



Hello World Part 2: Service Components and Web Interfaces

Note

Before using this information and the product it supports, read the information in “Notices” on page 35.

Contents

Chapter 1. Introduction 1

Overview 1

Chapter 2. Build it Yourself 5

Import the Hello World Part 1 sample 5

Create an integration solution 6

Create new interfaces 7

Create a new module 9

Populate the module. 9

Create the human task implementation 11

Create the business process implementation 13

Chapter 3. Run the sample 21

Explore the solution 21

Deploy and test the modules 23

Remove the modules from the server. 32

Chapter 4. Import 33

Notices 35

Terms of use 39

Chapter 1. Introduction

In the Hello World Part 1 sample, you created a mediation service that accepted a title, first name and last name as input and returned *"Hello title firstName lastName"* as output. In this Hello World Part 2 sample, you extend the Part 1 application by creating a module that contains a business process. The business process uses a human task to prompt users for their name information and it then invokes the module from Hello World Part 1 to produce the resulting string. You will also expose this process as a Web service.

The Hello World Part 1 sample contains only mediation modules, which enables the sample to be deployed and run on either WebSphere Process Server or WebSphere Enterprise Service Bus. However, the Hello World Part 2 sample incorporates a standard business integration module to contain the business process, which means that the sample can only be deployed and run on WebSphere Process Server.

In the sample, you learn how to complete the following activities:

- Create a module for deployment to WebSphere Process Server
- Create a human task to interact with your application
- Create an import to invoke another module
- Create a business process to orchestrate service invocations
- Expose a business process as a Web service to remote clients
- Create an integration solution to group related projects and see their relationships
- Deploy and test module applications on the server
- Interact with the Web-based user interface of the Business Process Choreographer Explorer

Overview

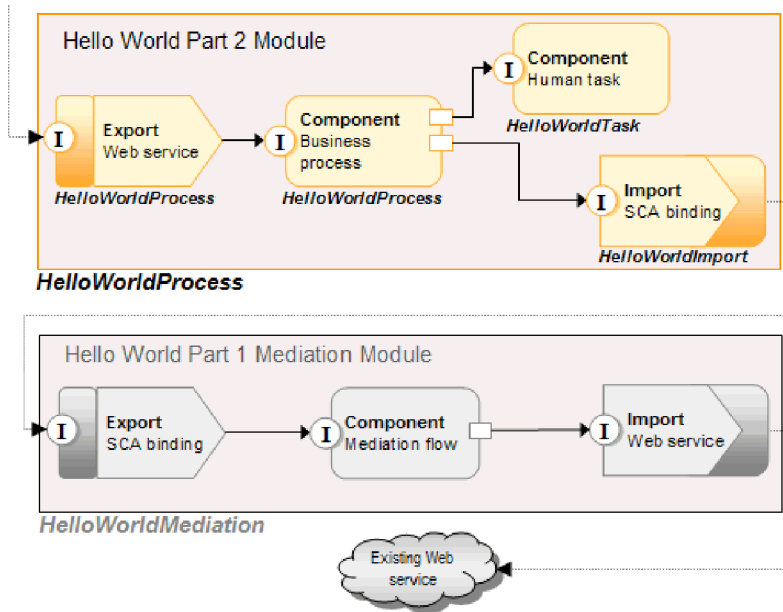
In this sample you create an SCA import component for invoking the module from Hello World Part 1, a human task component to prompt for the name information, and a business process component that invokes the first two components. This gives you an introduction to some of the capabilities (activities) of a business process and provides an introduction to the other core tools that supplement the business object, interface and mediation flow tools introduced in the Hello World Part 1 sample.

In Hello World Part 1 you built a mediation module that provides an interface to an existing Web service. In Hello World Part 2 you will build a business process that uses this interface.

At a conceptual level, the Hello World Part 2 sample application is comprised of the following elements:

- A module named **HelloWorldProcess** that contains a business process that is also named **HelloWorldProcess**
- A human task component named **HelloWorldTask**
- An import named **HelloWorldImport** and an export named **HelloWorldExport**.

These elements are shown in the following figure:

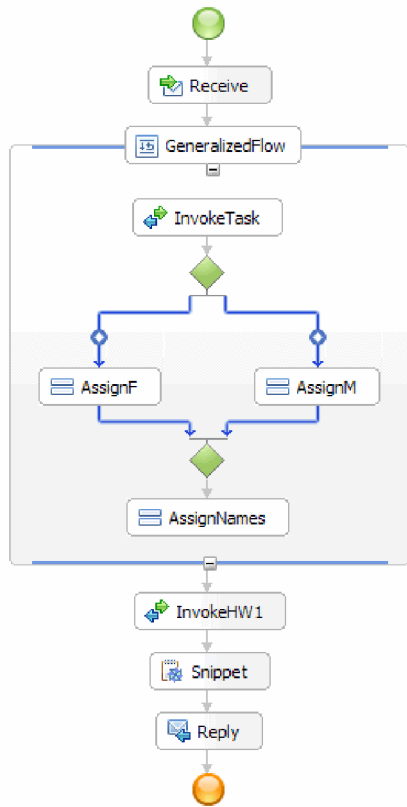


A *human task* is a unit of work performed by a person. In this sample, the HelloWorldProcess business process assigns a task for completion by a person -- you!

The HelloWorldProcess business process is a long-running process, which means that it can come to a complete stop while waiting for input or instructions. The most common form of this interruption would be a human interaction or decision, but it could also be the result of calling a long-running asynchronous service.

In the sample, the business process receives text input indicating gender and then sends this text to a To-do human task that displays it while prompting for user name information. You will find and claim the human task in the supplied Web-based user interface (that is used for working with human tasks) and then you will provide the requested name information. When you are finished with the task, you complete it to send the response back to the business process. The business process subsequently decides on an appropriate title string based on the gender input text. It then sends the title and name information to the service implemented by the module from the Hello World Part 1 sample, which results in a concatenated name prefixed by the word "Hello". This result is written to the console using a small snippet of visual Java code and it is returned to the waiting caller of the process.

This figure shows what the business process for this sample looks like:



Chapter 2. Build it Yourself

You can use the many tools of WebSphere Integration Developer to build the sample yourself.

To build the application, complete the following steps:

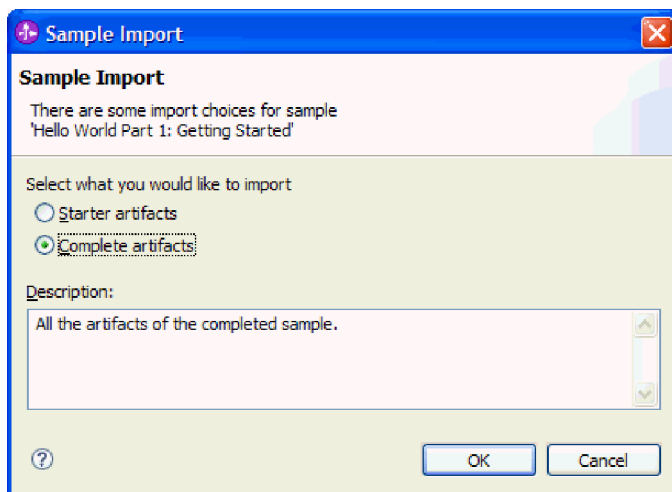
- Import the Hello World Part 1 sample.
- Create an integration solution.
- Create new interfaces for the process and the human task.
- Create a new module to contain the process and human task components.
- Populate the new module with the process and human task components, as well as the import and export components.
- Create the human task implementation.
- Create the business process implementation.

Import the Hello World Part 1 sample

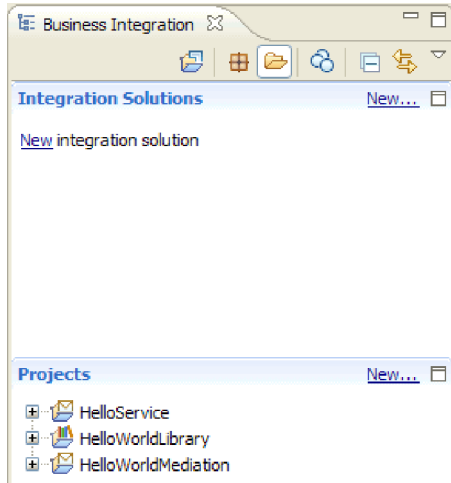
To build the Hello World Part 2 sample application, you need the Hello World Part 1 sample application in your workspace. If you did not build the Part 1 sample yourself and you did not import the ready-made version of the Part 1 sample, you should import it into your workspace now.

To import the ready-made Hello World Part 1 sample into your workspace:

1. In WebSphere Integration Developer, select **Help > Samples and Tutorials > IBM WebSphere Integration Developer 7.0**. The Samples and Tutorials page opens.
2. Under the **Hello World Part 2: Service Components and Web Interfaces** section, click the **Import** link.
3. You are presented with two possible options for import, as shown here:



4. Select **Starter artifacts** and click **OK**. This imports the Hello World Part 1 sample that you must have in your workspace to build the Hello World Part 2 sample. You should now see two module projects and one library project in your Business Integration view, as shown here:

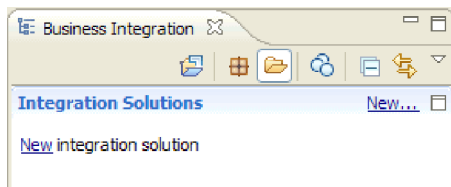


Create an integration solution

When you have numerous projects, it is helpful to group related projects together using an *integration solution*. Integration solutions are simply "super projects" that are used to reference other projects. They enable you to more easily test a group of related projects and work with them in a team environment. To help you visualize the relationships between the projects in an integration solution, an integration solution editor is provided.

