Create a data-driven functional test

Contents

Create a data-driven functional test

In this tutorial, you will learn how to create a data-driven functional test using the Rational Functional Tester data-driver wizard.

Data-driven testing puts a layer of abstraction between the data and the test script, eliminating literal values in the test script. Because data is separated from the test script, you can:

- Modify test data without affecting the test script
- Add new test cases by modifying the data, not the test script
- Share the test data with many test scripts

Learning objectives

After completing this tutorial, you will be able to:

- Create a project and record a test script
- Data-drive a test
- Add descriptive headings to the data
- Create a verification point with a datapool reference
- Add data to the datapool
- Play back the test

Time required

30 minutes.

Related information

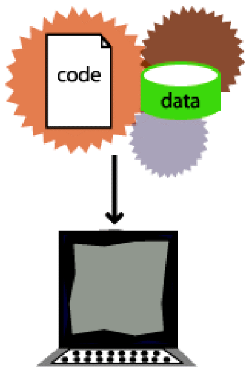
Introduction: Create a data-driven test

In this tutorial, you will learn how to create a data-driven test using a variety of realistic data to test the application with the Rational Functional Tester data-driver wizard.

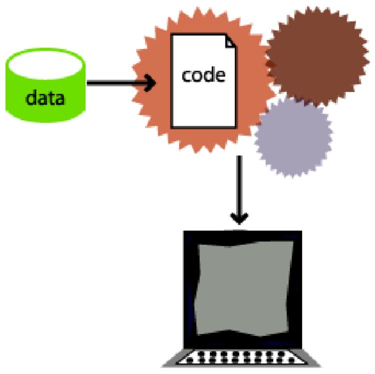
You will use the ClassicsCD sample application to create a project and record a test script to verify that the ClassicsCD sample application correctly totals an order. You will also create a verification point with a datapool reference to check that the total amount of the order is correct in the Classics CD application.

Learn more about datapools: A datapool is a collection of related data records. A datapool supplies data values to the variables in a test script during test script playback. Data-driven testing uses data from an external file, a datapool, as input to a test.

The diagram on the left shows a test script that uses data with hard-coded, literal references in the test script. The diagram on the right shows a data-driven test script that uses data from an external file, a datapool.



Hard-coded test script



Data-driven test script

Learning objectives

After completing this tutorial, you will be able to:

- Create a project and record a test script
- Data-drive a test
- Add descriptive headings to the data
- Create a verification point with a datapool reference
- Add data to the datapool
- Play back the test

Note: Consider printing the tutorial before you begin and using the printed copy as you work through the lessons. You can print the PDF version of the tutorial or print each individual lesson by right-clicking inside each topic and then clicking **Print**.

Time required

This tutorial should take approximately 30 minutes to finish. If you explore other concepts related to this tutorial, it could take longer to complete.

Lesson 1: Creating a project and recording a test script

In this lesson, you will use the Classics CD sample application to create a new project and start recording a test to verify that the sample application correctly totals the amount of music CDs purchased.

What is a project?: A project is a collection of test assets such as test scripts, object maps, verification points, and datapools, that can facilitate the testing of one or more software components. You must create a Functional Tester project before you can record a test.

Creating a project

Create a project to store the test assets that you need to test the Classics CD sample application.

1. Start Functional Tester.
2. Click **File → New → Functional Test Project**.
3. Type DataDriveTutorial for the name of the new project.
4. Click **Finish**.

Starting to record

Start recording a test script to verify that when a customer orders a music CD, the total amount charged to the credit card is the correct amount listed in the application.

1. On the Functional Test toolbar, click **Record a Functional Test Script**( ).
2. Type OrderTotal for the name of the test script.
3. Click **Next**.

   The Select Script Assets page opens.

   When you create a test script, Functional Tester creates a test datapool and other test assets. Use the defaults for **Private Test Datapool** and **Sequential**. A private test datapool is associated with only one script and is not available to any other scripts. When you use the sequential order, the test script accesses datapool records in the order that they appear in the datapool.
4. Click **Finish**.

The Functional Tester window minimizes and the Recording Monitor opens.

Starting the ClassicsCD application

Start the ClassicsCD application and navigate through the application to the dialog box that you will data-drive.

