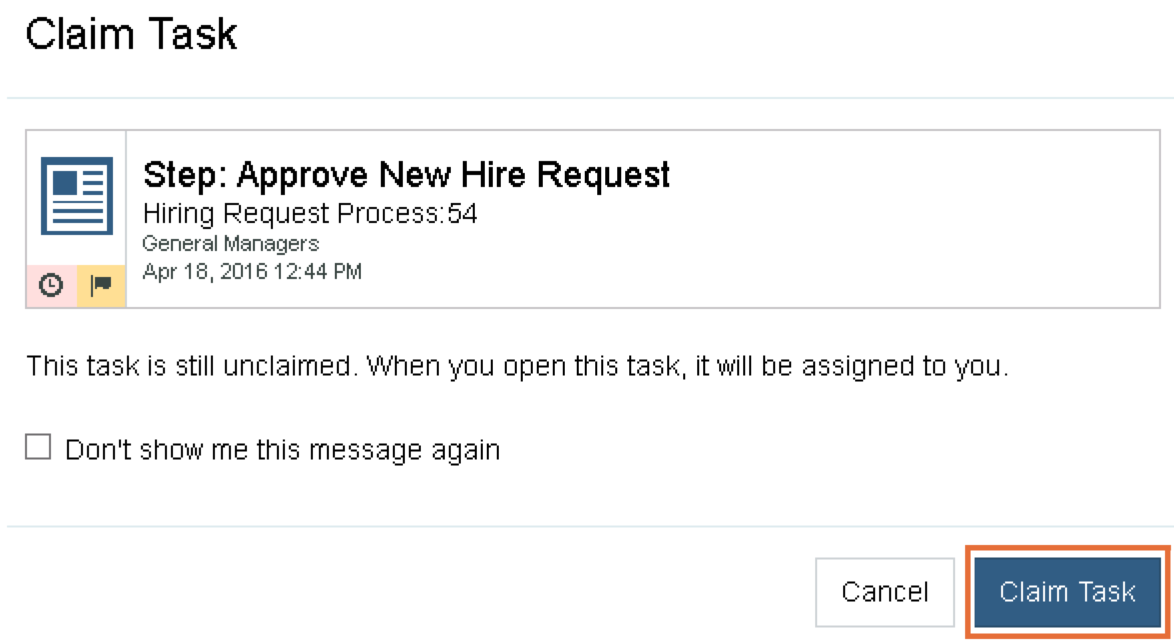
__ 8. Run the **Approve New Hire Request** task.

__ a. 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.



__ b. Click the **Step: Approve New Hire Request** task.

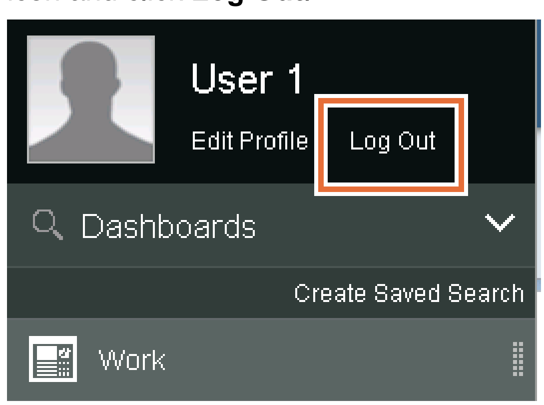
__ c. Click **Claim Task**.



__ d. A human service coach is displayed for the **Approve New Hire Request** activity. The data from the last coach is displayed in the new coach.

__ e. Complete this task by clicking **Submit** in the human task service.

__ f. Click the **Library** icon and click **Log Out**.



Part 4: 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. Not all of 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 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.

__ 1. Log on to the Process Portal.

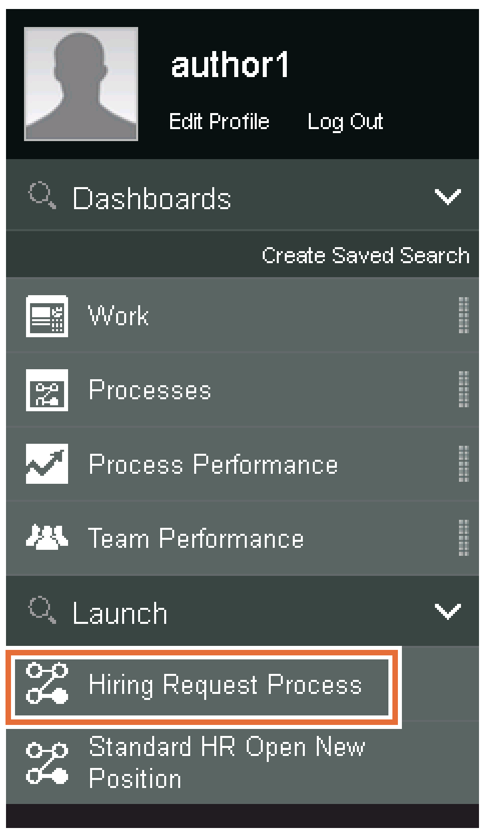
__ a. Maximize the **IBM Business Process Manager Quick Start** browser window.

__ b. Click the **Process Portal** link to start the Process Portal.

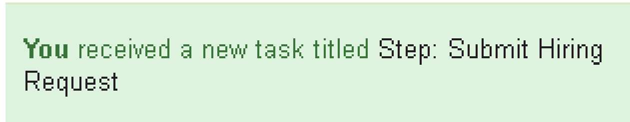
__ c. Log in with `author1` as the **Username** and `author01` as the **Password**.

__ 2. Create an instance of the **Hiring Request Process** in the Process Portal.

__ a. In the left frame, click **Hiring Request Process** to start an instance of the process.

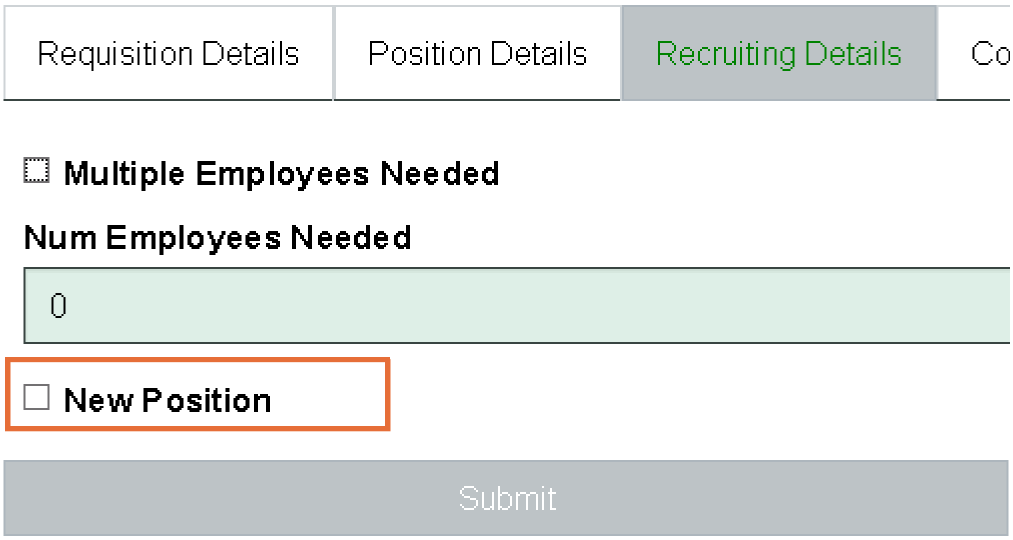


__ b. A notice is displayed to indicate that the process was started.



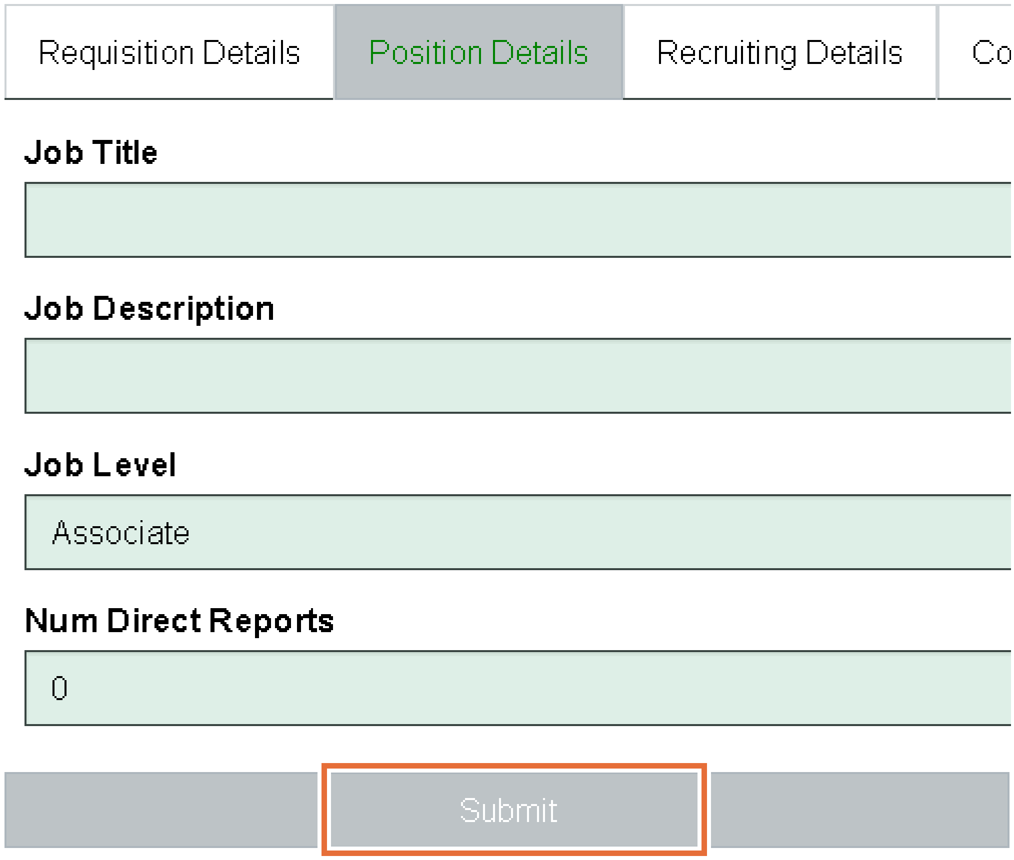
__ 3. Complete the coach.

