HP Unified Functional Testing

For the Windows ® operating systems

Software Version: 12.00

API Testing Tutorial

Document Release Date: March 2014

Software Release Date: March 2014

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe® and Acrobat® are trademarks of Adobe Systems Incorporated.

Apple and the Apple logo are trademarks of Apple Computer, Inc., registered in the U.S. and other countries.

Google™ and Google Maps™ are trademarks of Google Inc

Intel® and Pentium® are trademarks of Intel Corporation in the U.S. and other countries.

Microsoft®, Windows®, Windows® XP, and Windows Vista ® are U.S. registered trademarks of Microsoft Corporation.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document, is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to: www.hp.com/go/livenetwork. This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to: **http://h20229.www2.hp.com/passport-registration.html**

Or click the **New users - please register** link on the HP Passport login page.

Support

Visit the HP Software Support Online web site at: **http://www.hp.com/go/hpsoftwaresupport**

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

HP Software Solutions Now accesses the HPSW Solution and Integration Portal Web site. This site enables you to explore HP Product Solutions to meet your business needs, includes a full list of Integrations between HP Products, as well as a listing of ITIL Processes. The URL for this Web site is **http://h20230.www2.hp.com/sc/solutions/index.jsp**

Contents

About the Tutorial for Testing

The UFT Tutorial for API Testing is a self-paced printable guide designed to lead you through the process of creating tests for Web services, REST services, and other GUI-less applications.

After completing the tutorial, you can apply the skills you have learned to testing the GUI-less layer of your own application or Web site.

Note: To learn more about creating and running GUI tests, see the UFT Tutorial for GUI Testing, available from the `< UFT installation folder>\help` directory.

Tutorial Audience and Scope

This tutorial is intended for users who are new to UFT. No prior knowledge of UFT or Service Test is required. A general understanding of testing concepts and functional testing processes may be helpful, but is not mandatory. UFT enables you to create API tests and business process tests. This tutorial reviews topics related to API testing only. While performing the lessons in this tutorial, you may notice unfamiliar menu items or other UFT GUI elements that are not described in the tutorial. These may be relevant for BPT or GUI testing only, and are not relevant for API testing at all. For details about these elements, see the *HP Unified Functional Testing User Guide*.

Note: This tutorial refers to file system paths that are relevant for Windows 7 operating systems. The paths in other operating systems may be slightly different.

HP UFT Guides and References

The following tables provide a list of the UFT guides, online help and references:

Note: To check for recent updates of any of the guides below, visit the HP Software Product Manuals Web site (http://h20230.www2.hp.com/selfsolve/manuals).

Getting started

Reference	Description
What's New?	Describes the newest features in the latest version of Unified Functional Testing. You can also access the **What's New** from the **Unified Functional Testing Help** menu.
Product Movies	Click the link or select **Help > Product Feature Movies** to view short movies that demonstrate the main product features.

Reference	Description
Readme	Provides last-minute news and information about Unified Functional Testing. For the latest readme file, go to the HP Software Manuals Web site (requires an HP Passport) at http://support.openview.hp.com/selfsolve/manuals.
UFT PAM	The Product Availability Matrix (PAM) provides current information about technologies and integrations supported for this version of UFT.
GUI Testing Tutorial	The GUI Testing Tutorial is a self-paced printable guide, designed to lead you through the process of creating GUI tests and familiarize you with the testing environment.
API Testing Tutorial	The API Testing Tutorial is a self-paced printable guide, designed to lead you through the process of creating API tests in the Windows environment.

PDF guides 

Guide	Description
UFT User Guide	The HP Unified Functional Testing User Guide describes how to use UFT to test your applications. It provides step-by-step instructions to help you create, debug, and run tests, and report defects detected during the testing process.
Run Results Viewer	The HP Run Results Viewer User Guide explains how to use the Run Results Viewer to interpret and use the test results from your GUI or API tests.
UFT Installation Guide	The HP Unified Functional Testing Installation Guide provides a complete, step-by-step instructions on how to install and set up UFT on a standalone computer.
UFT QuickStart	The UFT Installation QuickStart Sheet explains the steps to perform a basic installation of UFT.
License Server Installation Guide	The Concurrent License Server Installation Guide provides the information you need to install and maintain the HP Functional Testing Concurrent License Server.
UFT Add-ins Guide	The HP Unified Functional Testing Add-ins Guide explains how to set up support for UFT add-ins and standard Windows testing support. Add-ins enable you to test any supported environment using GUI tests and business components.

References

Links to the references are available from the UFT online help home page.

Reference	Description
Object Model Reference	The Object Model Reference for GUI Testing includes a description, a list of methods and properties, syntax, examples, and identification properties for each UFT test object.
VBScript Reference	Microsoft's Visual Basic Scripting documentation that describes objects, methods, properties, functions, and other elements that can be used when writing VBScript scripts.
Automation Object Model Reference	List the objects, methods, and properties that enable you to control UFT from within another application.
Object Repository Automation Reference	Describes the objects that enable you to manipulate UFT shared object repositories and their contents from outside of UFT.
Run Results Schema Reference	Provides details about the structure of the Run Results XML schema, and describes the elements and attributes used in the its XML reports.
Test Object Schema Reference	A reference describing the elements and attributes available for creating test object configuration XML content, for use when creating UFT extensibility projects.
Object Repository Schema Reference	Describes the elements and complex types defined for the object repository schema.

Additional Online Resources

The following additional online resources are available from the Unified Functional Testing Help menu:

Resource	Description
HP Software Support Online	Opens the HP Software Support Web site. This site enables you to browse the HP Software Self-solve knowledge base. You can also post to and search user discussion forums, submit support requests, download patches and updated documentation, and more. Choose **Help > HP Software Support**. The URL for this Web site www.hp.com/go/hpsoftwaresupport. • Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. • To find more information about access levels, go to: http://h20230.www2.hp.com/new_access_levels.jsp • To register for an HP Passport user ID, go to: http://h20229.www2.hp.com/passport-registration.html
Testing Forums	Opens the testing forums for GUI Testing, API Testing, and BPT Testing forums. where you can interact with other users of UFT and discuss topics related to GUI Testing, API Testing, and BPT. The URLs for these sites are: • GUI Testing: http://h30499.www3.hp.com/t5/Unified-Functional-Testing/bd-p/sws-Fun_TEST_SF • API Testing: http://h30499.www3.hp.com/t5/Service-Test-Support-and-News/bd-p/sws-Serv_TEST_SF • BPT: http://h30499.www3.hp.com/t5/Business-Process-Validation/bd-p/sws-BPT_SF
UFT Product Page	Opens the HP Unified Functional Testing product page, with information and related links about UFT.
Troubleshooting & Knowledge Base	Opens the Troubleshooting page on the HP Software Support Web site where you can search the HP Software Self-solve knowledge base. Choose **Help > Knowledge Base** or **Help > Troubleshooting**. The URL for the troubleshooting Web site is http://h20230.www2.hp.com/troubleshooting.jsp.
HP Software Community	Opens the HP IT Experts Community site, where you can interact with other HP software users, read articles and blogs on HP software and access downloads of other software products.

Resource	Description
HP Manuals Site	Opens the HP Software Product Manuals Web site, where you can search for the most up-to-date documentation for a selected HP Software product. The URL for this Web site is http://support.openview.hp.com/selfsolve/manuals (requires an HP Passport).
What's New	Opens the UFT What's New Help, describing the new features and enhancements in this version of UFT.
Product Movies	Opens a page on HPLN (HP Live Networks) displaying a list of all product movies.
HP Software Web site	Opens the HP Software Web site. This site provides you with the most up-to-date information on HP Software products. This includes new software releases, seminars and trade shows, customer support, and more. The URL for this Web site is www.hp.com/go/software.

You can access the following sample applications from the **Start** menu. These applications are the basis for many examples in this guide:

- Mercury Tours sample Web site. The URL for this Web site is http://newtours.demoaut.com.
- Mercury Flight application. To access from the Start menu, select **All Programs > HP Software > HP Unified Functional Testing > Sample Applications > Flight API / Flight GUI**.

Chapter 1: Introducing HP Unified Functional Testing - API Testing

HP Unified Functional Testingfor API testing contains an extensible framework for the construction and execution of functional tests of GUI-less applications or the non-GUI parts of an application.. This document describes how to get started with HP UFT API testing and create your first tests. It also introduces the major features in the product and how to incorporate them into your tests.

Note: It is recommended to work with an soft copy of this tutorial because there are sections in which you will be asked to copy and paste test information into UFT.

If you would like, you can open a PDF of this tutorial from your **<Unified Functional Testing installation folder>\help folder**.

You can access the HTML version of this tutorial, by selecting **Help > Unified Functional Testing Tutorial**.

This chapter contains the following sections:

Why Should You Automate API Testing?

Automated API Testing is a discipline that leverages products and processes to reduce the risks of application upgrades or deployment of new services. At its core, automated testing is about applying production workloads to predeployment systems while simultaneously measuring system performance and end-user experience. A well-constructed performance test answers questions such as:

- Does the service/application respond quickly enough for the intended users?
- Will the application's server respond with the correct values?
- How will the service/application handle exceptions and illegal values?
- Is the service/application stable under expected and unexpected user loads?

By answering these questions, you can design a test more effectively. An effective automated testing process helps you make more informed release decisions, reduces system downtime, and prevents availability problems.

Benefits of Automated API Testing

If you have ever tested applications manually, you are aware of the drawbacks of manual testing.

Manual testing is time-consuming and tedious, and requires a heavy investment in human resources. Worst of all, time constraints often make it impossible to manually test every feature thoroughly before the application is released. This leaves you wondering whether serious bugs have gone undetected.

Automated API testing with UFT addresses these problems by dramatically speeding up the testing process. You can create tests that check all aspects of your application, and then run these tests every time your application changes.

As UFT runs, it simulates the application running by performing application processes in your application. However, UFT does this faster than any human user.

Benefits of Automated Testing	
Fast	Automated tests are significantly faster than human users.
Reliable	Tests perform precisely the same operations each time they are run, thereby eliminating human error.
Repeatable	You can test how the application reacts after repeated performance of the same operations.
Programmable	You can program sophisticated tests that reveal hidden information.
Comprehensive	You can build a set of tests that covers every feature in your application.
Reusable	You can reuse tests even if the application being tested changes.

Testing Process

The UFT API testing process consists of the following main phases:

1. ## Analyzing your application

   The first step in planning your test is analyzing your application to determine your testing needs.

   - **What application processes or technologies does your application use?** You will need to create steps that target the specific processes your application runs or the specific technologies that your application uses.
   - **Does my application use standard application processes or custom-designed services?** Depending on what processes your application uses to perform its tasks, you can use the out-of-the box activities provided with UFT in the Toolbox pane or import/create custom activities.

2. ## Preparing the testing infrastructure

   Based on your testing needs, you must determine what resources are required.

   Examples of such resources include **WSDL** or **WADL** files, **REST Services**, external data sources, or virtualization projects for your service calls. You need to import or create these resources in UFT.

   You also need to configure UFT settings so that UFT will perform any additional tasks you may need, such as displaying a results report each time you run a test, enabling or disabling debugging for the test run, or configuring SAP server connection information.

3. ## Building your tests and adding steps to each test.

   After the testing infrastructure is ready, you can begin building your tests.

   You can create one or more empty tests, add test steps to these tests, and define the input, output, and checkpoint properties for these steps.

   You can also add all your tests (or components) to a single solution. A solution enables you to store, manage, and edit any related tests together, without having to close one test before opening another.

   You may also want to configure test-specific properties at this point.

4. ## Enhancing your test

You can enhance your tests in a number of ways:

- Validate test steps by selecting **checkpoint properties** and entering expected values for the step properties
- Broaden the scope of your test and test how your application performs the same activities with multiple sets of data by replacing fixed values with **parameters**.
- Add additional functionality to your test steps with **custom code** activities, **event handlers**, or custom activities created using UFT's Activity Wizard.

This tutorial includes a step for creating an event handler. For additional details about custom code, see the *HP Unified Functional Testing User Guide*.

5. ## Debugging, running, and analyzing your test

   You can debug your test using UFT's debugging functionality to ensure that it operates smoothly and without interruption. After the test is working correct, you run it to check the behavior of your application. While running, UFT performs each step of the test in your application.

6. ## Reporting defects

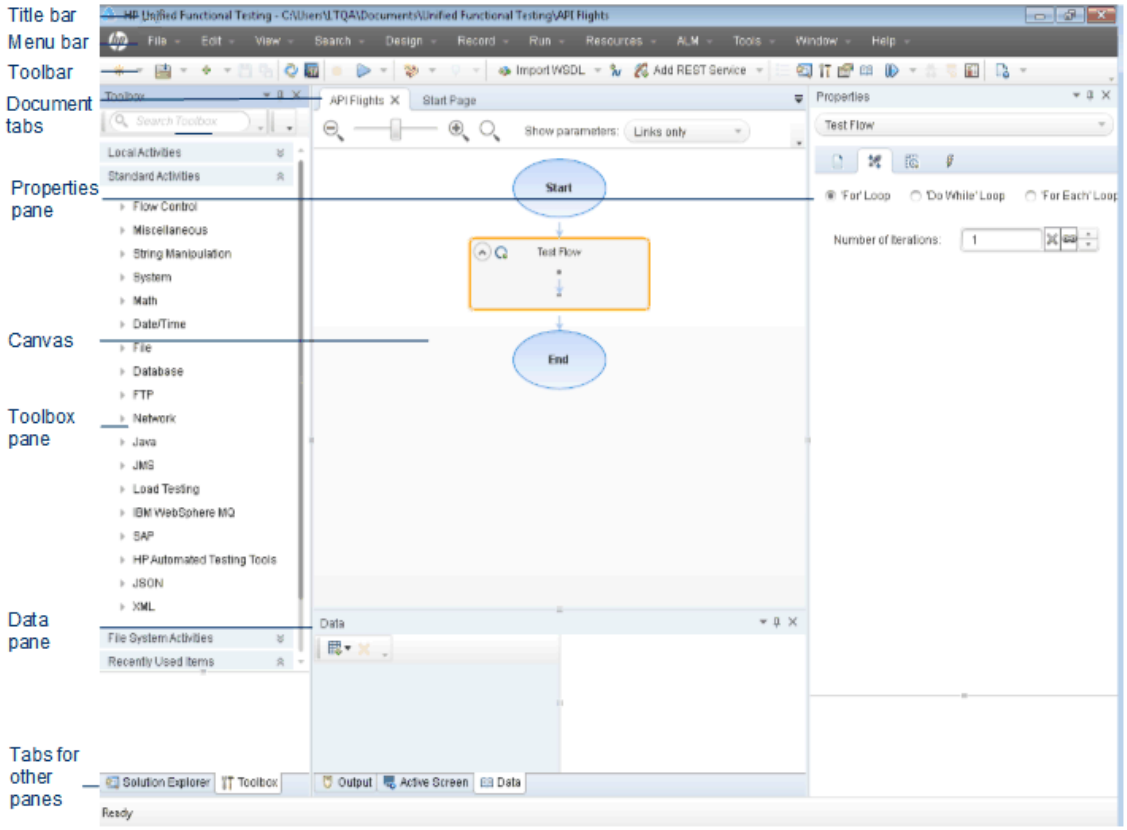
   If you have ALM installed, you can report defects you discover to a database. ALM is the HP test management solution.

   For details about working with ALM, see the *HP Application Lifecycle Management User Guide*. For details about using UFT with ALM, see the section on ALM integration in the *HP Unified Functional Testing User Guide*.

UFT Window

Before you begin creating your test, familiarize yourself with the main UFT window.

The image below shows a UFT window as it would appear right after you create a test, with the test flow shown in the canvas, and with the toolbar, Toolbox pane, Data pane, and Properties pane displayed.



Take a few minutes to explore the components of the UFT window. You can then continue to analyze the application you will be testing in this tutorial.

The UFT window displays the following elements:

Document Types

UFT displays open documents in the document pane. Use the document tabs located just below the toolbar to navigate to open documents and bring them into focus.

The document pane can display the following types of files:

- **Tests/Business Components.** You can create, view, and modify your test or business component in the canvas, which enables you to edit the flow of your test or component steps.
- **User Code Files.** Enables you to enter your custom code as an event handler for an existing test step (in the `TestUserCode.cs` file) or in other documents you import into UFT.
- **Start Page.** Welcomes you to UFT and provides links to recent files, descriptions of new features, product forums, and other support links. You can use the shortcut buttons to open new and existing documents.
- **Internal Browser pages.** Enables you to open internet pages for forums and other product related pages, such as those accessible from the Start Page or the Help menu.

Toolbars and Menus

In addition to the document pane, the UFT window contains the following key elements:

- **Title bar.** Displays the path of the current test.
- **Menu bar.** Displays menus of UFT commands.

