IBM Rational University

Rational software

TST279 Essentials of IBM Rational Functional Tester, Java Scripting, v6.1

Student Workbook

Part No. 800-027153-000

IBM Corporation
Rational University
TST279 Essentials of IBM Rational Functional Tester, Java Scripting
Student Workbook
Version 6.1

January 2005

Copyright © International Business Machines Corporation, 2005. All rights reserved.

This document may not be reproduced in whole or in part without the prior written permission of IBM.

The contents of this manual and the associated software are the property of Rational Software and/or its licensors, and are protected by United States copyright laws, patent laws, and various international treaties. For additional copies of this manual or software, please contact Rational Software.
IBM and the IBM logo are trademarks or registered trademarks of IBM Corporation, in the United States, other countries or both.

Rational, the Rational logo, ClearQuest, ClearCase, ClearCase LT, ClearCase MultiSite, Unified Change Management, Rational Developer Network, Rational Rose, Rational XDE, Purify, PureCoverage, Rational Unified Process, ClearDDTS, and Quantify are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries or both.

Microsoft Visual Basic, Windows NT, Windows 2000, Windows 95/98, Windows XP, Microsoft Word, Windows Explorer, DOS, PowerPoint, and Visual SourceSafe, among others, are trademarks or registered trademarks of Microsoft Corporation.

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

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

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

Printed in the United States of America.

This manual prepared by:
IBM Rational Software
18880 Homestead Road
Cupertino, CA 95014-0721
USA

Contents

Lab 0
Become Familiar with the Sample Application

The ClassicsCD application is used in the lab exercises for this course. We will use Functional Tester to test this application.

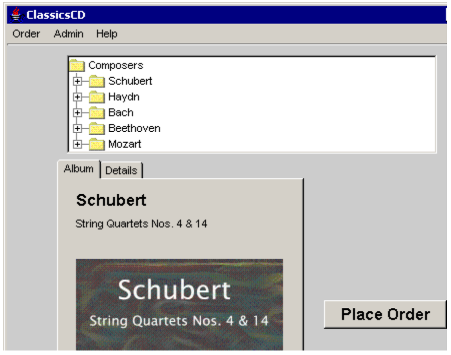
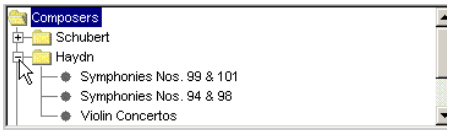
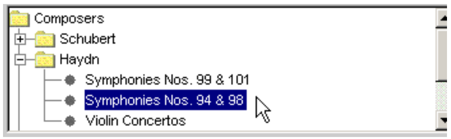
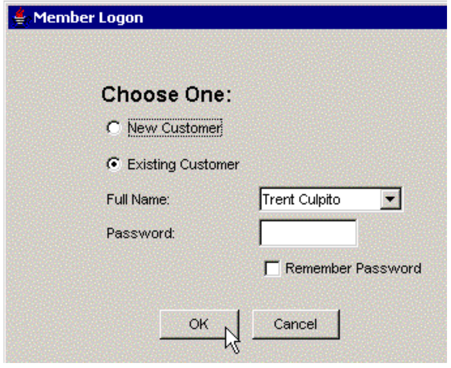
Objective

In this lab, you will start the ClassicsCD application and perform some user actions to get familiar with the application under test.

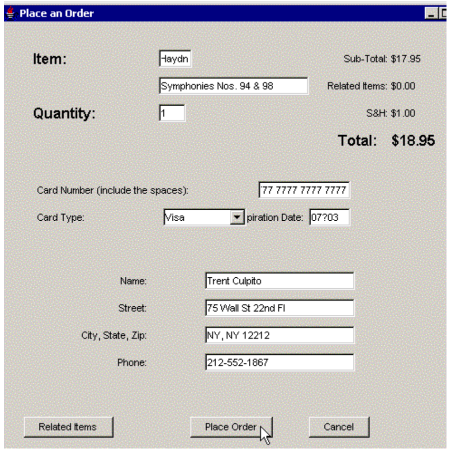
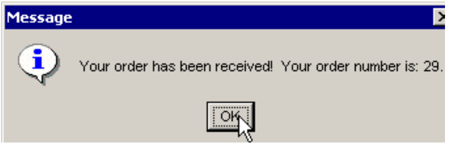
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

0.1 Become Familiar with the ClassicsCD Application

In this exercise, you will start the ClassicsCD application and perform the types of user actions you will record in later exercises with Functional Tester.

Steps	Comments
1. Start the ClassicsCD application. Use the desktop shortcut to ClassicsJavaA. Version A is the build of the application that is ready for testing by using Functional Tester. Later, we will see version B of the application under test.	 ClassicsCD Order Admin Help Composers Schubert Haydn Bach Beethoven Mozart Album Details Schubert String Quartets Nos. 4 & 14 Place Order
2. Click the **+** next to **Haydn** to expand the list.	 Composers Schubert Haydn Symphonies Nos. 99 & 101 Symphonies Nos. 94 & 98 Violin Concertos
3. Click **Symphonies Nos. 94 & 98**.	 Composers Schubert Haydn Symphonies Nos. 99 & 101 Symphonies Nos. 94 & 98 Violin Concertos
4. Click **Place Order**.	
5. In the Member Logon dialog box, keep the default settings of Existing Customer and Trent Culpito. Do not click either of the password fields at this time. Click **OK**.	 New Customer [✓ Existing Customer] Full Name: Trent Culpito Password: ☐ Remember Password

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
6. Enter a credit card number and expiration date. You can type anything in here, for instance, 7777 7777 7777 7777, expiration date 07/07.	 Place an Order Item: Haydn Symphonies Nos. 94 & 98 Quantity: 1 Sub-Total: $17.95 Related Items: $0.00 S&H: $1.00 Total: $18.95 Card Number (include the spaces): 77 7777 7777 7777 Card Type: Visa piration Date: 07?03 Name: Trent Culpito Street: 75 Wall St 22nd Fl City, State, Zip: NY, NY 12212 Phone: 212-552-1867
7. Click **Place Order**.	
8. Click **OK** in the confirmation message box.	 Message Your order has been received! Your order number is: 29.
9. Close the application by clicking the **X** in the upper right corner of the window.	
10. Take about 10 minutes to restart the application and place some other orders.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 1
Getting Started with IBM Rational Functional Tester

In this exercise, you will explore the Functional Tester interface and practice using it. Additionally, you will use a Functional Tester project. This is the project you will use throughout the lab exercises for this course.

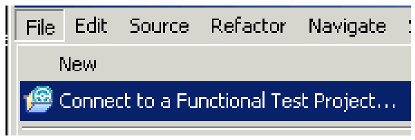
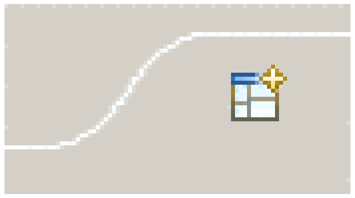
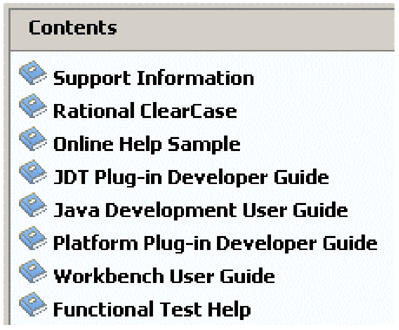
Objectives

In this lab, you will perform the following tasks:

- View Functional Test perspective.
- Record and play back a simple script.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

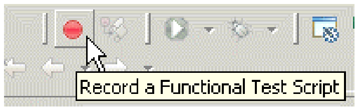
1.1 View the Functional Test Perspective

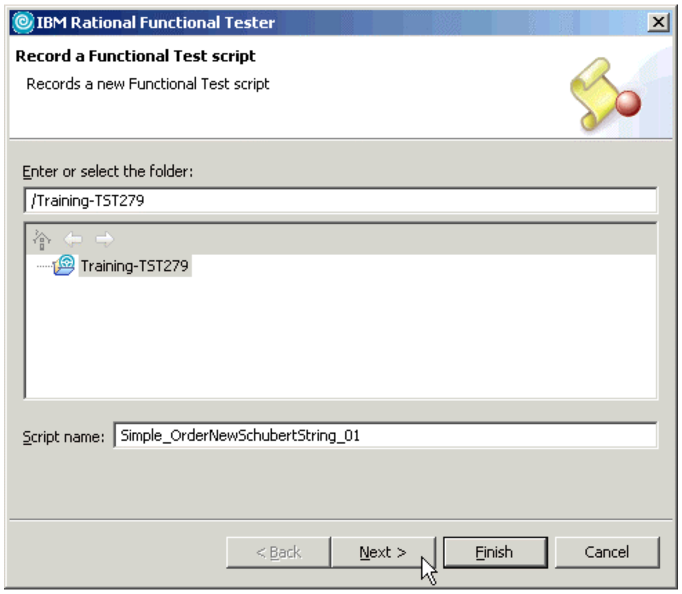
Steps	Comments
1. Click **Start** > **Programs** > **IBM Rational > IBM Rational Functional Tester V6.1** > **Java Scripting**.	
2. In the Workspace Launcher, check the **Use this as the default and do not ask again** checkbox and then click **OK**.	
3. Connect to the **Training-TST279** project, if necessary. If the project does not appear in the **Project** list, click **File > Connect to a Functional Test Project.** Browse to **C:\Training-TST279\Training-TST279** and click **Finish**.	 File Edit Source Refactor Navigate New Connect to a Functional Test Project...
4. If you are not already in the Functional Test perspective, click the **Open a Perspective** button on the right tab. (If the button is not displayed, click **Window** > **Open Perspective** > **Other** > **Functional Test**.)	
5. In the Functional Test Projects Explorer, locate the **Training-TST279** project that was created for you.	
6. Click **Help** > **Help Contents**. Click Rational Functional Tester Help and expand the User Guide book. Practice navigating the Help topics.	 Contents Support Information Rational ClearCase Online Help Sample JDT Plug-in Developer Guide Java Development User Guide Platform Plug-in Developer Guide Workbench User Guide Functional Test Help
7. Close the **Help** window.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

1.2 Become Familiar with the Script Recording and Playback Process

The purpose of this exercise is to introduce you to Functional Tester and step you through the process of recording and playing back a script. This is a preview of what you will be doing in later lab exercises.

Steps	Comments
1. If necessary, start Functional Tester: a. Click **Start** > **Programs** > **IBM Rational** **> IBM Rational Functional Tester V6.1** > **Java Scripting**. b. If you need to, connect to the **Training-TST279** project.	
2. In the Test perspective, click the **Record a Functional Tester Script** icon.	
3. In the **Record a Functional Tester Script** dialog box: a. Select the **Training-TST279** project. b. Name the script Simple_OrderNewSchubertString_01. c. Do not check the **Add the script to Source Control** checkbox if it is displayed. d. Click **Next**.	