__ a. Leave the **New Position** check box as false (not selected).



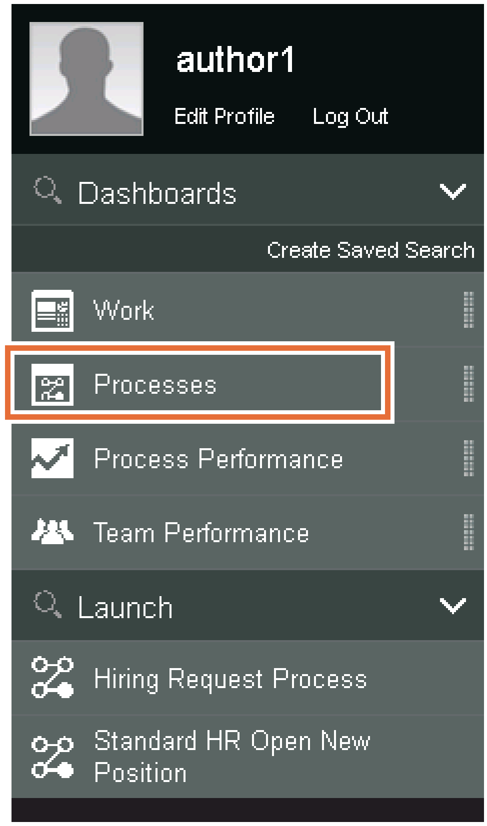
__ b. In the Compensation Details section, in the **Salary To Offer** field, type: `45000` (a number in the range `40000 – 60000`). In the Position Details section, in the **Job Level** field, type: `Associate`

__ c. Complete the remainder of the coach with any data you want. Click **Submit** when done.



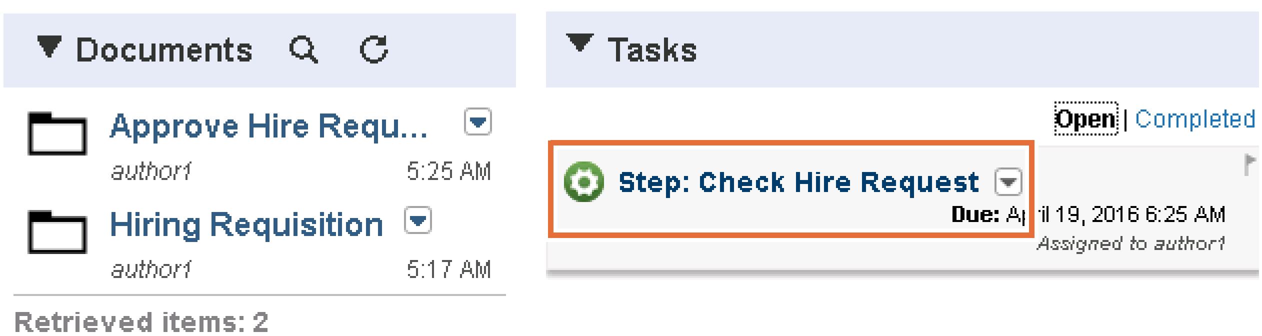
__ d. Click the **library** icon on the upper-left corner to open the Library.

__ e. Click **Processes**.

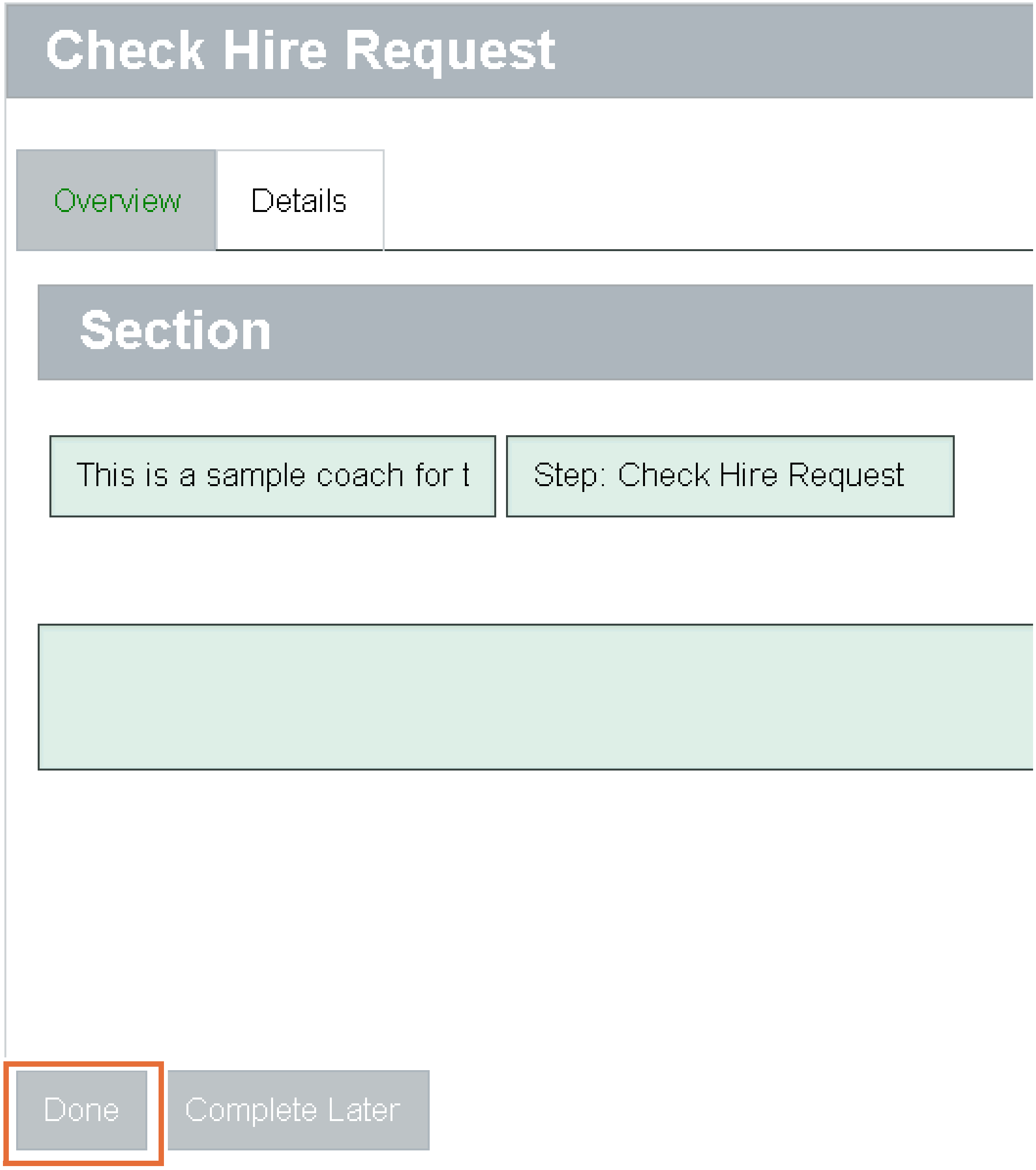


__ f. Click **Hiring Request Process** with the latest instance ID.

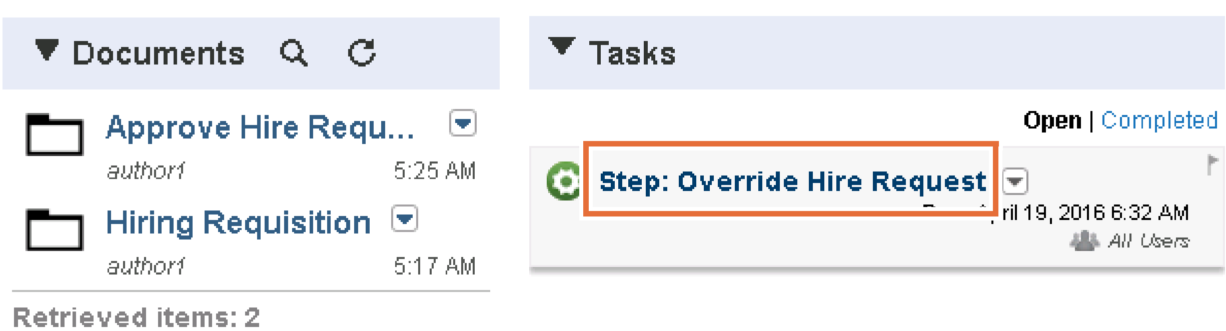
__ g. Click the **Step: Check Hire Request** task link.