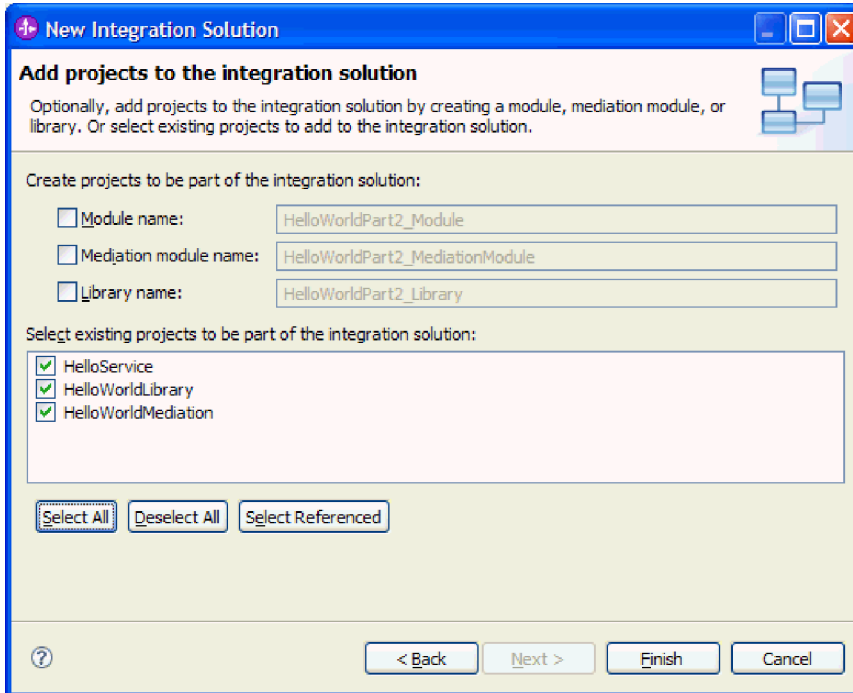
To create an integration solution:

1. In the **Integration Solutions** section of the Business Integration view, click the **New** link shown in the following figure:



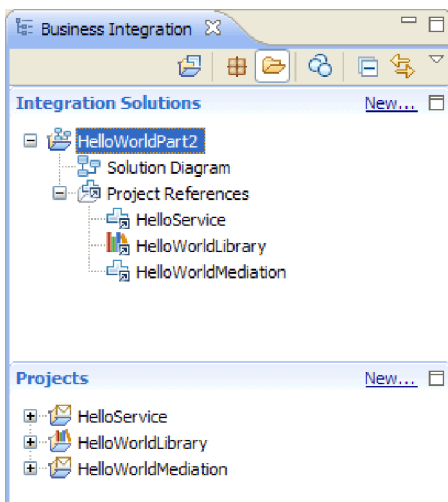
The New Integration Solution wizard opens.

2. In the **Integration solution name** field, type HelloWorldPart2 and click **Next**.
3. Click the **Select All** button to select the module and library projects **HelloService**, **HelloWorldLibrary**, and **HelloWorldMediation**, as shown in the following figure:



If there are other projects listed that are also selected, clear the check boxes beside their names.

4. Click **Finish** to add the new integration solution to the workspace. In the **Integration Solutions** section of the Business Integration view, expand the **HelloWorldPart2** integration solution and expand **Project References**. You can see that the **HelloService**, **HelloWorldLibrary**, and **HelloWorldMediation** projects are all referenced by the integration solution, as shown in the following figure:



Note: If necessary, you can grab the sash that separates the **Integration Solutions** section from the **Projects** section in the Business Integration view, then drag the sash up or down as needed.

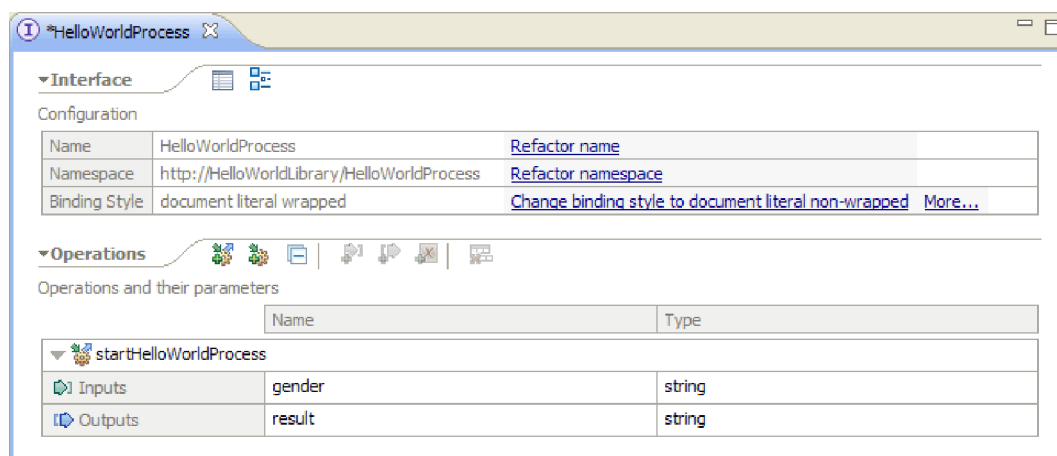
Create new interfaces

Your business process needs an interface of its own that takes a string for the caller's gender as input and returns the final concatenated result string.

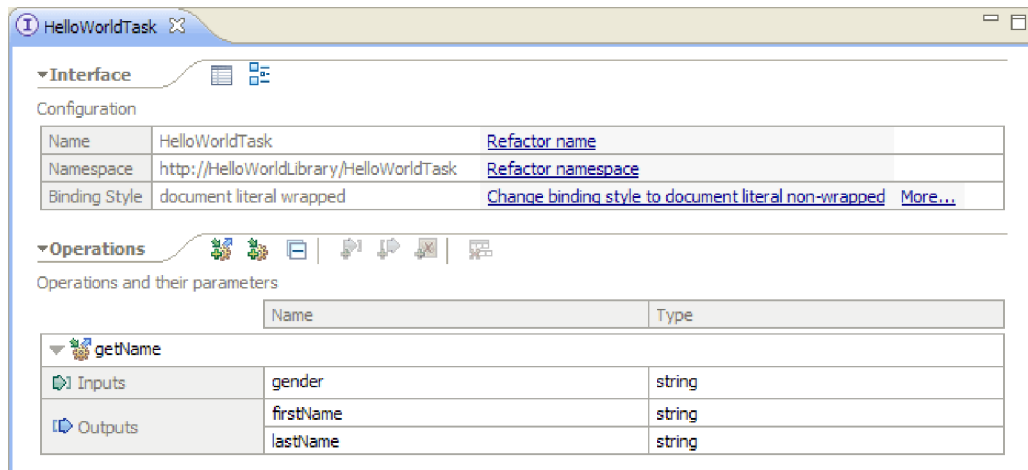
Your human task also needs its own interface. It will take a string as input and will be displayed in the user interface form of the claimed task, which will prompt the user for their first name and last name and return it as output from the human task after it is completed.

To create the two new interfaces in the existing HelloWorldLibrary library project:

1. Create the interface for the business process by completing the following steps:
 - a. In the **Projects** section of the Business Integration view, expand **HelloWorldLibrary**. then right-click on the **Interfaces** category and select **New > Interface**. The New Interface wizard opens.
 - b. In the **Name** field, type HelloWorldProcess and then click **Finish**. The interface editor opens.
 - c. In the **Operations** section, click the **Add Request Response Operation** icon to add a request response operation, then rename the default name **operation1** to **startHelloWorldProcess**.
 - d. Change the default input name from **input1** to **gender** and change the default output name from **output1** to **result**. The interface should now resemble the following figure:



- e. Press **Ctrl+S** to save the interface and then close the interface editor.
2. Create the interface for the business process by completing the following steps:
 - a. In the **Projects** section of the Business Integration view, expand **HelloWorldLibrary**, then right-click on the **Interfaces** category and select **New > Interface**. The New Interface wizard opens.
 - b. In the **Name** field, type HelloWorldTask and then click **Finish**. The interface editor opens.
 - c. In the **Operations** section, click the **Add Request Response Operation** icon to add a request response operation, then rename the default operation name **operation1** to **getName**.
 - d. Right-click anywhere in the **Outputs** row and select **Add Output**. A new output is added that is named **output2**.
 - e. Change the default input name from **input1** to **gender**.
 - f. Change the default output name **output1** to **firstName** and then change the default output name **output2** to **lastName**. The interface editor should now resemble the following figure:



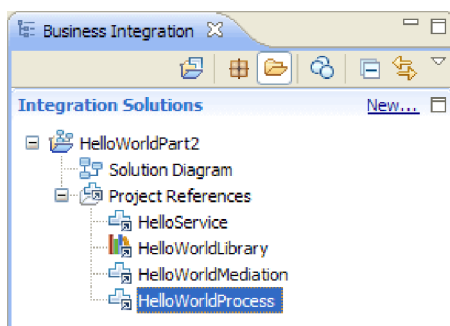
g. Press **Ctrl+S** to save the interface and then close the interface editor.

Create a new module

A module is required to hold the components that are necessary for defining your business process.

To create a new module:

1. In the **Projects** section of the Business Integration view, click the **New** link. The New Business Integration Project wizard opens.
2. Select **Module** and click **Next**. The New Module wizard opens.
3. In the **Module name** field, type HelloWorldProcess and click **Next**.
4. Select the **HelloWorldLibrary** check box (to enable the business process to access the interfaces in the library), then click **Next**.
5. Select the **Select an integration solution** check box.
6. In the **Integration solution** field, ensure that **HelloWorldPart2** is selected and then click **Finish**. The assembly editor opens for the new HelloWorldProcess module and a reference to the module is displayed in **HelloWorldPart2** integration solution, as shown in the following figure:



Populate the module

In the assembly editor for the new HelloWorldProcess module, you now need to add the business process component, human task component, and import and export components.

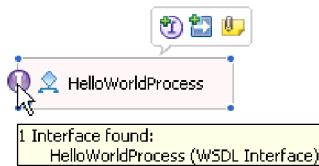
To populate the module:

1. If the assembly editor is not currently open for the HelloWorldProcess module, expand the **HelloWorldProcess** module in the **Projects** section of the Business Integration view, then double-click **Assembly Diagram** to open the assembly editor.

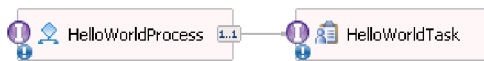
2. In the palette of the assembly editor, click the **Process** component and then drag it to the editor canvas.
3. In the canvas, click the new process component and then rename it to **HelloWorldProcess** and press the **Enter** key.

Note: If you accidentally double-click the process component, an Open dialog box will open to ask you whether you want to implement the component now. If the Open dialog box appears, click **No** (or press the **Esc** key).

4. In the canvas, click the new **HelloWorldProcess** component to display the hover bar above the component, then click the circled **I** icon to open the Add Interface dialog box.
5. In the **Matching interfaces** list, select **HelloWorldProcess** and click **OK**. The new **HelloWorldProcess** interface is added to the **HelloWorldProcess** component, as shown in the following figure:



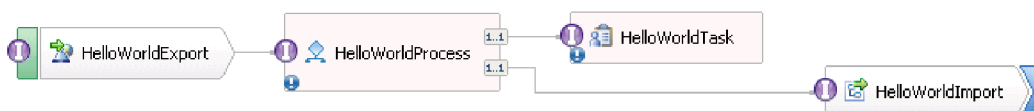
6. Drag and drop **Human task** from the palette to the canvas.
7. Rename the new human task component to **HelloWorldTask**.
8. Using essentially the same steps that you used to add an interface to the HelloWorldProcess component, add the interface **HelloWorldTask** to the new human task component.
9. Wire the **HelloWorldProcess** process component to the **HelloWorldTask** human task component and click **OK**. When the **Add Wire** dialog box opens, click **OK**. Here is what you should have so far:



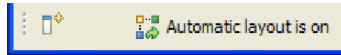
10. Press **Ctrl-S** to save your work.
11. Right-click the **HelloWorldProcess** process component and select **Generate Export > Web Service Binding**, then select the first transport option of **SOAP1.2/HTTP** and click **Finish**. An export is created in the assembly diagram.
12. Rename the new export by right-clicking on the export and selecting **Refactor > Rename**. (If the Save Modified Resources dialog box opens, click **OK**.) When the Rename Artifact dialog box opens, type **HelloWorldExport** in the **New name** field and click **Refactor**.