  Note: Some menu options that relate to GUI testing functionality are disabled when working with an API test.

- **UFT toolbar.** Contains commonly used buttons to assist you in designing your testing documents.

  Note: Some toolbar buttons that relate to GUI testing functionality are disabled when working with an API test.

Panes

The main UFT window displays the following panes:

Name	Toolbar Button	Description	Default Location

Solution Explorer		Displays all tests, components, and user code files contained in the currently open solution, and enables you to manage these resources.	A tab on the left side of the UFT window. To display: • Select **View > Solution Explorer**. • Click the **Solution Explorer** button in the toolbar.
Toolbox		Displays all the activities available to use in your test, and enables you to drag and drop these activities on the canvas.	A tab on the left side of the UFT window. To display: • Select **View > Toolbox**. • Click the **Toolbox** button in the toolbar.
Document Pane	N/A	Displays all open documents. Each document has a tab that you can click to bring the document into focus.	An unlabeled pane in the center of the UFT window. Each document tab is labeled with the document name. You can display a document in the document pane by double-clicking the document's name in the Solution Explorer.

Properties		Displays all properties for the selected test step or test flow or the properties of the selected data source (in the Data pane).	A pane on the right side of the UFT window. To display: • Select **View > Properties**. • Click the **Properties** button in the toolbar. • Double-click a step in the canvas. • Right-click a step in the canvas and select **Properties**.
Data		Assists you in parameterizing your test.	A tab at the bottom of the UFT window. To display: • Select **View > Data**. • Click the **Data** button in the toolbar.
Output	N/A	Displays the compilation information for your test when running or importing custom activities.	A tab at the bottom of the UFT window. To display, select **View > Output**.
Errors	N/A	Displays a list of missing property values for test steps, missing references for your current test, or syntax errors found in your custom code.	A tab at the bottom of the UFT window. To display, select **View > Errors**.

Debug		Assists you in debugging your test. There are multiple available debug panes: the **Breakpoints** pane, **Call Stack** pane, **Local Variables** pane, **Console**, **Watch** pane, **Threads** pane, and the **Loaded Modules** pane.	Tabs at the bottom of the UFT window. These panes are not displayed by default. To display: • Select **View > Debug** • Click the **Debug** button in the toolbar, and then select from the individual debug pane from the drop-down list.
Tasks	N/A	Displays and enables you to manage the tasks defined for the current test. Displays the TODO comments included in your test's custom code.	A tab at the bottom of the UFT window. To display, select **View > Tasks**.
Search Results	N/A	Displays all occurrences of the search criteria you define in the Find dialog box or using other Search menu items.	A tab at the bottom of the UFT window. To display: • Select **View > Search Results**. • Perform a search.
Bookmarks	N/A	Displays the location of bookmarks in your code documents, and enables you to navigate to these bookmarks.	A tab at the bottom of the UFT window. To display, select **View > Bookmarks**.
Run Step Results	N/A	Displays the run results for a test step run from the canvas.	A tab at the bottom of the UFT window. To display: • Select **View > Run Step Results**. • Run a step by right-clicking a test step and selecting **Run Step**.

Accessing UFT in Windows 8 Operating Systems

UFT applications and files that were accessible from the **Start** menu in previous versions of Windows are accessible in Windows 8 from the **Start** screen or the **Apps** screen.

- **Applications (.exe files).** You can access UFT applications in Windows 8 directly from the **Start** screen. For example, to start UFT, double-click the **HP Unified Functional Testing** shortcut  .

  Other examples of applications accessible from the **Start** screen include:

  - The Run Results Viewer
  - All UFT tools, such as the Password Encoder and the License Validation Utility
  - The API testing sample Flight applications

- **Non-program files.** You can access documentation and the link to the Mercury Tours Website from the **Apps** screen.

Note: By default, the Start and Apps screens on Windows 8 are set to open Internet Explorer in Metro Mode. However, if User Account Control is turned off on your computer, Windows 8 will not open Internet Explorer in Metro mode. Therefore, if you try to open an HTML shortcut from the Start or Apps screen, such as the UFT Help or Readme file, an error will be displayed.

To solve this, you can change the default behavior of Internet Explorer so that it never opens in Metro mode. In the **Internet Properties** dialog box > **Programs** tab, select **Always in Internet Explorer on the desktop** for the in the **Choose how you open links** option. For more details, see http://support.microsoft.com/kb/2736601 and http://blogs.msdn.com/b/ie/archive/2012/03/26/launch-options-for-internet-explorer-10-on-windows-8.aspx.

Where to Go From Here

Now that you have invoked the application, you can begin creating tests for your headless applications using the sample API Flights application. The following lessons will walk you through the process of creating a test for basic activities, Web, and REST services.

Chapter 2: Analyzing Your Application and Preparing Your Test

"Introducing HP Unified Functional Testing - API Testing" on page 10 gave you an overview of automated API testing and UFT.

In this lesson, you will analyze an application to determine what needs to be tested.

This lesson includes:

Getting to Know Your Application

Before you begin creating tests, you must determine exactly what you want to test in your application. To do this, you need to analyze your application in terms of its application processes - the distinct activities that the application performs in order to complete a specific task.

For the purpose of this tutorial, you need to become familiar with the Flight API Application. The Flight API application is a sample service-based flight reservation application that works with a flight reservation database. Using this sample application, you can retrieve flights for specific destinations, create customer orders, update reservations, or delete them. The Flight API application is available both as a Web Service or a REST Service.

For details about the service's methods and operations, type `help` in the Flight API application's command prompt window.

Note: You must have administrator privileges to run the sample **API Flights** application. If you are working as a non-administrator user, the application prompts you to run the application as an administrator.

As you think about how to plan your test, consider the following:

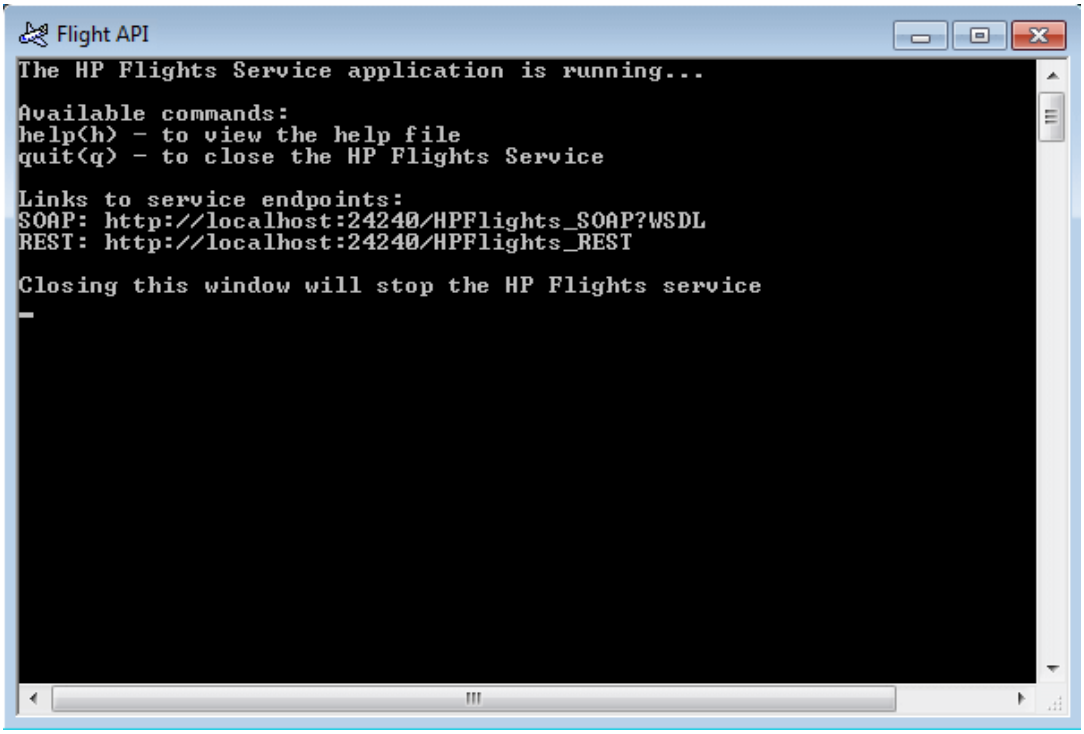
- **How is the application organized?** Are there separate application processes for each application task? What are these processes? What are the expected outcomes for each of the application processes?
- **Are there specific operations that are repeated in multiple processes/activities?** One example of such a process is connecting to a login database to verify a user's credentials before performing application activities. Think of these operations as "reusable" parts.
- **What business processes need to be tested?** UFT provides a number of API testing activities technologies. However, if your application uses custom activities not supported out-of-the-box by UFT, you need to import or create these activities inside UFT.

Explore the Flight API Application

The first step is to invoke the sample flight application so that it will be available for your test.

1. Make sure you have administrator privileges. These are required by Windows to run the sample HP Flights service.

2. Select **Start > (All) Programs > HP Software > HP Unified Functional Testing > Sample Applications > Flight API > Sample Application**. A Command window opens indicating that the application is available.



Note: When working in Windows 8 or Windows Server 2012, you can access UFT and UFT tools directly from the **Start** screen. For more information about working with UFT in Windows 8, see the *HP Unified Functional Testing User Guide*.

3. If the window issues a message that the default port 24240 is unavailable, edit the `<installation_directory>SampleApplication\HPFlights_Service.exe.config` file in a text editor. In the **appSettings** section, replace the 24240 port key with a valid one.

4. Type `help` in the command window to view the methods included in the application.

   As you explore the list of methods included in the application, note the property details provided for each method. You will need this data later in this tutorial to provide property values for these

methods.

5. Minimize the sample application's Command window. Do not close the Command window, as this will stop the service.

You are now ready to use this tutorial to create tests on the Flight API application using UFT. Continue with "Building a Simple Test" on page 26 to create your test.

Create a New Solution and Test

In this exercise, you will create anew solution and test for the Flight API that you explored in the "Explore the Flight API Application" on page 22.

The solution you create in this exercise will be used for the other tests created as part of subsequent lessons.

1. **Start UFT.**

   If UFT is not currently open:

   - Double-click the UFT icon on your desktop.
   - Select **Start > All Programs > HP Software > HP Unified Functional Testing**

   > **Note:** When the Add-in Manager is displayed, click **OK** to continue. The Add-in Manager is relevant for GUI testing only.

   The UFT splash screen is displayed while UFT loads. This may take a few seconds.

2. **Explore the Start Page.**

   The Start Page provides links to recent files, information about new features in this version of UFT, as well as links to helpful support and community forums. In the top-right corner of the page, you can define options for displaying and closing the Start Page.

   a. If it is not already displayed, in the Document pane, click the Start Page tab.

   b. In the top-right corner of the Start Page, select the **Options** drop-down arrow, and then select **Close Start Page after test loads**. The Start Page will close automatically after you create a test.

3. **Create a new test and solution.**

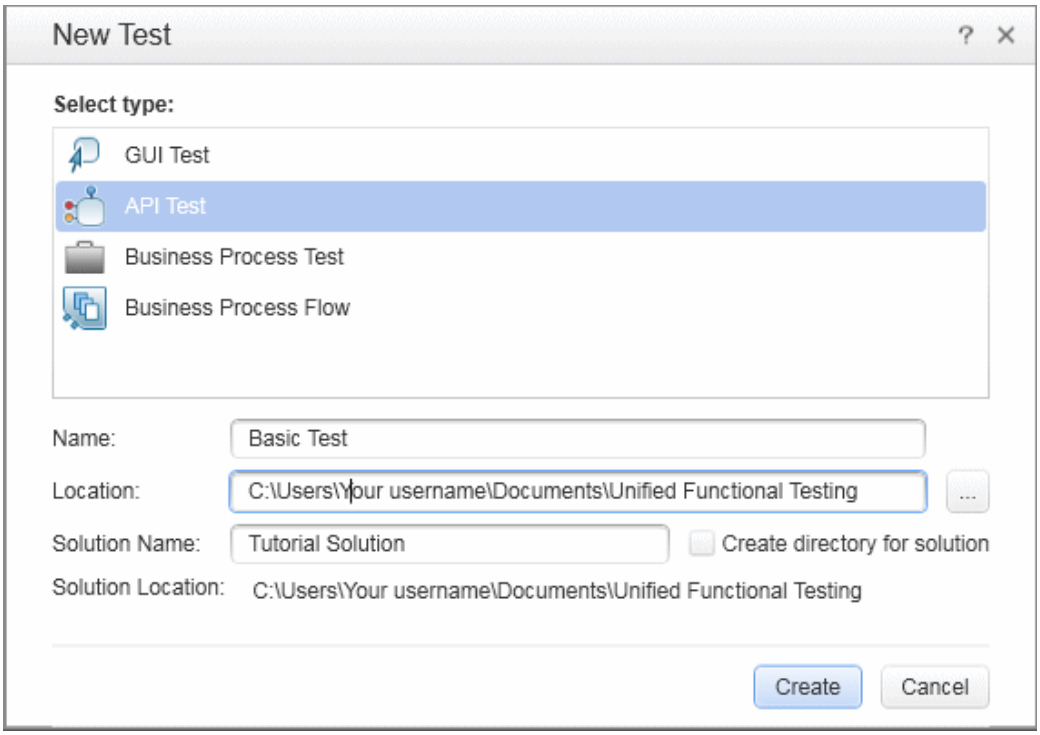
   Click the **New** button  . By default, UFT assumes that you want to create a new test, and the New Test dialog box opens.

   a. Select **API Test**. Populate the fields as follows:

      - **Name:** Enter Basic Test.
      - **Location:** Tests are automatically saved at **C:\%HOMEPATH%\My Documents\Unified Functional Testing**, and you do not need to modify this path. An example of a default test location is **C:\Documents and Settings\<user name>\My Documents\Unified Functional Testing**
      - **Solution Name:** `Tutorial`

b. Enter the solution name.

In the Solution Name field, enter the name of your solution. By default, UFT saves the solution in the same directory as the folder containing your test. If you want to create an additional directory for solution items, select the Create directory for solution option.



c. Click **Create**. A blank test opens, with an empty test flow in the canvas.

The test name (`Basic Test`) and path are displayed in the title bar of the main UFT window.

In the Solution Explorer, you can see that the test is loaded as part of the **Tutorial Solution** solution. Later in this tutorial, you will add additional tests to this solution.

If the Solution Explorer is hidden, click the Solution Explorer button  or select **View > Solution Explorer** to display it.

You are now ready to begin adding steps to your test.

Chapter 3: Building a Simple Test

In "Introducing HP Unified Functional Testing - API Testing" on page 10 you learned about the Flight API application and determined what needs to be tested. You then created a solution, and a test.

This lesson will guide you through the steps of creating tests with standard API activities.

This lesson contains the following sections:

Creating Test Steps for Your Test - Overview

After analyzing your application and planning what you need to test, you need to create test steps for your test. You create test steps by dragging available activities from the Toolbox pane to the canvas to make a test flow.

UFT has two different types of activities for use in your test:

- **Standard API activities**

  UFT supports a number of standard API activities that model common application processes, including:

  - **Flow Control** activities, such as **Wait**, **Break**, and **Conditional** steps.
  - **String Manipulation** activities, such as **Concatenate String** and **Replace String**.
  - **File** system activities for processes involving file system processes
  - **Database** activities, for your application's interaction with a database
  - **FTP** activities for application processes that involve using FTP (File Transfer Protocol)
  - **Network** activities, such as **HTTP Request** and **SOAP Request**
  - **JSON** and **XML** string activities for application processes that require conversion of XML and JSON
  - **Math** and **Date/Time** activities
  - Other **Miscellaneous** activities, including **Custom Code** activities, **Run** and **End** program activities, and **Report** activities.

  In addition, there are various common activities directed at testing application processes using specific technologies, including:

  - A **Call Java Class** activity to use Java-based application processes
  - **JMS** (Java Message Service) activities
  - **IBM Websphere MQ** activities
  - **SAP** activities to access an SAP iDOC or RFC from a SAP server
  - **Load Testing** activities to enable your test to run in HP LoadRunner
  - **HP Automated Testing Tools** activities, which enable you to call a GUI test or action, API test or action, or Virtual User Generator script from UFT, QuickTest Professional, Service Test, or LoadRunner to use as part of your test.

- **Custom activities**

  If the standard API activities do not suit your testing needs, you can also create or import custom activities into your test. You can import a number of different types of custom activities:

  - **Web Service methods.** These activities are stored in a WSDL file that you import into UFT.
  - **REST Service methods.** These activities are created using UFT's REST Service editor and then used in your test.
  - **Web Application methods.** These activities are stored in a WADL file that you import into UFT.
  - **User-designed activities.** These activities are user-created using UFT's Activity Wizard, and then imported into a test.
  - **.NET Assembly activities.** These activities are created when you import a .NET assembly in a test.