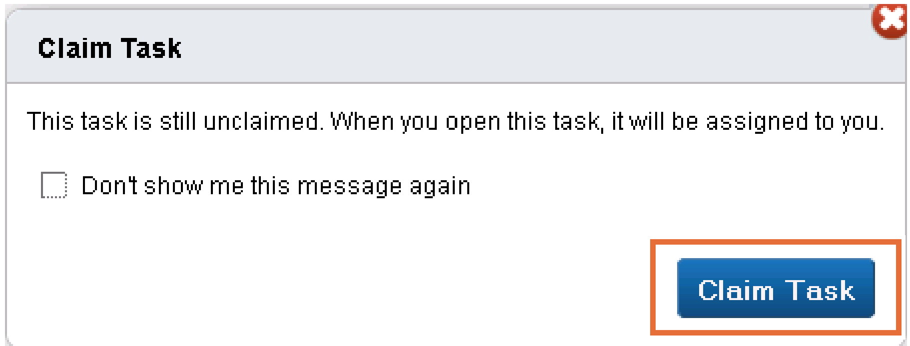
__ h. Click **Done** to complete the task.



__ i. Now the process moves to the next task. Click **Step: Override Hire Request**.



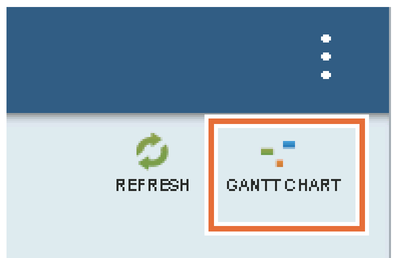
__ j. Click **Claim Task**.



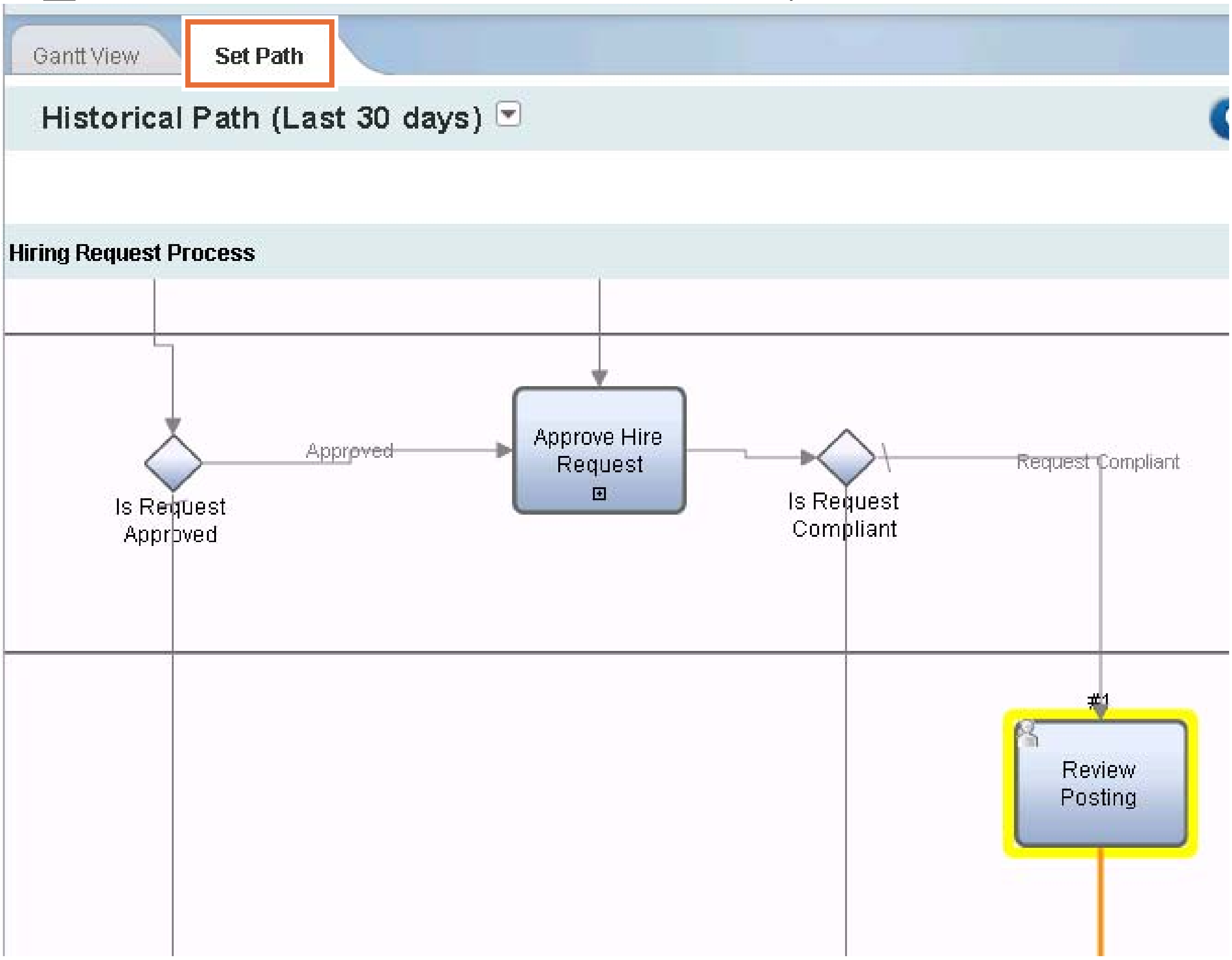
__ k. Click **Done** to complete the task and the instance.

__ l. The process is now moved to **Step: Review Posting**.

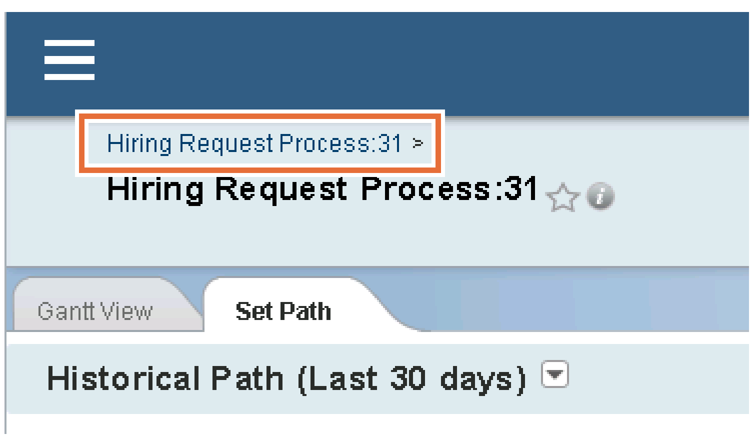
__ m. Click the **GANTT CHART** icon on the upper-right corner of the window.



__ n. Click the **Set Path** tab for a model view of the same process instance.



__ o. Click the **Hiring Request Process** in the breadcrumb to close the diagram view and return to the previous screen.



__ p. Complete the task for the next activity and follow the process flow.

__ 4. Create another instance that follows a different path in the process.

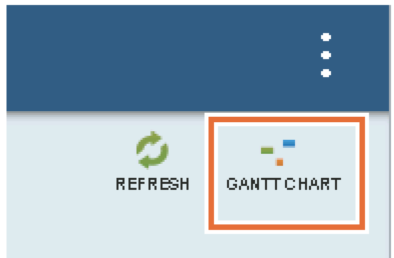
__ a. In the library, select the **Hiring Request Process** to start a new instance of the process.

__ b. Complete the coach with any data you want. This time, change `isNewPosition` to true (selected). Set the job level to `Director`, and the salary to offer under the **Compensation Details** must be set to `110000` (a number not within the `70000` – `95000` range). Click **Submit** when done.