Note: Refactoring is an action available throughout WebSphere Integration Developer that allows you to make a change and propagate that change through all of the impacted files. In this case, it is necessary to use refactoring in order to change the WSDL port.

13. In the **Projects** section of the Business Integration view, expand the **HelloWorldMediation** project and the **Assembly Diagram** category, then select **HelloWorldMediationExport** and drag it to the assembly editor canvas for the HelloWorldProcess module. The Component Creation dialog box opens.
14. Select **Import with SCA Binding** and click **OK**. An SCA import component is generated that can be used to invoke the module from the Hello World Part 1 sample.
15. Rename the new import to **HelloWorldImport**.
16. Wire the **HelloWorldProcess** process component to the **HelloWorldImport** import. When the **Add Wire** dialog box opens, click **OK**. Your assembly diagram should now look like this:



- By default, the assembly editor canvas is in automatic layout mode and each component is positioned automatically. However, if you manually adjust the position of a component, the automatic layout capability is switched off. Look on the status line at the bottom of the workbench to see whether automatic layout is on or off. If the status is **off**, you can turn automatic layout on again by right-clicking in the assembly editor canvas and selecting **Automatic Layout**, as shown in the following figure:



Alternatively, you can leave automatic layout off and perform a one-time layout by selecting **Layout Contents**.

- Save the contents of the assembly editor.

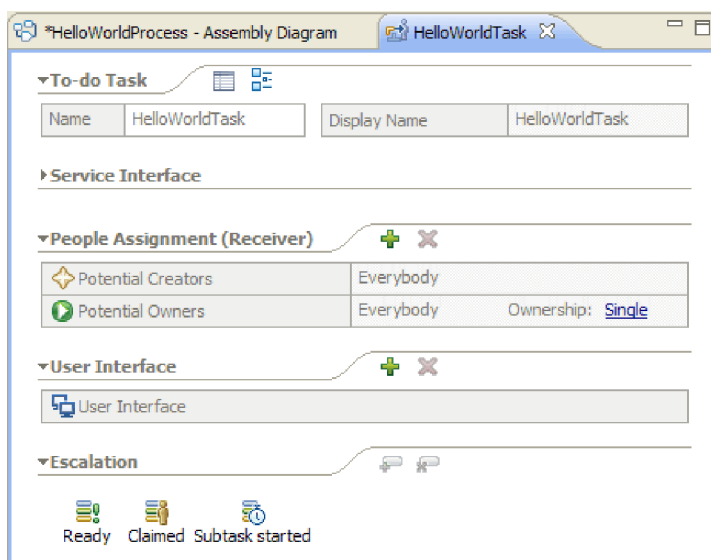
Create the human task implementation

You will now create the implementation of the human task component.

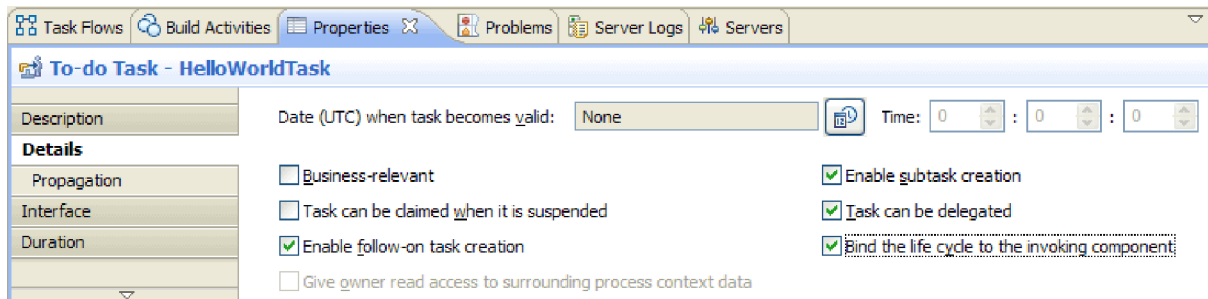
A *human task* is a unit of work performed by a person. There are several kinds of human tasks, such as a *To-do* task where the system assigns a task to a person who subsequently supplies the necessary information and completes the task, which results in the information being returned to the process.

To create the human task implementation:

- In the assembly editor, double-click on the **HelloWorldTask** component. Click **Yes** on the Open dialog indicating you do want to create the implementation. Click **OK** on the Generate Implementation dialog indicating you want to create the implementation file in the project's root folder. The human task editor opens, as shown here:

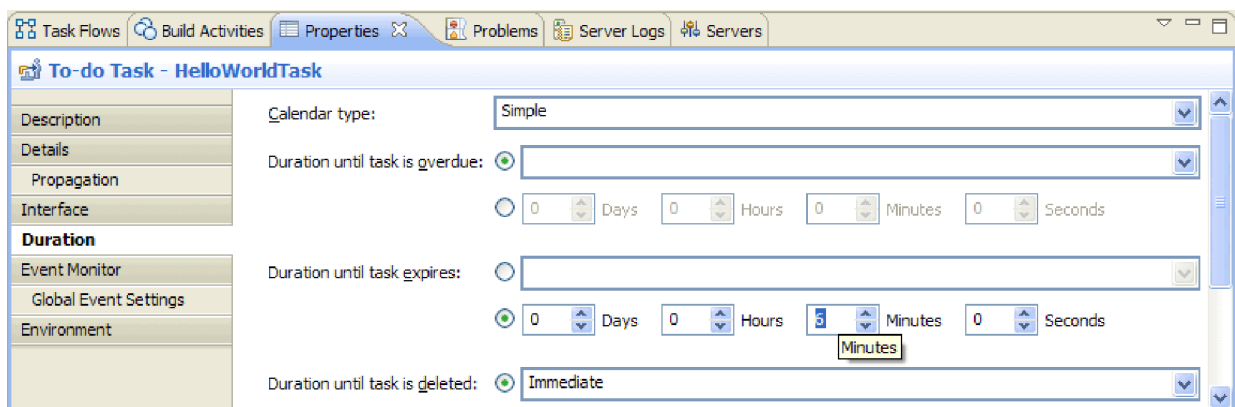


- Strictly speaking, there is nothing more you need to do here, but you want to learn! So click in the white space of the human task editor canvas near the bottom. Then select the **Properties** view below the editor and click the **Details** tab.
- In the bottom right of the Properties view, select the check box **Bind the life cycle to the invoking component**, as shown here:



This action ensures that outstanding instances of this task (to-dos) will be cleaned up when the process that invoked the task is cleaned up.

4. Select the **Duration** tab in the Properties view. Set the **Duration until task expires** to 6 minutes, as shown here:

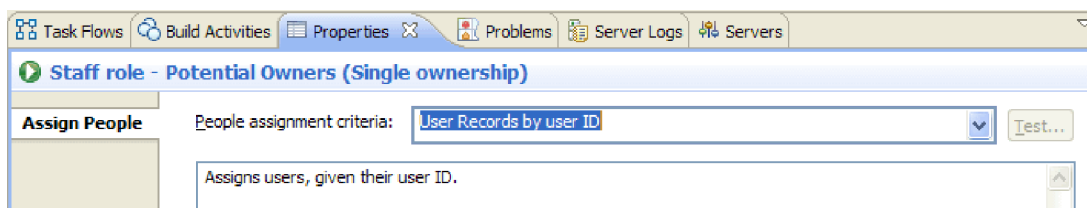


This way, it will clean itself up if you forget to claim and complete it in a reasonable amount of time.

5. In the editor canvas, note the **People Assignment (Receiver)** section, as shown here:



By default, anyone can create instances of this human task (in other words, create to-dos) and anyone can claim those instances and work on them. However, you can restrict this capability. Select the **Everybody** cell in the **Potential Owners** row and focus the Properties view. There is only the **Assign People** tab. In the **People assignment criteria** field, select **User Records by user ID**, as shown here:



In the Assign People page, scroll down to the table and set the value of **userID** to the user ID that was specified for the server at install time. If you did not change the user ID, then specify the default user ID admin, as shown in the following figure:

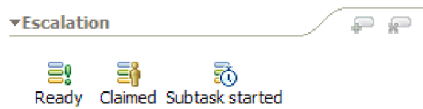
☒ If only one person qualifies, claim the task automatically.

Name	Value
UserID *	admin
AlternativeID1	
AlternativeID2	

In the **People Assignment** section near the top of the editor, you see that the table has changed, as shown in the following figure:

▼People Assignment (Receiver) + ×	
Potential Creators	Everybody
Potential Owners	User Records by user ID Ownership: Single
	UserID * admin

6. Optional: Note the **Escalations** section, as shown here:



In addition to specifying a duration for how long users must process a task before it expires, you can also specify a series of escalation actions in case the task is not claimed in a certain amount of time after it is created (Ready state), or in case it is not completed in a certain amount of time after it is claimed (Claimed state).

Actions include creating a new to-do task for someone else or sending an e-mail notification. This is where you set these escalations up, using the green “plus” button when one of the state icons is selected.

For both durations and escalations, you can specify elapsed time not only with absolute hours, minutes or days, but you can also specify it with a *business calendar*. By creating and specifying a business calendar, you can identify noncontiguous time. For example, you can specify that escalation should only occur after two business days have elapsed.

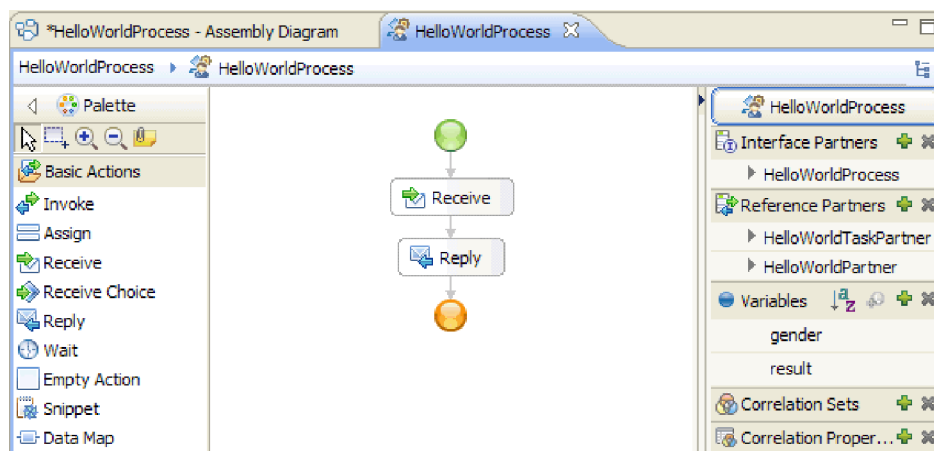
- 7.
8. Save and close the human task editor and then save your work in the assembly editor.

Create the business process implementation

You will now create the implementation of the business process component, which is more complex than the implementation of the human task.