1. On the Recording toolbar, click **Start Application**( ).
2. If necessary, click the **Application Name** arrow to see the options, and then select **ClassicsJavaA - java**.
3. Click **OK**.

   ClassicsJavaA is build 1 of the sample application, ClassicsCD, which comes with Functional Tester. The sample application starts.
4. In the ClassicsCD application, under **Composers**, double-click **Schubert** to open the list of CDs for sale by that composer, and then click **String Quartets Nos. 4 & 14**.
5. Click **Place Order**.
6. Click **OK** to close the Member Logon window.

   The Place an Order window opens.
7. In the ClassicsCD application, type 1234567890 in the **Card Number** field and then type 09/09 in the **Expiration Date** field.

Lesson 2: Data-driving a test

In this lesson, you will use the data-driver to populate a datapool with data from the sample application. A datapool is a collection of related data records. A datapool supplies data values to the variables in a test script during test script playback.

1. On the Recording toolbar, click **Insert Data Driven Commands** ( ).

   The recording pauses and the Insert Data Driven Actions page opens.
2. From the Insert Data Driven Actions page, use the mouse to drag the Object Finder ( ) to the title bar of the **Place an Order** window on the **ClassicsCD** application.

   Functional Tester outlines the entire Place an Order window with a red border.
3. Release the mouse button.

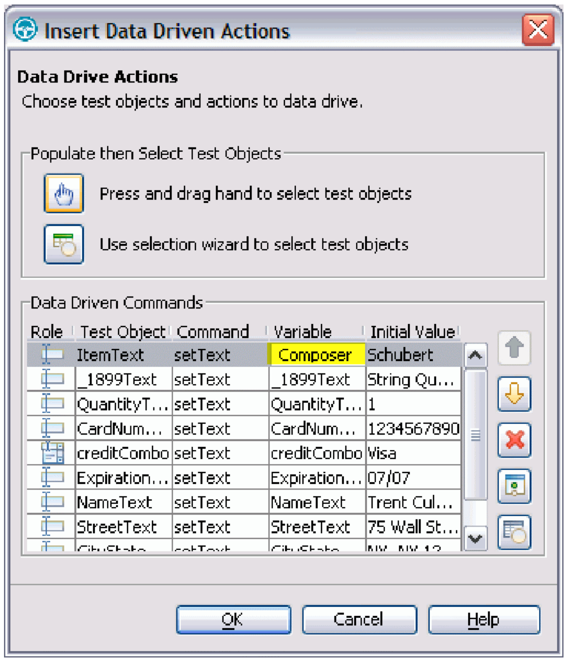
   The Data Drive Actions page opens. In the Data Drive Actions page, under the **DataDriven** Commands table, information appears about the objects that you selected.

You can hover over a row in this table to view the line of code that Functional Tester inserts into the test script to data-drive the test script.

Lesson 3: Adding descriptive headings to the data

In this lesson, you will add descriptive headings to the datapool you created in the previous lesson. Descriptive headings make it easier to add data to the datapool.

1. In the **Data Driven Commands** table, under the **Variable** header, replace **ItemText** with Composer.



Insert Data Driven Actions

Data Drive Actions

Choose test objects and actions to data drive.

Populate then Select Test Objects

Press and drag hand to select test objects

Use selection wizard to select test objects

Data Driven Commands

Role	Test Object	Command	Variable	Initial Value
	ItemText	setText	Composer	Schubert
	_1899Text	setText	_1899Text	String Qu...
	QuantityT...	setText	QuantityT...	1
	CardNum...	setText	CardNum...	1234567890
	creditCombo	setText	creditCombo	Visa
	Expiration...	setText	Expiration...	07/07
	NameText	setText	NameText	Trent Cul...
	StreetText	setText	StreetText	75 Wall St...

2. Repeat sequentially, replacing each cell in the **Variable** column with a descriptive name for each heading in the **Variable** field. Use the text in the following variables list as descriptive names.

   Note: Do not use spaces in **Variable** names. Typically, you would look at the application to determine the appropriate headings for each row, but we have done that for you in the following variables list:

Variable
Composer
Item
Quantity
CardNo
CardType
ExpiryDate
Name
Street
CityStateZip
Phone

   Functional Tester automatically updates the test script as you change each of the **Variable** names.