IBM Rational Functional Tester

Record a Functional Test script

Records a new Functional Test script

Enter or select the folder:

/Training-TST279

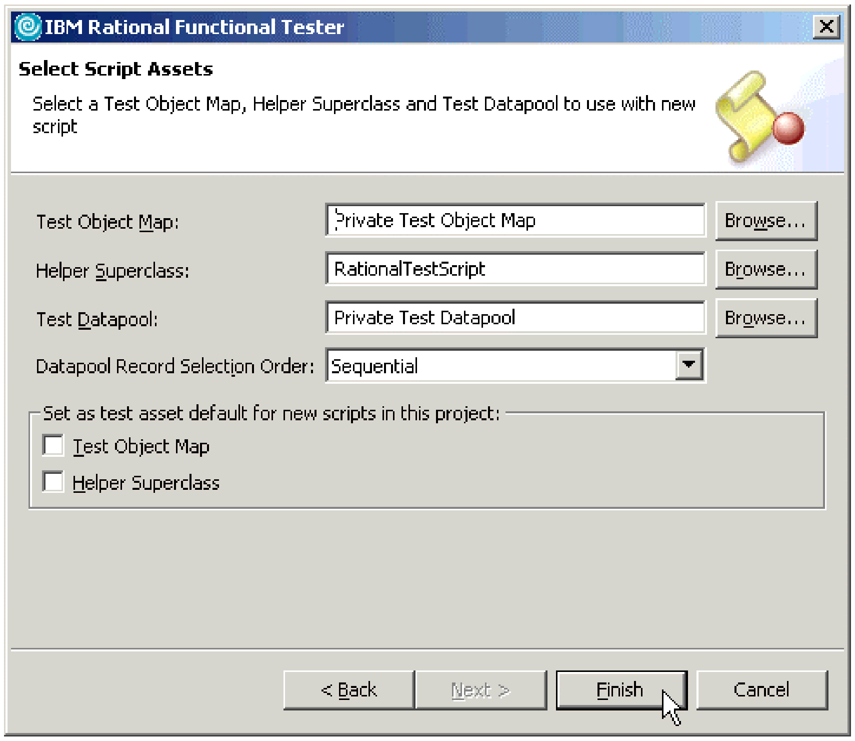
Training-TST279

Script name: Simple_OrderNewSchubertString_01

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
4. In the **Select Script Assets** dialog box, make sure **Private Test Object Map** is selected in the Test Object Map field. Click **Finish**.	We will discuss test object maps later.


IBM Rational Functional Tester

Select Script Assets

Select a Test Object Map, Helper Superclass and Test Datapool to use with new script

Test Object Map: Private Test Object Map

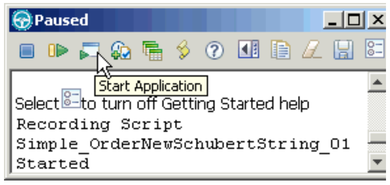
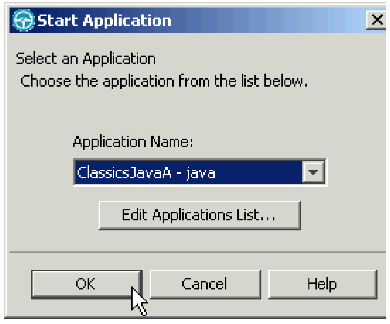
Helper Superclass: RationalTestScript

Test Datapool: Private Test Datapool

Datapool Record Selection Order: Sequential

Set as test asset default for new scripts in this project:

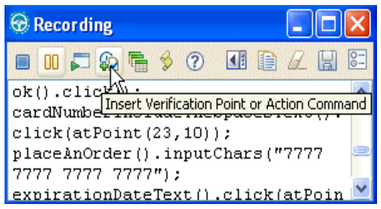
- [ ] Test Object Map
- [ ] Helper Superclass

Steps	Comments
5. You are now ready to begin recording a script. In the Functional Test Recording Monitor, click **Start Application**.	 Paused Select to turn off Getting Started help Recording Script Simple_OrderNewSchubertString_01 Started
6. In the **Start Application** dialog box, select **ClassicsJavaA – java** and click **OK**.	 Start Application Select an Application Choose the application from the list below. Application Name: ClassicsJavaA - java

© Copyright IBM Corp. 2005

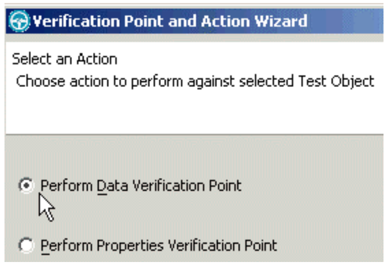
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

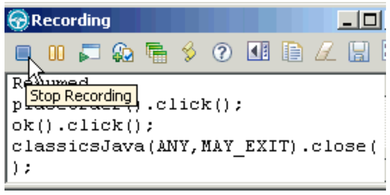
Steps	Comments
7. Perform the following user actions to place an order in the ClassicsCD application: a. Click the **+** next to **Schubert** to expand the list. b. Click **String Quartets Nos. 4 & 14**. c. Click **Place Order**. d. In the Member Logon dialog box, keep the default settings of Existing Customer and Trent Culpito. Do not click either of the password fields. Click **OK**. e. Type *7777 7777 7777 7777* in the Card Number field. f. Type 07/07 in the Expiration Date field.	These are the same actions you performed in Exercise 0.1.

8. In the Recording Monitor, click the **Insert Verification Point or Action Command** button. The script pauses, and the Verification Point and Action Wizard dialog box opens.	
	`ok().clic`
	`cardNumbe`
	`click(atPoint(23,10));`
	`placeAnOrder().inputChars("7777`
	`7777 7777 7777");`
	`expirationDateText().click(atPoin`

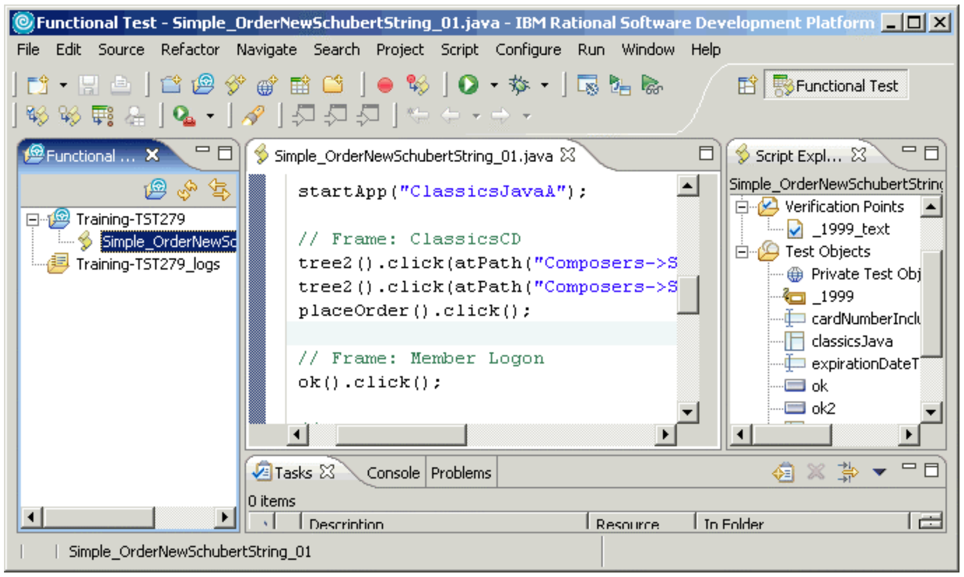
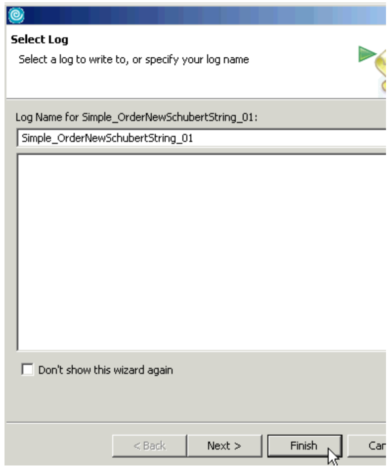
9. Clear the **After selecting an object advance to next page** check box.	
10. To insert a verification point, you must first select the object to test. a. Grab the **Object Finder** tool icon by holding down your mouse over the icon. b. Select the object to test. Drag the **Object Finder** tool icon over the dollar amount displayed as the total for the order in the application. c. Release the mouse button. d. Click **Next**.	The Insert Verification Point or Action Command dialog disappears. The object is outlined with a red border, and the object name is displayed.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
11. In the Select an Action dialog, click **Next**.	The Perform Data Verification Point radio button is already selected.  Verification Point and Action Wizard Select an Action Choose action to perform against selected Test Object (●) Perform Data Verification Point ( ) Perform Properties Verification Point
12. In the Select an Action dialog, click **Next** and then **Finish**.	The verification point is recorded, and the script recording resumes.
13. Click the **Place Order** button.	
14. Click **OK** on the dialog that indicates that your order has been received.	
15. Click the **X** in the upper right corner of the ClassicsCD window to close the application.	

16. Click the **Stop Recording** button in the Functional Tester Recording Monitor.	 Recording
	Stop Recording
	`.click();`
	`ok().click();`
	`classicsJava(ANY,MAY_EXIT).close(`
	`);`

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
17. In the Test perspective, your script is listed in the Project Explorer, and its contents are displayed in the editor pane. Its test objects are shown in the Script Explorer.	We will look closely at the contents of Functional Tester scripts in later modules.
	
18. Now you are going to play back the script you just recorded. On the Functional Tester toolbar, click the **Runs Functional Tester script** icon and select **Simple_OrderNewSchubertString_01**.	
19. In the **Select Log** dialog box, click **Finish** to accept the default settings.	

```
startApp("ClassicsJavaA");

// Frame: ClassicsCD
tree2().click(atPath("Composers->S
tree2().click(atPath("Composers->S
placeOrder().click();

// Frame: Member Logon
ok().click();
```

Select Log

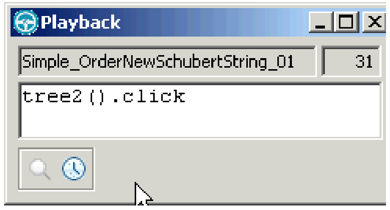
Select a log to write to, or specify your log name

Log Name for Simple_OrderNewSchubertString_01:

Simple_OrderNewSchubertString_01

☐ Don't show this wizard again

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
20. Observe the playback actions and the messages displayed in the Functional Tester Playback Monitor.	 Playback Simple_OrderNewSchubertString_01 31 `tree2().click`
21. When playback finishes, examine the test log. If the log does not appear automatically, expand the Training-TST279_logs directory in the Functional Test Projects Explorer and then double-click the **Simple_OrderNewSchubertString_01** log.	We will discuss logs in a later module. For now, just become familiar with the types of information contained in the log.
22. Close the test log.	
23. Close the Simple_OrderNewSchubertString_01 script.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 2
Recording a Script

In this lab, you record scripts that help with the challenges discussed in Module 2. In later labs, you will learn more about structuring scripts and debugging them when they fail.