Using these activities, you can create test steps for many different types of application processes.

Creating Test Steps

You create test steps by dragging activities from the **Toolbox** pane into the canvas and setting the step properties in the Properties pane.

In this section you will create a simple test step to illustrate the use of the Toolbox and Properties panes.

1. **Locate the Replace String activity.**

   a. Click the Toolbox tab to display the Toolbox pane.

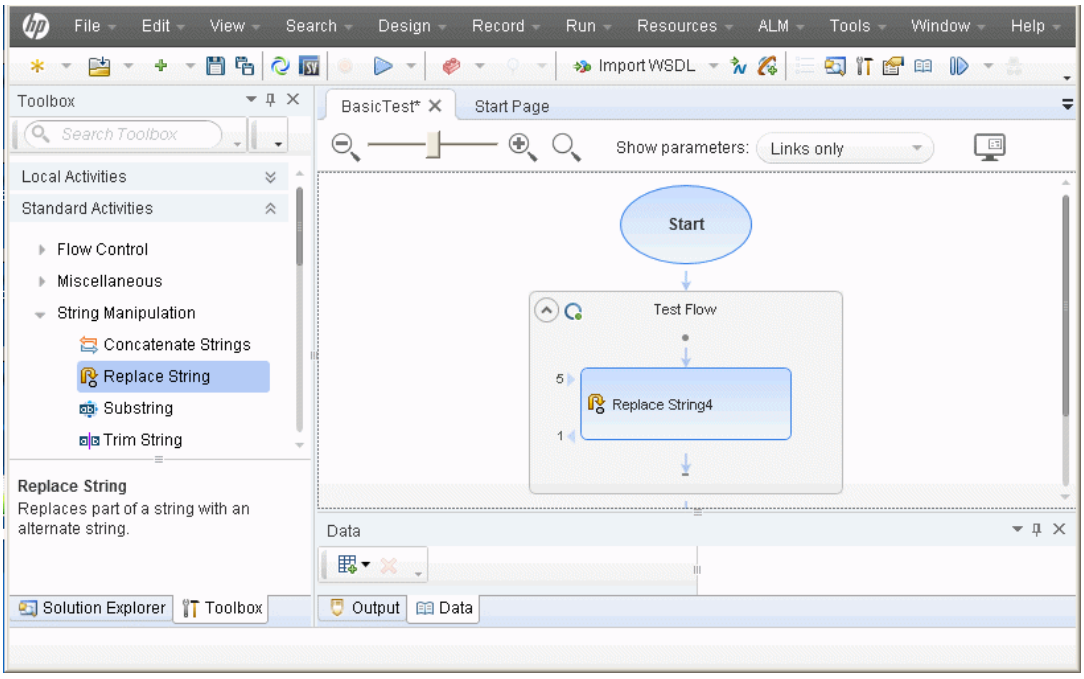
   b. In the Toolbox pane, expand the **String Manipulation** category and find the **Replace String** string activity.

      This activity searches for a specified text string and replaces the string with alternate text. The text strings to find and replace will be specified in the Properties pane.

2. **Create a step.**

   From the Toolbox pane, drag the **Replace String** activity onto the canvas and drop it inside the **Test Flow** area.

   Note: You can also double-click an activity in the Toolbox pane to add it to the canvas.



3. **Change the step's display name.**

   a. Select **View > Properties** to display the Properties pane.

   b. Select the **Replace String** step in the canvas.

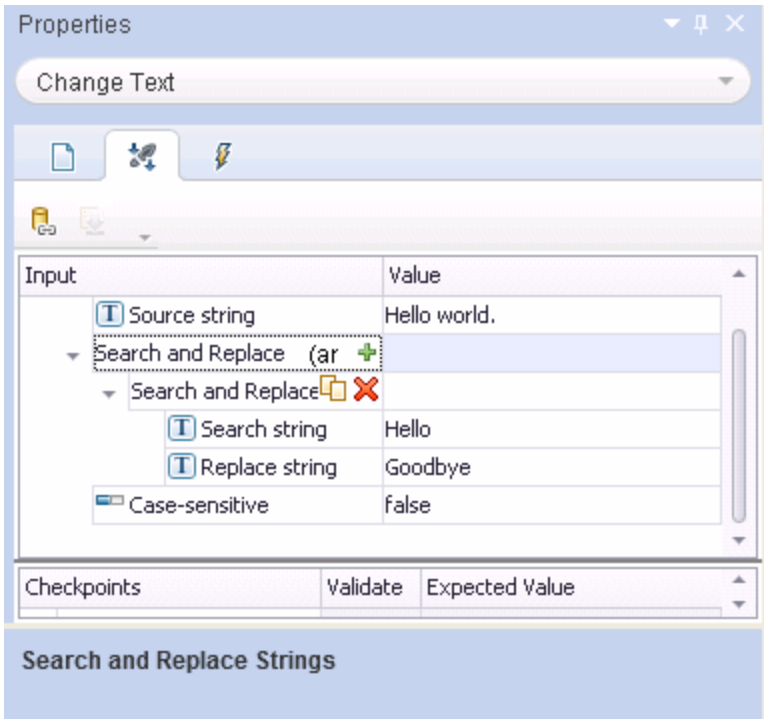
   c. In the Properties pane, click the **General** tab .

   d. In the **Name** row, type **Change Text** and press ENTER. This changes the step name in the canvas.

4. **Set the input properties for the Change Text step.**

   In the Properties pane, select the **Input/Checkpoints** tab. Enter the following values in the Input section of the Input/Checkpoints tab:

   - **Source string:** `Hello world.`
   - **Search string:** `Hello`
   - **Replace string:** `Goodbye`
   - **Case-sensitive:** `false`



5. **Run the test.**

   a. Click the **Run** button or press F5 to open the Run dialog box.

   b. In the Run dialog box, click **Options** to expand the dialog box.

   c. Select the **Temporary run results folder** option.

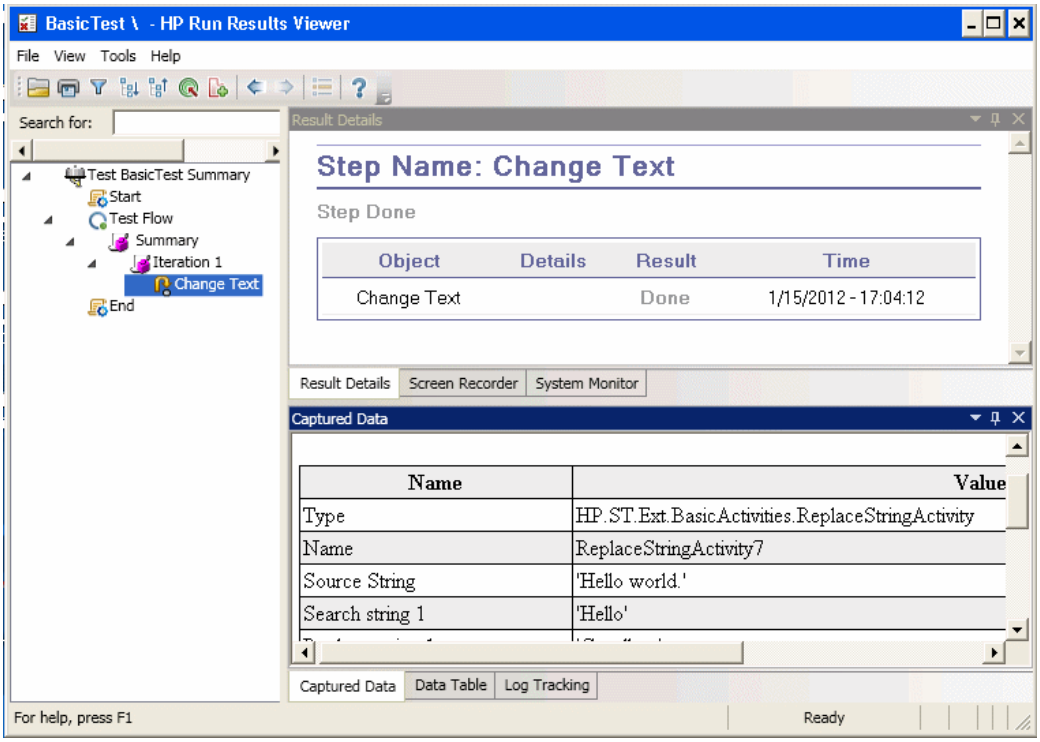
   d. Click **Run** to compile and run the test.

6. **View the results.**

   After your test runs, the Run Results Viewer opens.

   a. Select **View > Expand All** or click the Expand All toolbar button to view all the test steps.

   b. In the test step hierarchy in the left pane, click the **Change Text** node. The step results for the Change Text step are displayed in the Result Details and Captured Data panes.



   c. In the Captured Data pane, view the source and replacement strings and note the result string, **Goodbye world**. This is in fact the expected string—the test passed.

   d. When you are finished reviewing the results, close the Run Results Viewer.

7. **Set a checkpoint for the Change Text step.**

   In the previous step, you manually viewed the output to check if the result of the step matched the expected value for the step. In this step, you will create a checkpoint for your test step.

   Checkpoints allow you to see whether a step is successful without having to manually check the result. Checkpoints are the means to validate the test—a success or failure is determined by its checkpoints.

   a. Select the Change Text step in the canvas.

   b. In the Properties pane, open the **Input/Checkpoints** tab.

   c. In the **Checkpoints** section (lower section), select the **Validate** check box in the **Results** row to enable the checkpoint.

   d. In the **Expected value** column, type the expected string, **Goodbye world**.

   e. Run the test again. In the Run Results Viewer, expand the test step nodes, and note the checkmarks by the test step name. This indicates that the checkpoint passed since the result matched the expected value.

   f. When you are finished reviewing the results, close the Run Results Viewer.

Now that you have learned how to create test steps, including adding input and checkpoint properties for your test steps, you can continue to enhance your tests by linking test steps to one another. Continue to "Linking Test Steps" on the next page to learn more.

Linking Test Steps

When your application runs, it sometimes can pass a property or parameter from one process to the next. As a result, when testing your application processes in UFT, you need to be able to link test properties to each other.

In this section, you will use the output of one step as input for another test step.

1. **Add a Concatenate String step to the Test Flow**

   In the **Toolbox** pane, select **Concatenate String** from the **String Manipulation** category. Drag the activity into the canvas and drop it below the `Change Text` step in the Test Flow.

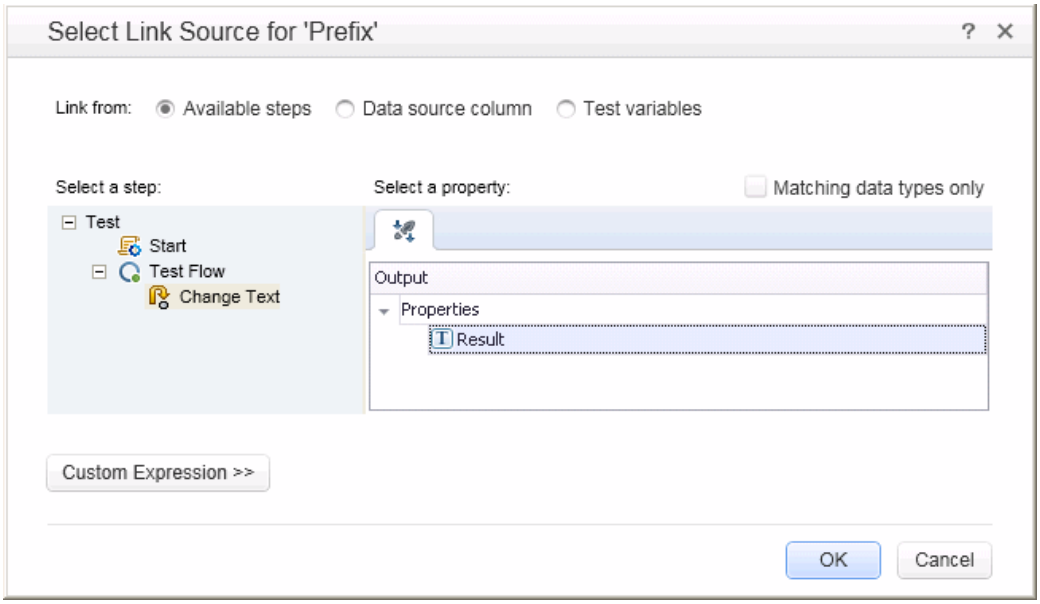
2. **Set the prefix input property for the Concatenate String step.**

   a. In the canvas, select the **Concatenate String** step.

   b. In the **Properties** pane, click the Input/Checkpoints tab.

   c. In the **Input** section (upper section), select the **Value** cell of the **Prefix** row.

   d. In the Value cell of the Prefix row, click the **Link to a data source** button . The Select Link Source dialog box opens.

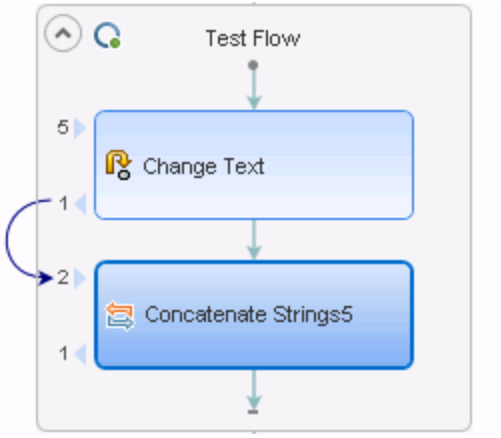


3. **Link the Concatenate String step to the Change Text Step.**

   a. In the Select Link Source dialog box, select the **Available steps** option.

   b. In the left pane, select the **Test Flow > Change Text** node. The list of available properties for the Change Text step is displayed in the right pane.

   c. In the right pane, double-click the **Results** node. The canvas now reflects that data is moving from **Change Text** to **Concatenate String**.



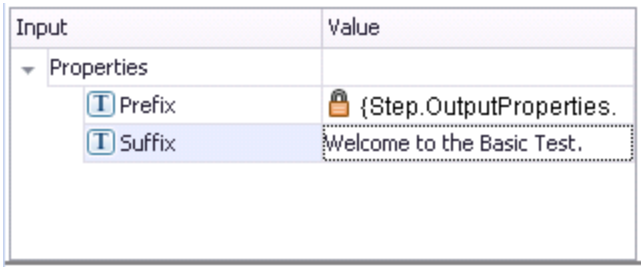
Note that the property value for the Prefix step is also displayed as `{Step.OutputProperties.ReplaceStringActivity4.Result}`. This indicates that this property is the result of the output of the Replace String (Change Text) activity.

> **Note:** Although you renamed the Replace String activity in the previous section, UFT still refers to this activity by its standard (programmatic) name.

4. **Configure the suffix properties for the Concatenate Strings step.**

   a. In the **Input** section of the Input/Checkpoints tab, select the **Suffix** row.

   b. In the suffix row, enter the text `Welcome to the Basic Test.` into the property's **Value** field.



5. **Run the test.**

   Click the **Run**  button or press **F5** to run the test.

6. **View the run results.**

   a. Expand the Run Results tree and select the **ConcatenateStringsActivity** node. The report shows the result of the concatenated strings: **Goodbye World.Welcome to the Basic Test.**

   b. When you are finished reviewing the results, close the Run Results Viewer.

Now that you have learned the basics of linking test steps to one another, you can learn how to link your test steps to multiple input sources, Continue to "Mapping Test Steps to Multiple Sources" on the next page to learn more.

Mapping Test Steps to Multiple Sources

Using the Select Link Source dialog box, you can link to different types of data sources to provide input values for your test steps: **Available steps**, **Data source column**, and **Test variables**. By linking your test steps to multiple sources, you can test those application processes whose input is linked to the output from multiple other processes.

In the above section, you used the **Available steps** source for one value, and manually typed in the data for another value. In this section, you will create a custom expression to use multiple data sources as a property value. You will use the Select Link Source dialog box to create an expression for the **Suffix** property that uses both manual entry and automatic values from the **Available steps** option.

1. **Set the prefix for the Concatenate String step.**
   a. In the canvas, select the **Concatenate String** step.
   b. Open the Input/Checkpoints tab  in the Properties pane.
   c. In the Input/Checkpoints tab, select the **Value** cell of the **Prefix** row.
   d. In the Value cell, click the  to clear the contents.
   e. In the Value cell, enter a new prefix `Hello world`.
2. **Open the Select Link Source dialog box.**
   a. In the Input/Checkpoints tab, select the **Value** cell of the **Suffix** row
   b. Click  to clear its contents.
   c. In the Value cell, click the **Link data to source** button  . The Select Link Source dialog box opens.
3. **Edit the suffix property for the Concatenate String step.**
   a. In the Select Link Source dialog box, click the **Custom Expression** button to display the custom expression.

b. In the **Expression** box, type the following:" `was replaced with "` (adding a space before and after the phrase to improve the readability).