__ 5. Show the different path that is taken based on how the rule is assessed as a complex claim.

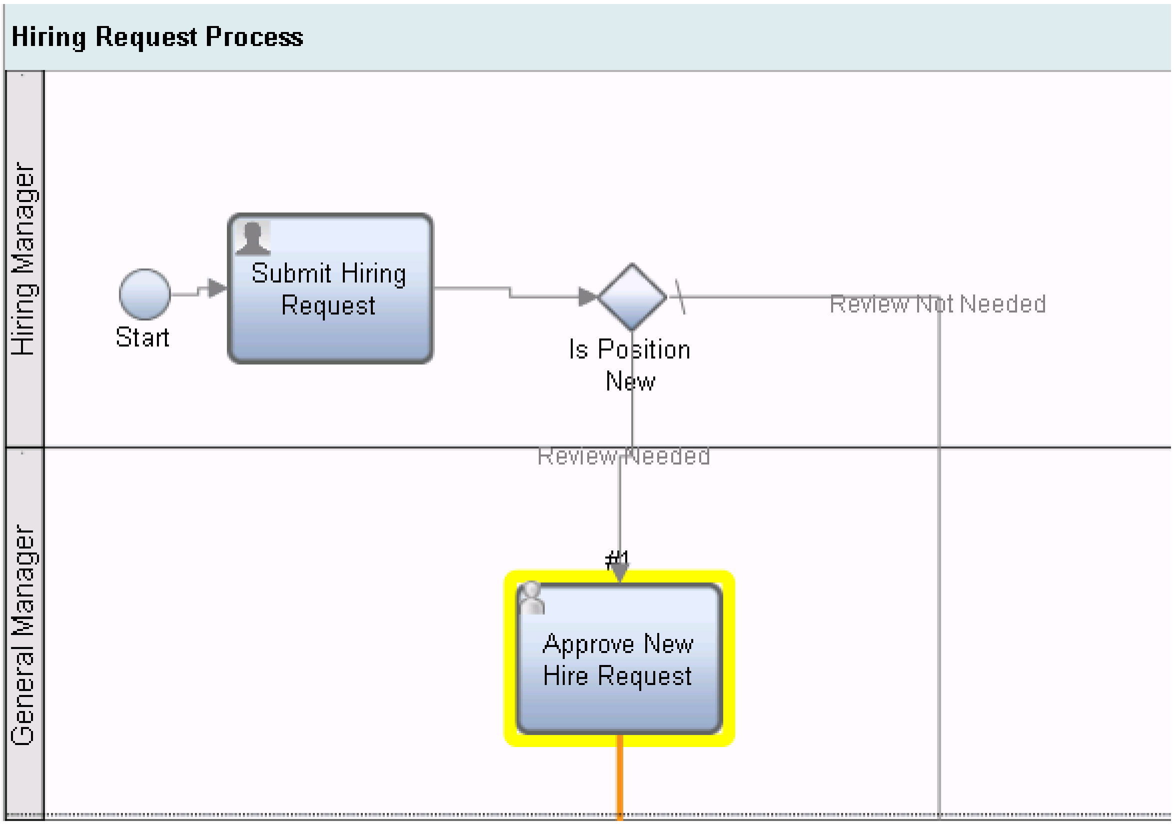
__ a. Click **Processes** and click the most recent instance.

__ b. Click the **GANTT CHART** icon in the upper-right corner to view the Gantt chart view.

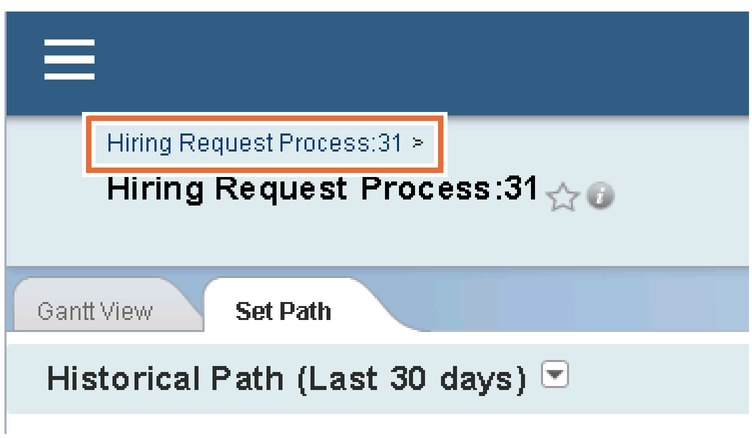


__ c. Click the **Set Path** tab for a model view of the same process instance.

__ d. The model view of the process instance is displayed. Notice the different path that was taken, and how the flow went to the **Approve New Hire Request** because you set the new position to true.



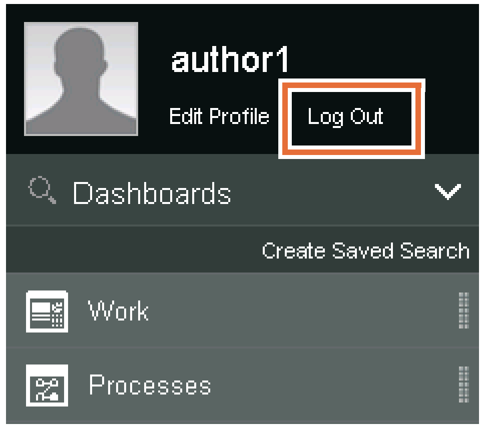
__ e. Click the **Hiring Request Process** in the breadcrumb to close the diagram view and return to the previous screen.



__ f. Complete the task for the next activity and follow the process flow.

__ g. Try the same data without the **isNewPosition** selected. View the different flow results from the rule and the decision gateway **Is salary compliant**.

__ h. When the tasks are completed, log out of the Process Portal and minimize the browser window.



Part 5: Conduct 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 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.

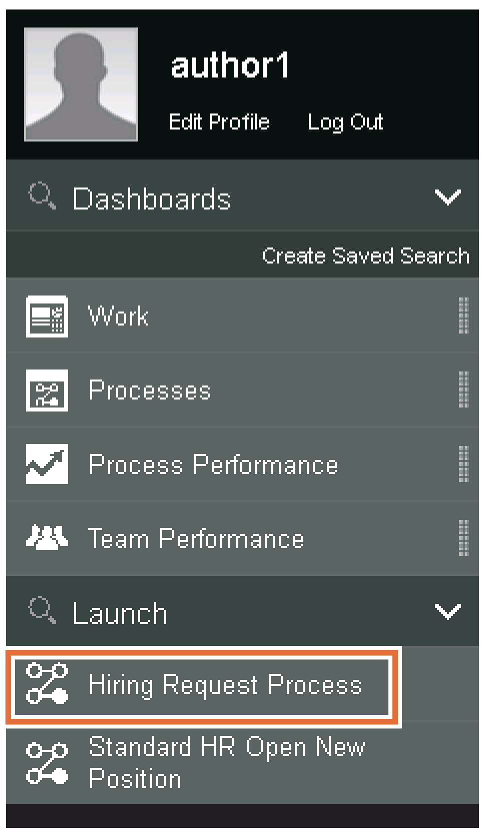
The new integrations and added features and functions are in the Hiring Requisition Process coaches, so it is now necessary to get a sign-off for the process application. Conduct a playback of your process to review the added enhancements.

__ 1. Show the coach enhancements.

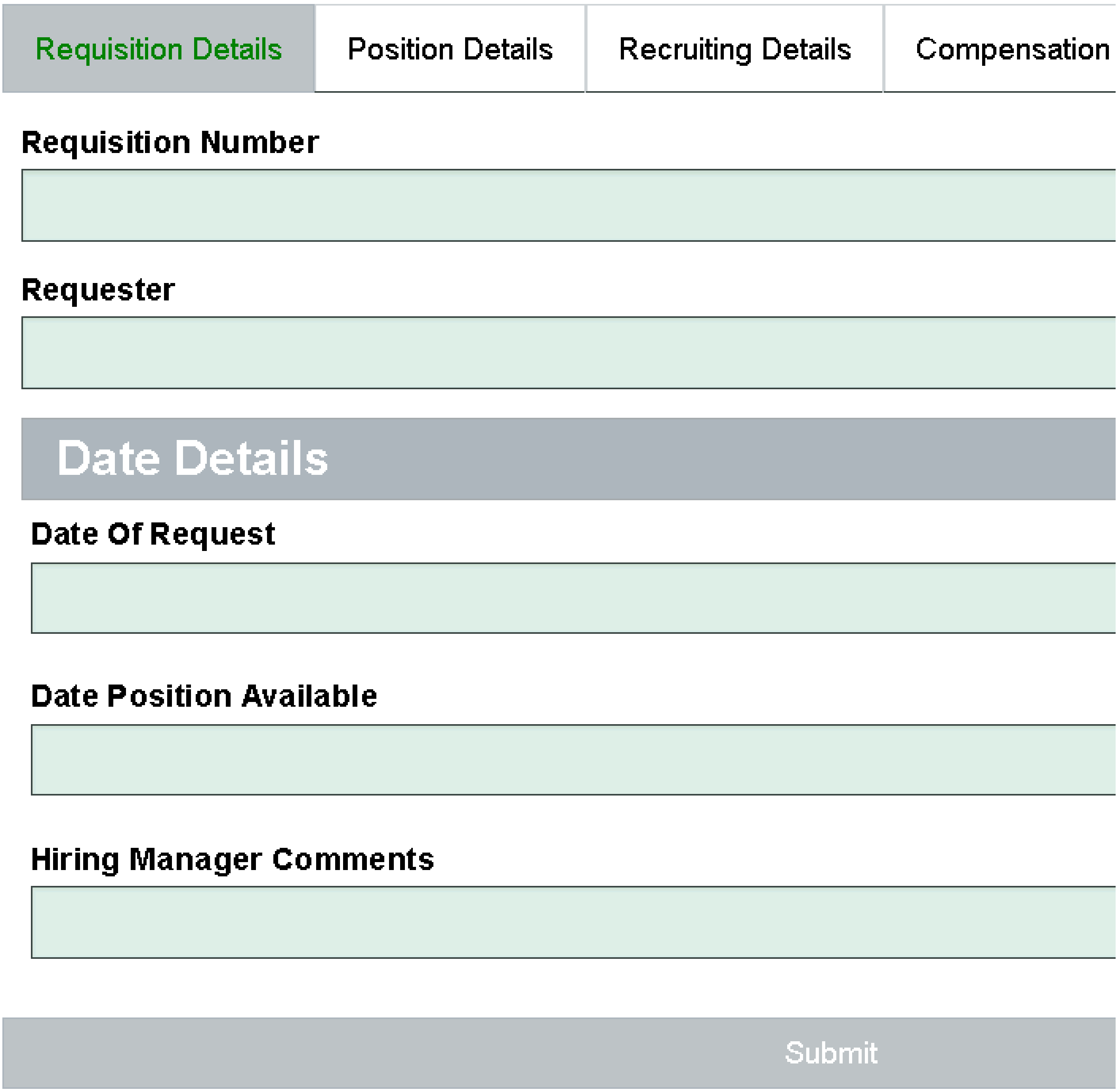
__ a. Maximize the **Process Portal** browser window.