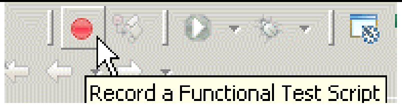
Objectives

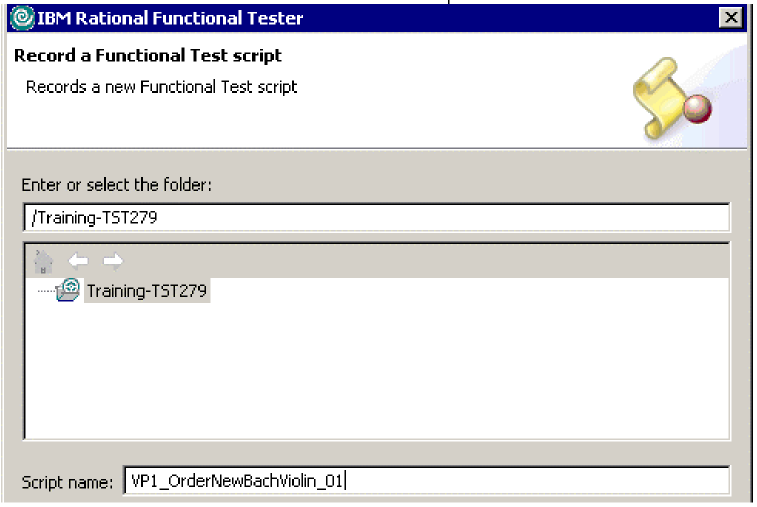
In this lab, you will perform the following tasks:

- Record a Data verification point.
- Record a Properties verification point.
- Include script support functions in a script.
- Include a timer in a script.
- Insert recording into a script.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.1 Record a Data Verification Point

Steps	Comments
1. In the Functional Test perspective, click the **Record a Functional Test Script** icon.	 Record a Functional Test Script
2. In the **Record a Functional Test Script** dialog box: a. Select the **Training-TST279** project. b. Name the script **VP1_OrderNewBachViolin_01**. c. Do not check the **Add the script to Source Control** checkbox if it is displayed. d. Click **Next**.	



IBM Rational Functional Tester

Record a Functional Test script

Records a new Functional Test script

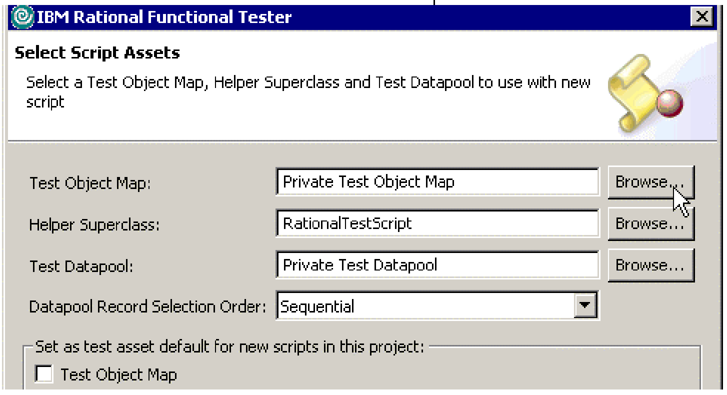
Enter or select the folder:

/Training-TST279

Training-TST279

Script name: VP1_OrderNewBachViolin_01

Steps	Comments
3. In the **Select Script Assets** dialog box, click the Test Object Map **Browse** button.	



IBM Rational Functional Tester

Select Script Assets

Select a Test Object Map, Helper Superclass and Test Datapool to use with new script

Test Object Map: Private Test Object Map Browse...

Helper Superclass: RationalTestScript Browse...

Test Datapool: Private Test Datapool Browse...

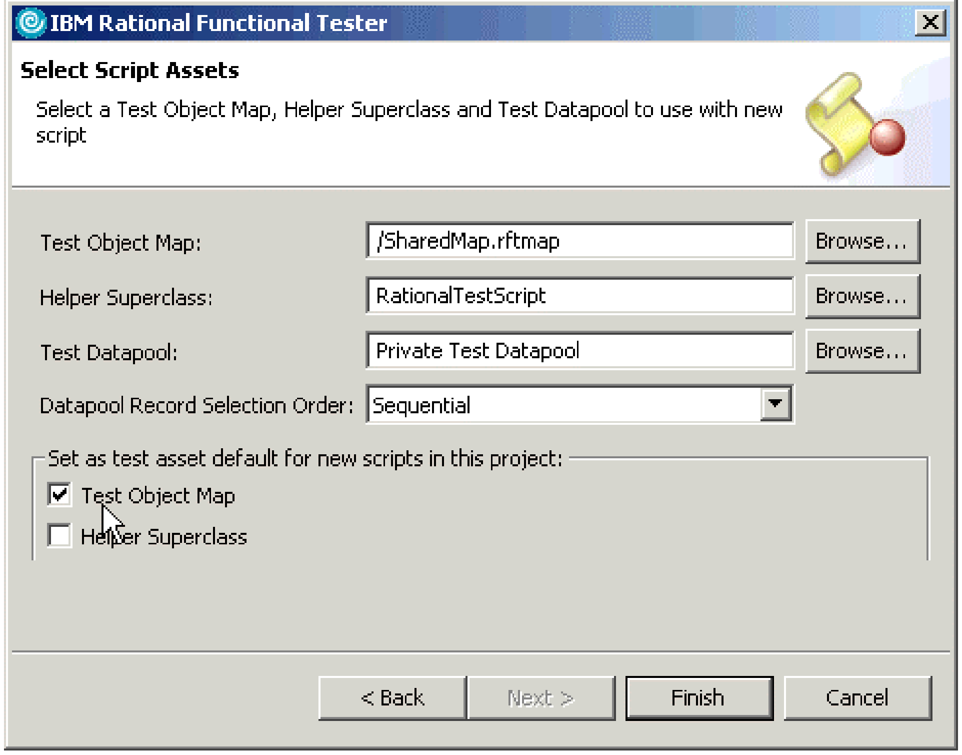
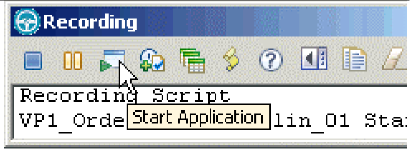
Datapool Record Selection Order: Sequential

Set as test asset default for new scripts in this project:

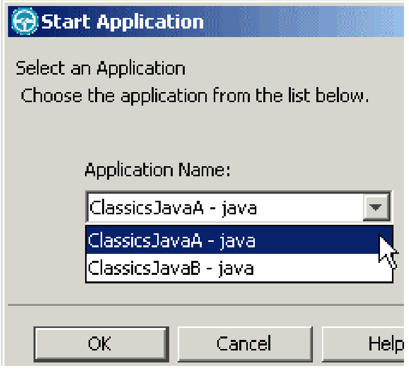
☐ Test Object Map

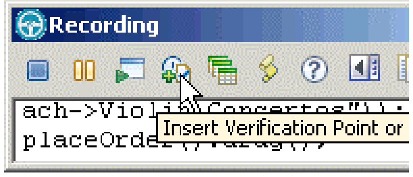
© Copyright IBM Corp. 2005

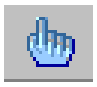
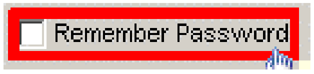
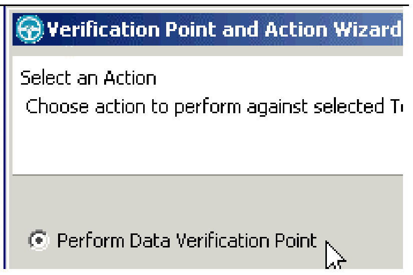
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
4. In the Select Test Object Map dialog box, select the /**SharedMap.rftmap** icon and then click **OK**. In the **Set as test asset default for new scripts in this project** section, select the checkbox for **Test Object Map**.	
 IBM Rational Functional Tester **Select Script Assets** Select a Test Object Map, Helper Superclass and Test Datapool to use with new script Test Object Map: /SharedMap.rftmap Helper Superclass: RationalTestScript Test Datapool: Private Test Datapool Datapool Record Selection Order: Sequential Set as test asset default for new scripts in this project: ☑ Test Object Map ☐ Helper Superclass 	
5. In the Select Script Assets dialog box, click **Finish**.	
6. In the Functional Test Recording Monitor, click **Start Application**.	 Recording Script VP1_Orde lin_01 Sta

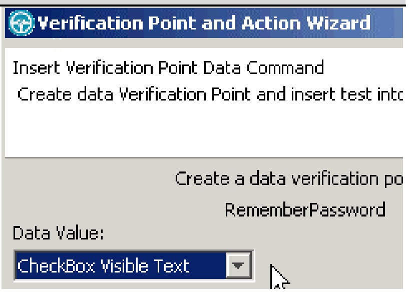
© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
7. In the Start Application dialog box, select **ClassicsJavaA - java** and click **OK**.	 Start Application Select an Application Choose the application from the list below. Application Name: ClassicsJavaA - java ClassicsJavaA - java ClassicsJavaB - java OK Cancel Help
8. Expand the **Bach** folder and click **Violin Concertos**.	
9. Click the **Place Order** button.	

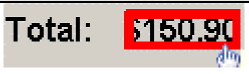
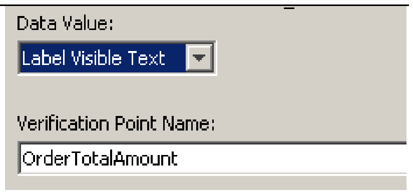
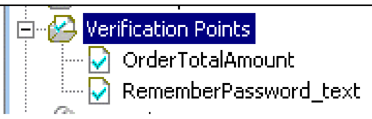
10. In the Recording Monitor, click the **Insert Verification Point or Action Command** button.	 Recording
	ach->Violin Concertos"));
	placeOrder().click();
	Insert Verification Point or

11. To insert a verification point, you must first select the object to test. a. Grab the **Object Finder** tool icon by holding down your mouse over the icon. b. Drag the icon over the **Remember Password** text. c. Release the mouse button. d. Click **Next**.	 The object is outlined with a red border, and the object name is displayed.  Remember Password
12. Now you must specify the action to record. Perform Data Verification Point should already be selected. If not, click it and then click **Next**.	 Verification Point and Action Wizard Select an Action Choose action to perform against selected T Perform Data Verification Point

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

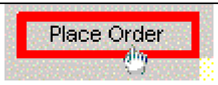
Steps	Comments
13. In the Data Value list, select **CheckBox Visible Text** and then click **Next**.	 Verification Point and Action Wizard Insert Verification Point Data Command Create data Verification Point and insert test into Create a data verification po RememberPassword Data Value: CheckBox Visible Text
14. In the next screen, the object's recognition properties are displayed. Click **Finish**.	You can see the verification point recorded in the Recording Monitor.
15. In the ClassicsCD application, click **OK** on the Member Logon dialog.	
16. Perform the following user actions in the ClassicsCD application: a. Click the **Quantity** field. b. Press the **Home** key. c. Hold down the **Shift** key and press the **End** key. d. Press the **Delete** key. e. Enter 10 as the quantity. f. Press the **Tab** key. (**Note**: If the tabbing order in the application changes, the script may fail. The alternative is to click in the desired field.) g. Type 1234 1234 1234 1234 in the **Card Number** field. h. Press the **Tab** key twice to get to the **Expiration Date** field. i. Type 12/07 in the **Expiration Date** field.	
17. In the Recording Monitor, click the **Insert Verification Point or Action Command** button to record another verification point.	
18. If it is not already checked, check the **After selecting an object advance to the next page** option.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