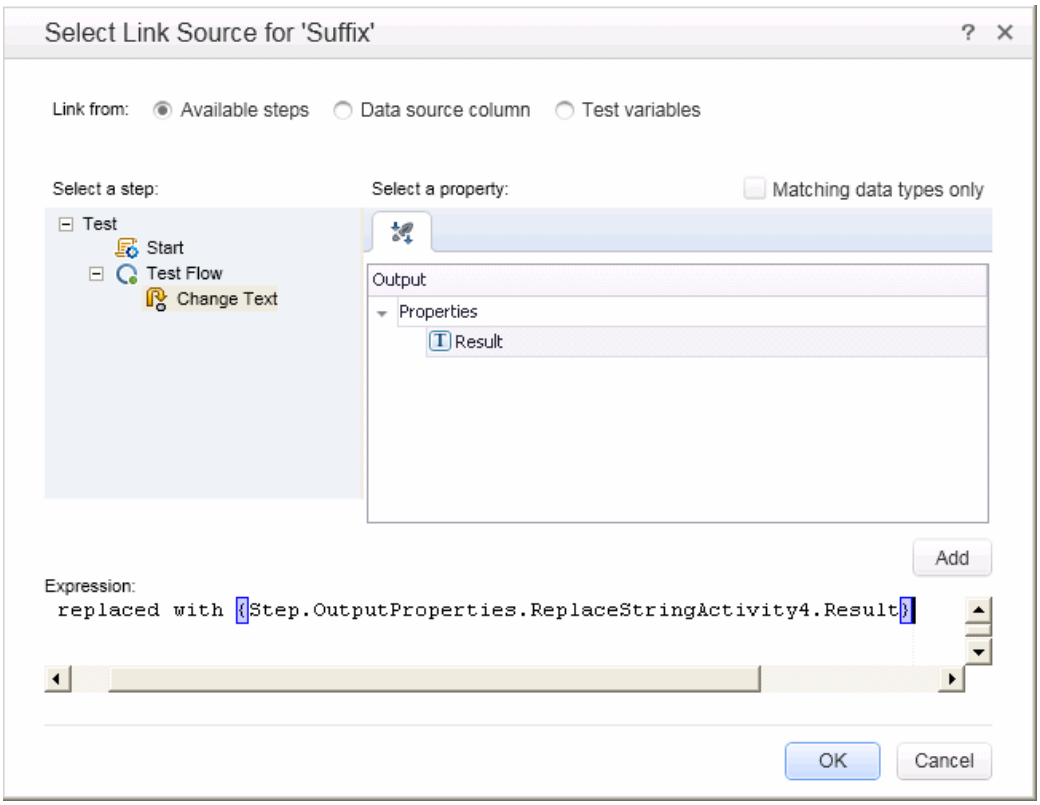
4. **Add another source to the custom expression.**

   a. In the Select Link Source dialog box, select the **Available steps** option (if not already selected). A list of available steps for this property is displayed in the left pane.

   b. In the left pane, select the **Change Text** node. The available properties for the Change Text step are displayed in the right pane.

   c. In the right pane, select the **Result** node in the right pane, and click **Add**.

   The **Expression** box shows both sources: your manually-entered expression and the output from the Replace String (Change Text) activity. This string will be added as the suffix property for the Concatenate Strings step.

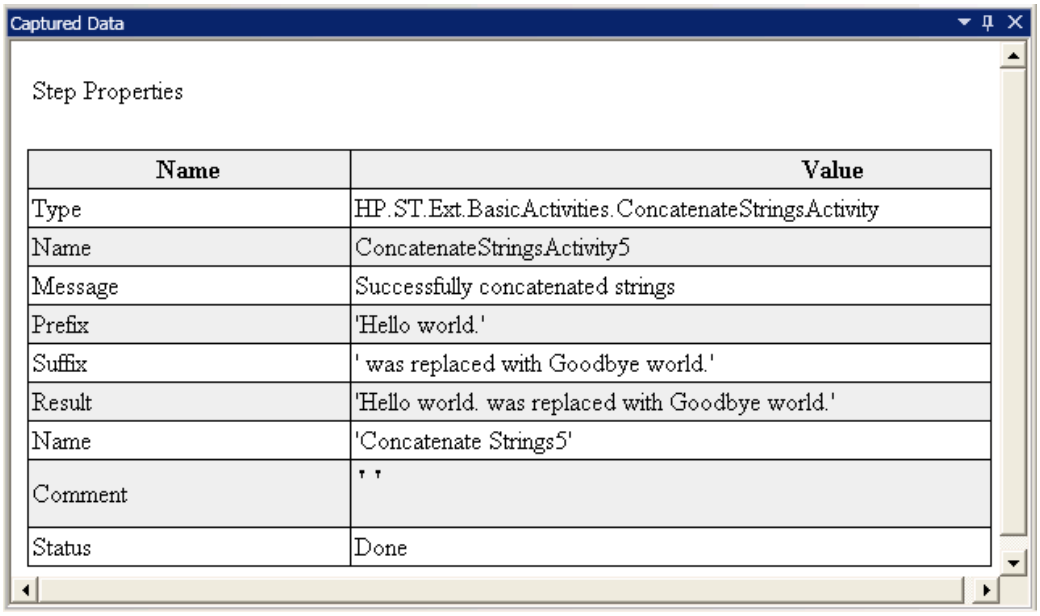


d. Click **OK** to add the custom expression as the Suffix property for the Concatenate String steps.

5. **Run the test and view the report.**

a. Click the **Run** button to run the test.

b. In the Run Results Viewer, expand the results and select the **ConcatenateString** node. The report shows the result of the concatenated strings.



6. When you are done viewing the results, close the Run Results Viewer.

Data Driving a Test Step

Data driving is the assigning of data to test steps from a data source, such as an Excel file, an XML file, a database, or a local table. The goal of data driving is to run the same application process with different values. It allows you to check your application in different scenarios simply by modifying the data values used for the step properties

In this section, you will learn how to data drive your test steps.

1. **Data drive the input properties for the Change Text step.**

   a. Select the **Change Text** step in the canvas.

   b. Open the **Input/Checkpoints** tab in the Properties pane

   c. In the Properties pane, click the **Data Drive**  button. The Data Driving dialog box opens.

2. **Specify a data provider for the data.**

   a. In the Data Driving dialog box, select the following options:

      - Set the **Data Provider** type to `Excel`.
      - Enable data driving for **Both Input and Checkpoints**.
      - Clear the **Configure 'Test Flow' as a ForEach loop using the new data source** option.

        This option repeats the Test Flow according to the number of data rows. You will manually set the number of iterations in a later step.

   b. Click **OK** to close the Data Driving dialog box.

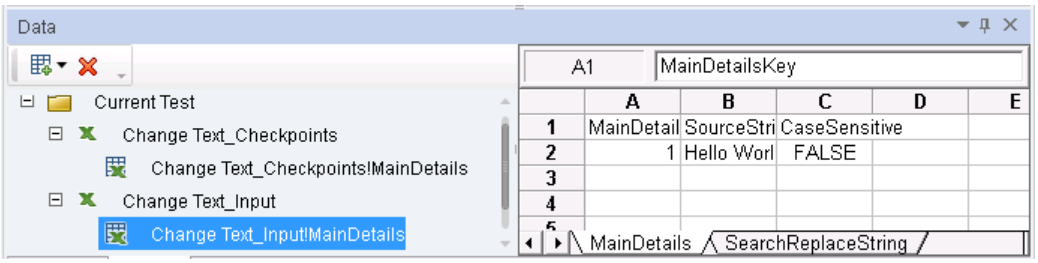
   c. Accept the popup message. The data driving mechanism replaces the constant values with the new expressions, `{DataSource.Change Text_ Input!MainDetails.SourceString}`.

3. **View the Data pane.**

   a. Open the Data pane **View > Data**.

   b. In the left pane of the Data pane, expand the **Change Text_Input** node and select the **Change Text_Input!MainDetails** node.

   The Data pane shows a data table with a column for each input property, and one row of values corresponding to the input property. In this example, you see the **Hello World.** input string and **FALSE** (or an empty check box if you do not have Excel installed) that you entered earlier.



4. **Add new data.**

   In the **Change Text_Input!MainDetails** sheet, add two additional rows. Make sure to copy the text exactly, including punctuation where included.

Entry Number	MainDetailsKey	SourceString	CaseSensitive
1	1	Hello world.	FALSE
2	2	I like eating broccoli.	TRUE
3	3	The product version is 11.	FALSE

5. **Add new search and replace data.**

   In the Data pane, select the **ChangeText_Input!SearchReplaceString** node and add two additional rows to the table. Make sure to copy the text exactly, including punctuation where included.

MainDetailsKey	Key	Value
1	Hello	Goodbye
2	broccoli	ice cream
3	11	12

6. **Add checkpoint values.**

   a. In the Data pane, expand the **Change Text_Checkpoints** node and select the **Change Text_Checkpoints!MainDetails** node.

   b. In the first column (under the **Result** cell), add values to this column as shown below:

   > **Note:** In the third row, we will intentionally insert an exclamation point (!) to generate an error.

Result
Goodbye world.
I like eating ice cream.
The product version is 12!

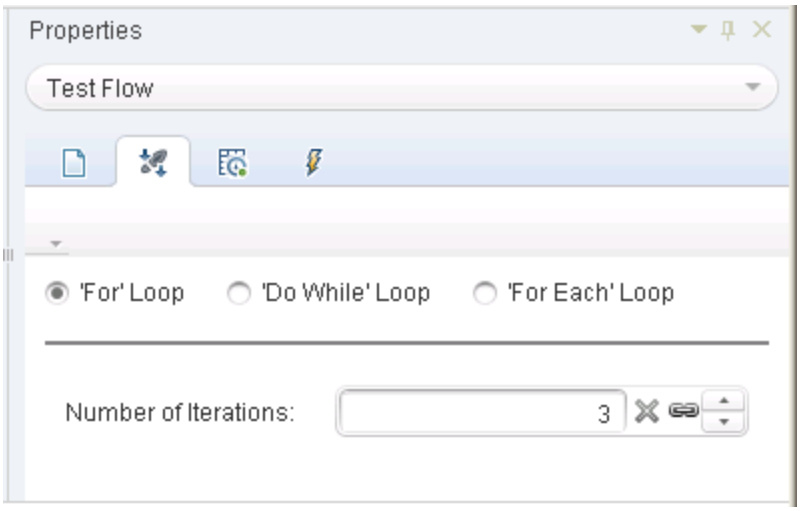
7. **Set the number of iterations.**

   The number of iterations is the number of times to repeat the step. We will set it to 3, corresponding to the number of rows of data in our table.

   a. Return to the canvas and click inside the Test Flow frame—but not within a test step.

   b. Open the Input tab in the Properties pane.

   c. In the Input tab, select **'For' Loop** and set the **Number of Iterations** to **3**.



8. **Run the test and view the report.**

   Click the **Run**  button or press **F5** to compile and run the test. The test runs three times, using the three lines of data in the table.

   When the Run Results Viewer opens, expand the **Test Flow** node and drill down to the row with the red **X**, indicating a failed checkpoint. The checkpoint failed because the expected result contained an exclamation point, which was not present in the source string.

9. **Correct the error and rerun the test.**

   a. In the Data pane, select the **Change Text_Checkpoints!MainDetails** node

   b. In the third row of the **Results** column replace the exclamation point with a period.

   c. Run the script again and verify that you have no errors in the report.

Where To Go From Here

In this lesson, you learned how to create test steps from standard activities, link the step properties together, and use data to drive the values of test step properties.

In the next lesson, you will apply these lessons when working with a custom activity using an imported Web service.The following lessons will walk you through the process of importing WSDLs and creating Web service tests.

Chapter 4: Building a Web Service Test

In "Building a Simple Test" on page 26, you learned how to create a test using standard API activities and learned some basic techniques for enhancing your test steps.

However, there will also be times where the standard activities do not match the processes your application performs. In these cases, you will need to use custom activities that you import or create in UFT, such as Web Services. Using UFT, you can also create tests for your WSDL-based Web services.

This lesson contains the following sections:

Importing a Web Service

When you want to test your Web service application, you must import this service into UFT. You import your Web Service using a WSDL file, which defines the operations/methods in a Web service. UFT reads the WSDL file and creates the service's methods as activities in the Toolbox pane.

This section shows you how to import the Flight API application's WSDL file.

1. **Start the Sample Flight application.**

   Make sure that the Flight Application service is available, as described in "Explore the Flight API Application" on page 22.

2. **Add your new test to the solution.**

   a. Select **File > Add > New Test**.

   b. In the Add New Test to Solution dialog box, select **API Test**.

   c. Name the **WebServiceTest** and click **Add**.

   The test is added to the **Tutorial Solution** along with the `Basic Test` created in the previous lesson.

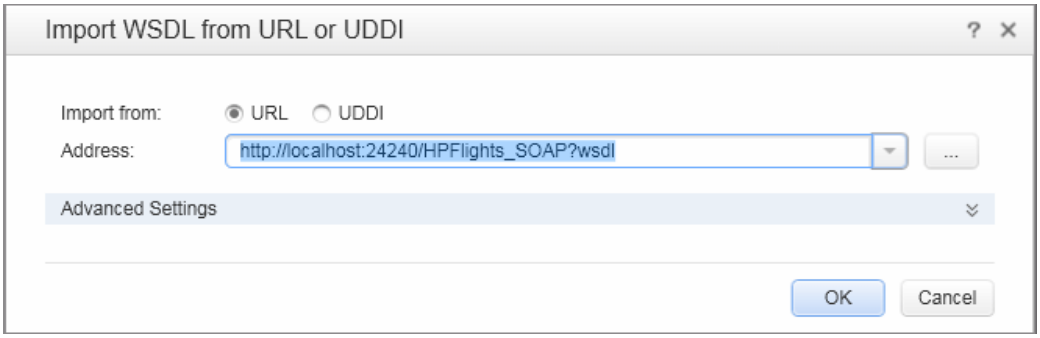
3. **Open the Import WSDL dialog box.**

   In UFT, select **Import WSDL > Import WSDL from URL or UDDI** from the toolbar. The Import WSDL from URL or UDDI dialog box opens.

4. **Specify an import source.**

   a. In the Import WSDL from URL or UDDI dialog box, Select the **URL** option.

   b. In the address field, enter the URL for the Web Service: **http://localhost:24240/HPFlights_SOAP?wsdl**



   c. Click **OK**.

The service is imported into UFT and its methods are displayed in the Toolbox pane under the Web Services node.

Now that you have imported the service's methods into UFT, you can begin to create test steps using these methods. Continue to "Building a Web Service Test" on the next page to learn more.

Building a Web Service Test

After you import a Web Service's methods, its methods are available for use in a test.

In this section, you will create a new flight order using the **HPFlights** Web service.

In order to create a flight order, you must first know the available flights. First you will run the **GetFlights** step that retrieves all of the flights to your destination. In the next step, you will use the first flight number returned, as input for the **CreateFlightOrder** step.

1. **Create a GetFlights step.**

   In the Toolbox pane, expand the **Web Services**, **HPFlights_Service**, and **FlightServiceMethods** nodes and drag the **GetFlights** activity into the Test Flow.