__ b. Log in with `author1` as the **Username** and `author01` as the **Password**.

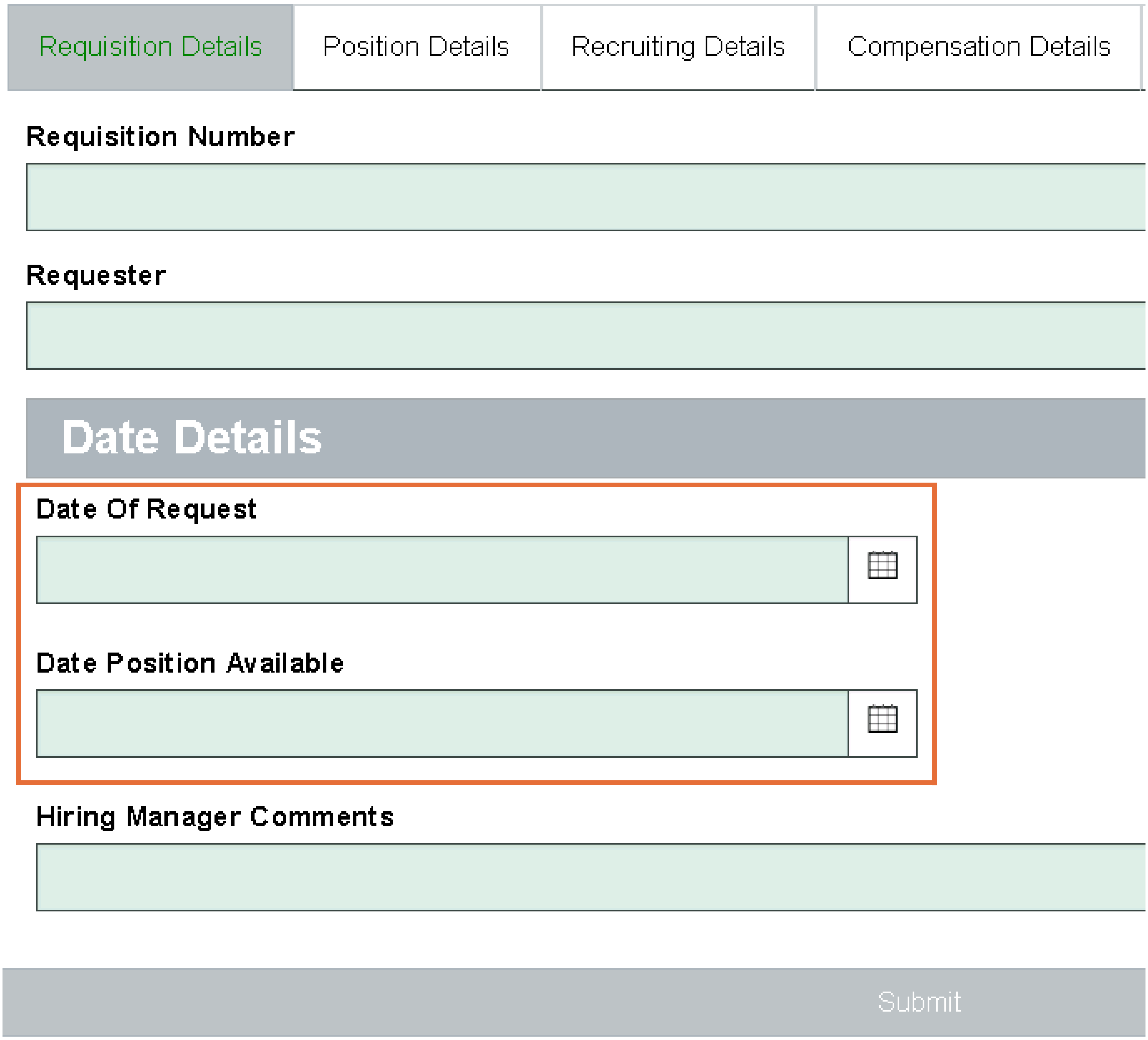
__ c. Create an instance of the **Hiring Request Process**.



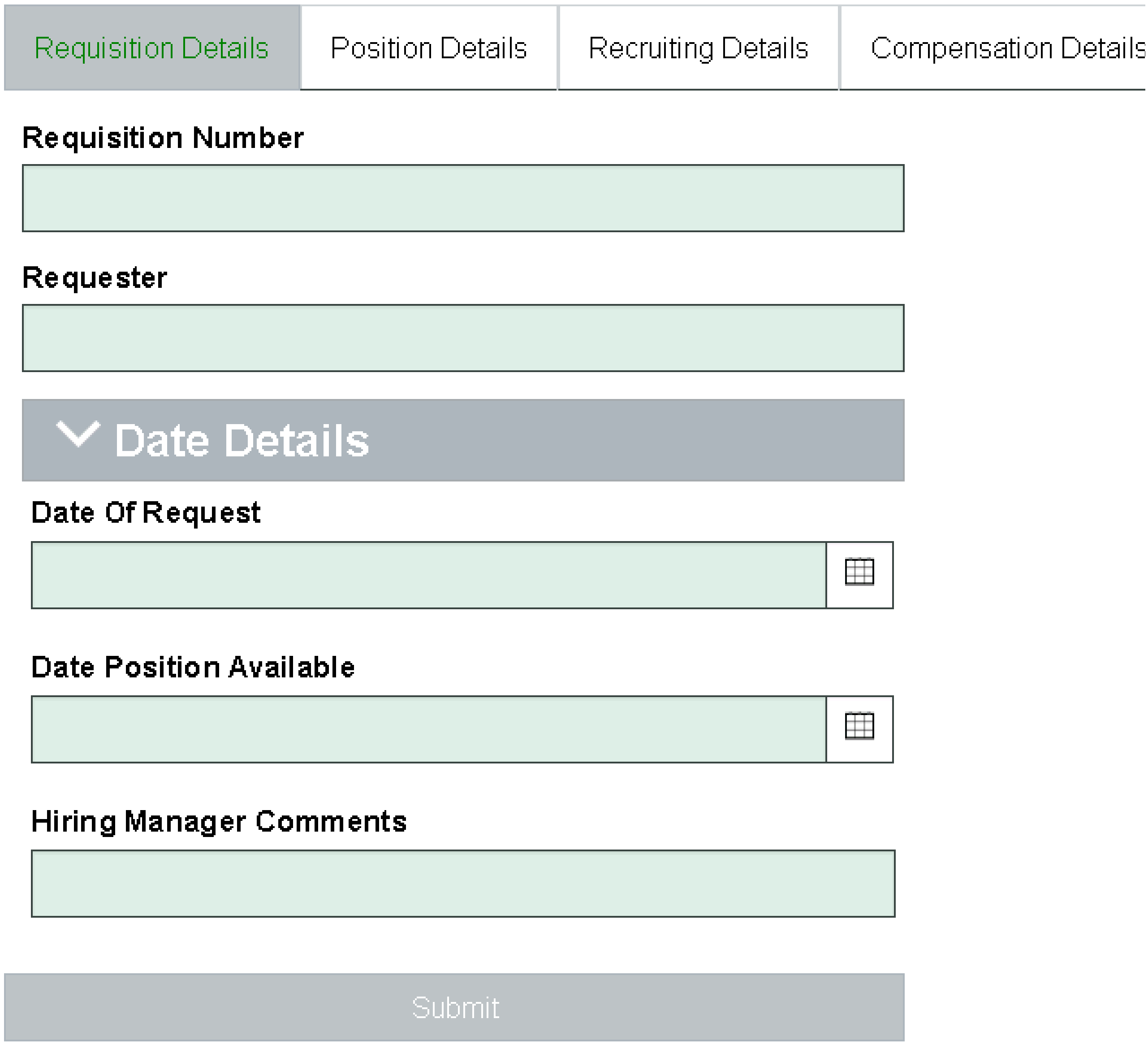
__ d. On the hiring form, demonstrate the tabs and how they are intended to move the user along in a wizard-like experience.