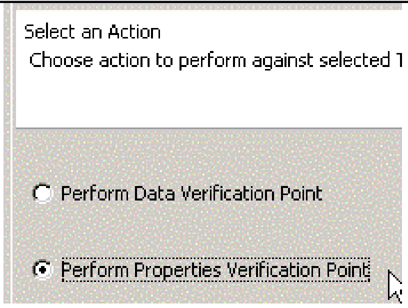
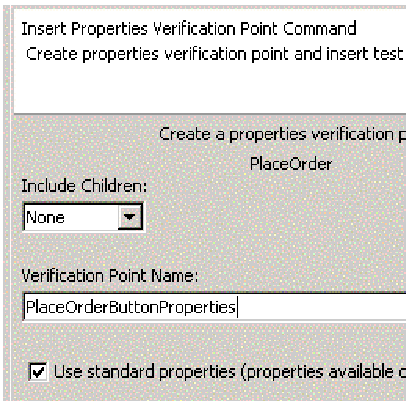
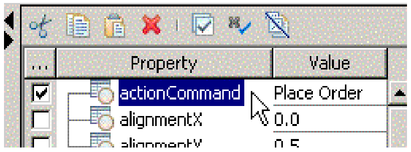
Steps	Comments
19. Select the object to test. Drag the **Object Finder** tool icon over the dollar amount displayed as the total for the order in the application and then release the mouse button. Click **Next**.	 Total: $150.90
20. In the Insert Verification Point Data Command page, change the **Verification Point Name** to **OrderTotalAmount**. Click **Next**.	 Data Value: Label Visible Text Verification Point Name: OrderTotalAmount
21. In the Verification Point Data page, click **Finish**.	The verification point is recorded.
22. In the application, click the **Place Order** button.	
23. Click **OK** on the dialog that indicates your order has been received.	
24. Close the ClassicsCD window and then stop recording.	
25. In the Functional Test perspective, find your new script listed in the Functional Test Projects view and double-click to open it in the Java editor.	A later lab exercise will use this script.
26. Now, locate your verification points in the script.	The verification point lines end with performTest(); and you should see this in the script.
27. Locate your verification points in the Script Explorer listing.	 Verification Points OrderTotalAmount RememberPassword_text
28. Close the VP1_OrderNewBachViolin_01 script.	

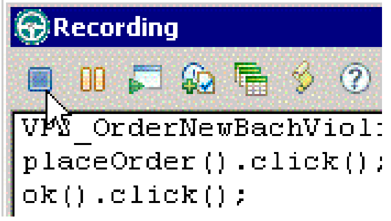
© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.2 Record a Properties Verification Point

Steps	Comments
1. Record a new script. a. In the Functional Test perspective, click the **Record a Functional Test Script** icon. b. In the **Record a Functional Test Script** dialog box, select the **Training-TST279** project. c. Name this script **VP2_OrderNewBachViolin_02**. d. Do not check the **Add the script to Source Control** checkbox if it is displayed. e. Click **Finish.** (The SharedMap is used by default.)	
2. Start the ClassicsJavaA application.	
3. Expand the **Bach** folder and click **Violin Concertos**.	
4. Click the **Place Order** button.	
5. Click **OK** on the Member Logon dialog.	
6. In the ClassicsCD application, perform the following user actions: a. Click in the **Card Number** field. b. Type 1234 1234 1234 1234 in the **Card Number** field. c. Click in the **Expiration Date** field. d. Type 12/07 in the **Expiration Date** field.	**Quantity** field should already have a value of 1.
7. In the Recording Monitor, click the **Insert Verification Point or Action Command** button.	
8. Select the object to test. Drag the **Object Finder** tool icon over the **Place Order** button and then release the mouse button.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
9. In the Select an Action page, click the **Perform Properties Verification Point** option to indicate that you are recording a Properties verification point and then click **Next**.	 Select an Action Choose action to perform against selected 1 Perform Data Verification Point [✓ Perform Properties Verification Point]
10. In the Insert Properties Verification Point Command page, you will define the properties to test for the **Place Order** button. a. The **Include Children** field should indicate **None**. b. Change the **Verification Point Name** to **PlaceOrderButtonProperties**. c. Click **Next.** d. Resize the Verification Point and Action Wizard window as necessary to see the **Property** and **Value** text in the right-hand pane. e. Find the **actionCommand** property and check the checkbox for it. f. Find the **enabled** property and check the checkbox for it. g. Click **Finish**.	 Insert Properties Verification Point Command Create properties verification point and insert test Create a properties verification p PlaceOrder Include Children: None Verification Point Name: PlaceOrderButtonProperties ✓ Use standard properties (properties available c  Property | Value ✓ actionCommand | Place Order alignmentX | 0.0 Do not check all of the properties. Only the properties of interest for the test should be checked.
11. In the AUT, click the **Place Order** button.	

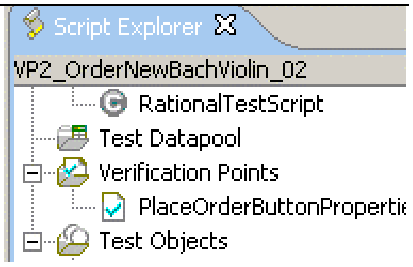
12. Click **OK** on the dialog that indicates your order has been received.
Close the ClassicsCD application and then stop recording.
	 Recording
	VP_OrderNewBachViol:
	placeOrder().click();
	ok().click();

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

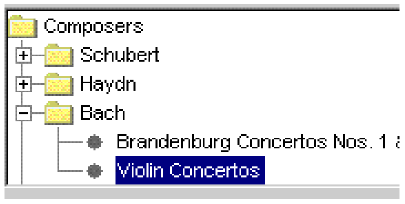
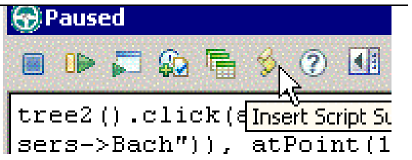
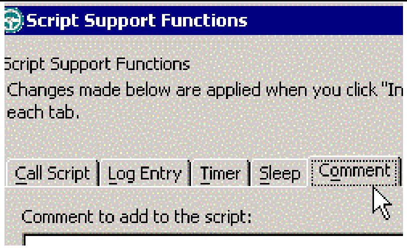
13. In the Functional Test perspective, find your new script listed in the Projects view.	
14. Locate your verification point in the script.	

```
PlaceOrderButtonPropertiesVP().performTest();
```

15. Locate your verification point in the Script Explorer listing.	 Script Explorer VP2_OrderNewBachViolin_02 RationalTestScript Test Datapool Verification Points PlaceOrderButtonPropertie Test Objects
16. Close the VP2_OrderNewBachViolin_02 script.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.3 Include Script Support Functions in a Script

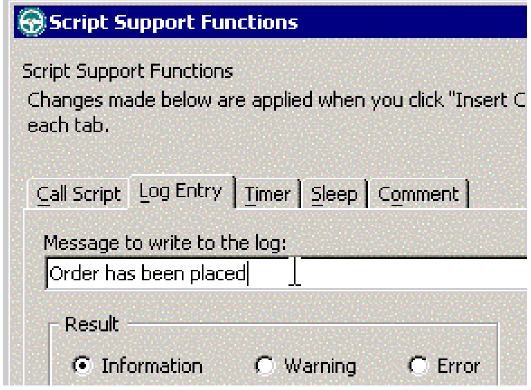
Steps	Comments
1. Record a new script. a. In the Functional Test perspective, click the **Record a Functional Test script** icon. b. In the **Record a Functional Test script** dialog box, select the **Training-TST279** project. c. Name this script SCRIPTSUPPORT_OrderNewBachViolin_03. d. Do not check the **Add the script to Source Control** checkbox if it is displayed. e. Click **Finish**.	
2. Start the ClassicsJavaA application.	
3. Perform the following user actions: a. Click the **+** next to **Bach** to expand the list. b. Click **Violin Concertos**. c. Click **Place Order**.	 Composers Schubert Haydn Bach Brandenburg Concertos Nos. 1 & Violin Concertos
4. In the Recording Monitor, click the **Insert Script Support Commands** button.	 tree2().click(a sers->Bach")), atPoint(1
5. In the Script Support Functions dialog box, click the **Comment** tab.	 Script Support Functions Call Script Log Entry Timer Sleep Comment Script Support Functions Changes made below are applied when you click "In each tab. Comment to add to the script:
6. Type a comment, such as "Logon." Click **Insert Code** and then click **Close**.	The comment is inserted into your script.
7. Back in the application, click the **OK** button.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
8. In the Recording Monitor, click the **Insert Script Support Commands** button.	
9. In the **Script Support Functions** dialog box, click the **Comment** tab.	
10. Type a comment, such as "Enter credit card number." Click **Insert Code** and then click **Close**.	
11. In the ClassicsCD application, type 1234 1234 1234 1234 in the **Card Number** field.	
12. Use the Script Support Functions to insert a comment, such as "Select AMEX as the credit card type." Click **Insert Code** and then click **Close**.	
13. Perform the following user actions in the ClassicsCD application: a. Select **AMEX** from the pull-down menu in the **Card Type** field. b. Type 12/07 in the **Expiration Date** field. c. Click the **Place Order** button.	
14. In the Recording Monitor, click the **Insert Verification Point or Action Command** button.	
15. Drag the **Object Finder** tool icon over the Message dialog so that the whole message window is selected and then release the mouse button. Click **Next**.	In the Select an Action page, notice that the **Perform Properties Verification Point** option is selected by default.
16. Define the properties to test for the confirmation dialog: a. The **Include Children** field should indicate **None**. b. Change the **Verification Point Name** to **DialogTitle**. c. Click **Next.** d. Find the **title** property and check the checkbox for it. e. Click **Finish**.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
17. In the ClassicsCD application, click **OK** on the dialog that indicates your order has been received.	
18. Click the **Insert Script Support Commands** button, click the **Log Entry** tab, and insert "Order has been placed" as an informational log entry into your script.	(Click **Insert Code** and then click **Close**.)