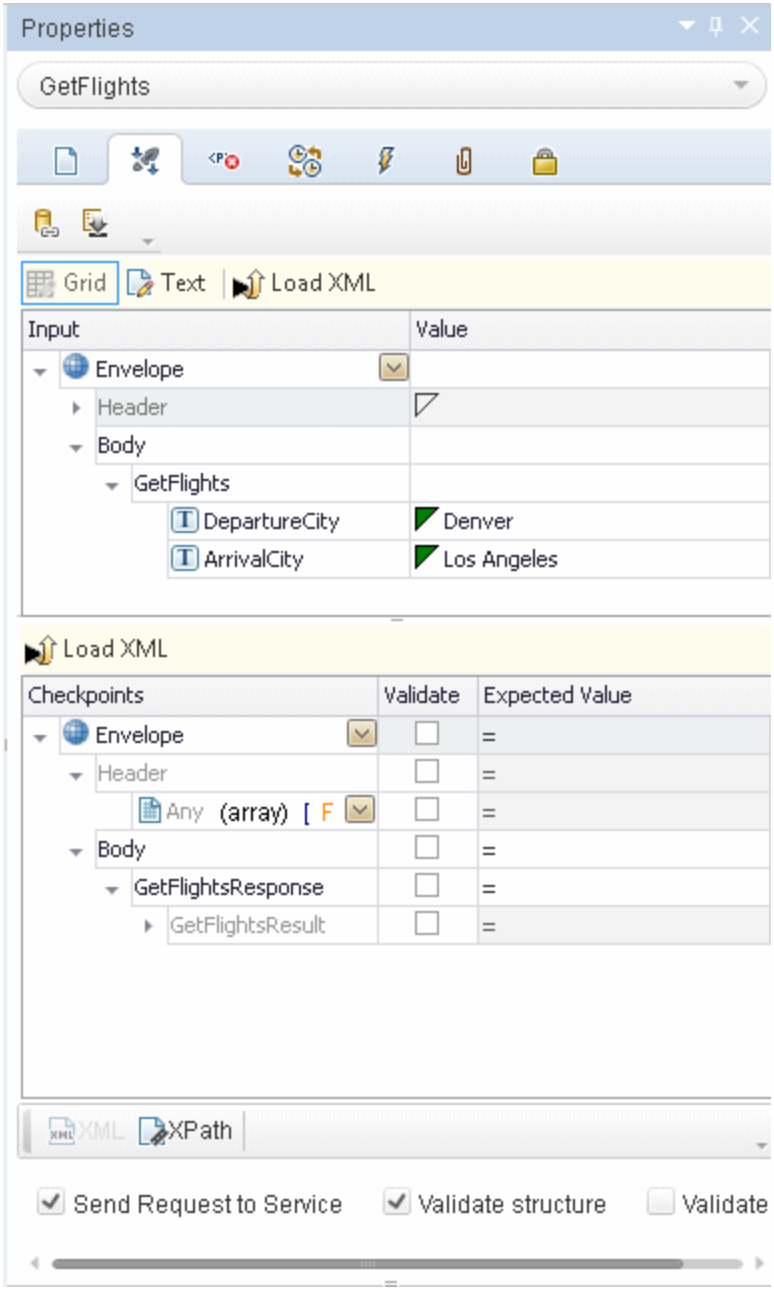
2. **Assign values for the DepartureCity and ArrivalCity input properties.**

   a. In the canvas, select the **GetFlights** step.

   b. In the Properties pane, open the **Input/Checkpoints** tab  and expand the **Body > GetFlights** node.

c. Select the following values from the drop-down list:

- **DepartureCity:** `Denver`
- **ArrivalCity:** `Los Angeles`



3. **Create a CreateFlightOrder step.**

   From the Toolbox pane, drag the **CreateFlightOrder** activity into the Test Flow,below the **GetFlights** step.

4. **Set the input property values for the CreateFlightOrder step.**

   a. In the canvas, select the **CreateFlightOrder** step.

   b. In the Properties pane, open the **Input/Checkpoints** tab  , and fully expand the **CreateFlightOrder > FlightOrder** node.

   c. Set the input property values for creating a flight order:

      - **Class**—Select a class, such as `Business`from the drop-down list.
      - **CustomerName**—enter any value
      - **DepartureDate**—use the drop-down to open a calendar and select a date at least two days in the future.
      - **FlightNumber**—leave this field blank for now. You will set its value in the following steps.
      - **NumberofTickets**—select any value.

5. **Link the output of the GetFlights stepto the CreateFlightOrder step.**

   a. In the canvas, select the **CreateFlightOrder** step.

   b. In the Properties pane, open the Input/Checkpoints tab.

   c. Expand the **CreateFlightOrder** node and select the **Value** cell in the **FlightNumber** row.

   d. In the Value cell, click the **Link to a data source** icon  . The Select Link Source dialog box opens.

   e. In the Select Link Source dialog box, select **Available steps** option.

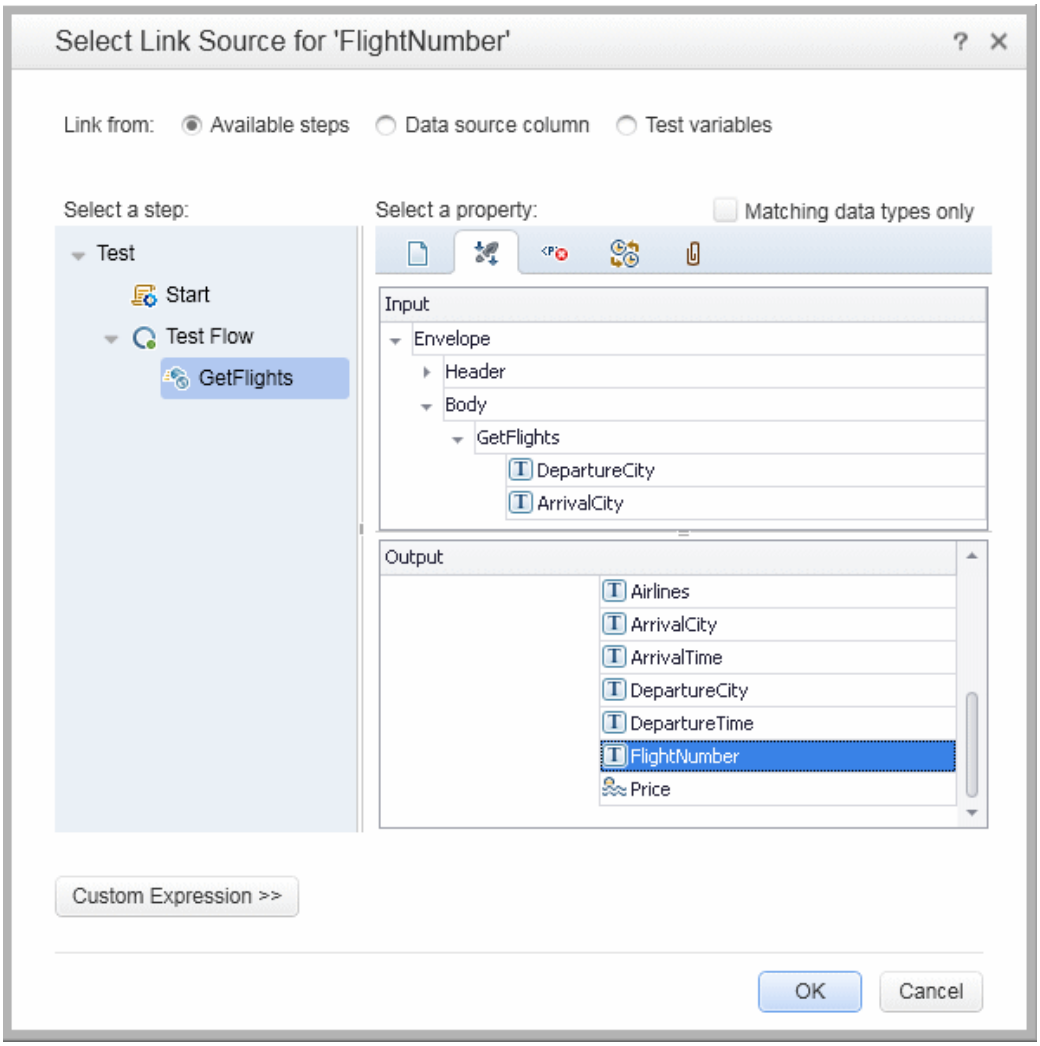
   f. From the left pane, select the **GetFlights** node.

   g. In the right pane, select the **Input/Checkpoints** tab.

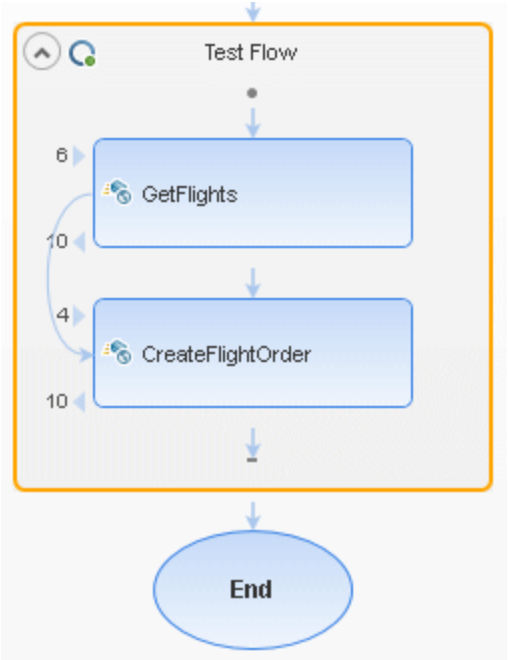
   h. In the **Output** section, expand all nodes under the **Body** node by clicking the GetFlightsResult node and clicking the **Add** button  in the **Flight (array)** node row to create the **Flight[1]** array.

   i. Expand the **Flight[1]** node.

j. Select the **FlightNumber** element, and click **OK**. When UFT asks if you want to enclose the target step in a loop, select **No**.



The canvas indicates a connection between the two steps.



6. **Reset the number of iterations.**

   a. Return to the canvas and click inside the Test Flow frame but not within a test step.

   b. In the Properties pane's, open the **Input/Checkpoints** tab .

   c. Select **For Loop** and set the **Number of Iterations** to **1**.

7. **Run the test.**

   Click the **Run** button. Observe the build log in the **Output** tab.

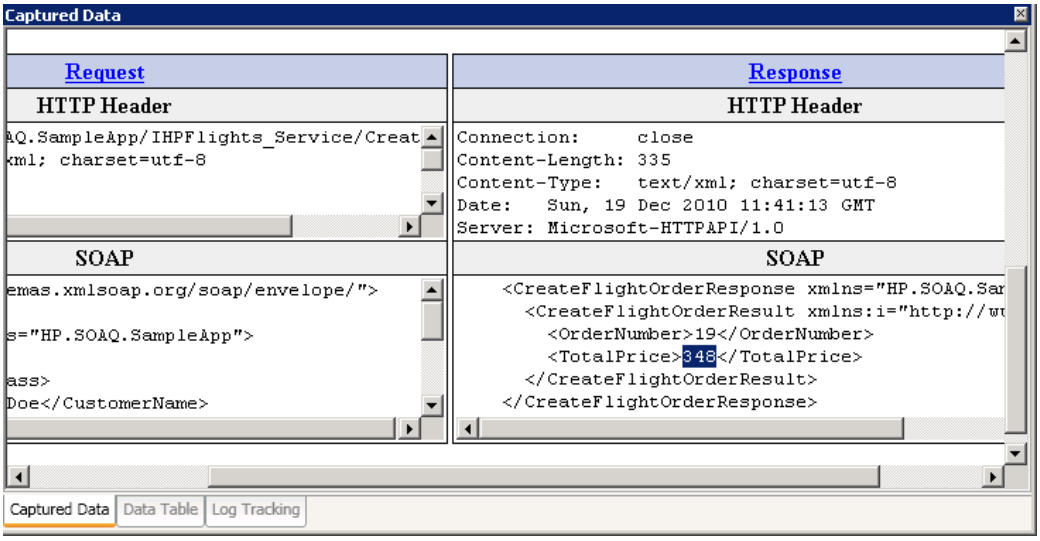
   After the test run is complete, the Run Results Viewer opens automatically.

8. **Check the results.**

   a. In the Run Results tree (left pane), right- click the parent node and select **Expand All**.

   b. Click the **CreateFlightOrder** node. The result details for the CreateFlightOrder step are displayed.

c. In the **Captured Data** pane, scroll down to the Web service Call HTTP Snapshot section and look at the Response pane. Note the output of the request—**OrderNumber** and **TotalPrice**. Copy the **TotalPrice** value to the clipboard for use in the next step.



Tip: Click the **Request** or **Response** links to open the SOAP in a separate browser.

When you are finished viewing the results, close the Run Results Viewer.

9. **Set a checkpoint for the CreateFlightOrder step.**

   a. Select the **CreateFlightOrder** step in the canvas.

   b. In the Properties pane's, open the **Input/Checkpoints** tab.

   c. In the Checkpoints section (lower section), expand the **CreateFlightOrderResponse > CreateFlightOrderResult** node.

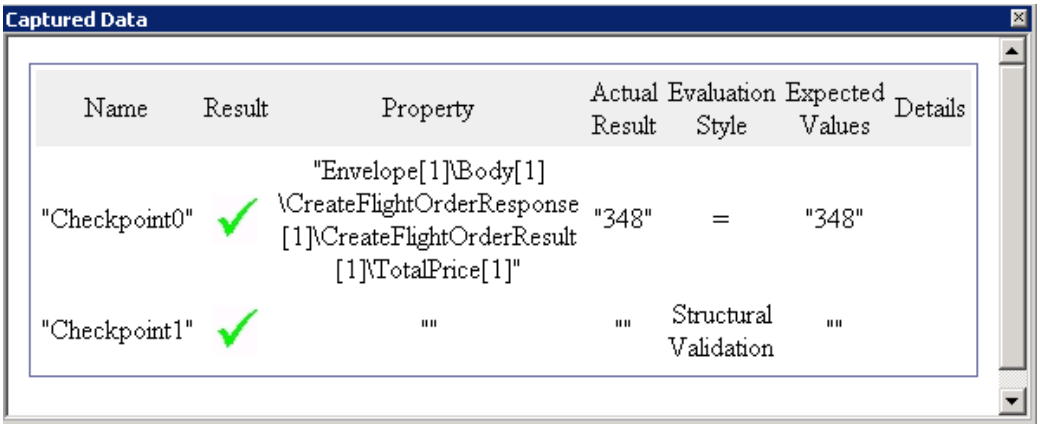
   d. Paste the total price amount from the previous step into the **TotalPrice** field and select the **Validate** check box in the **TotalPrice** row.

10. **Run the test and view the checkpoint results.**

    a. Run the test again

    b. When the Run Results Viewer opens after the test run, expand the Run Results tree and Select the **Checkpoints** node for **CreateFlightOrder** step.

    The report shows a checkmark and indicates the expected and actual values. If the expected value was not returned by the server, the report indicates a failure.



    When you are finished viewing the results, close the Run Results Viewer.

Now that you have created a test for your Web Service, you can enhance your Web Service test by using data. Continue to "Integrating Data into Your Web Service Tests" below to learn more.

Integrating Data into Your Web Service Tests

In this section you will learn how to integrate data from an existing source, and how to data drive the test. When you data drive a test, the Data pane automatically creates a data table whose values you can edit.

1. **Import a data source for your test**

    In the Data pane, click the **New Data Source** button  select **Excel**. The Add New Excel Data Source dialog box opens.

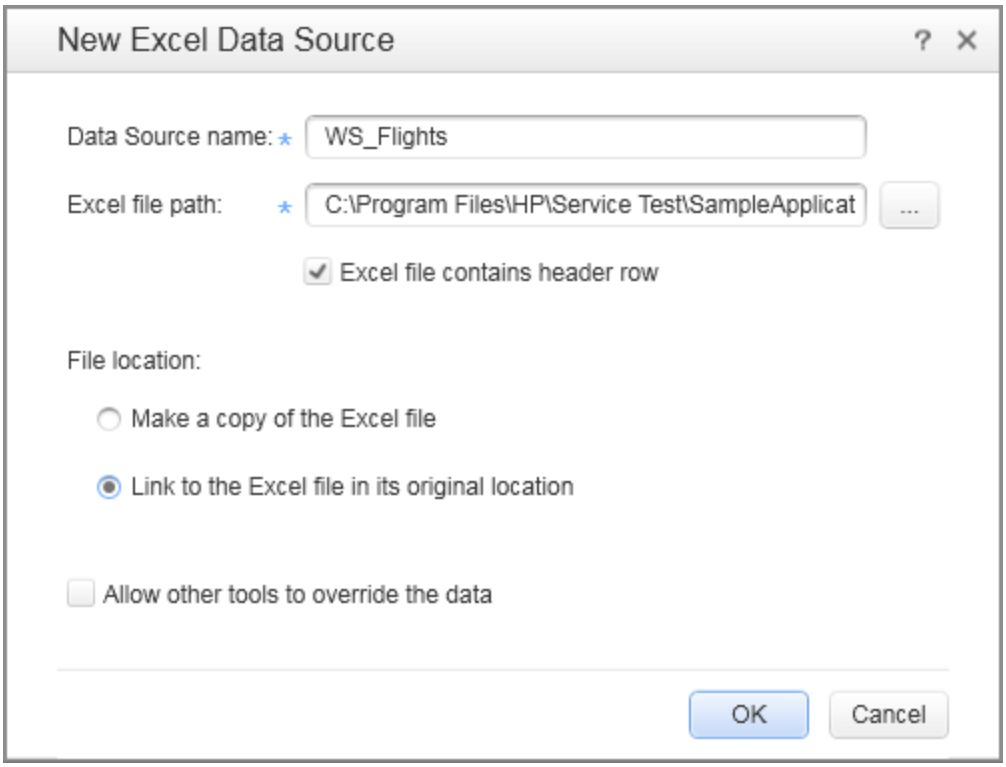
    a. Browse for the sample application's Excel file, `SampleAppData.xlsx`, in the `<installation directory>\SampleApplication` folder..

    b. Enable the **Excel file contains header row** option, since the sample file contains a header row.

    c. Enter `WS_Flights` as a **Data source name.**

d. Select **Link to the Excel file in its original location** as the mode of import. This links to the Excel file at its original location, so that if data changes, your data source will be current.

e. Click **OK**.