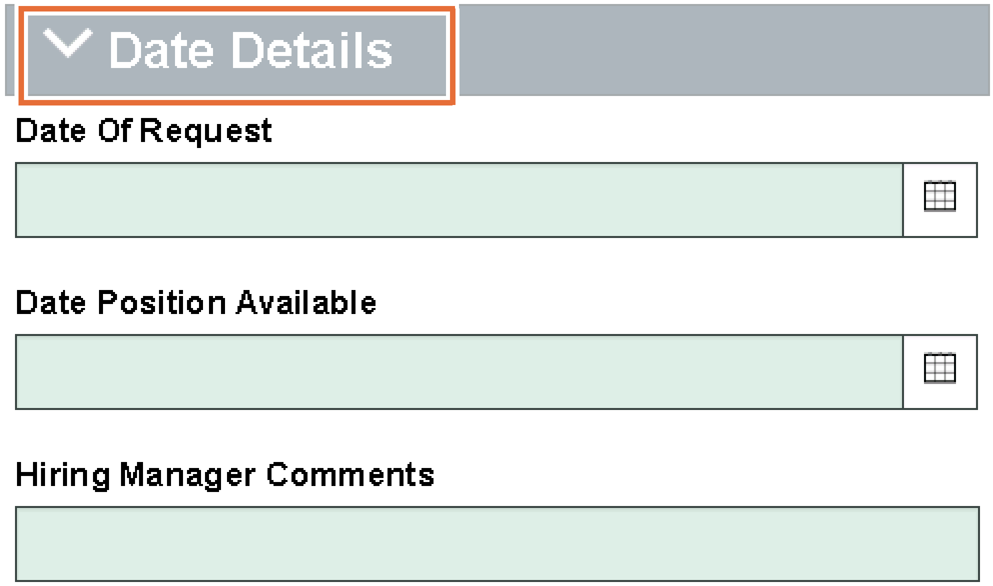
__ e. Show the shortened **Date of Request** input, which is necessary because you set the width of this input control to 50% for large screen.



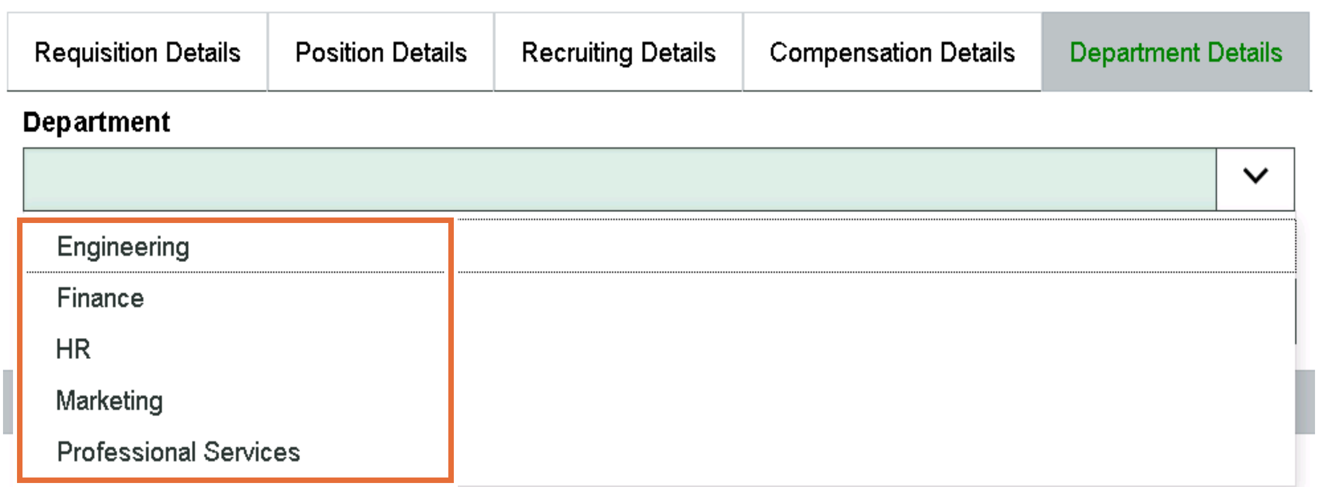
__ f. Slowly reduce the width of the browser. When the browser is 650 pixels or less, the **Date Details** section changes and the **Date Of Request** control in the section changes its width to 100%. This change is because you set the width of this input control to 100% for small and medium screen.



__ g. Demonstrate the collapsible icon added next to the **Date Details** section. You can use this icon to collapse and expand your Date Details section.



__ h. Demonstrate the select menu on the **Department Details** tab.



Information

This playback represents some of the items that you show during Playback 3. Most likely, you would continue to refine your UI so that you show the polished appearance, and make the coaches production-ready.

Part 6: 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.

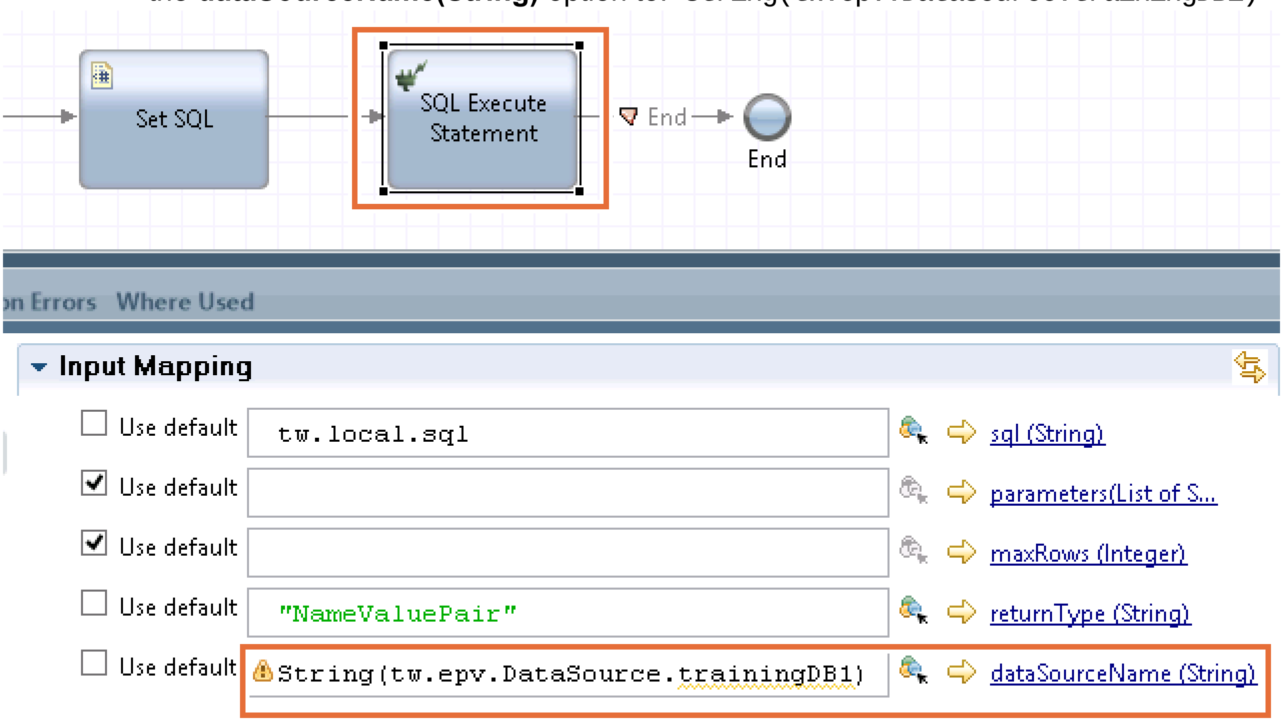
__ 1. Change the Retrieve Incident Categories service to throw an error.

__ a. For this demonstration, you want to show what would happen if a database is unreachable, so you remove the connection information to cause this error.