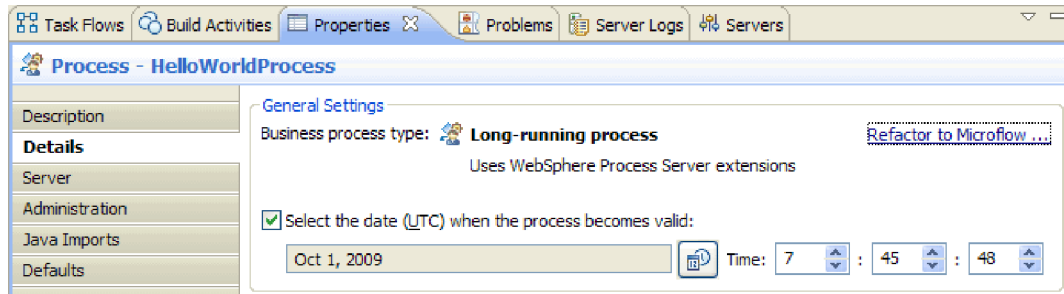
Create the business process implementation:

1. In the assembly editor, double-click the **HelloWorldProcess** process component to start work on it. Click **Yes** in the Open dialog and click **OK** in the Generate Implementation dialog. The business process editor opens, as shown here:



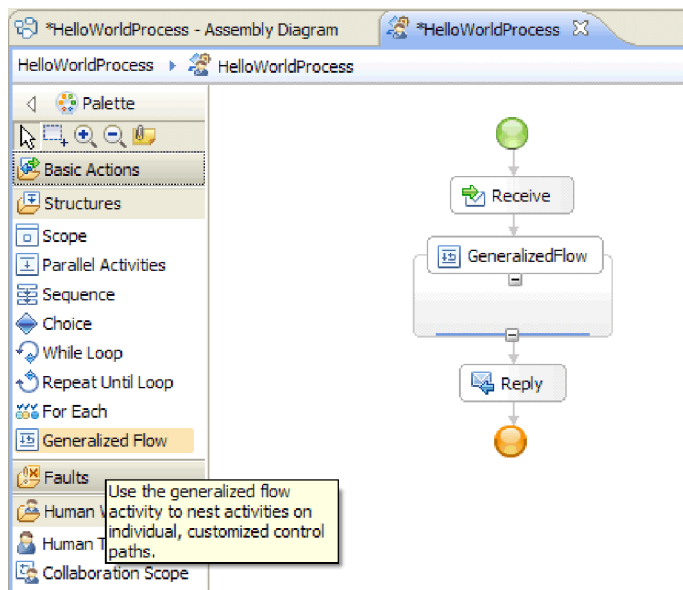
2. Your process is going to invoke a human task, which could take a long time to respond. As a result, you need the process to be defined as **long running**. Click somewhere in the white space of the

editor and go to the **Properties** view and select the **Details** tab. Notice that the process is identified as a **Long-running process**, as shown here:



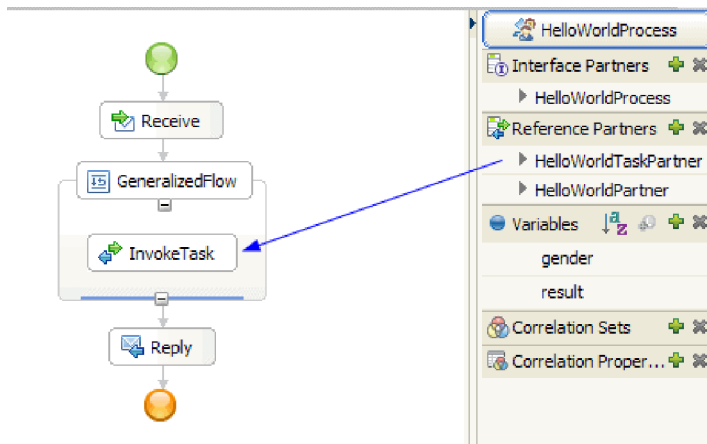
If you want to change the process from a long-running process to a microflow, click the **Refactor to Microflow** link. This ensures that not only the process definition is changed, but also the process component in the assembly editor. This means that the process changes and all downstream artifacts impacted by the change are updated.

3. A business process is made of *activities* or individual steps. When a process is initially created for an interface with a request response operation, it has two activities pre-supplied: a **Receive** activity for starting the process through the operation and a **Reply** activity for returning the response to the caller. Your own activities will be inserted between these two activities. On the palette, expand the **Structures** category and drag **Generalized Flow** and drop it between **Receive** and **Reply**, as shown here:



Note that some activities like Generalized Flow are structured, meaning they are intended to contain other activities. To produce a flow that branches, you must use Parallel Activities, Generalized Flow, or Collaboration Scope.

4. In the tray on the right side of the canvas, expand the **Reference Partners** category and drag and drop **HelloWorldTaskPartner** into the **GeneralizedFlow** structure, then set the name to **InvokeTask**, as shown here:

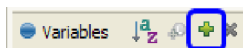


This creates a configured Invoke activity that calls this reference, which is wired to the human task component back in the assembly editor. This means that at run time, the activity results in the creation of an instance of your human task; that is, a **to do** task is created and awaits being claimed by one of its potential owners. It is not fully configured yet, however. You need to define where to get its input parameter data and store its output results.

- On the canvas, select the **InvokeTask** activity and open the **Properties** view below the editor. Select the **Details** tab. In the **Inputs** rows of the table, click on **none** in the cell of the **gender** row and of the last column named **Read from Variable**. You see a drop-down list showing all variables currently defined in this process that have a matching type, which at this point consists only of the input and output parameters of the process interface. Select the **gender** variable, which is the input parameter to this process. This selection means that you will pass this variable's text data as input to the human task.
- Similarly, in the **Outputs** rows, click on **none** in the cell of the **firstName** row and of the last column named **Store into Variable**, but this time select **New** and create a new variable named **firstname**. Do the same for the **lastName** row and create a new variable named **lastname**. Note that the variables are created with the appropriate type for this reference partner parameter. Your table should look like this:

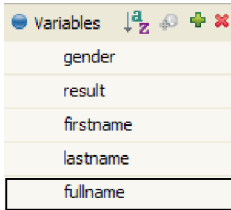
	Name	Type	Read from Variable
Inputs	gender	string	gender
	Name	Type	Store into Variable
Outputs	firstName	string	firstname
	lastName	string	lastname

- At run time when the **InvokeTask** returns, you will have variables containing the first name and last name of the user. The Hello World Part 1 sample service that you need to invoke also requires a title like Mister, which means that you need to define a variable to hold the value. Beside the **Variables** category in the tray, click the plus (+) icon as shown in the following figure:



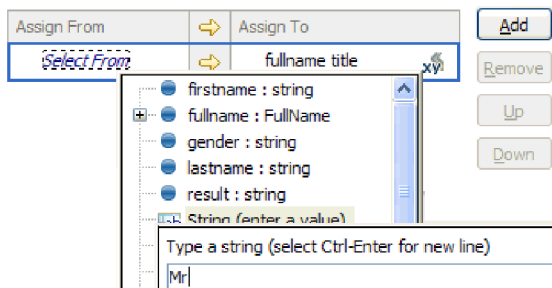
The Add a Variable dialog box opens.

- In the **Name** field, type the variable name **fullname** and select the type **FullName**, which is a business object created in the Hello World Part 1 sample. Click **OK**. This is the matching data type of the variable that the service from the first sample requires as input. Your variable list should now look like this:

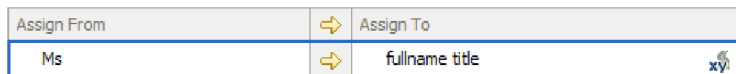


The fullname variable has three fields defined in it, all of type string: **title**, **firstName** and **lastName**. You will need to set a value for all three fields before you can invoke the service, as described in the following steps.

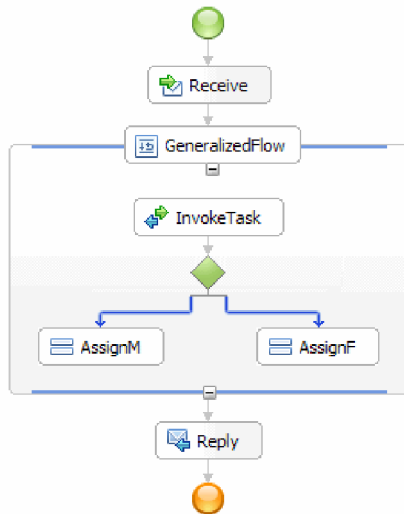
9. From the palette, expand the **Basic Actions** category and drag and drop the **Assign** activity to the inside of the **GeneralizedFlow** structure on the canvas. Rename it to **AssignM**.
10. While **AssignM** is selected, go to the **Properties** view and the **Details** tab. In the **Assign To** column of the table, click on **Select To**. In the drop-down list, expand **fullname** and select **title**.
11. In the same table, in the **Assign From** column, click on **Select From**. Select **String (enter a value)** from the drop-down list and type Mr, as shown here:



12. In the canvas, wire the **InvokeTask** activity to the **AssignM** activity and select **Add a Link** from the pop-up dialog box.
13. Once again, drag and drop the **Assign** activity from the palette to the interior of the **GeneralizedFlow** structure and name it **AssignF**. Configure it to assign **Ms** to the title field, as shown here:



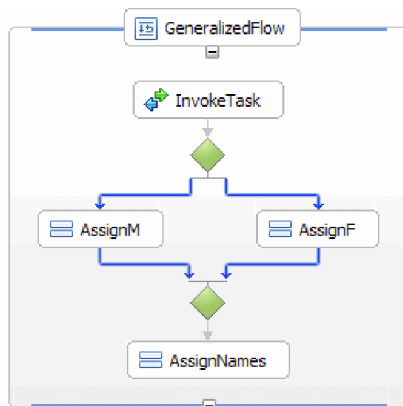
14. Wire the **InvokeTask** activity to the **AssignF** activity. In the resulting popup, select **Add a Link**. Your flow should now look like this:



15. It is time to assign values to the remaining two fields in fullname: namely **firstName** and **lastName**. You will set these from the values that are returned from the human task. Drag and drop **Assign** again from the palette to the inside of the **GeneralizedFlow** structure and then rename it to **AssignNames**. Using the **Add** button to add another row to the Assign table, configure it as shown in the following figure:

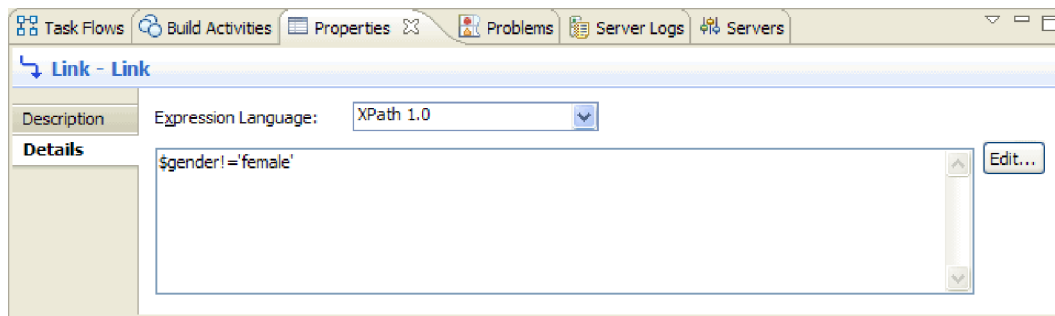
Assign From	⇒	Assign To		Add
firstname	⇒	fullname firstName	xy!	Remove
lastname	⇒	fullname lastName	xy!	Up