2. **Link the input properties for the CreateFlightOrder step to the data.**

   a. Select the **CreateFlightOrder** step in the canvas.

   b. In the Properties pane, open the **Input/Checkpoints** tab.

   c. In the **Input** section of the Input/Checkpoints tab, expand all the FlightOrder node and select the **Class** row.

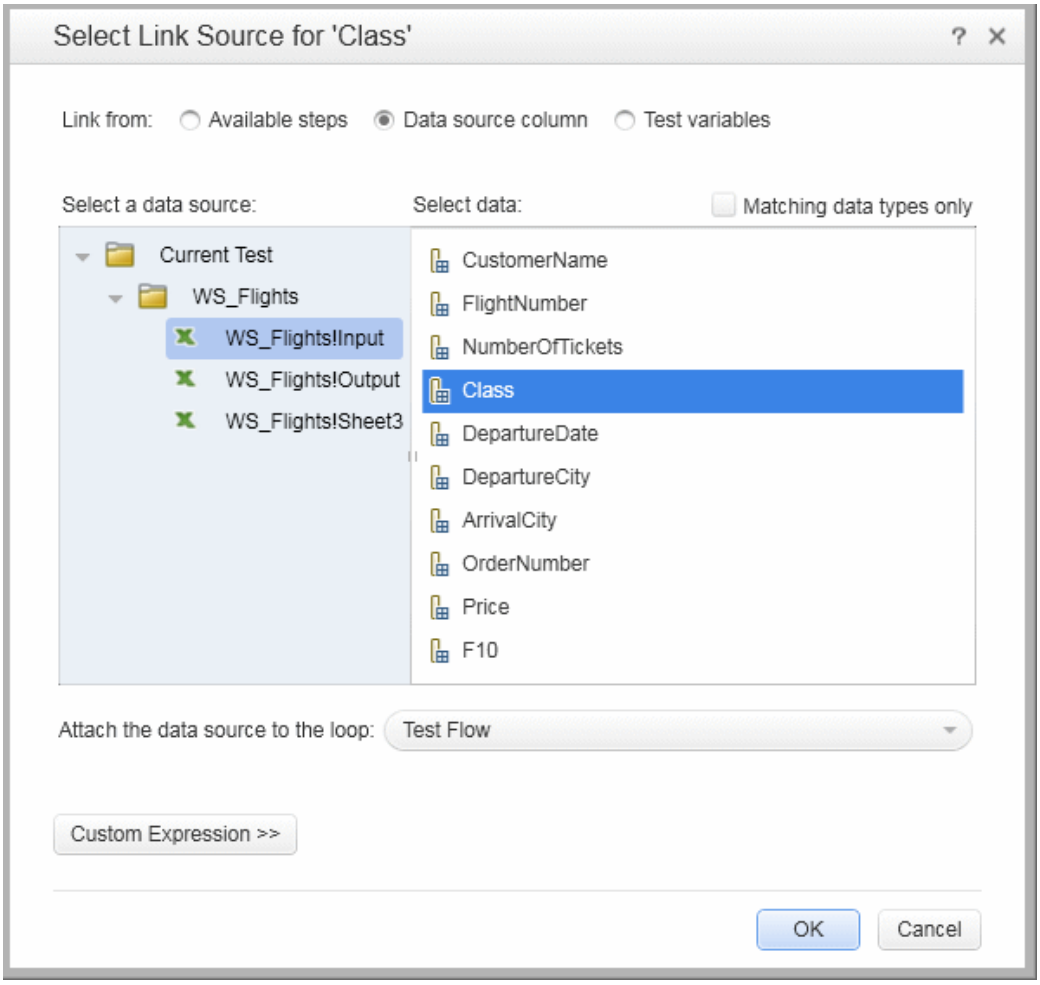
   d. In the Class row, click the **Link to a data source** icon . The Select Link Source dialog box opens.

3. **Link the Class input property to the data source.**

   a. In the Select Link Source dialog box, select the **Data source column** option. The list of all available data sheets is displayed.

   b. Select the **WS_Flights!Input** node. The list of all available data columns is displayed.

c. From the list of data columns, select **Class** in the right pane and click **OK**. This instructs the test to refer to this column in the sample data during the test run.



d. Repeat this for the other input parameters: **CustomerName**, **DepartureDate**, **FlightNumber**, and **NumberofTickets**.

4. **Disable the CreateFlightOrder checkpoint.**

a. Select the **CreateFlightOrder** step in the canvas.

b. In the Properties pane, open the Input/Checkpoints tab.

c. In the Checkpoints section (lower section) of the Input/Checkpoints tab, clear the checkbox in the TotalPrice row. This instructs UFT not to check this property during the test run.

5. **Set the navigation settings for your data.**

   The navigation settings let you indicate how to use the data in your data source. You can specify from which row to begin, how many rows to advance, and in what direction to move for the next set of values. You can also specify what to do when reaching the end of the data table—wrap around or continue using the last line.

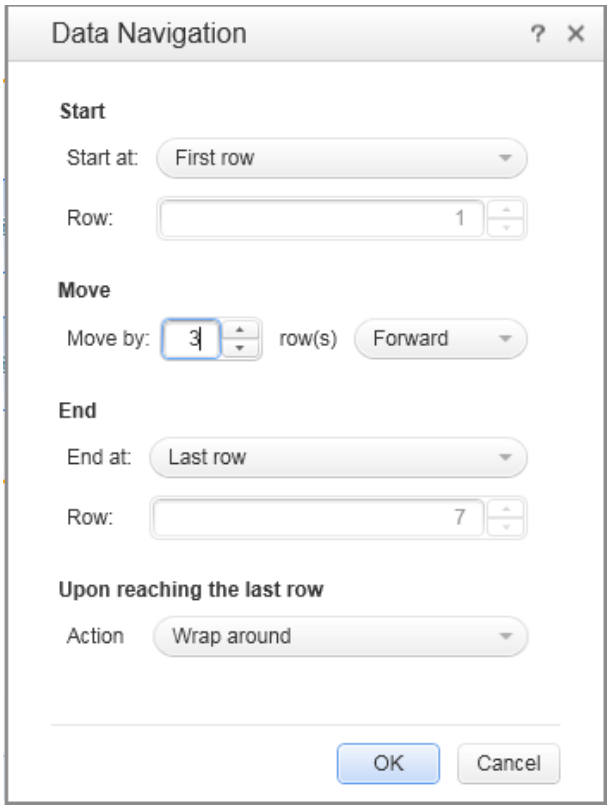
   a. In the canvas, click in the **Test Flow** but not within a step.

   b. In the Properties pane, open the **Data Sources** tab  .

   c. In the Data Sources tab, select the **WS_Flights!Input** entry in the table and click **Edit** to open the Data Navigation dialog box.

   d. In the Data Navigation dialog box, specify the data navigation details:

      - **Start at:** `First row`
      - **Move:** Move by `3` rows `Forward`
      - **End at:** `Last row`
      - **Upon reaching the last row:** `Wrap around`

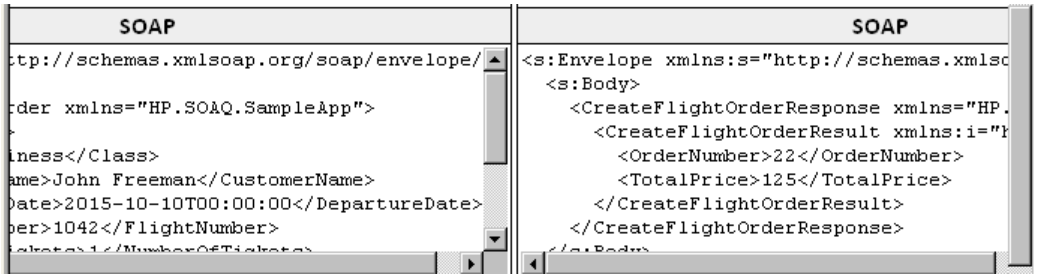


e. Click **OK**.

6. **Run the test and view the results.**

   a. Click the **Run** button and observe the results in the Output window.

   b. In the Run Results Viewer, expand the result tree and select the **CreateFlightOrder** step. Scroll down within the **Captured Data** tab and note the data from the Excel file in the SOAP request (left pane), and the result in the SOAP response (right pane).



   When you are finished viewing the results, close the Run Results Viewer.

Now that you have learned how to integrate data into your Web Service tests, you can further enhance your tests by linking steps to multiple sources and using custom code. Continue with "Using Multiple Data Sources and Custom Code in Your Web Service Tests" on the next page.

Using Multiple Data Sources and Custom Code in Your Web Service Tests

In the previous section, you learned how to use data in your Web Service test steps. This section extends this by describing how to define data using multiple data sources and sending information to the report through a custom code step.

1. **Create a new test.**

   Add a new test called `WebServicesCustom` to your solution and import the HP Flights Services WSDL as described in "Importing a Web Service" on page 45.

2. **Create test steps for your test.**

   From the Toolbox pane, drag the activities into the canvas in the following order:

   - **GetFlights** (found under the Web Services node)
   - **CreateFlightOrder** (found under the Web Services node)
   - **Custom Code** (found under the Miscellaneous node)

3. **Add a data source to your test.**

   In the Data pane, select **New > Excel**. In the Add New Excel Data Source dialog box:

   a. Browse for the sample application Excel file in the `installation directory>\SampleApplication` folder.

   b. Select the **Excel file contains header row** check box.

   c. Enter `WS_Flights` as a **Data source name**.

   d. Select the **Link to the Excel file in its original location** mode.

   e. Click **OK** to add the data source to the test.

4. **Assign values for the input properties of the GetFlights step.**

   a. In the canvas, select the **GetFlights** step.

   b. In the Properties pane, open the **Input/Checkpoints**  tab.

   c. In the Input/Checkpoints tab, in the **Input** section, select **DepartureCity=** `Denver`, and **ArrivalCity=**`Los Angeles`.

5. **Assign values for the input properties of the CreateFlightOrder step.**

   In the canvas, select the **CreateFlightOrder** activity.

   In the Properties pane, open the **Input/Checkpoints** tab.

   In the Input/Checkpoints tab, expand the **FlightOrder** node (found under the Body node) and set the input properties as follows:

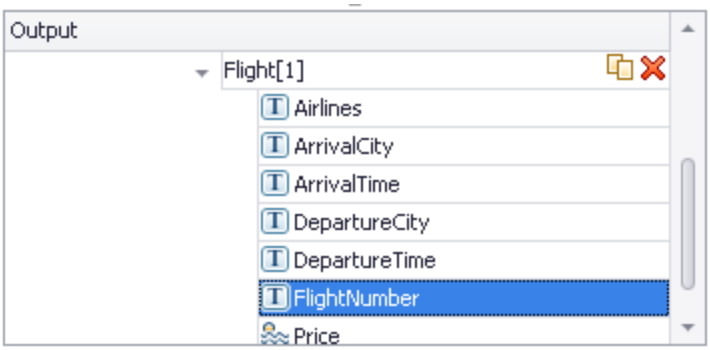
   - **Class:** Economy
   - **CustomerName:** Click the **Link to a data source** button in the right corner of the **CustomerName** row. In the Select Link Source dialog box, select **Data source column**, and expand the tree to show the **WS_Flights!Input** node. In the right pane, select the **CustomerName** parameter and click **OK**.
   - **DepartureDate.** A date at least two days ahead of the current date.
   - **NumberofTickets.** 3
   - **FlightNumber:** Link from the previous step:

     i. Click the **Link to a data source** button in the right corner of the **FlightNumber** row.

     ii. In the Link to Source dialog box, select **Available steps**, expand the **Test Flow** node, and click **GetFlights**.

     iii. In the right pane, select the **Input/Checkpoints** button.

     iv. In the **Output** section, expand the **GetFlightsResult** node and click the **Add** button in the **Flight (array)** node row to create the **Flight[1]** array. Expand the array, select **FlightNumber**, and click **OK.** When prompted to enclose the target step in a loop, select **No**.



6. **Create an input property for the custom code step.**

   a. In the canvas, select the **Custom Code** activity.

   b. In the Properties pane, open the **Input/Checkpoints** tab.

   c. In the Input/Checkpoints tab, click the **Add Property** button and select **Add Input Property**. The Add Input Property dialog box opens.

   d. In the Add Property dialog box, create a new **String** type property called **FlightInfo**.

   e. Click **OK** to add the input property.

7. **Define values for the custom code step.**

   In this step you will define a value using multiple sources. In this example, you will set a value which is a combination of the **CustomerName**, a constant string, and the **OrderNumber**:

   a. In the canvas, select the **Custom Code** step.

   b. In the Properties pane, open the **Input/Output Properties** tab.

   c. In the Input/Output Properties tab select the **FlightInfo** row.

   d. In the **Value** column of the FlightInfo row, click the **Link to a data source** button  . The Select Link Source dialog box opens.

   e. In the Select Link Source dialog box, click **Custom Expression** to show the Expression area.

   f. At the top of the Select Link Source dialog box, select the **Data source column** option. The list of available data sheets is displayed.

   g. In the data sheet list (left pane), select the **WS_Flights!Input** node. The list of available data columns is displayed.

   h. In the list of data columns, select **CustomerName** and click **Add**.

   i. In the **Expression** area, type `_OrderNumber_` (with the underscores) after the existing expression.

   j. At the top of the dialog box, select the **Available steps** and expand the **Test Flow** branch. The list of available steps is displayed.

   k. In the list of test steps, select the **CreateFlightOrder** node.

   l. In the right pane, select the **Input/Checkpoints** button  .

m. In the lower pane of the Input/Checkpoints tab, expand the Output **Body** node, expand the **CreateFlightOrderResponse** node, then the **CreateFlightOrderResult** node, select the **OrderNumber** element, and click **Add**.

n. The the Expression box contains the following value:

```
{DataSource.WS_Flights!Input.CustomerName}_OrderNumber_{Step.OutputPro
perties.StServiceCallActivity(x).Body.CreateFlightOrderResponse.Create
FlightOrderResult.OrderNumber}
```

o. Click **OK** to close the dialog box.

8. **Create an event.**

   In this step you will create an event handler in order to use custom C# code to your test step. Defining events let you adapt your test to your custom requirements, and perform actions that are not built-in to UFT. In this example, you will add code that sends a custom string to the report.

   a. In the canvas, select the **Custom Code** step.

   b. In the Properties pane, open the **Events** tab  . A list of default event handlers is displayed.

   c. In the **ExecuteEvent** row of the Events tab, click the drop down arrow and select **Create a default handler**.

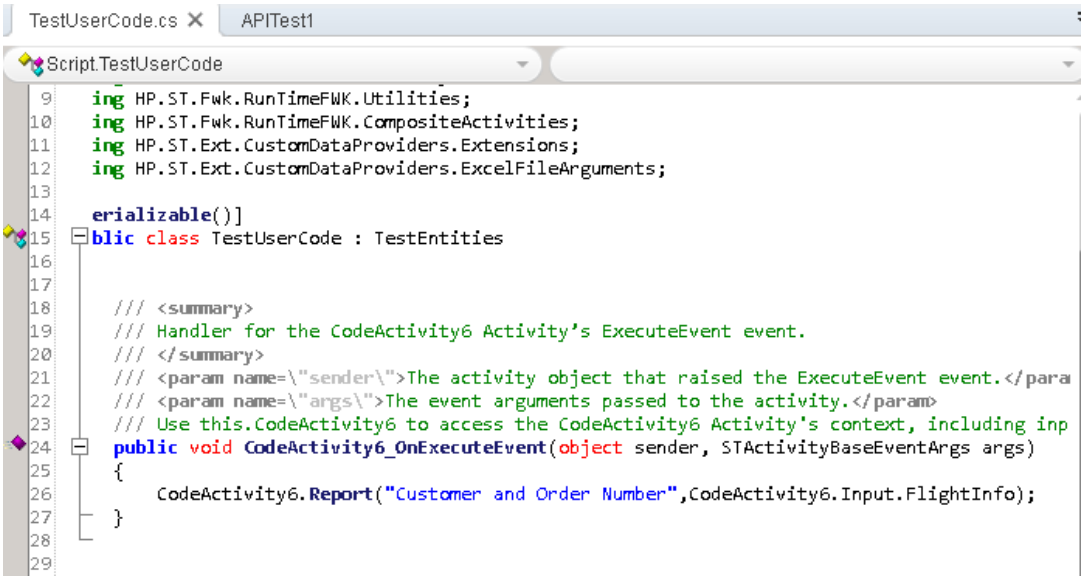
      UFT creates an event called **CodeActivity(x)_OnExecuteEvent** and opens a new tab called `TestUserCode.cs`.