__ b. In the Process Designer client application, open the **Hiring Request Toolkit**.

__ c. In the library, click the **Implementation** category and open the **Retrieve Department Categories** service.

__ d. Click the **SQL Execute Statement** service. In the **Properties** tab, change the value of the **dataSourceName(String)** option to: `String(tw.epv.DataSource.trainingDB1)`



Note

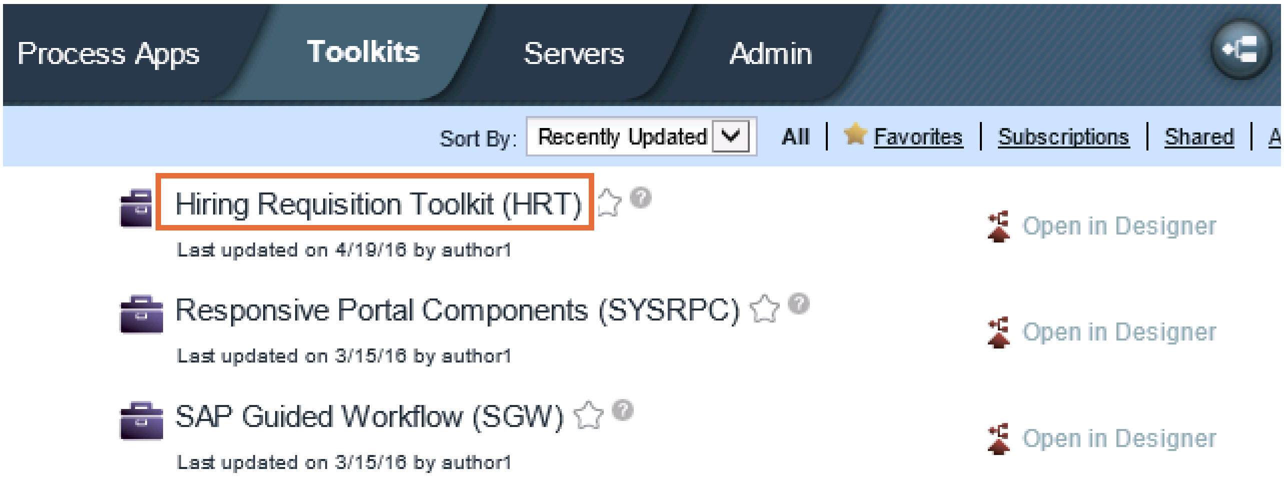
You changed the dataSourceName to `trainingDB1`. This change makes the database unreachable and throws an error message for the demonstration purpose.

__ e. Save this service.

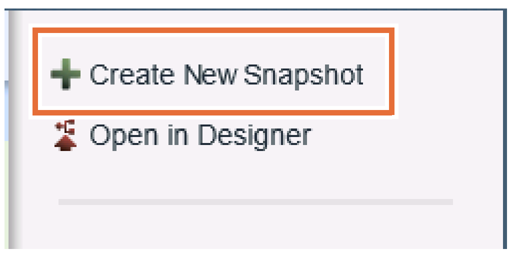
__ 2. Create a snapshot and upgrade process application dependency.

__ a. Return to the Web Process Designer.

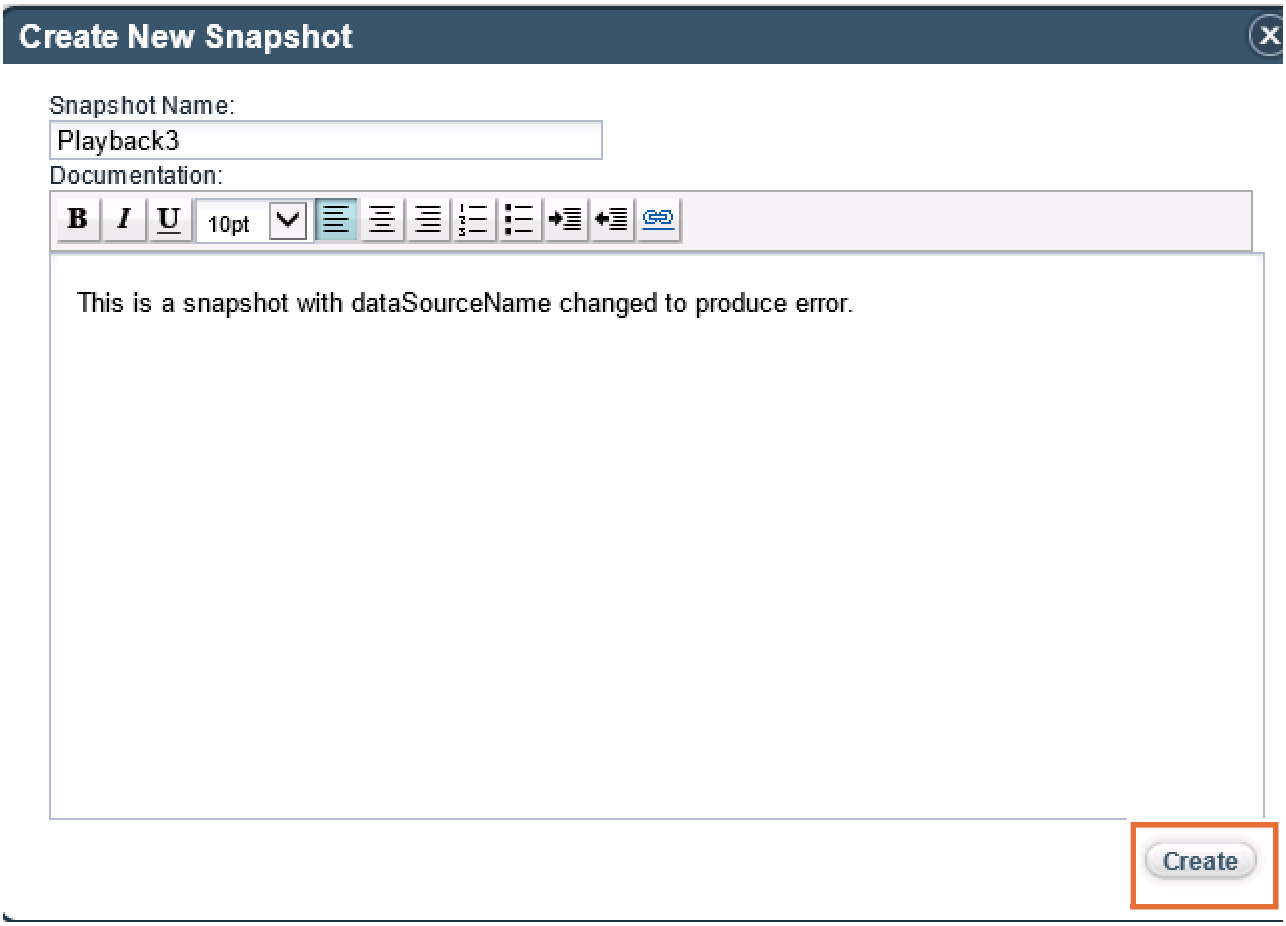
__ b. Click the **Toolkits > Hiring Requisition Toolkit (HRT)** link.



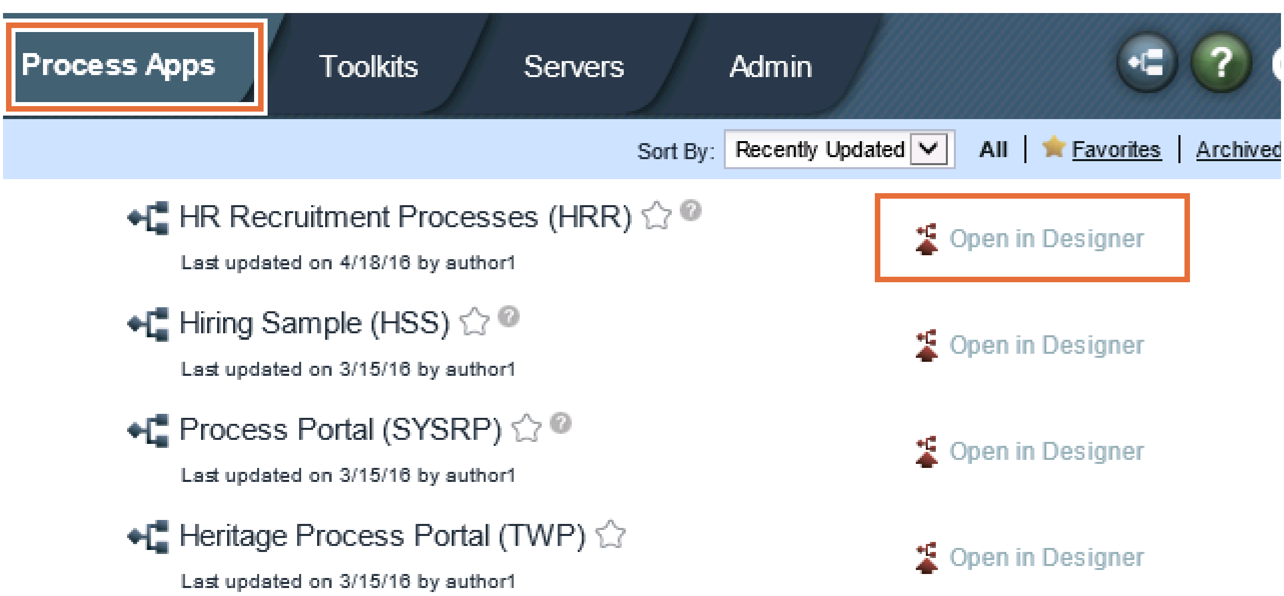
__ c. On the right, click **Create New Snapshot**.



__ d. Name the snapshot `Playback3` and provide the following documentation: `This is a snapshot with dataSourceName changed to produce error.` Click **Create**.