3. Click **OK**.

   The Insert Data Driven Actions page closes.

Now the datapool has descriptive headings that make it easier to add more data. You will add more data to the datapool after you finish recording the test script.

Lesson 4: Creating a verification point with a datapool reference

In this lesson, you will create a verification point with a datapool reference to check that the total amount due for the order is correct in the Classics CD application.

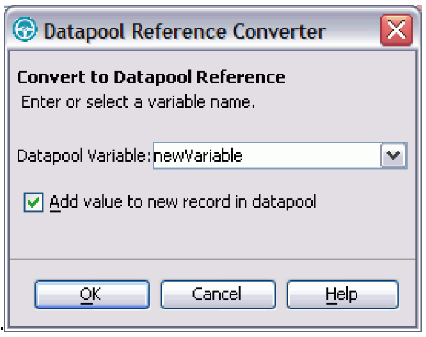
What is a verification point?: A verification point captures object information and literal values from the application-under-test and stores it as the baseline for comparison during playback. When you play back the script, a verification point captures the object information again to compare it to the baseline and see if any changes have occurred, either intentionally or unintentionally. Comparing the actual object information in a script to the baseline is useful for identifying potential defects.

You will use a datapool reference instead of a literal value for the value that you are testing in the verification point. Using datapools with verification points gives you more flexibility to test realistic data with your test scripts.

Create a verification point with a datapool reference

1. On the **Recording** toolbar, click **Insert Verification Point or Action Command** ( ). The Verification Point and Action Wizard opens.
2. From the Verification Point and Action Wizard, use the mouse to drag the **Object Finder** ( ) to $19.99, which is next to "Total" in the Classics CD application.
   Functional Tester outlines $19.99 with a red border.
3. Release the mouse button.
4. If the Select an Action page is not displayed, click **Next**.
5. On the Select an Action page, if necessary, click **Perform Data Verification Point** to test whether the total amount equals the expected amount.
6. Click **Next**.

   The Insert Verification Point Data Command page is displayed.
7. Click **Next**.
8. On the Verification Point Data page toolbar, click **Convert Value to Datapool Reference** ( ) to use a datapool instead of a literal value in a verification point. (If you cannot see the **Convert Value to Datapool Reference** button on the toolbar, make the page larger by dragging a corner of the page).

   
   Datapool Reference Converter

   Convert to Datapool Reference
   Enter or select a variable name.

   Datapool Variable: newVariable

   ☑ Add value to new record in datapool

   The Datapool Reference Converter dialog box opens.
9. In the Datapool Variable field, type Total to replace the **newVariable** for the heading in the datapool.
10. If necessary, select the **Add value to new record in datapool** check box to add the **Total** to the existing datapool record you created in the previous exercise.
11. Click **OK**.
12. Click **Finish**.

Place the order and close the ClassicsCD application

1. In the **ClassicsCD** application click **Place Order** to place the order, and then click **OK** to close the message confirming your order.
2. Click **X** in the upper right corner of the **Classics CD** application to close the application.

Stop recording

On the **Recording** toolbar, click **Stop Recording** (■) to write all recorded information to the test script.

The test script is displayed in the editor window.

Lesson 5: Adding data to the datapool

In this lesson, you will add data to the datapool to test that the ClassicsCD sample application correctly totals each order placed in the application.

1. In the Script Explorer, double-click **Test Datapool** and then double-click **Private Test Datapool**. In the test script editor, double-click the **Test Datapool** tab to expand the datapool editor so that you can work.

   The datapool editor opens and should look similar to the following table:

	Composer	Item	Quantity	Card#	CardType	ExpDate	Name	Street	CityStZip	Phone	Total
0	Schubert	String Quartets Nos. 4 & 14	1	1234567890	Visa	09/09	Trent Culpito	75 Wall St.	Ny, Ny 12212	212-552-1867	$19.99

2. Position your mouse pointer in the datapool editor, right-click, and then click **Add Record**. Click **OK** to add a row after the first row.
3. To add a second empty row, right-click **Add Record**.