9. **Edit the "Todo" section for your event handler.**

   In the `TestUserCode.cs` file, in the TODO section of the **CodeActivity(x)_OnExecuteEvent** section, replace the **Todo** section with the following:

```
CodeActivity(x).Report("Customer and Order Number",CodeActivity(x).Input.
FlightInfo);
```

In the following example, the index assigned to the event was 12, so the string is `CodeActivity12.Report("Customer and Order Number",CodeActivity12.Input.FlightInfo);`.



10. **Run test and check the results.**

    In the Run Results Viewer, expand the Test Results tree to the **Custom Code** step. Note the new entry in the **Captured Data** pane: `Customer and Order Number`.

    Tip: You can also use the **Report Message** activity under the **Miscellaneous** folder, to send text and property values to the report.

Where to Go From Here

Now that you have learned to create a test for a Web service, you can relate this to other types of services and application components. The next lesson will walk you through the process of creating a test for a REST service.

Chapter 5: Building a REST Service Test

Similar to what you did when importing your Web Service, you can test your REST Service application processes in UFT. When you test REST Service processes, you must create them inside UFT as a prototype and then use the methods in your test.

This section will teach the basic steps in creating your REST Service methods and using them in your test.

This lesson contains the following sections:

Creating a REST Service Activity

Before you can use a REST Service activity in your tests, you must create the necessary activities and their properties inside of UFT.

This section describes how to model a REST service activity using the Flight API application. Once you create a REST Service method, you can reuse the method for different test steps.

1. **Start the Sample Flight application.**

   Make sure that the Flight Application service is running, as described in "Explore the Flight API Application" on page 22.

2. **Obtain the REST service modeling document.**

   In the Flight API application command window, type **h**, and press ENTER. A browser opens with the modeling information for the REST service.

   > **Note:** The file containing this information (`index.htm`) is located in the `<installation_folder>\SampleApplication\Help` folder.

3. **Save the ReserveOrder Request body.**

   a. Copy the Request Body for the **FlightOrders > ReserveOrder (POST)** method. Only copy the XML code.

   ```
   <FlightOrderDetails xmlns="HP.SOAQ.SampleApp" >
     <Class>Business</Class>
     <CustomerName>John Doe</CustomerName>
     <DepartureDate><future date></DepartureDate>
     <FlightNumber>1304</FlightNumber>
     <NumberOfTickets>21</NumberOfTickets>
    </FlightOrderDetails>
   ```

   b. Create a new file in a text editor and paste the contents of the clipboard. Make sure to modify the date in the `<DepartureDate>` line to a future date.

   c. Save the file as `body.xml` to any location.

4. **Add a new test to the solution.**

   Add a new test called `RESTServiceTest` to your solution as described in "Importing a Web Service" on page 45.

5. **Create a REST service.**

   a. Click the **Add REST Service** toolbar button . The Add REST Service dialog box opens.

   b. In the left pane of the Add REST Service dialog box, click on the **New Service** node and rename it to **SamplesRESTService**.

6. **Add a resource to the REST Service method.**

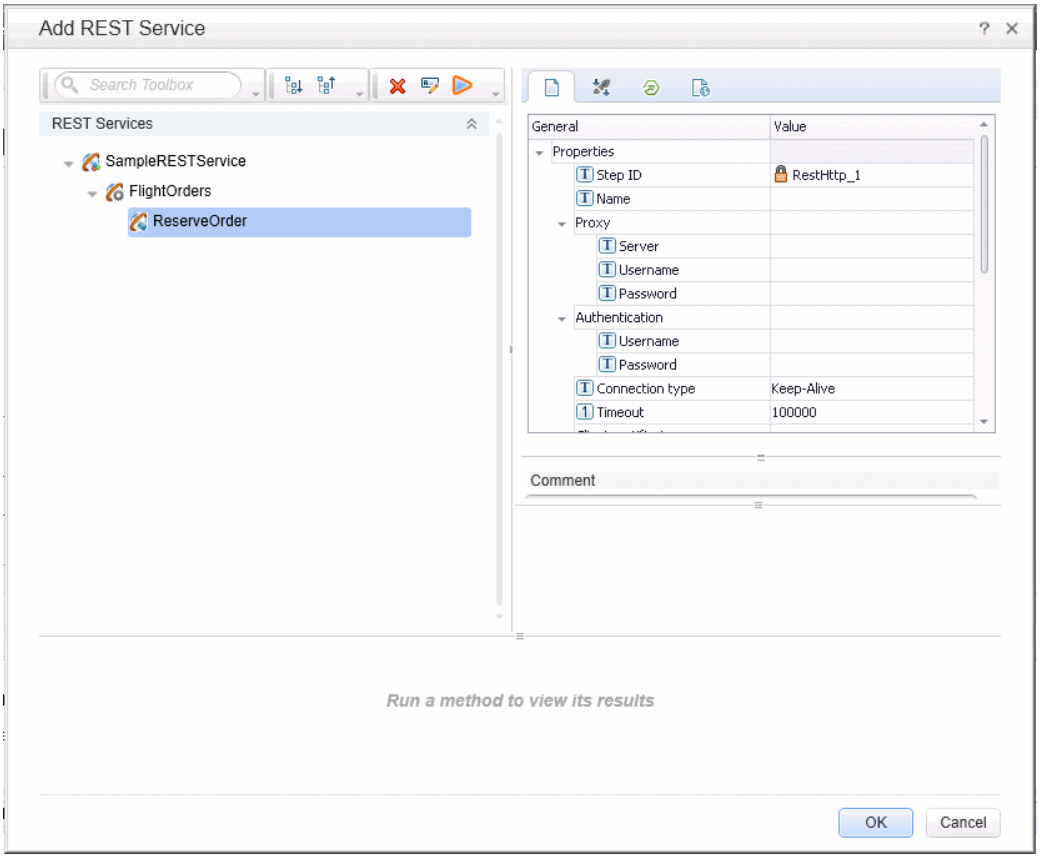
   a. In the Add REST Service dialog box, click the toolbar's **Add Resource** button . A new resource is added under your SampleRESTService.

   b. Rename the resource to **FlightOrders**.

7. **Add a method to the REST service.**

   a. In the Add REST Service dialog box, click the **Add Method** button . A new method is added under the FlightOrders resource.

   b. Rename the method to **ReserveOrder**.



8. **Configure the REST Service URL.**

   a. In the left pane, select the **SampleRESTService** node .

   b. In the right pane, select the **General** tab.

   c. In the URL property row, paste the URL prefix: `http://localhost:24240`.

   d. Return to the left pane and select the **FlightOrders** node. Note that the value you pasted in the URL property field for the **SampleRESTService** node has been passed to the **FlightOrders** resource.

   e. In the **General** tab of the right pane, paste `/HPFlights_REST` in the **Relative URL** property row. After you paste this value here and select an area outside the property value row, the `/HPFlights_REST` is added to the URL prefix value from the **SampleRESTService** node.

   f. In the left pane, select the **ReserveOrder** node.

   g. In the **HTTP Input/Checkpoints** tab  of the right pane, paste `/FlightOrders/` in the **Relative URL** value row. This addition is appended with the URL property value passed from the **FlightOrders** node of your REST service.

9. **Configure additional HTTP properties.**

   a. In the Add REST Service dialog box, select the **ReserveOrder** method node.

   b. In the right pane, open the **HTTP Input/Checkpoints** tab  in the right pane.

   c. Set the **HTTP method** to **POST**.

   d. Open the HTTP tab  in the right pane.

   e. In the **Request Body** section, click the **Load XML** button.

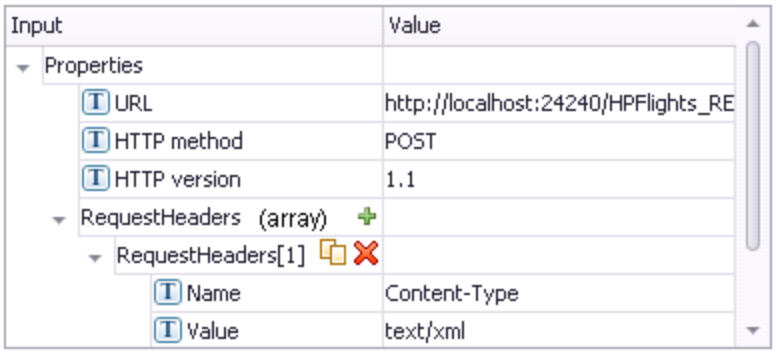
   f. Navigate to the `body.xml` file you saved in a previous step and click OK to add this XML to your REST method.

      Note that the XML structure saved in the body.xml file is displayed in the Request Body section (either in Grid form or Text form, depending on which view you have selected).

   g. Open the **HTTP Input/Checkpoints**  tab.

h. Expand the **Request Headers** array. The following values are now displayed in the Request Headers properties:

- **Name** row: `Content-Type`
- **Value** row: `text/xml`



10. **Create input properties for the REST Service.**

a. In the Add REST Service dialog box, select the SampleRESTService node.

b. In the right pane, open the **Input/Checkpoints** tab.

c. In the Input/Checkpoint tab, select **Add > Add Input Property**. The Add Input Property dialog box opens.

d. In the Add Input Property dialog box, add a **String** type property called **Class**. You do not need to provide a default value.

e. Add another **String** type property called **Customer_Name**.

f. Add another **DateTime** type property called **Departure_Date**.

g. Add an **Int** type property called **Flight_Number**.

h. Add another **Int** type property called **Number_of_Tickets**.

11. **Create output properties for the REST Service.**

a. In the Add REST Service dialog box, select the **ReserveOrder** node.

b. In the right pane, open the **Custom Input/Checkpoints** tab.

c. Select **Add > Add Output Property**. The Add Output Property dialog box opens.

d. In the Add Output Property dialog box, add a **Int** type property called **Total_Price**.

e. Add another **Int** type property called **Order_Number**.

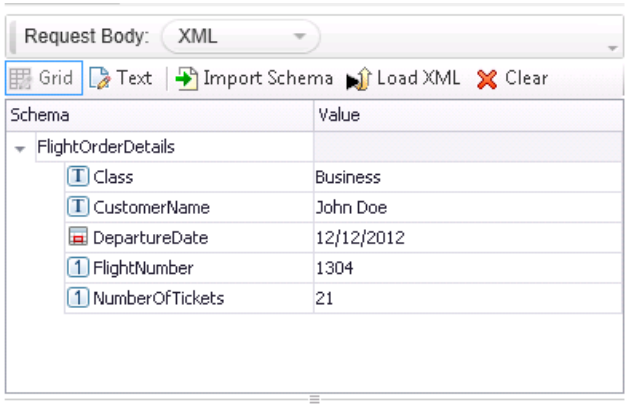
12. **Import the Request body.**

a. In the Add REST Service dialog box, select the **ReserveOrder** node.

b. In the right pane, open the **HTTP** tab .

c. Select **XML** as the **Request Body** type.

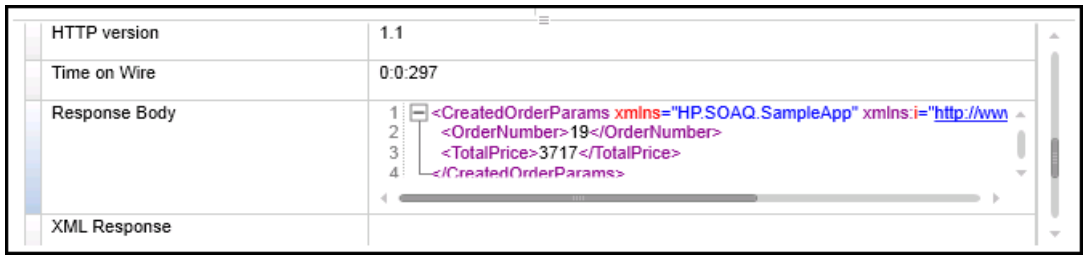
d. Click the **Load XML** button and load the `body.xml` file that you saved earlier.



The property values as defined in the XML from the `body.xml` file are displayed in the Request Body grid.

13. **Test the method.**

In the Add REST Service dialog box, click the **Run Method** toolbar button to check the validity of the method. Scroll through the results and verify that the response body contains an order number and price.



14. **Add the method to the Toolbox.**

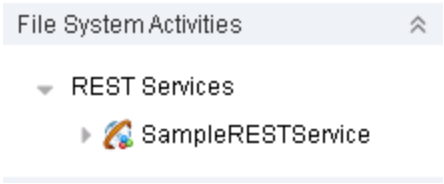
    Click **OK** in the Add REST Service dialog box. The REST designer adds the REST service, along with its resource and method to the **Toolbox** pane, under the **Local Activities** category.

15. **Share the REST activity to make the activity available for all tests.**

    a. In the **Toolbox** pane, select the parent node of the REST service, **SampleRESTService**

    b. Right-click the SampleRESTService node and select **Move to > File System Activities** from the right-click menu.

    The REST service activity is now moved to the **File System Activities** section in the **Toolbox** pane. Any other test can now use this activity.



You have now created a prototype activity for your REST service, complete with input parameters and the HTTP information. You can now use the methods in your tests. Continue with "Running a REST Service Test" on the next page to learn how to run the test with these newly-created REST methods.

Running a REST Service Test

In the previous section, you learned how to create a REST Service in UFT using the Add REST Service dialog box (also called the REST Service editor).

This section describes how to run the REST service method created in the previous section. You will incorporate data into the test, using the sample data file included with the product.

1. **Create a test step.**

   a. In the Toolbox pane, expand the **File System Activities** node.

   b. In the File System Activities node, expand all the nodes in the **SampleRESTService** node.

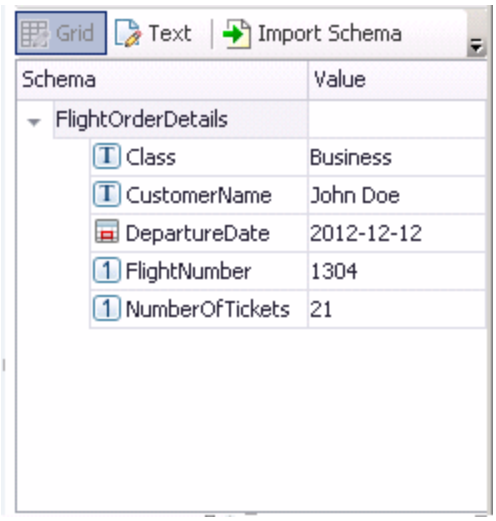
   c. Select the **ReserveOrder** method and drag it to the canvas.

2. **Check the HTTP response properties for the ReserveOrder REST method**

   a. In the canvas, select the **ReserveOrder** step.

   b. In the Properties pane, open the **HTTP** tab in the Properties pane.

   c. Check to make sure that the property values for this step are those which we imported in the previous section from the `body.xml` file. These values were used in the test run that we just performed.



3. **Run the test.**

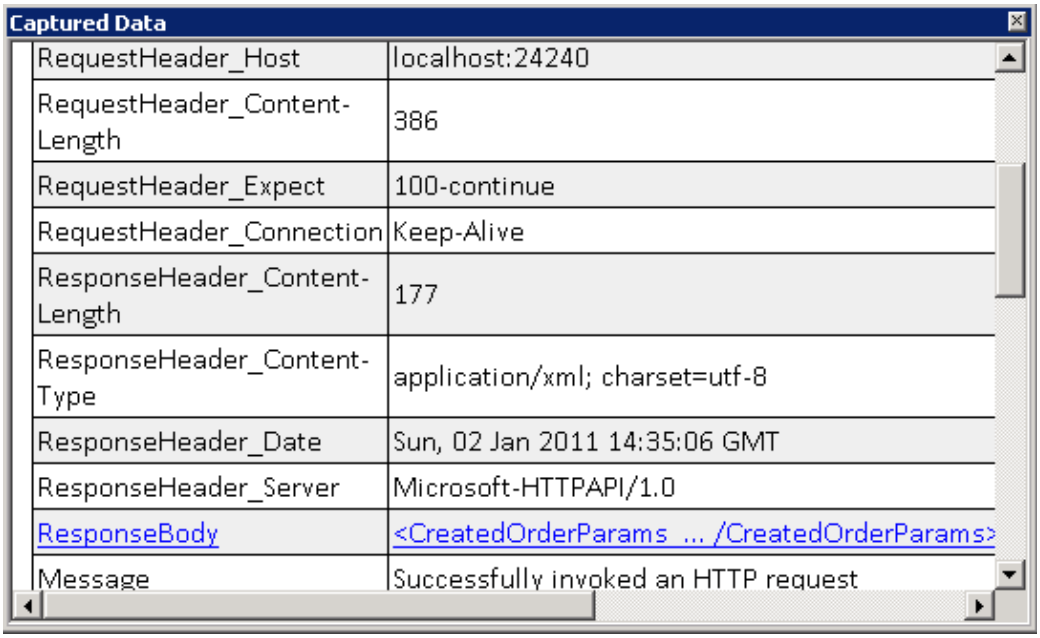
   Select **Run > Run** to run the test.