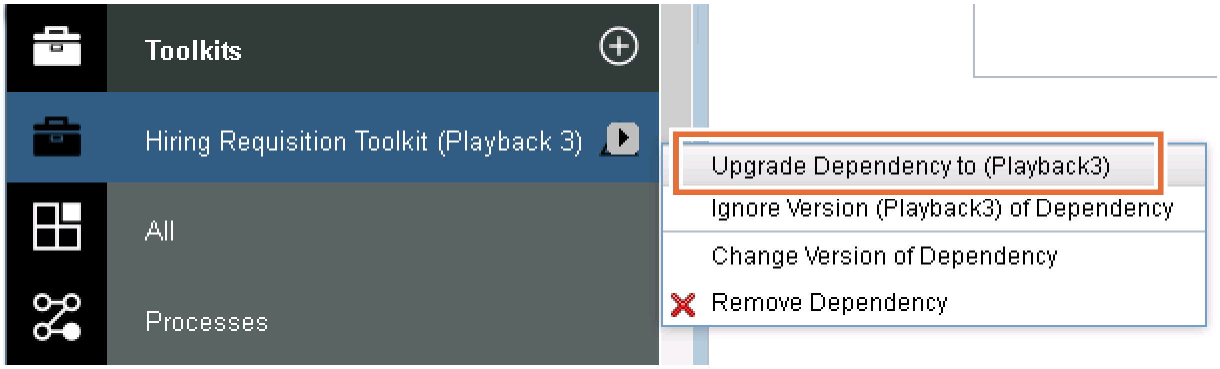


__ e. Click the **Process Apps** tab. Open the **HR Recruitment Processes** process application.



__ f. Click **Toolkits** in the library.

__ g. Click the arrow next to **Hiring Requisition Toolkit (Playback3)** and click **Upgrade Dependency to Playback3**.

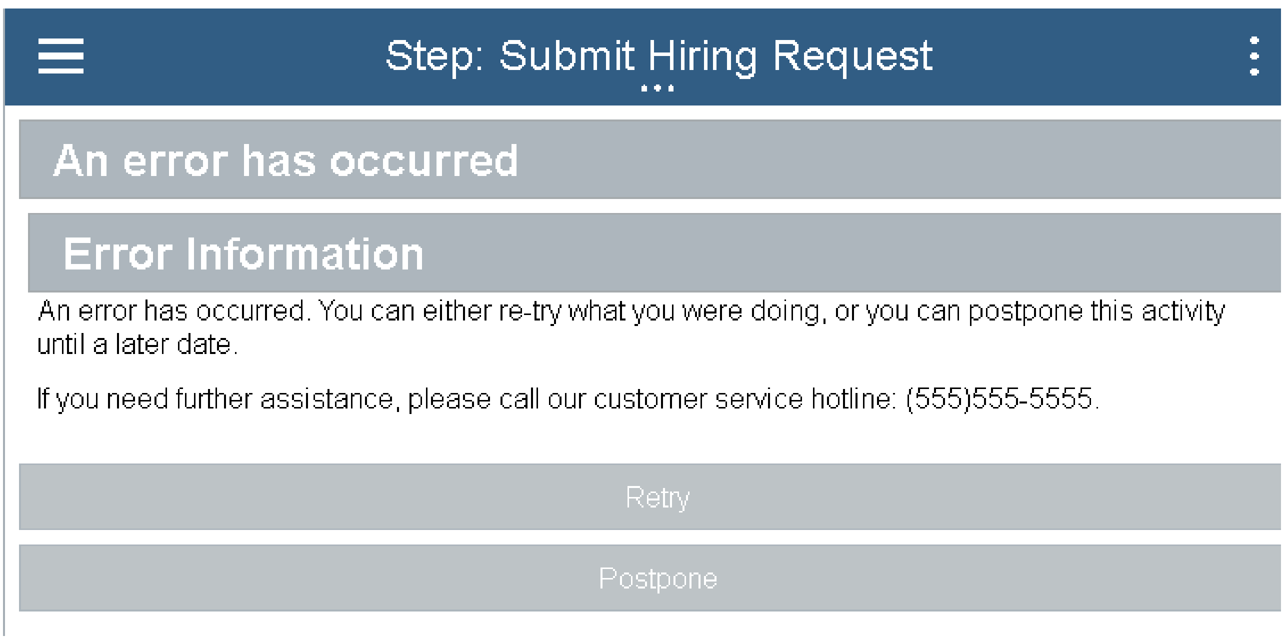


__ 3. Show the user experience when an error occurs in a service.

__ a. Log on to the Process Portal with `author1` as the user name and `author01` as the password.

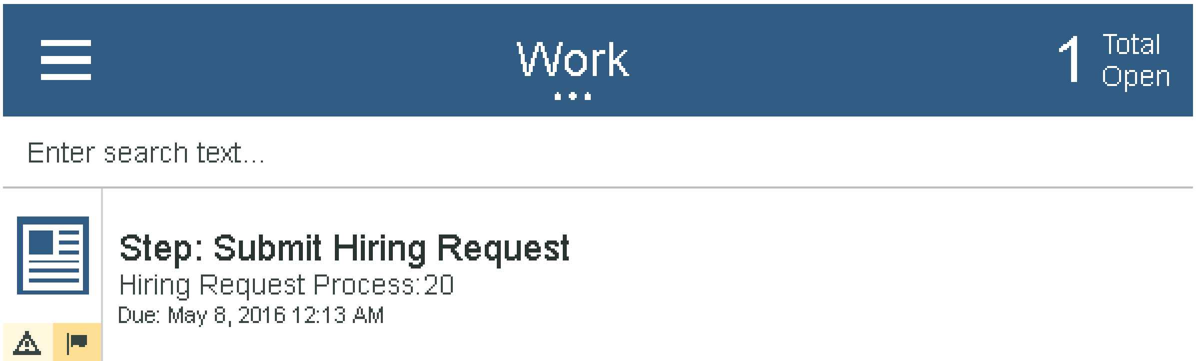
__ b. Click **Hiring Request Process** to create an instance of the process.

__ c. You immediately see the error display. Click **Retry**.



__ d. You immediately see the error display again. Click **Postpone**.

__ e. The task remains in the inbox until the error is resolved. Any data that is filled out on a coach is saved when a task is postponed, and is presented to the user when the task is resumed.



Information

You can correct the problem and show how a user would recover by going back into the Portal and running the task again. This time the coach is displayed as expected.

Important

Return to the **Retrieve Department Categories** service and remap the dataSourceName (String) input to `String(tw.epv.DataSource.trainingDB1)` to fix the problem with the service.

End of exercise

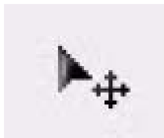
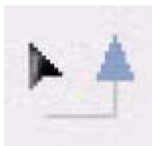
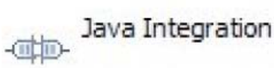
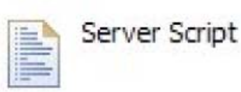
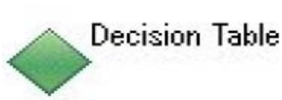
Exercise review and wrap-up

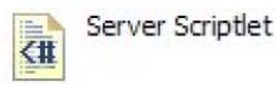
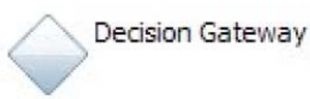
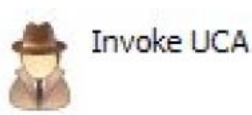
This exercise conducted all the playbacks from Playback 1 to Playback 3.

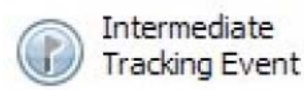
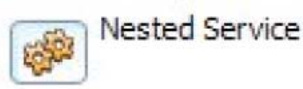
Appendix B. Modeling and implementation artifacts

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 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.
General system service	Use a general system service 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 general system service. General system services cannot include Java or web service integrations directly. You can call a general system service from any other type of service, and a general system service can call other nested services.
Ajax service	Use an Ajax service when you want to include a control in a coach to implement dynamic data selection such as automatically populating lists and automatically completing edit boxes. An Ajax service can pull data dynamically from a connected data source, such as a database. You cannot call an Ajax service from other types of services, but an Ajax service 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.
Integration service	Use an integration service when you want to integrate with an external system. An integration 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.
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
	All service types	Use to select and move components on the diagram.
	All service types	Use to connect service components to establish the order in which the steps in the service occur.
	Integration service only	Use to start an external web service. Use this component to supply a WSDL URI and then use any of the available services.
	Integration service only	Use to call methods from a Java class. You can use the methods to complete tasks like reading or writing files or sending SMTP mail.
	Human service only	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**.
	All service types	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.
	Decision service only	Use to build conditions for your decision services.
	Decision service only	Use to include decision services available on an ILOG JRules Rule Execution Server.

	All service types	Use 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.
	Human service only	Use to change the priority, due date, status, or other aspects of a task; for example, if you want the status of a task to change to **Closed** each time that a user completes a task. Use this component to properly set the status and move the task into each **Closed** folder in IBM Process Portal.
	Human service only	Use to halt processing without changing the status of a task.
	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.
	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.
	All service types	Use to add information about the overall service or each step in the service to the diagram. Adding notes helps other developers understand your design.
	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).
	All service types	Use to start an undercover agent (UCA) from your service.
	All service types	Use to listen for exceptions from the service component to which it is attached.

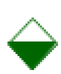
	All service types	Use to indicate a point in a service at which you want IBM Business Process Manager to capture the runtime data for reporting purposes. For examples of how tracking events are implemented, see **Creating a basic custom report** and **Creating a more advanced custom report**.
	All service 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.
	All service types	Use to send task-related alerts to IBM Process Portal.
	Integration service only	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