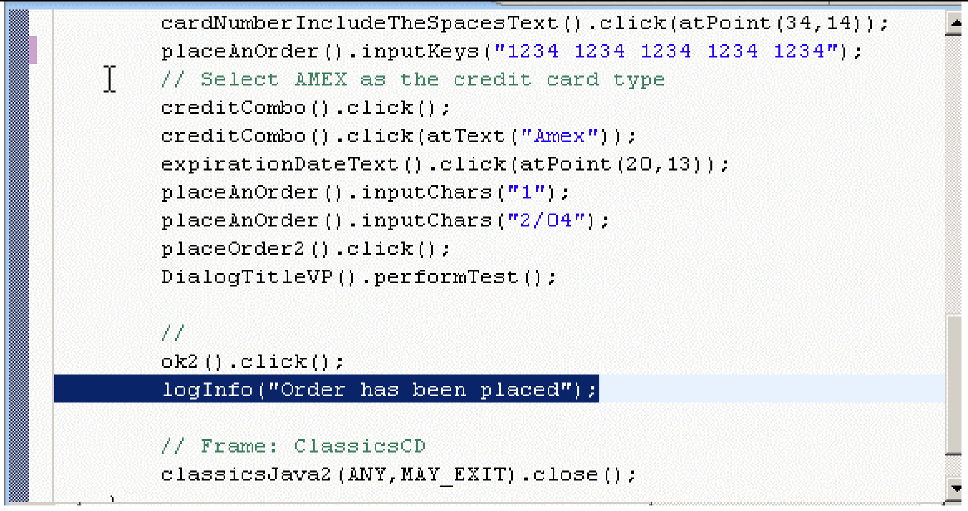

Script Support Functions

Script Support Functions
Changes made below are applied when you click "Insert C
each tab.

Message to write to the log:
Order has been placed

Result
[✓ Information] Warning Error

Steps	Comments
19. Close the ClassicsCD window and then stop recording.	
20. In the Functional Test perspective, find your new script listed in the Projects view. In the Java editor, locate your comments and log entry in the script.	

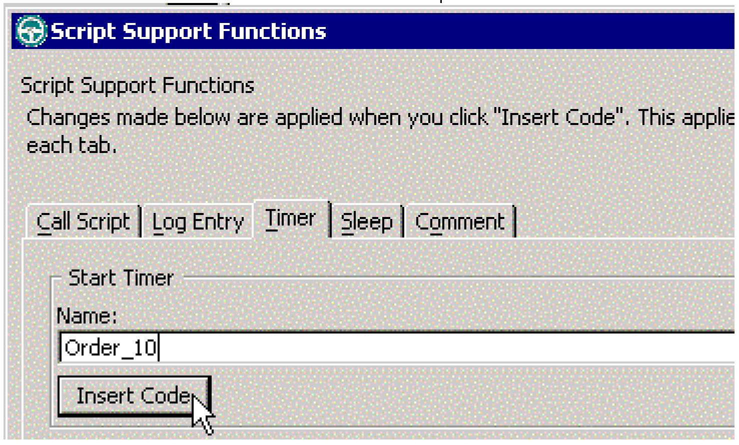

```
cardNumberIncludeTheSpacesText().click(atPoint(34,14));
placeAnOrder().inputKeys("1234 1234 1234 1234 1234");
// Select AMEX as the credit card type
creditCombo().click();
creditCombo().click(atText("Amex"));
expirationDateText().click(atPoint(20,13));
placeAnOrder().inputChars("1");
placeAnOrder().inputChars("2/04");
placeOrder2().click();
DialogTitleVP().performTest();

//
ok2().click();
logInfo("Order has been placed");

// Frame: ClassicsCD
classicsJava2(ANY,MAY_EXIT).close();
```

Steps	Comments
21. Close the SCRIPTSUPPORT_OrderNewBachViolin_03 script.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.4 Include a Timer in a Script

Steps	Comments
1. Record a script called **TIMER_OrderNewSchubertString_02**.	
2. Start the **ClassicsJavaA** application.	
3. Expand the **Schubert** folder and click **String Quartets Nos. 4 & 14**. a. Click **Place Order**. b. Log in as Trent Culpito.	
4. In the Recording Monitor, click the **Insert Script Support Commands** button.	
5. In the **Script Support Functions** dialog box, click the **Timer** tab.	
6. In the **Start Timer** field, enter **Order_10** as the name. Click **Insert Code** and then click **Close**.	Make sure you put the underscore between Order and 10.


Script Support Functions

Script Support Functions

Changes made below are applied when you click "Insert Code". This applie each tab.

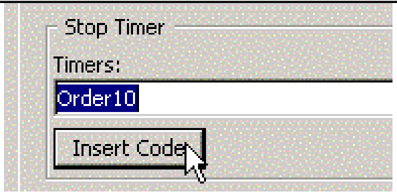
Start Timer

Name: Order_10

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
7. Perform the following user actions in the ClassicsCD application: a. Click the **Quantity** field. b. Press the **Home** key. c. Hold down the **Shift** key and press the **End** key. d. Press the **Delete** key. e. Enter 10 as the quantity. f. Press the **Tab** key. (**Note**: If the tabbing order in the application changes, the script may fail. The alternative is to click in the desired field.) g. Type 1234 1234 1234 1234 in the **Credit Card number** field. h. Click in the **Expiration Date** field. i. Type 12/07 in the **Expiration Date** field.	
8. In the Recording Monitor, click the **Insert Verification Point or Action Command** button.	
9. Select the object to test. Drag the **Object Finder** tool icon over the dollar amount displayed as the total for the order in the application and then release the mouse button.	
10. Create a Data verification point and name it TotalSchubertString10. Click **Next** and then **Finish**.	
11. In the ClassicsCD application: a. Click the **Place Order** button. b. Click **OK** on the dialog that indicates your order has been received.	
12. In the Recording Monitor, click the **Insert Script Support Commands** button again.	
13. In the **Script Support Functions** dialog box, click the **Timer** tab again.	Now you are going to stop the timer.

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
14. In the **Stop Timer** field, select the **Order_10** timer. Click **Insert Code** and then click **Close**.	
15. Close the ClassicsCD window and then stop recording.	
16. In the Functional Test perspective, find your new script listed in the Projects view.	A later lab exercise will use this script.

17. Locate your timer code in the script.	There will be two lines, one for the start and one for the stop.
	`timerStart("Order_10");`
	.
	.
	.
	`timerStop("Order_10");`

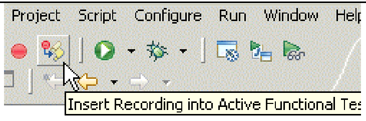
18. Close the TIMER_OrderNewSchubertString_02 script.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

2.5 Insert Recording into a Script

Steps	Comments
1. In the Projects view, double-click the **VP2_OrderNewBachViolin_02** script to select and open it.	

2. In the Java editor, insert a blank line right after the input for the Expiration Date of 12/07 and **type stop();** on the new line.	`// Frame: Place an Order`
	`cardNumberIncludeTheSpac`
	`placeAnOrder().inputKeys`
	`placeAnOrder().inputChar`
	`expirationDateText().cli`
	`placeAnOrder().inputKeys`
	`stop();`

3. Run the script using the default log information.	
4. Examine the test log and then close it.	Notice that the application is now in the desired state to add a VP to check the order total.
5. Delete the **`stop();` line** from the script.	This stop example is one instance of a handy coding feature.
6. Position the cursor in a blank line immediately following the input for the Expiration Date of 12/07.	
7. Click the **Insert Recording into Active Functional Test script** button.	 Project Script Configure Run Window Help Insert Recording into Active Functional Tes
8. Create a Data verification point. a. Drag the icon over the total amount for this order. b. Name the verification point TotalBachViolin01.	Pause the recording, if necessary, to make the **Place an Order** window have focus.
9. Stop recording.	
10. Cancel the Object Map: Help Page and close the object map dialog, if necessary.	
11. Cancel the order in the ClassicsCD window, and then close the ClassicsCD application.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
12. In the Functional Test perspective, look at the new code that has been added to the VP2_OrderNewBachViolin_02 script.	Note that the entire script did not have to be recorded again. The new recording placed the code in the desired location to validate the amount of the order.
13. Close the VP2_OrderNewBachViolin_02 script, saving your changes.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 3 Playing Back a Script and Viewing Results

After you record a script, play it back on the same build of the system-under-test to ensure that it performs properly. A script should play back without error on the same software build on which it was recorded.

Before playback, always verify and reset the playback environment for testing consistency.

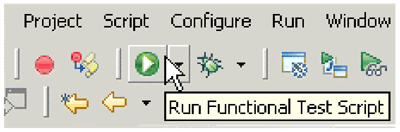
Objectives

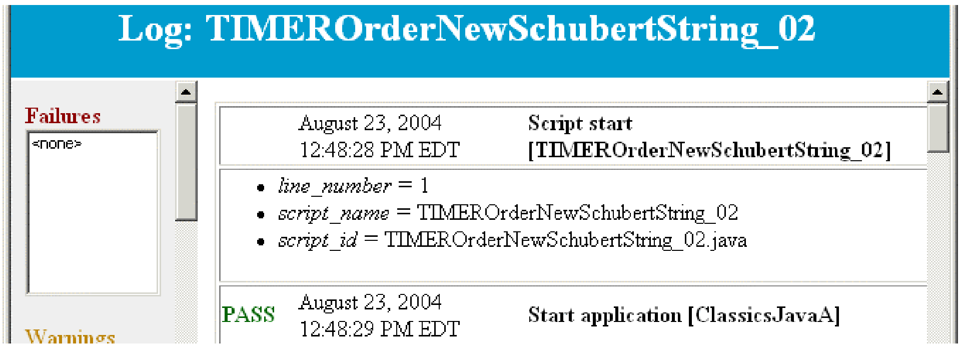
In this lab, you will perform the following tasks:

- Play back a script and view results.
- View a specific log.
- View results from a script containing a verification point.
- Use the Verification Point Comparator.
- Insert a breakpoint into a script.
- Set Functional Tester options and preferences.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3.1 Play Back a Script and View Results

Steps	Comments
1. Display the Functional Test perspective.	
2. In the Functional Test Projects view, select the **TIMER_OrderNewSchubertString_02** script by double-clicking its name.	This is the script containing the timer. You must double-click it to open the script in the java editor.
3. On the Functional Test toolbar, click the **Run Functional Test script** icon to play back the selected script.	 Project Script Configure Run Window Run Functional Test Script
4. In the Select Log dialog box, use the defaults. Click **Finish**.	
5. When playback finishes, view the log. Scroll through the log to see all information.	



Log: TIMEROrderNewSchubertString_02

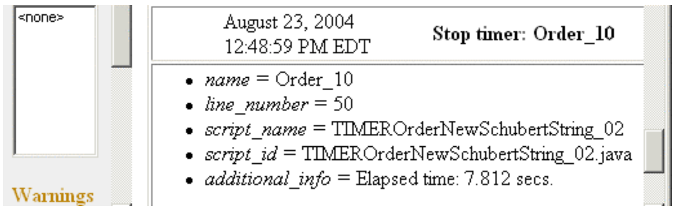
Failures
<none>
Warnings

August 23, 2004
12:48:28 PM EDT
Script start
[TIMEROrderNewSchubertString_02]

- *line_number* = 1
- *script_name* = TIMEROrderNewSchubertString_02
- *script_id* = TIMEROrderNewSchubertString_02.java

PASS
August 23, 2004
12:48:29 PM EDT
Start application [ClassicsJavaA]

Steps	Comments
6. Scroll through the log to the **Stop timer** event. Note the kinds of information provided for each event. Specifically, look at the **Additional Information** property that shows the elapsed time that the timer calculated.	It is okay if your elapsed time is different from what is in the example.