16. Wire both the **AssignF** and **AssignM** activities to this new **AssignNames** activity and select **Add a link** when prompted. The **GeneralizedFlow** should look like this:



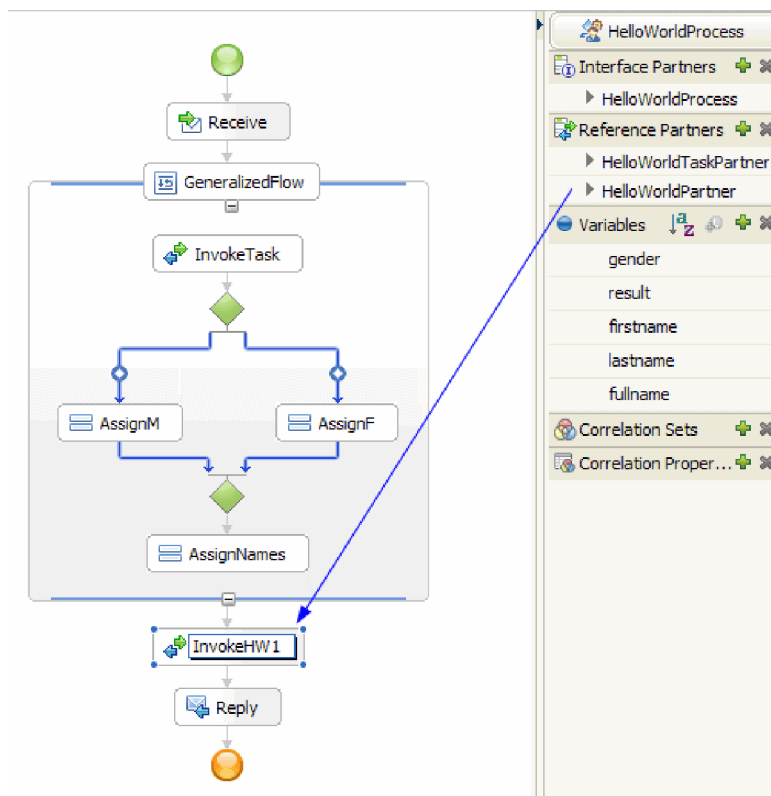
17. You need to condition the two links coming from **InvokeTask** so that there is some criteria about when to follow each link. Select the blue link that goes into **AssignF** and go to the **Details** tab of the Properties view. In the **Expression Language** field, select **XPath 1.0** and then click the **Edit** button. The XPath Expression Builder opens.
18. Set the gender condition to **female** using one of the following approaches:
- Type the expression `$gender='female'` directly and click **OK**.
 - Click **Insert Simple XPath** and select **gender** in the list of data types, then select the equals sign (=) as the **Operator** in the **Add an optional condition** area and type **female** in the corresponding **Value** field. Click **OK** twice.

19. Select the link that goes into **AssignM**, and for its properties set the Expression Language to XPath 1.0 and the Condition to `$gender!='female'` as shown here:



Optional: It is possible to give your links a label by setting the **Display Name** in the **Description** tab of the Properties view, then showing the label by right-clicking the link and selecting **Show Labels on Links**.

20. In the tray, drag and drop **HelloWorldPartner** to a position immediately above the **Reply** activity and name it **InvokeHW1**, as shown here:



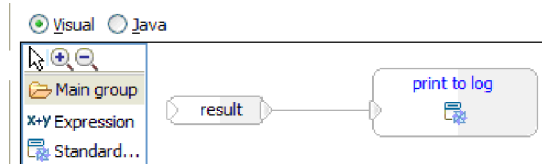
21. In the **Details** tab of the Properties view, for the **InvokeHW1** activity, bind the input and output parameters to the **fullname** and **result** variables, as shown here:

		Name	Type	Read from Variable	
Inputs		fullname	FullName	fullname	⇒
		Name	Type	Store into Variable	
Outputs		result	string	⇒	result

22. From the Basic Actions category in the palette, drag **Snippet** to immediately above **Reply**. In the **Details** tab of the Properties view, there is a visual Java snippet editor. Double-click the **Properties**

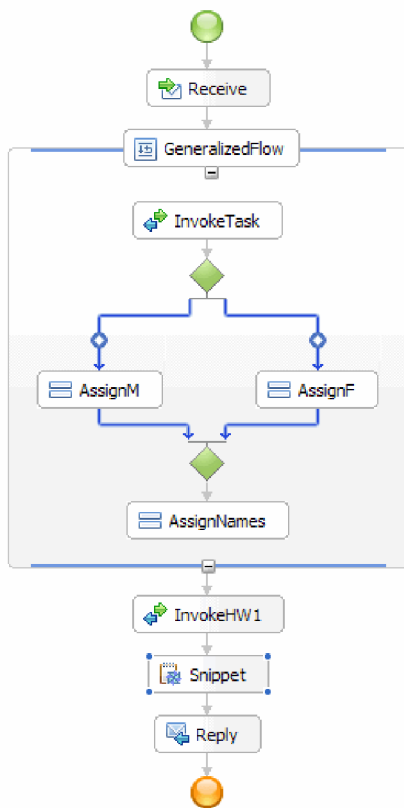
tab to go full screen. Click on **Standard** in the palette. The Add a Standard Visual Snippet dialog appears. Expand **utility** and select **print to log** and then click **OK**. Click in the visual snippet editor canvas. A **print to log** node appears.

23. Drag the **result** variable from the tray on the right to the visual snippet editor canvas. Wire from the **result** node to the **print to log** node, so that your snippet looks like this:

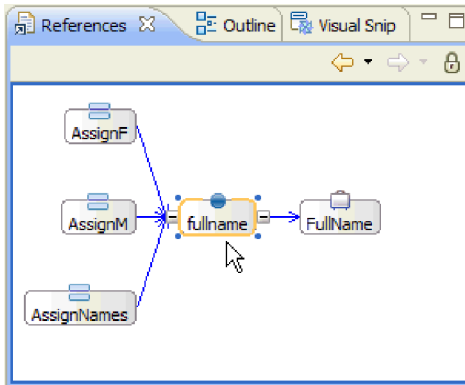


Restore the Properties view to its normal size. You have now visually authored Java code to emit the contents of the **result** variable to SystemOut. Technically, by wiring one node to another, you supply an input parameter to a method.

24. You are done authoring the process! Save your process. It should look like this:



25. Switch to the assembly editor and save your changes, then switch back to the business process editor.
26. Optional: In your process, you sometimes want to know which activities use a particular variable. There is a way to do that. In the tray, select the **fullname** variable. From the **Window** menu, select **Show View > References**. The References view opens in the lower left of the perspective, where you see a graph showing as input to the variable all those activities that set the variable, as shown in the following figure:



Congratulations! The authoring steps are complete. It is now time to test.

Chapter 3. Run the sample

After you have finished building the sample, you can run it.

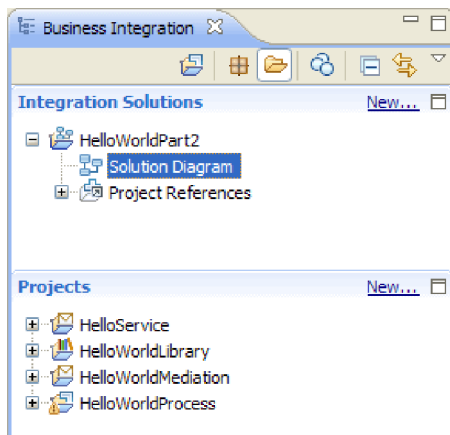
However, you will first explore the integration solution to view the projects that make up the entire application and to learn about the relationships between the projects.

Explore the solution

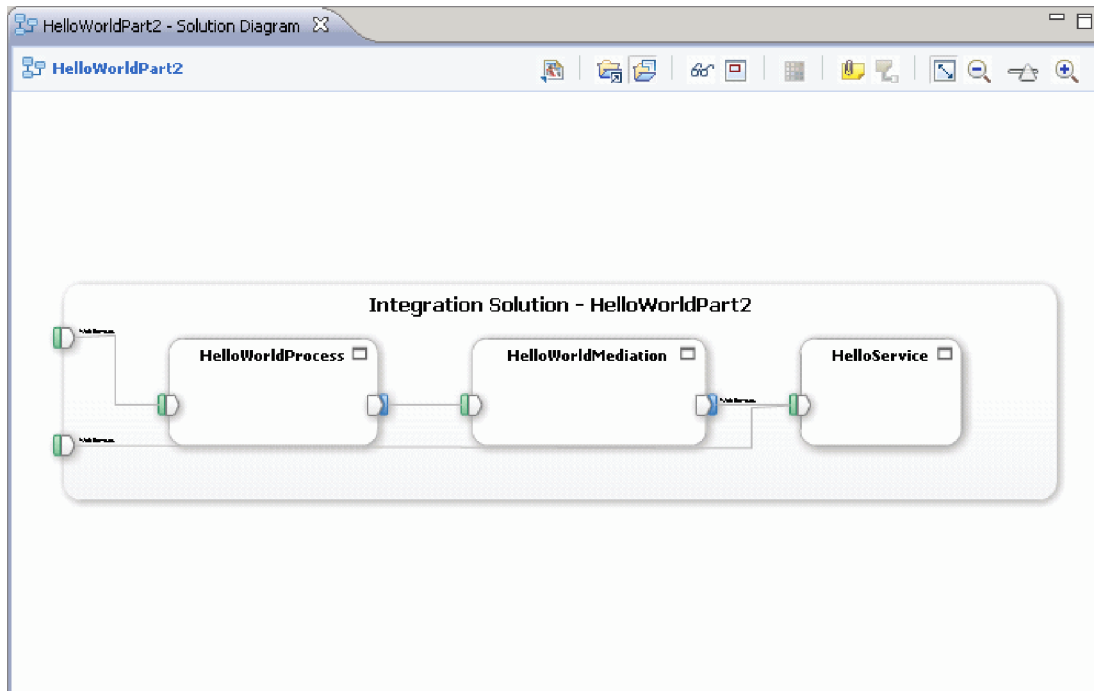
Sometimes you need to obtain a high-level view of your application to see all of the various elements and how they fit together. The integration solution editor can help with this task.