4. **View the run results.**

   a. In the Run Results tree, click the **Expand All** button or right-click the Run Results Tree and select **Expand All**.

   b. In the Run Results tree, select the **ReserveOrder**node. The step result details are displayed for the ReserveOrder step.

   c. In the **Captured Data** pane, click the **ResponseBody** link to open the response in a separate browser window.



   d. Verify that the **Response Body** contains values for the **OrderNumber** and **TotalPrice** elements. This corresponds to the operation's description in the REST Service Help page that indicated the following: `It creates a new flight order and returns OrderNumber and TotalPrice.`

```
- <CreatedOrderParams xmlns="HP.SOAQ.SampleApp" xmlns:i="http://www.w3
.org/2001/XMLSchema-instance">
<OrderNumber>18</OrderNumber>
<TotalPrice>3717</TotalPrice>
</CreatedOrderParams>
```

Continue to "Assigning Data to a REST Service Test" on the next page to learn how to enhance your REST method test steps by assigning data values for the method's properties.

Assigning Data to a REST Service Test

In previous sections, you have seen how you can assign data values to your test steps. In this section, you will also assign data from an imported Excel data source to your REST Service test steps.

1. **Import an Excel data source into your test.**

   a. In the Data pane, expand the Data Source button and select **Excel**. The New Excel Data Source dialog box opens.

   b. Click the Browse button adjacent to the **Excel file path** field, and browse to the sample application's **SampAppData** Excel file in the <`installation directory>\SampleApplication` folder.

   c. Select the **Excel file contains header row** check box, because the sample contains a header row.

   d. In the **Data source name** field, enter `REST_Flights`.

   e. Select the **Make a copy of the Excel file** option. This saves a copy of the data file with the test.

   f. Click **OK**. The data is imported into the Data pane and the data sheets are displayed in the Data pane.

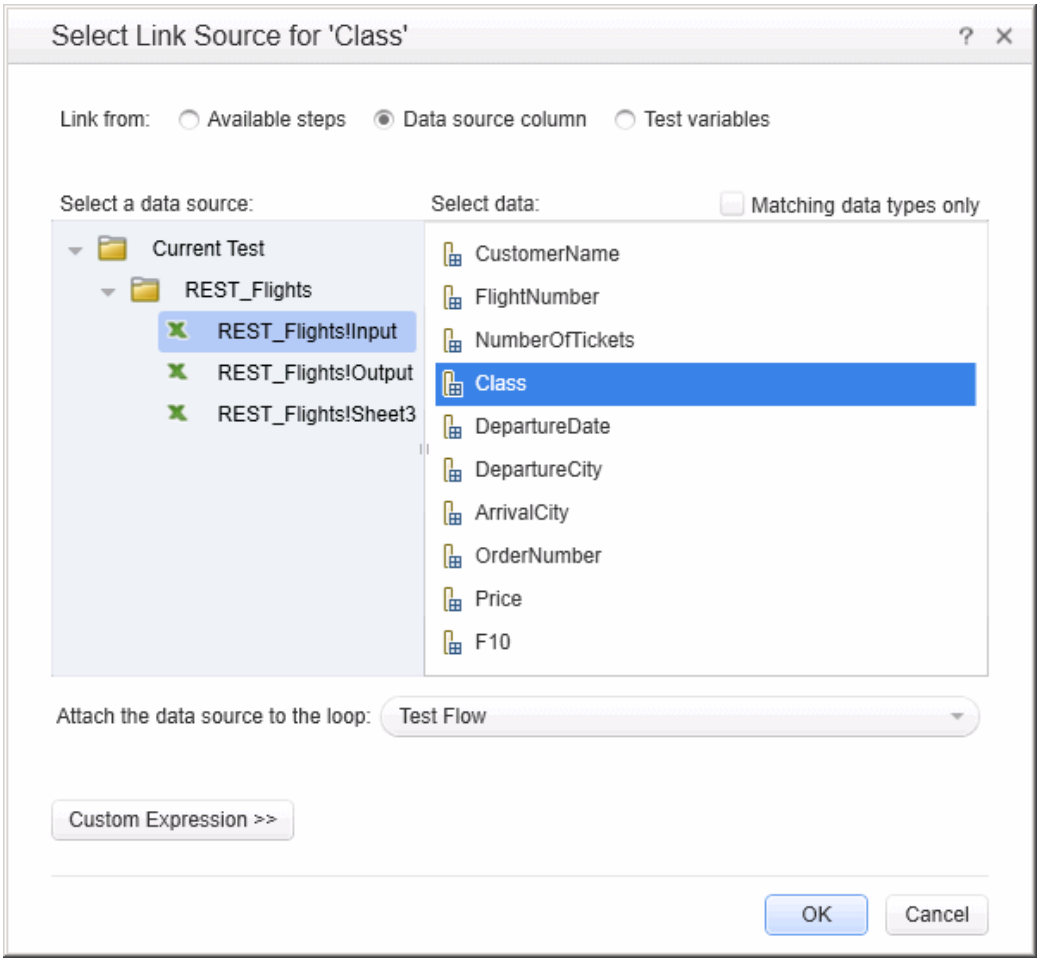
2. **Link the input properties for the ReserveOrder step to the data source.**

   a. In the canvas, select the **ReserveOrder** step.

   b. In the Properties pane, select the **Input/Checkpoints** tab.

   c. In the input section (upper section) of the Input/Checkpoints tab, click the **Link to a data source** button in the **Class** row. The Select Link Source dialog box opens.

d. In the Select Link Source dialog box, select the **Data source column**option.



e. In the list of data sources in the left pane, select the **REST_FlightsInput** node in the left pane.

f. In the right pane, choose the **Class** property in the right pane and click **OK**.

g. Repeat the above step for the other input properties: **Customer_Name**, **Departure_Date**, and **Flight_Number.**

h. Enter 2 for the **Number_of_Tickets**.

The resulting properties view shows the new links.

Input	Value
Properties	
Class	{DataSource.REST_Flights!Input.Class}
Customer_Name	{DataSource.REST_Flights!Input.CustomerName}
Departure_Date	{DataSource.REST_Flights!Input.DepartureDate}
Flight_Number	{DataSource.REST_Flights!Input.FlightNumber}
Number_of_Tickets	2

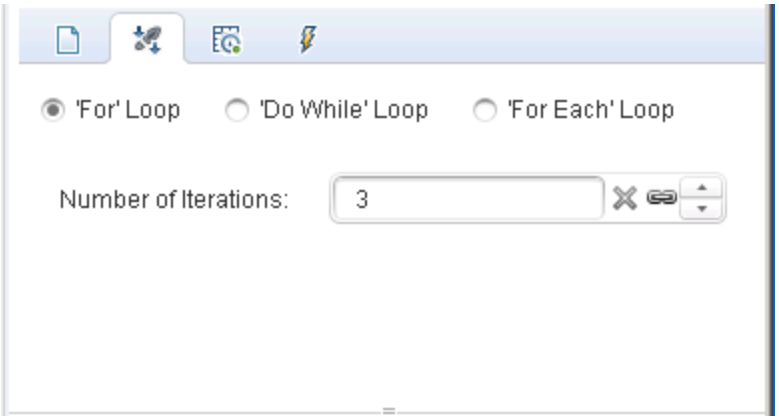
3. **Set the number of test iterations.**

By setting iterations, we will see how our REST method used multiple sets of data from the data source.

a. In the canvas, select the **Test Flow** frame in the canvas.

b. In the Properties pane, open the **Input/Checkpoints** tab.

c. Set a **For Loop** with 3 iterations.



4. **Run the test.**

Select **Run > Run** to run the test.

5. **Verify that the request used the table data.**

a. In the Run Results tree, click the **Expand All** button.

b. In the Run Results tree, select the **ReserveOrder** nodes (one under each iteration). The result details for the ReserveOrder step are display.

c. In the Captured Data pane, scroll down and click the **Request Body** link. In the browser window that opens, note that the test used the data from the Data pane for the properties that we assigned: `Class`, `CustomerName`, `DepartureDate`, and `FlightNumber`.

```
- <FlightOrderDetails xmlns="HP.SOAQ.SampleApp" >
<Class>Business</Class>
<CustomerName>John Freeman</CustomerName>
<DepartureDate>2012-12-12</DepartureDate>
<FlightNumber>1304</FlightNumber>
<NumberOfTickets>21</NumberOfTickets>
</FlightOrderDetails>
```

6. **Save the response data.**

   We will save the response data from this run for use in future steps.

   a. In the Run Results tree, select a **ReserveOrder** node. Click the **Response Body** link in the Captured Data pane. A browser window opens with the XML response.

   b. Save the entire contents of the window to a file `Response.xml`. Close the Run Results Viewer.

Checkpoints for REST Service Test Steps

In previous sections, you used checkpoints to verify your test steps, both for a basic test and a Web Service test. In this section, you will verify that the output of our REST method is correct by using checkpoints.

1. **Insert a checkpoint for the ReserveOrder step.**

   a. In the canvas, select the **ReserveOrder** step.

   b. In the Properties pane, open the **HTTP** tab .

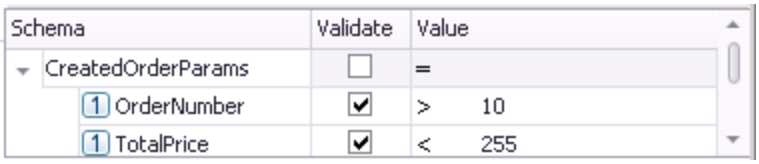
   c. The lower section of the HTTP tab contains the output properties or output schema. We will use these as our checkpoints to check the server response.

      In the lower section, select **XML** from the **Body** drop-down.

   d. Click the **Load XML** button and load the `Response.xml` file that you saved earlier. The checkpoint properties and their values from the `response.xml` file are displayed.

   e. In the Checkpoints section (lower section), select the **Validate** checkbox in the **OrderNumber** and **TotalPrice** rows.

   f. Set the value of **OrderNumber** to `Greater than (>) 10`, and the value of **TotalPrice** to `Less than (<) 255`.

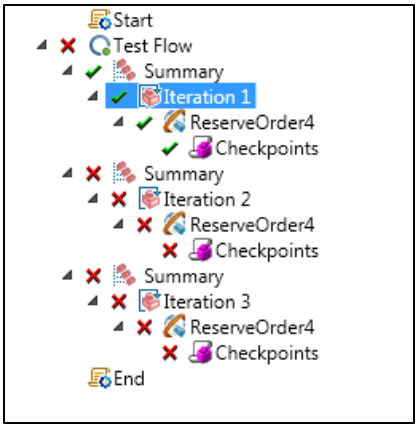


2. **Run the test.**

   Select **Run > Run** to run the test.

3. **Verify that the checkpoint passed.**

   In the Run Results tree, click the **Expand All** button  . Note that some of the checkpoints passed, while others did not.



4. **Determine why checkpoints failed.**

   a. In the Run Results tree, select one of the failed checkpoints nodes.

   b. In the Captured Data pane, note the Actual Results and Expected Values. In the following example, the **OrderNumber** was valid, but the **TotalPrice** was not valid because it exceeded 255.

Name	Result	Property	Actual Result	Evaluation Style	Expected Values	Details
"Checkpoint 2"	✓	"CreatedOrderParams [1]/OrderNumber[1]"	"32"	>	"10"	""
"Checkpoint 3"	✗	"CreatedOrderParams [1]/TotalPrice[1]"	"328"	<	"255"	""

5. **Save the test.**

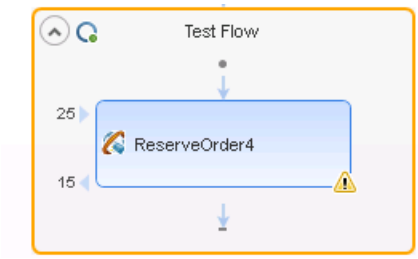
Close the Run Results Viewer and save the test.

Resolving Changes in a REST Service

Initially, we created a prototype REST service method **ReserveOrder**, with specific properties, such as the URL and property names. If these properties changed after you created a test, your test step will no longer match the prototype. The Resolve Conflict wizard detects changes in the method's properties and helps you resolve them.

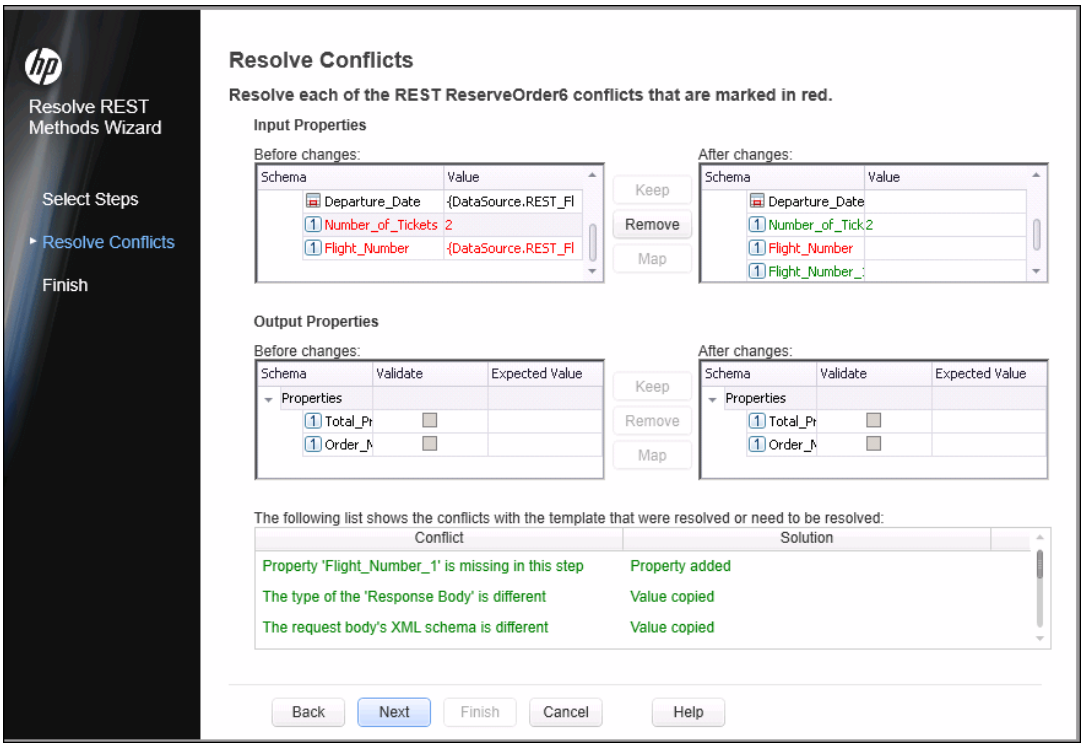
In this lesson, you will learn to use the Resolve Conflict wizard to resolve these differences.

1. In the Toolbox pane, right-click the **SampleRESTService** node and select **Edit Service**. The Edit REST Service dialog box opens.
2. In the Edit REST Service dialog box, select the **SampleRESTService** node and open the **Custom Input/Checkpoint** tab in the right pane.
3. Select the **Flight_Number** property, and click the **Edit Property** toolbar button. The Edit Property dialog box opens.
4. In the Edit Property dialog box, rename the property to **Flight_Number_1** and click **OK**.
5. In the Custom Input/Checkpoints tab, select the **Number_of_Tickets** property, and click the **Delete Property** button in the toolbar. Confirm the warning and click **OK**.
6. Click **OK** in the Edit REST Service dialog box.
7. View the canvas. Note the alert icon in the bottom right corner of the **ReserveOrder** REST method frame.



8. Click on the drop-down arrow adjacent to the alert icon, and select the text message: `This step should be resolved. Resolve step`. The Resolve REST Methods Wizard opens.
9. The wizard's first screen shows the problematic steps. If multiple steps were affected, you could choose which steps to resolve and which to ignore. In this lesson, there is only one step in your test, so by default , the **ReserveOrder** step is selected. Click **Next**.

10. In the Resolve Conflicts screen, in the Input Properties section (top area of the window), select the **Number_of_Tickets** property in the right pane (**After changes**). Click **Keep**. This instructs the existing step to keep the property, even though it was removed from the method's prototype.



11. In the **After changes** pane, select the old property **Flight_Number** and click **Remove**.

    The **Flight_Number** property is now obsolete. Instead, the method will contain the new, automatically detected property, **Flight_Number_1**.

12. Scroll through the lower section of the wizard screen to see a log of all of the conflicts and their resolutions.

13. Click **Next.** Click **Finish** to close the wizard and return to your test.

Chapter 6: Where To Go From Here

Now that you have learned to create tests with standard activities, Web services, and REST services, and learned how you can UFT's testing functionality to enhance these steps, you can create your own tests for your GUI-less applications.

We appreciate your feedback!

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on API Testing Tutorial (Unified Functional Testing 12.00)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to sw-doc@hp.com.


Java™
COMPATIBLE