<none>
Warnings

August 23, 2004
12:48:59 PM EDT
Stop timer: Order_10

- *name* = Order_10
- *line_number* = 50
- *script_name* = TIMEROrderNewSchubertString_02
- *script_id* = TIMEROrderNewSchubertString_02.java
- *additional_info* = Elapsed time: 7.812 secs.

Steps	Comments
7. Close the Log Event window.	
8. Close the TIMER_OrderNewSchubertString_02 script.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3.2 View a Specific Log

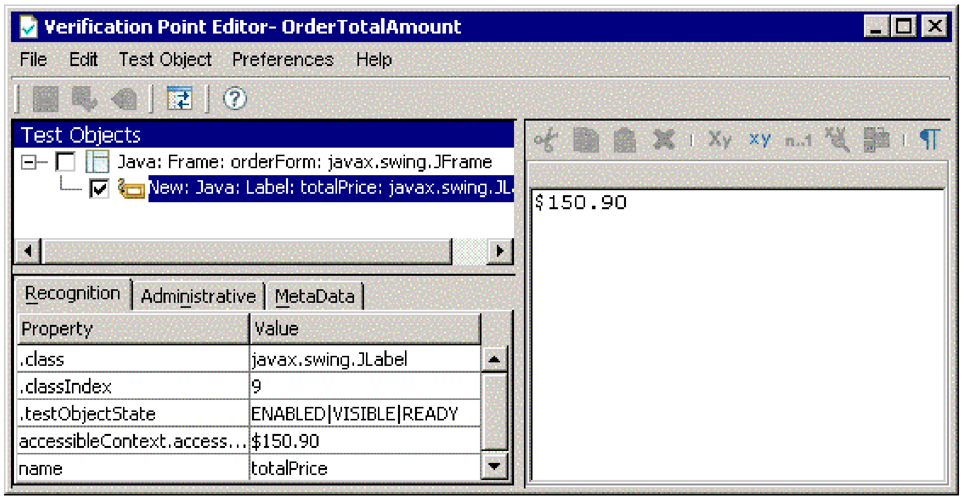
You may need to view a log from an earlier playback. In this exercise, you will select and open a specific HTML log.

Steps	Comments
1. In the Functional Test Projects view, expand the **Training-TST279_logs** icon.	
2. In the list of logs, double-click **Simple_OrderNewSchubertString_01** to display the log.	The contents of the HTML log open in a new window.
3. Now double-click the name of a different log.	This is how you select a specific log to be displayed.
4. Close all open log windows.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3.3 View Playback Results from a Script Containing a Verification Point

Steps	Comments
1. Run the VP1_OrderNewBachViolin_01 script.	This is the script containing a verification point that we will examine closely.
2. In the Select Log dialog box, use the defaults. Click **Finish**.	
3. Observe the playback actions and the messages displayed in the Functional Tester Playback Monitor.	
4. Scroll through the log to locate your verification points.	
5. Look at the information for the OrderTotalAmount verification point. Click the **View Results** hyperlink for this verification point.	This starts the Verification Point Editor.
6. Observe the value for the verification point in the right-hand pane.	



Java: Frame: orderForm: javax.swing.JFrame
New: Java: Label: totalPrice: javax.swing.JL

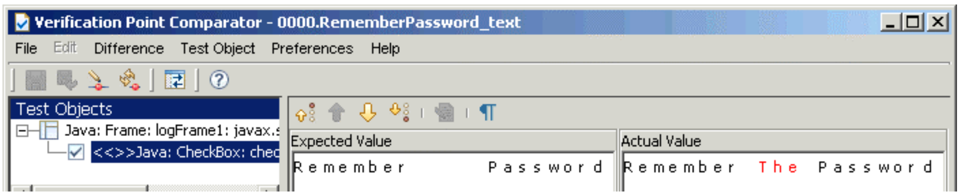
Property	Value
.class	javax.swing.JLabel
.classIndex	9
.testObjectState	ENABLED|VISIBLE|READY
accessibleContext.access...	$150.90
name	totalPrice

```
$150.90
```

Steps	Comments
7. Close the Verification Point Editor.	
8. Close the test log.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3.4 Use the Verification Point Comparator

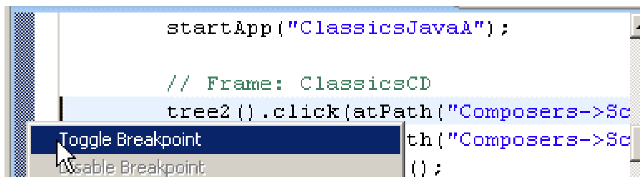
Steps	Comments
1. If the VP1_OrderNewBachViolin_01 script is not already open, open it in the Functional Test perspective.	
2. Edit the start application line to use the JavaB application. `startApp("ClassicsJavaB");`	JavaB is the second build of the ClassicsCD application.
3. Run the VP1_OrderNewBachViolin_01 script.	This is the script containing the verification point.
4. Click **OK**. In the Select Log dialog box, add **JavaB** to the end of the log name and then click **Finish**.	
5. Observe the playback actions and the messages displayed in the Playback Monitor.	The playback will take longer on the Member Logon dialog due to a difference in JavaB.
6. When playback finishes, the log opens. Locate your verification points.	
7. Click the **View Results** hyperlink for the verification point that failed.	This launches the Verification Point Comparator.
8. Observe the difference between the expected value and the actual value.	This is a difference between Build 1 and Build 2 of the application under test.
 Verification Point Comparator - 0000.RememberPassword_text File Edit Difference Test Object Preferences Help Test Objects Java: Frame: logFrame1: javax.s <<>>Java: CheckBox: chec Expected Value: Remember Password Actual Value: Remember The Password	
9. Close the Verification Point Comparator window.	
10. Close the test log.	
11. Close the VP1_OrderNewBachViolin_01 script.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3.5 Insert a Breakpoint into a Script

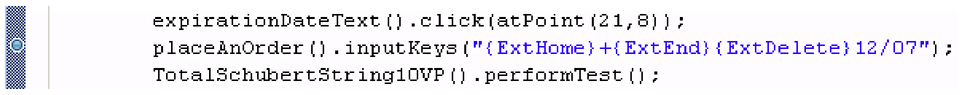
This exercise introduces some of the Functional Tester debugging features.

Steps	Comments
1. In the Projects view, double-click the **TIMER_OrderNewSchubertString_02** script to open it.	This is the script that contains the timer.
2. In the Java Editor, locate the line that follows the `startApp` command. Position your cursor on the marker bar to the left of this line. Right-click and select **Toggle Breakpoint** in the shortcut menu.	The line contains a comment. Functional Tester inserts the breakpoint at the next command.

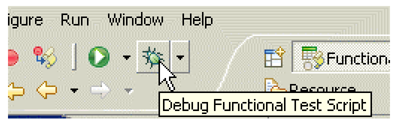


```
        startApp("ClassicsJavaA");

        // Frame: ClassicsCD
        tree2().click(atPath("Composers->Sc
                        th("Composers->Sc
                        ();
```

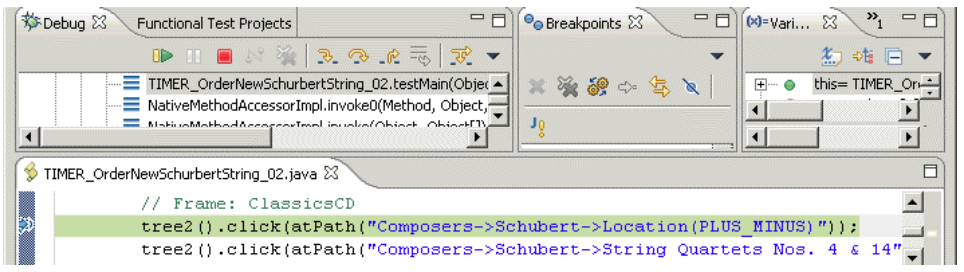
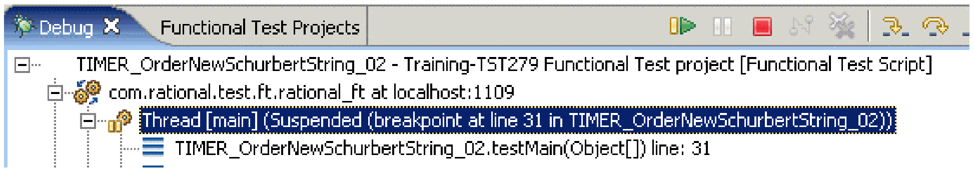
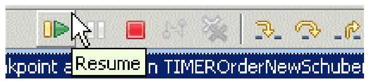
Steps	Comments
3. In the script display, locate the line that enters the credit card expiration date. Insert another breakpoint at this line.	



```
        expirationDateText().click(atPoint(21,8));
        placeAnOrder().inputKeys("{ExtHome}+{ExtEnd}{ExtDelete}12/07");
        TotalSchubertString1OVP().performTest();
```

Steps	Comments
4. Click the **Debug Functional Test Script** button.	 igure Run Window Help Function Resource Debug Functional Test Script
5. In the Select Log dialog box, click **Finish**.	Playback action begins.
6. When the Confirm Perspective Switch dialog box opens, click **Yes** to play back the selected script in Debug mode.	You can wait until the playback action stops immediately after the start of the application.
7. Execution stops at the first breakpoint encountered. In the Functional Test Debug perspective, click tabs and expand panes to see what information is displayed in the Debug, Breakpoints, Script, and Console views. If the Outline view is not open, open it.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
 Debug Functional Test Projects TIMER_OrderNewSchurbertString_02.testMain(Objec NativeMethodAccessorImpl.invoke0(Method, Object, Breakpoints Vari... this= TIMER_Or TIMER_OrderNewSchurbertString_02.java `// Frame: ClassicsCD` `tree2().click(atPath("Composers->Schubert->Location(PLUS_MINUS)"));` `tree2().click(atPath("Composers->Schubert->String Quartets Nos. 4 & 14"`	
8. In the Java editor, note that a checkmark overlays the breakpoint icon, indicating that it has been verified by the JRE.	The line number in your display may be different. The line at which the script is suspended has green highlighting.
9. Double-click the **Debug** tab to expand it. Note the “Suspended” message.	Execution is suspended at the breakpoint.
 Debug Functional Test Projects TIMER_OrderNewSchurbertString_02 - Training-TST279 Functional Test project [Functional Test Script] com.rational.test.ft.rational_ft at localhost:1109 Thread [main] (Suspended (breakpoint at line 31 in TIMER_OrderNewSchurbertString_02)) TIMER_OrderNewSchurbertString_02.testMain(Object[]) line: 31	
10. Minimize the Functional Test window. Observe that the application has been started.	Breakpoints may also be useful to get the application under test into a desired state.
11. Restore the Functional Test Debug window.	
12. Click the Debug view **Resume** button to continue executing the script.	 kpoint a Resume n TIMEROrderNewSchube
13. The script continues until the next breakpoint. Look at the ClassicsCD window to see where it stopped on the application.	The cursor should be in the **Expiration Date** field.
14. Return to the Functional Test Debug perspective and click the Debug view **Resume** button to finish executing the script.	
15. Close the test log.	Note that Functional Tester switches back to the Test perspective.
16. In the Functional Test perspective, in the script, remove the breakpoints by right-clicking the breakpoint and then clicking **Toggle Breakpoint** in the shortcut menu.	