To explore the solution:

1. In the Business Integration view, expand the integration solution **HelloWorldPart2**, as shown here:

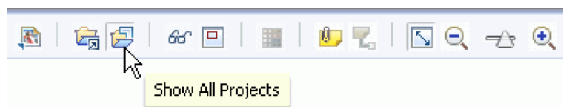


2. Double-click **Solution Diagram**. The integration solution diagram opens in the integration solution editor, as shown in the following figure:

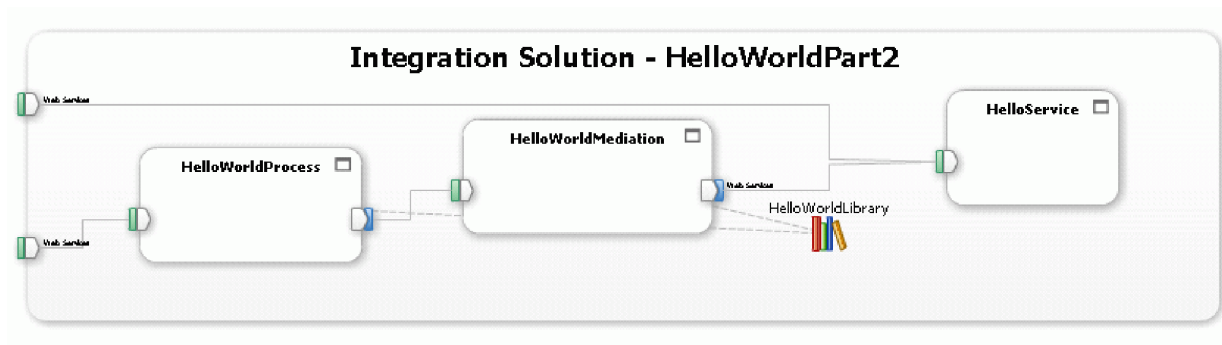


In the solution editor, you see all three modules inside the integration solution. The left edge of each module displays an export and the right edge of each module displays an import (except for the HelloService module). Any non-SCA imports and exports are connected to the edge of the integration solution. SCA imports and exports are not connected to the edge of the integration solution because they are only designed for direct communication between modules. If a module invokes another module through an SCA binding, a connection is displayed between the two modules.

3. In the solution editor, display the library **HelloWorldLibrary** by clicking the **Show All Projects** icon, as shown in the following figure:



The integration solution diagram now displays the **HelloWorldLibrary** library with connections to the modules that reference it, as shown in the following figure:



If you want to see the contents in a module, click the **Expand module** icon in the upper right corner of the module. For example, if you click the **Expand module** icon on the HelloWorldMediation module, you should see the contents shown in the following figure:



If you want, you can explore the other icons on the toolbar as well as the context menu items. For example, the icons and menu items enable you to open the assembly editor, add sticky notes for comments, set the background color of a module, and perform numerous other tasks.

4. Close the solution diagram. If you are prompted to save your settings, click **No**.

Now that you have some familiarity with the integration solution, it is time to test it.

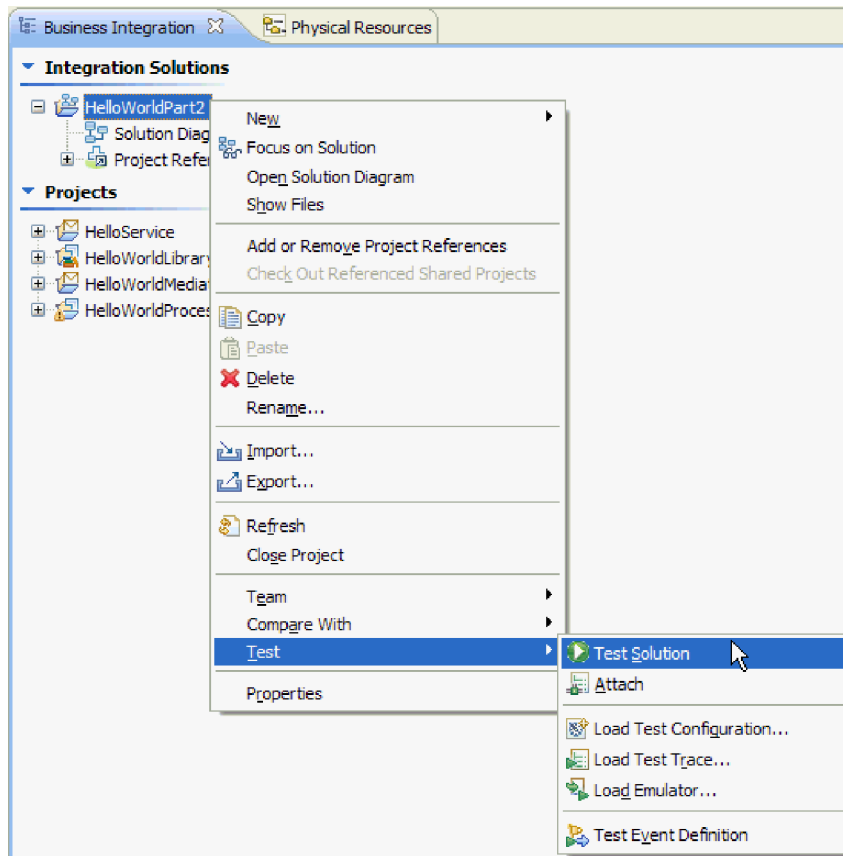
Deploy and test the modules

You must deploy (publish) your module to your test environment server before you can run and test your mediation.

In the Hello World Part 1 sample, you learned how to use the Servers view and the **Add and Remove Programs** dialog box to deploy module applications to the server. You also learned how to use the Build Activities view to subsequently republish the modules to the server whenever it was needed. However, in the Hello World Part 2 sample, you do not use these tools to deploy or republish the modules. Instead, you open the integration test client to perform your testing and the test client performs the deployment and republishing of your modules.

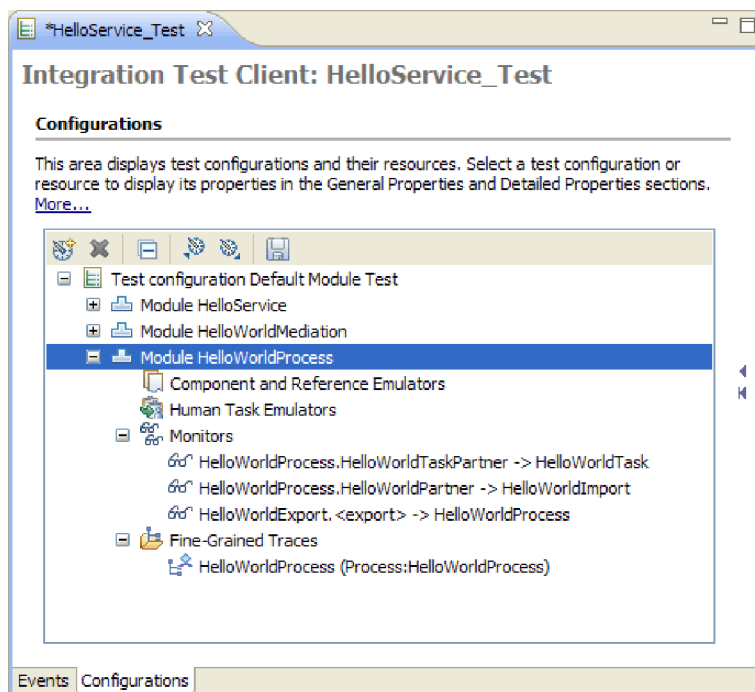
To deploy and test the modules:

1. In the Business Integration view, right-click the **HelloWorldPart2** integration solution and select **Test > Test Solution**, as shown here:



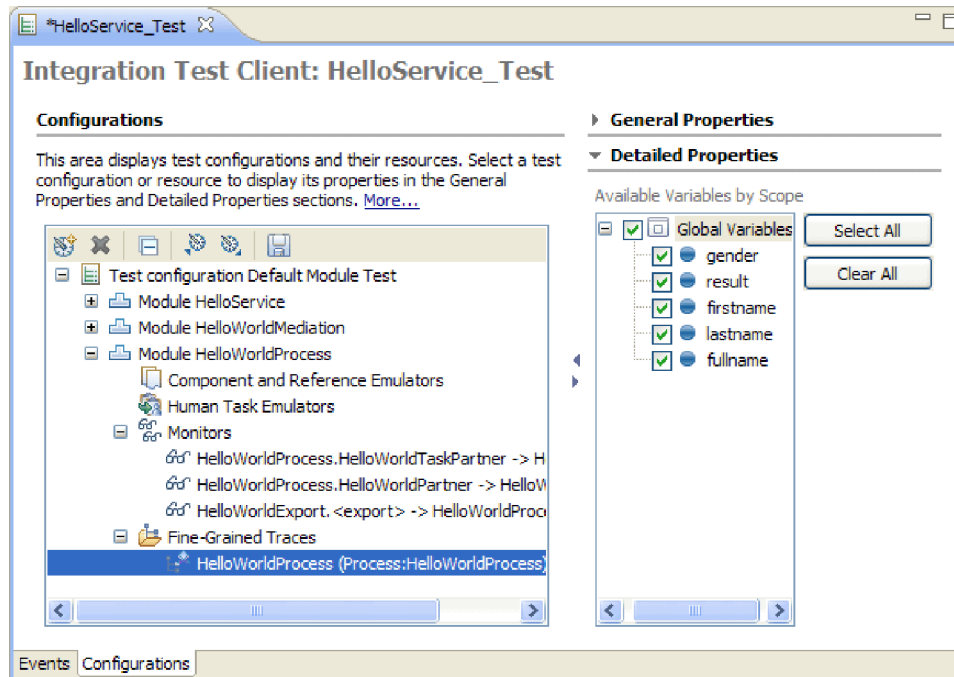
The integration test client opens.

2. In the test client, click the **Configurations** tab. The Configurations page opens and lists all of the modules in your solution. Ensure that only the **HelloWorldProcess** module is expanded, as shown here:



The Configurations page is where you can configure your test session, such as identifying the modules to include in the test, which ensures that the modules are deployed and up-to-date on the server. (By launching the test client from an integration solution, the modules are pre-populated in the test configuration.)

3. You can also configure your test session so that variable data is shown for each event when a process is tested, which can be very helpful. Under the **Fine-Grained Traces** category of the module, select **HelloWorldProcess**. On the right side of the Configurations page, you see all of the variables in the selected business process. Click **Select All**, as shown here:



Another useful capability (which is not explored in this sample) is the emulation of components and imports, which means that you do not actually need to run them during your testing. This capability encompasses the testing of human task components, where you can emulate different users and claim and complete the tasks.

Note: You can also save your integration test client sessions to reuse them at another time.

4. Click the **Events** tab. In the Detailed Properties section, specify the module and component that you want to test: namely, the **HelloWorldProcess** module and the **HelloWorldProcess** process. In the value editor, specify a value of male for the **gender** variable. The Detailed Properties section should look like the following figure:

▼ Detailed Properties

Specify the component, interface, operation, and input parameter values for the Invoke event, then click the Continue icon in the Events area to run the test. [More...](#)


Configuration:	Default Module Test
Module:	HelloWorldProcess
Component:	HelloWorldProcess
Interface:	HelloWorldProcess
Operation:	startHelloWorldProcess

Initial request parameters:

☒ Value editor ☐ XML editor

Name	Type	Value
gender	string	male

To edit values, start typing or press F2.