   To save time, copy the data from row 0 in the datapool into the two empty rows that you created in steps 2 and 3.
4. Position the mouse pointer in the row 0 cell, right-click, and then click **Copy**.
5. Position the mouse pointer in the row 1 cell, right-click, and then click **Paste**.
6. Click **Yes** to paste the data into the empty row.
7. Position the mouse pointer in the row 2 cell, right-click, and then click **Paste**.
8. Click **Yes** to paste the data into the empty row.
9. Change the value in the **Quantity** and **Total** columns to test that the ClassicsCD sample application correctly totals each order:
   a. In row 1, in the Quantity column, select the cell and type 2.
   b. In row 1, in the Total column, select the cell and type $38.98.
   c. In row 2, in the Quantity column, select the cell and type 3.
   d. In row 2, in the Total column, select the cell and type $57.97.

   The data in the datapool should look like the following table:

	Composer	Item	Quantity	Card#	CardType	ExpDate	Name	Street	CityStZip	Phone	Total		Composer	Item	Quantity	Card#	CardType	ExpDate	Name	Street	CityStZip	Phone	Total
0 75	Schubert	String Quartets Nos. 4 & 14	1	1234567890	Visa	09/09	Trent Culpito	Wall St.	Ny, Ny 12212	212-552-1867	$19.99 75	0	Schubert	String Quartets Nos. 4 & 14	1	1234567890	Visa	09/09	Trent Culpito	Wall St.	Ny, Ny 12212	212-552-1867	$19.99
1 75	Schubert	String Quartets Nos. 4 & 14	*2*	1234567890	Visa	09/09	Trent Culpito	Wall St.	Ny, Ny 12212	212-552-1867	$19.99 75	0	Schubert	String Quartets Nos. 4 & 14	1	1234567890	Visa	09/09	Trent Culpito	Wall St.	Ny, Ny 12212	212-552-1867	*$38.98*

	Composer	Item	Quantity	Card#	CardType	ExpDate	Name	Street	CityStZip	Phone	Total	Composer	Item	Quantity	Card#	CardType	ExpDate	Name	Street	CityStZip	Phone	Total
2 75	Schubert	String Quartets Nos. 4 & 14	*3*	1234567890	Visa	09/09	Trent Culpito	Wall St.	Ny, Ny 12212	212-552-1867	$19.99 75	0	Schubert	String Quartets Nos. 4 & 14	1	1234567890	Visa	09/09	Trent Culpito	Wall St.	Ny, Ny 12212	212-552-1867	$57.97

10. On the **Test Datapool** tab, click **X** to close the datapool editor, and then click **Yes** to save the changes you made to the datapool.

Lesson 6: Playing back the test

In this lesson, you will play back the test you just recorded to see how easy it is to use a variety of data from a datapool to test the application.

Each time you play back a script with an associated datapool, the script accesses one record in the datapool. When you create a datapool reference for a verification point, the verification point uses the datapool reference to access a variable in that record. During playback, Functional Tester substitutes the variable in the datapool for the datapool reference and compares the variable in the datapool to the actual results.

During playback you can view the script name, the script line number that is executing, status icons, and a description of the action in progress in the Playback Monitor.

1. To play back the test script, click **Script→Run**.

   The Select Log window opens.
2. Click **Next**.
3. Click the **Datapool Iteration Count** arrow and then scroll to select **Iterate Until Done** to access all three records in the datapool.
4. Click **Finish** to use the default log name.

   The Functional Tester window minimizes, and the Playback Monitor is displayed in the top right area of your screen. Messages appear in the Playback Monitor as Functional Tester plays back all of the recorded actions in the test script and enters data from the datapool.

   When the test script finishes playing back, Functional Tester displays a log with the test results. A log is a file that contains the record of events that occur while playing back a script. A log includes the results of all verification points executed that can be used to test the application.
5. Click **X** to close the log.

Summary: Create a data-driven test

This tutorial has shown you how to create a data-driven test.

You have created a data-driven test script, created descriptive headings for the data collected, added data to the datapool, created a data verification point with a datapool reference, played back a test script, and viewed the log.

Lessons learned

By completing this tutorial, you learned how to:

∨ Create a project and record a test script
∨ Data-drive a test
∨ Add descriptive headings to the data
∨ Create a verification point with a datapool reference
∨ Add data to the datapool
∨ Play back the test

Additional resources

If you want to learn more about the topics covered in this tutorial, see the Data-Driving Tests section of the Functional Tester Help.

Related information

ibm.com

eclipse.org