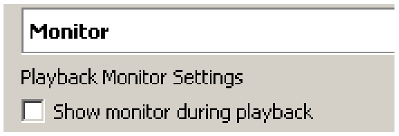
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

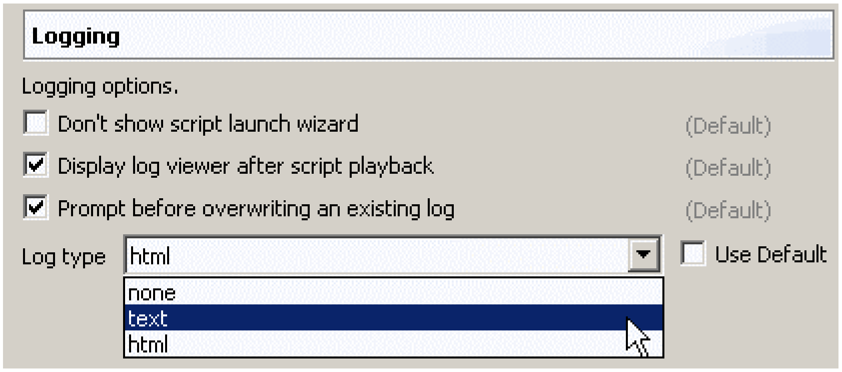
Steps	Comments
17. If necessary, close the AUT. Close the TIMER_OrderNewSchubertString_02 script	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

3.6 Set Functional Tester Preferences and Options

Steps	Comments
1. In the Functional Test perspective, select the **Simple_OrderNewSchubertString_01** script and run it, accepting the default log information.	This is so you can see how the script runs before you change any options.
2. Close the test log.	
3. On the Functional Test menu bar, click **Window** > **Preferences**.	
4. In the left pane, expand **Functional Test** by clicking the + sign.	This will display the categories of preferences that you can set.
5. In the left pane, click **Playback**.	The general playback settings are displayed in the right pane.
6. Expand **Playback** by clicking the + sign and then click **Delays**.	The playback delay options are displayed in the right pane.
7. In the left pane, click **Monitor**.	The playback monitor setting is displayed in the right pane.
8. Clear the **Show monitor during playback** checkbox.	 Monitor Playback Monitor Settings ☐ Show monitor during playback
9. In the left pane, click **Logging**.	The logging options are displayed in the right pane.
10. Clear the **Use Default** checkbox, click the dropdown menu, and select **text**.	



Logging

Logging options.

☐ Don't show script launch wizard (Default)

☑ Display log viewer after script playback (Default)

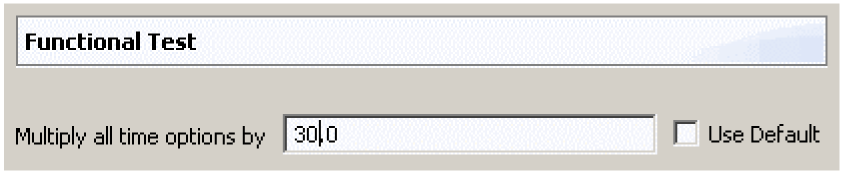
☑ Prompt before overwriting an existing log (Default)

Log type: html ☐ Use Default

11. Click **OK**.	This closes the Preferences window.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
12. Run the SIMPLE_OrderNewSchubertString_01 script and look at the resulting log file.	Note the difference between the text log and the HTML logs you have been viewing. The text log opens within the workspace in a tab labeled rational_ft_log.txt
13. Close text log window.	
14. On the Functional Test menu bar, select **Window > Preferences** again.	
15. Reset the log type to **HTML** and reset the playback monitor to **Show during playback**.	Click **Apply** after each change.
16. In the left pane, click **Functional Test**.	"Multiply all time options by" is displayed in the right pane.
17. Clear the **Use Default** box, change the value in the box to 30.0, and then click **OK**.	This causes all operations controlled by a time option to take 30 times as long to run.


Functional Test

Multiply all time options by 30.0 ☐ Use Default

Steps	Comments
18. Run the Simple_OrderNewSchubertString_01 script.	Observe how slowly the script runs.
19. Close the test log.	
20. On the Functional Test menu bar, select **Window > Preferences**. Restore the time options multiplier to the default value by checking the **Use Default** box. Click **OK**.	
21. Run the Simple_OrderNewSchubertString_01 script again.	Observe how much faster the script runs.
22. Close the test log.	
23. Close the Simple_OrderNewSchubertString_01 script.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 4
Extending Scripts

These lab exercises introduce some more functionality associated with Java code as the scripting language. This will enable better testing of the application. Also, these lab exercises will reinforce some of the recording and playback concepts you have learned.

Objectives

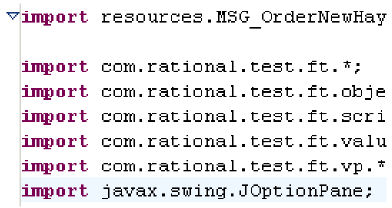
In this lab, you will perform the following tasks:

- Record a script, add code for a message box, and play back the script.
- Record a script, add code to override some preference settings, and play back the script.
- Run a sample script that has an unexpected active window.
- Edit the script to handle an unexpected active window and play back the script.
- Create a helper class and move the code to handle an unexpected active window into the helper class.

© Copyright IBM Corp. 2005

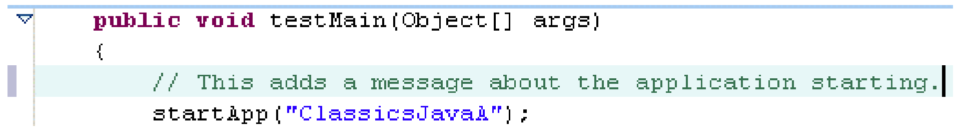
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

4.1 Create a Message Box

Steps	Comments
1. Record a script called **MSG_OrderNewHaydnViolin_01**.	
2. Start the ClassicsJavaA application.	
3. Expand the **Haydn** folder and click **Violin Concertos**.	
4. Click the **Place Order** button.	
5. In the Member Logon dialog box, click **OK**.	
6. In the ClassicsCD application, perform the following user actions: a. Click in the **Card Number** field. b. Type 1234 1234 1234 1234 in the C**ard Number** field. c. Click in the **Expiration Date** field. d. Type 12/07 in the **Expiration Date** field.	**Quantity** field should already have a value of 1.
7. In the Recording Monitor, click the **Insert Verification Point or Action Command** button.	
8. Create a Data verification point for the total order amount and name the verification point OrderTotalHaydnViolin.	
9. Place the order, acknowledge the message, close the application, and stop recording.	
10. Play back the script.	
11. Use the defaults for the log.	
12. Review the test log.	
13. Close the test log.	
14. In the Test Perspective, make sure the MSG_OrderNewHaydnViolin_01 script is displayed in the Java editor.	
15. Expand the import resources line at the beginning of the script. Remembering that Java is case-sensitive, insert the following new line after the last import line: `import javax.swing.JOptionPane;`	 `import resources.MSG_OrderNewHay` `import com.rational.test.ft.*;` `import com.rational.test.ft.obje` `import com.rational.test.ft.scri` `import com.rational.test.ft.valu` `import com.rational.test.ft.vp.*` `import javax.swing.JOptionPane;`

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
16. Insert the following comment line just above the startApp line: `// This adds a message about the application starting.`	



```
public void testMain(Object[] args)
{
    // This adds a message about the application starting.
    startApp("ClassicsJavaA");
```

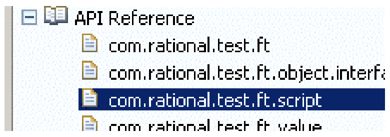
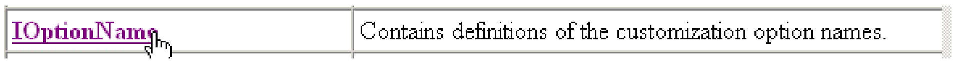
Steps	Comments
17. Insert the following line just after the comment line: `JOptionPane.showMessageDialog(null,"The application will start next.","Information",JOptionPane.INFORMATION_MESSAGE);`	Make sure to follow the code exactly.
18. Insert a blank line just before the verification point line. `// This checks the total order amount.`	

19. Insert the following comment lines just after the verification point line:
`// This adds a message indicating that the order is about to be`
`// placed.`	

20. Insert the following line just after the comment line: `JOptionPane.showMessageDialog(null,"The order is placed next.","Order Message",JOptionPane.INFORMATION_MESSAGE);`	
21. Play back the script.	
22. Accept the defaults for the log.	
23. Click **OK** on the Information message box when it appears. (You may need to bring the message box to the front by clicking the **Recording Monitor** or pressing Alt+Tab.)	This is the message that you created to display before starting the application.
24. Click **OK** on the Order Message message box when it appears. (You may need to bring the message box to the front by clicking the **Recording Monitor** or pressing Alt+Tab.)	You may have to move it to see the entire message box. This is the message that you created to display before placing the order.
25. Close the test log.	
26. Close the MSG_OrderNewHaydnViolin_01 script.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

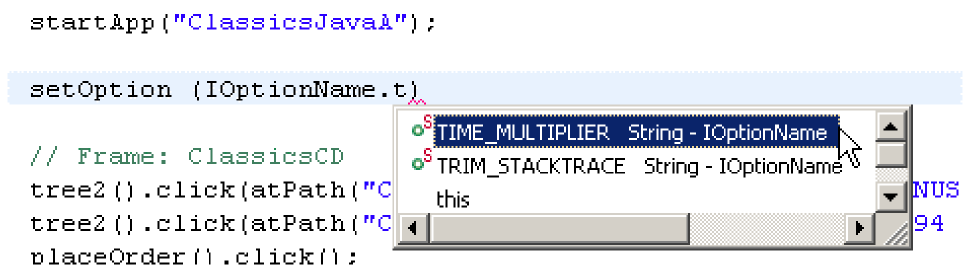
4.2 Override Preference Settings

Steps	Comments
1. Click **Help** > **Functional Test API Reference**.	
2. Expand the API Reference.	
3. Click **com.rational.test.ft.script**.	 API Reference com.rational.test.ft com.rational.test.ft.object.interf com.rational.test.ft.script com.rational.test.ft.value
4. Click **IOptionName** in the right pane.  IOptionName — Contains definitions of the customization option names.	
5. Scroll down in the **Field Summary** to see the options that can be customized.	We will use the **`BRING_UP_LOGVIEWER`** and **`TIME_MULTIPLIER`** fields in this lab exercise.
6. Close the Help window.	
7. In the Functional Test Perspective, record a script called PREF_OrderNewHaydnS94_01.	
8. Start the ClassicsJavaA application.	
9. Expand the **Haydn** folder and click **Symphonies Nos. 94 & 98**.	
10. Click the **Place Order** button.	
11. Log in as Susan Flontly.	Log in without a password.
12. Back in the ClassicsCD application, perform the following user actions: a. Click in the **Card Number** field. b. Type 5555 5555 5555 5555 in the **Card Number** field. c. Click in the **Expiration Date** field. d. Type 12/07 in the **Expiration Date** field.	The **Quantity** field should already contain a value of 1.
13. Create a Data verification point for the total order amount.	
14. Place the order, acknowledge the dialog, close the application, and stop recording.	
15. Play back the script.	
16. Accept the defaults for the log.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
17. Review the test log.	
18. Close the test log.	
19. Enter the following text on the line in the script after the `startApp` line: `setOption(IOptionName.`	Insert a new line by inserting your cursor at the beginning or ending of a line and pressing the **Enter** key.
20. The end parenthesis is inserted automatically, and the Content Assist option list pops up. Scroll down the options (or type the first letter or two) and double-click **`TIME_MULTIPLIER`**.	If the list doesn't automatically appear, you can bring it up by placing your cursor at the end of the text (after the period but before the end parenthesis) and then holding down the **Ctrl** key and pressing the space bar.



```
startApp("ClassicsJavaA");

setOption (IOptionName.t)

// Frame: ClassicsCD
tree2().click(atPath("C                                          NUS
tree2().click(atPath("C                                          94
placeOrder().click();
```