- At the top of the Events area, click the **Continue** icon . The Deployment Location dialog box opens, as shown here:

Deployment Location

Select Deployment Location per Module

For each module, specify a runtime location where this test will deploy.

Select a deployment location for each module:

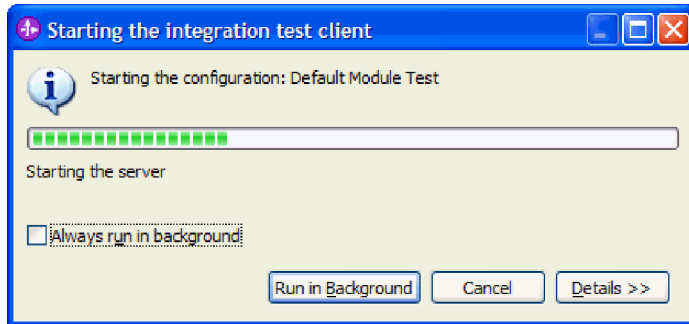
Module	Deployment Location
<input type="checkbox"/> HelloWorldService	WebSphere Process Server v7.0 at localhost
<input type="checkbox"/> HelloWorldMediation	WebSphere Process Server v7.0 at localhost
<input type="checkbox"/> HelloWorldProcess	WebSphere Process Server v7.0 at localhost

☐ Use this location as the default and do not ask again

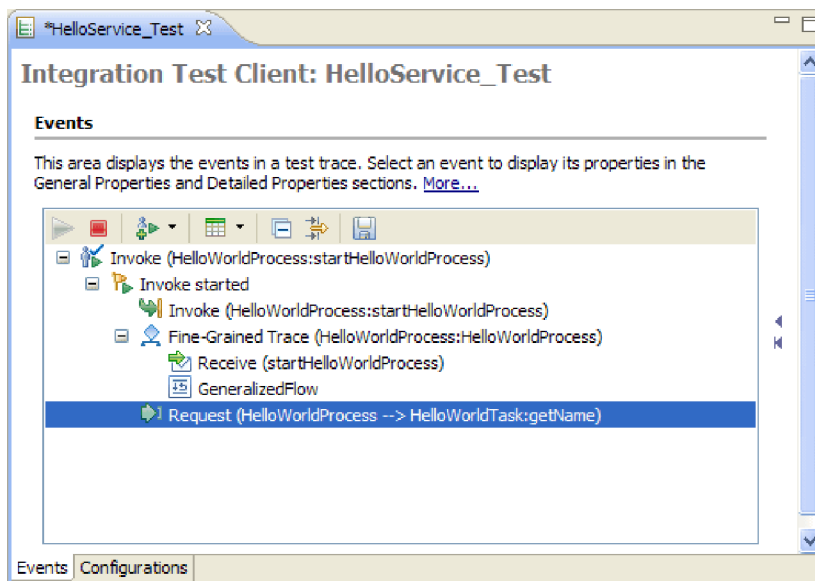
If you have different servers defined, you could choose to deploy your modules to different servers. However, for the purposes of this sample, simply click **Finish** to accept the default server associations. The User Login dialog box opens.

- If you did not change the default user ID and password for the server when you installed it, simply click **OK**. (The default user ID and password are both admin.) However, if you *did* change the default

user ID or password when you installed the server, specify the user ID and password and then click **OK**. The server is started (if required) and all of the modules in the test configuration are deployed or republished as necessary, as shown here:

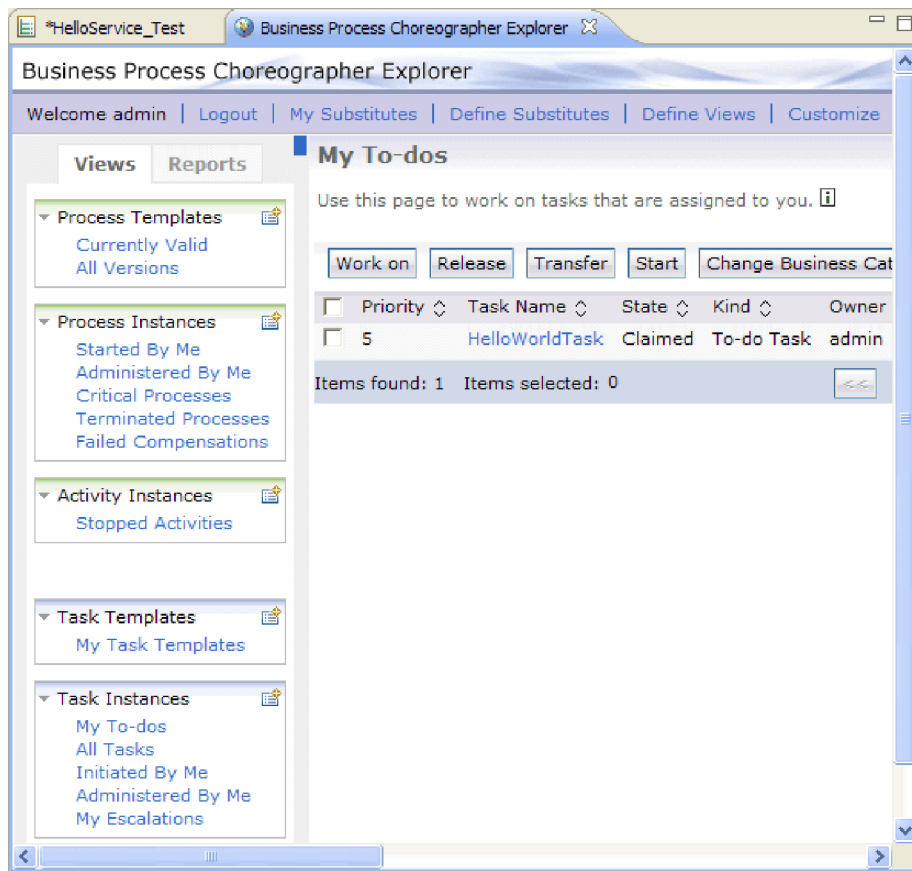


Finally, the actual test begins to run. In the **Events** area of the test client, you can see events that display the execution path through the components in the assembly diagram. You can also see fine-grained trace events that display the execution path through the activities in the business process, as shown in the following figure:



At this point, the execution pauses. The **Request** event (highlighted in the above figure) is appended with the text **(HelloWorldProcess -> HelloWorldTask:getName)**, which indicates that the component **HelloWorldProcess** has invoked the component **HelloWorldTask** through its **getName** operation. The Request event is currently the last event in the Events area, which means that the process is waiting for user input...yours! There is a to-do task that is waiting to be claimed and completed before the test can continue. In the following steps, you will locate and process the to-do task.

7. In the **Servers** view, right-click your server and select **Launch > Business Process Choreographer Explorer**. The Business Process Choreographer Explorer opens in the built-in Web browser.
8. When you are prompted for a user name and password, specify the user name and password that you have been using for administration and for which you are specified as the only potential owner for the human task. By default, the user name and password are both **admin**.
9. Click **Login**. The My To-dos page opens. This page contains a list of to-do tasks for the user name that you used to log into the Business Process Choreographer Explorer, as shown here:



Remember, it is possible for multiple people to see the same to-do task if they are part of the same Potential Owners list. However, once a person claims a to-do task, only that person will be able to see it.

10. Select the check box in front of the **HelloWorldTask** task and then click the **Work on** button to claim the task. The Task Message page opens and displays a form that contains the input parameter data as well as the prompts used for specifying the output parameter data, as shown here:

Task Message

Use this page to provide the data required to complete the task. ⓘ

Task Name: HelloWorldTask

Task Input Message

Form View

gender: male

Task Output Message

Form View

firstName:

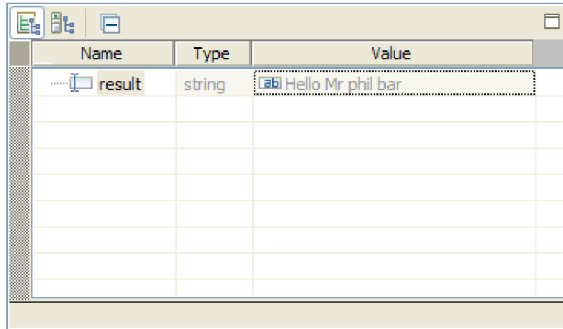
lastName:

11. In the Task Message page, specify phil in the **firstName** field and specify bar in the **lastName** field, then click the **Complete** button to complete your to-do task. The My To-dos page opens again, but the to-do task no longer appears.
12. At the top of the Business Process Choreographer Explorer, click **Logout** and then close the browser window.

Note: This Web user interface is the default one supplied for your convenience. There is also another supplied user interface for human tasks in the Business Space, which you can also launch from the Servers view. As well, you can create your own user interfaces for processing human tasks. This can be done by starting from scratch and using only the business process and human task APIs that IBM supplies, or you can get a head start on your custom user interface by using the module context menu item **Generate Human Task User Interface**. For each task, a default form will be generated if no customized forms are found. You can optionally create and customize the forms for each task in the User Interface section of the human task editor.

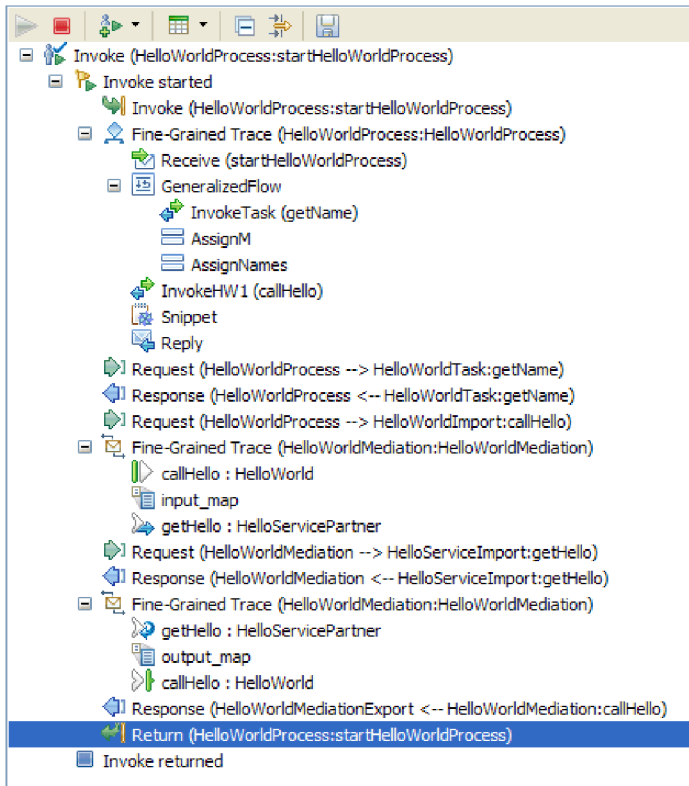
13. In the workbench, click the **HelloService_Test** tab to refocus the integration test client. You will note that the test automatically continued and ran to completion. In the value editor, you can see the contents that were returned for the **result** output variable, as shown in the figure below:

Return parameters:



Name	Type	Value
result	string	Hello Mr phil bar

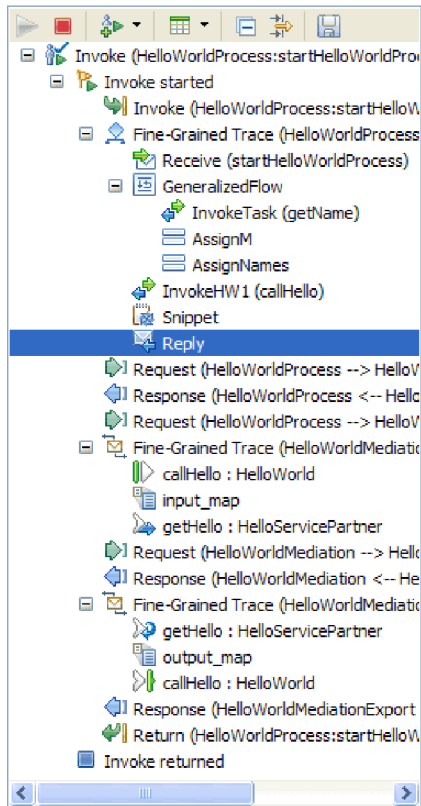
In the **Events** area, you can also see the fine-grained trace event flow for the business process as well as the request and response mediation flows from the mediation module in the Hello World Part 1 sample, as shown in the following figure:



14. In the Events area, click the **Reply** fine-grained trace event to see the contents of the business process variables as they existed at that point in the execution path, as shown in the following figure:

Events

This area displays the events in a test trace. Select an event to display its properties in the General Properties and Detailed Properties sections. [More...](#)



General Properties

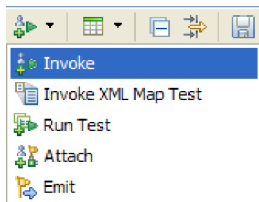
Detailed Properties

Module: [HelloWorldProcess](#)
Component: [HelloWorldProcess](#)
Interface: [HelloWorldProcess](#)
Operation: [startHelloWorldProcess](#)
Process: [HelloWorldProcess](#)
Activity: [Reply](#)

Variables Values

Name	Value
gender	male
result	Hello Mr phil bar
firstname	phil
lastname	bar
fullname	
lastName	bar
title	Mr
firstName	phil

- Optional: If you want, you can perform some additional testing. At the top of the Events area, click the drop-down arrow beside the **Invoke** icon and then select **Invoke**, as shown here:



A new **Invoke** event is displayed in the Events area and the original input data for the earlier test is displayed in the **Initial request parameters** value editor. In the value editor, change **male** to **female** and then click the **Continue** icon to run the test again. You will need to once again launch Business Process Choreographer Explorer to claim the task and you should again specify phil as the first name and bar as the last name for the output parameters. When you complete the task in Business Process Choreographer Explorer, you should see a result of **Hello Ms phil bar** returned in the test client.

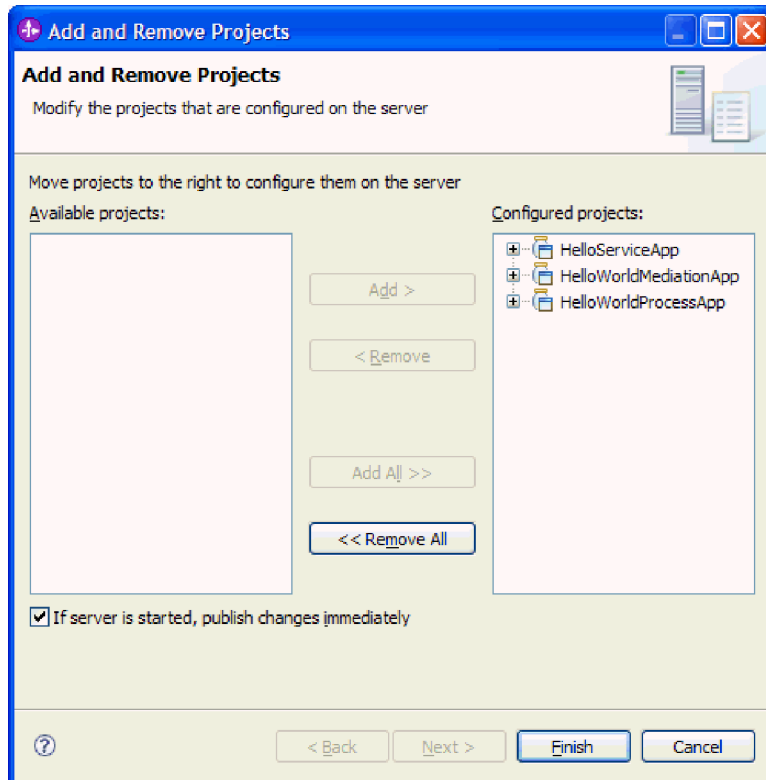
- Use **File > Close All** to close all of the open editors. When a Save Test Trace dialog box asks whether you want to save your changes, click **No**.

Remove the modules from the server

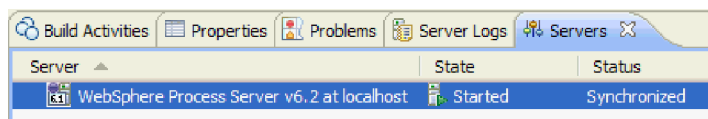
Generally, when you have finished testing a module, you should remove it from the server. This will ensure that the only modules that are deployed to the server are those that you are preparing to test, which will reduce the load on the server and enhance its performance.

To remove the modules from the server:

1. Click the **Servers** tab. The Servers view opens.
2. In the Servers view, right-click **WebSphere Process Server** or **WebSphere Enterprise Service Bus** and select **Add and Remove Projects**. The Add and Remove Projects dialog box opens, as shown in the following figure:



3. Click **Remove All**. The applications are removed from the **Configured projects** list.
4. Click **Finish**. If a dialog box opens and informs you that the project is being removed from the server, click **OK**. The applications no longer appear under the server in the Servers view, as shown here:



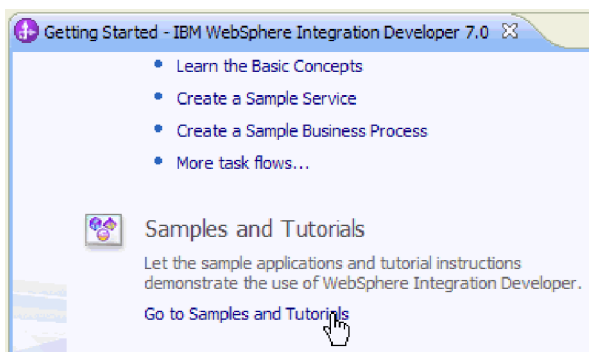
Congratulations! You have completed the Hello World Part 2: Service Components and Web Interfaces sample.

Chapter 4. Import

You can either import a complete ready-made version of the Hello World Part 2: Service Components and Web Interfaces sample, or you can import starter artifacts and build the sample yourself.

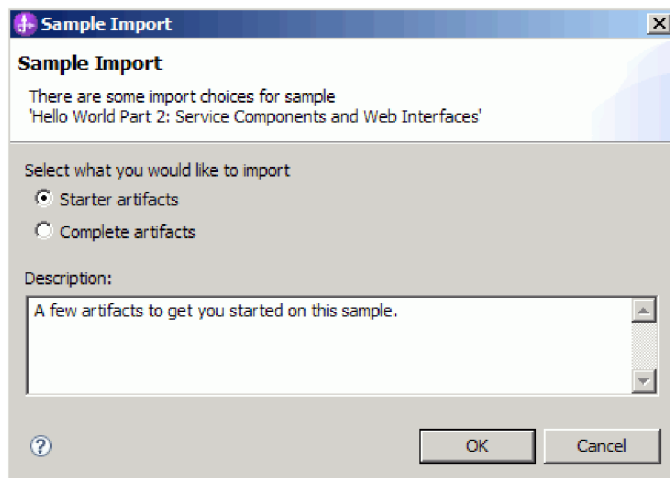
To import the sample:

1. Open WebSphere Integration Developer and select a new workspace.
2. If the Getting Started page is not open in the workspace, select **Help > Getting Started > IBM WebSphere Integration Developer**. The Getting Started page opens.
3. On the **Getting Started - IBM WebSphere Integration Developer** page, select the **Go to Samples and Tutorials** link, as shown in the following figure:



The Samples and Tutorials page opens.

4. Under the **Hello World Part 2: Service Components and Web Interfaces** section, click the **Import** link. You are presented with two options, as shown here:



5. If you want to build the sample yourself, select **Starter artifacts** and click **OK**. You should now see the following three projects in your Business Integration view:
 - HelloService
 - HelloWorldLibrary
 - HelloWorldMediation

Open the "Build it yourself" instructions and begin with the topic "Create an integration solution".

6. If you want to import the complete ready-made sample, select the option **Complete artifacts** and click **OK**. You will see the following projects in the Business Integration view:

- A mediation module named HelloService.
- A mediation module named HelloWorldMediation.
- A library named HelloWorldLibrary.
- A module named HelloWorldProcess.

Instructions for running the sample are found in the topic "Run the sample".

Notices

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

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this documentation 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 documentation. The furnishing of this documentation does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: 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 OR CONDITIONS OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states 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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*Intellectual Property Dept. for WebSphere Integration Developer
IBM Canada Ltd.
8200 Warden Avenue
Markham, Ontario L6G 1C7
Canada*

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this documentation and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2000, 2009. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

IBM, IBM Logo, WebSphere, Rational, DB2, Universal Database DB2, Tivoli, Lotus, Passport Advantage, developerWorks, Redbooks, CICS, z/OS, and IMS are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both.

UNIX is a registered trademark of The Open Group in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

Adobe is either a registered trademark or trademark of Adobe Systems Incorporated in the United States, other countries, or both.

Other company, product and service names may be trademarks or service marks of others.

Terms of use

Permissions for the use of publications is granted subject to the following terms and conditions.

Personal Use: You may reproduce these publications for your personal, non commercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM®.

Commercial Use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

© Copyright IBM Corporation 2005, 2009. All Rights Reserved.

Readers' Comments — We'd Like to Hear from You

Integration Developer

Version 7.0

Hello World Part 2: Service Components and Web Interfaces

Version 7 Release 0

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Send your comments to the address on the reverse side of this form.

If you would like a response from IBM, please fill in the following information:

Name

Address

Company or Organization

Phone No.

E-mail address



Cut or Fold
Along Line

Fold and Tape

Please do not staple

Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Canada Ltd. Laboratory
Information Development for WebSphere Integration
Developer
8200 Warden Avenue
Markham, Ontario
Canada L6G 1C7

Fold and Tape

Please do not staple

Fold and Tape

Cut or Fold
Along Line