TIME_MULTIPLIER   String - IOptionName
TRIM_STACKTRACE   String - IOptionName
this

Steps	Comments
21. Finish the line with **,10.0);**.	Make sure to include a decimal point with the new time value.
22. Add a line of code before the new setOption line instructing Functional Test to display a message about the current value and add a line of code after the new line to display a message about the new value. The finished code should look like this:	

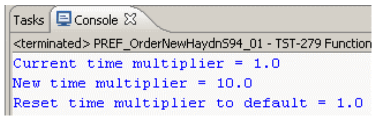
```
startApp("ClassicsJavaA");

System.out.println("Current time multiplier = "+
getOption(IOptionName.TIME_MULTIPLIER));

setOption(IOptionName.TIME_MULTIPLIER,10.0);

System.out.println("New time multiplier = "+
getOption(IOptionName.TIME_MULTIPLIER));
```

Steps	Comments
23. Go to the line just before the input keys for the expiration date and reset the time multiplier to the default value. Print the default value in the console window.	This is useful when you want the time value in question to be in effect for only part of the test.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
24. This part of the script should look like the following:	

resetOption(IOptionName.TIME_MULTIPLIER);
System.out.println("Reset time multiplier to default= "+ getOption(IOptionName.TIME_MULTIPLIER));
placeAnOrder().inputKeys("{ExtHome}+{ExtEnd}{ExtDelete}12/07");	

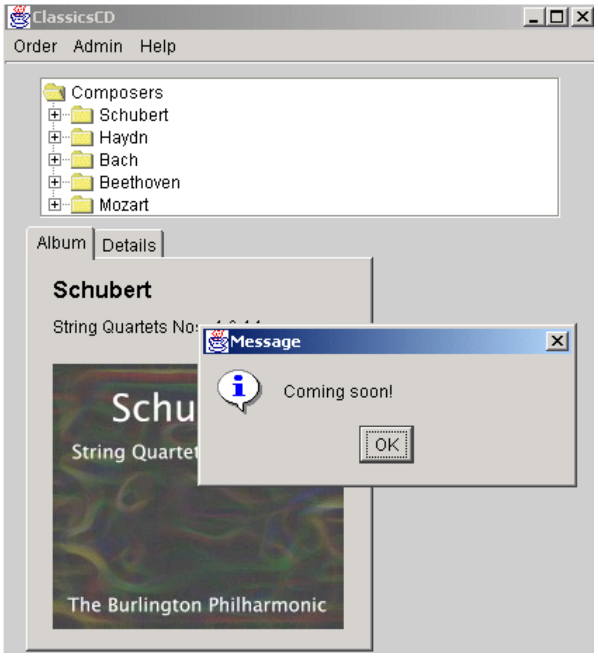
25. Play back the script and accept the defaults for the log.	
26. Review the test log and then close the log.	
27. In the Console view, look at the messages your script displayed.	 <terminated> PREF_OrderNewHaydnS94_01 - TST-279 Function Current time multiplier = 1.0 New time multiplier = 10.0 Reset time multiplier to default = 1.0
28. Close the PREF_OrderNewHaydnS94_01 script.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

4.3 Handle an Unexpected Active Window

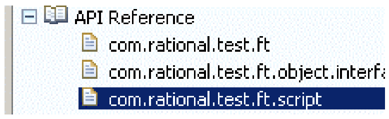
Suppose that you just started a test and an unexpected dialog appears. This lab will show you how to handle this condition.

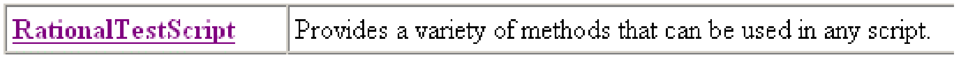


Coming soon!

Steps	Comments
1. Run the script called **UAW_OrderNewMozartS34_01**.	This script allows an unexpected active window to stay on the screen.
2. Accept the defaults for the log.	It may take up to a minute to finish and display the log.
3. Look at the Properties of the log message.	The commands to expand the tree could not execute because the message window was active. You will edit the script to handle this condition.
4. Close the test log.	
5. Click **OK** to close the Message box and then close the ClassicsCD application.	You may have to use the Windows Task Manager to end the ClassicsCD application.
6. In the Functional Test perspective, make sure the UAW_OrderNewMozartS34_01script is open.	

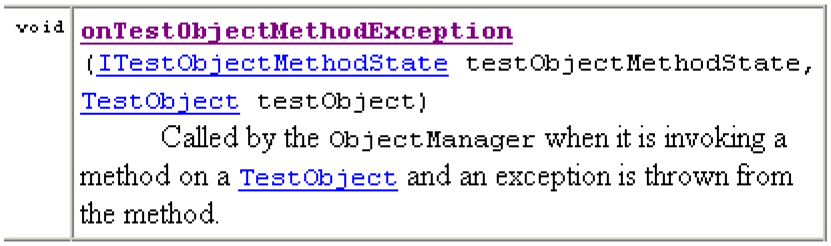
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
7. Insert two blank lines above the last brace **}** in the script.	This is where you will paste a method that handles test object exceptions.
8. Click **Help** > **Functional Test API Reference**.	
9. Expand the API Reference.	
10. Click **com.rational.test.ft.script**.	 API Reference com.rational.test.ft com.rational.test.ft.object.interfa com.rational.test.ft.script
11. Scroll down under the **Class Summary** and click **RationalTestScript**.	



RationalTestScript	Provides a variety of methods that can be used in any script.

Steps	Comments
12. Scroll down in the Method Summary and click **`onTestObjectMethodException`**.	



void	onTestObjectMethodException(ITestObjectMethodState testObjectMethodState, TestObject testObject) Called by the ObjectManager when it is invoking a method on a TestObject and an exception is thrown from the method.

Steps	Comments
13. Under **`Specified by`**, click the `onTestObjectMethodException` link.	
14. Under **`onTestObjectMethodException`**, select and copy all of the text at the top of the entry that starts with **`public`** and ends with **`foundObject)`**.	
15. Paste the text for this method into the script. The end of your script should look like the following:	

```
        // Frame: ClassicsCD
         classicsJava(ANY,MAY_EXIT).close();
    }
    public void onTestObjectMethodException(ITestObjectMethodState
                testObjectMethodState,TestObject foundObject)
}
```

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
16. On the next line after the public void line, insert a left brace **{**. Then press **Enter** to move to the next line.	
17. Enter the following text:	

```
if ( testObjectMethodState.getThrowableClassName().equals(
        "com.rational.test.ft.WindowActivateFailedException") )
```

Steps	Comments
18. On the next line, insert another left brace **{**. Then press **Enter** to move to the next line.	
19. Edit the script to look like the following text:	Make sure to surround the second Enter with braces: {Enter}

```
IWindow activeWindow = getScreen().getActiveWindow();
if (activeWindow !=null)
{
  System.out.println("Unexpected active window caption =
  "+activeWindow.getText());
  activeWindow.inputKeys("Enter{Enter}");
  testObjectMethodState.findObjectAgain();
```

Steps	Comments
20. Enter the following text on the next set of lines in the script:	

```
    else
    super.onTestObjectMethodException(testObjectMethodState, foundObject);
}
else
super.onTestObjectMethodException(testObjectMethodState, foundObject);
```

Steps	Comments
21. A sample of what the end of your script should look like is on the next page.	Make sure you do not have too many braces. If you cannot make the script run, look at and/or copy script lines from the Solutions project. See your instructor for help on this.
22. Close the Help window and run the script. See that it now handles the unexpected active window.	Accept default log information.
23. Close the test log.	
24. Close the UAW_OrderNewMozartS34_01 script.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

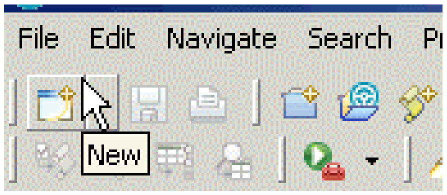
Sample Script (just the end)

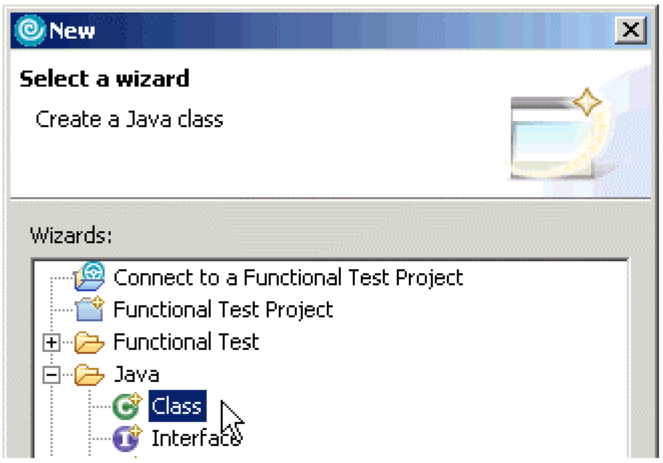
```
        // Frame: ClassicsCD
        classicsJava(ANY,MAY_EXIT).close();
    }
    public void onTestObjectMethodException(ITestObjectMethodState testObjectMethodState,
    TestObject foundObject)
    {
        if (testObjectMethodState.getThrowableClassName().equals("com.rational.test.ft.WindowActivateFailedException"))
        {
            IWindow activeWindow = getScreen().getActiveWindow();
            if (activeWindow !=null)
            {
                System.out.println("Unexpected active window caption = "+activeWindow.getText());
                activeWindow.inputKeys("Enter{Enter} ");
                testObjectMethodState.findObjectAgain();
            }
            else
            super.onTestObjectMethodException(testObjectMethodState, foundObject);
        }
        else
        super.onTestObjectMethodException(testObjectMethodState, foundObject);
    }
}
```

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

4.4 Create a Java helper class and put the unexpected active window code in it

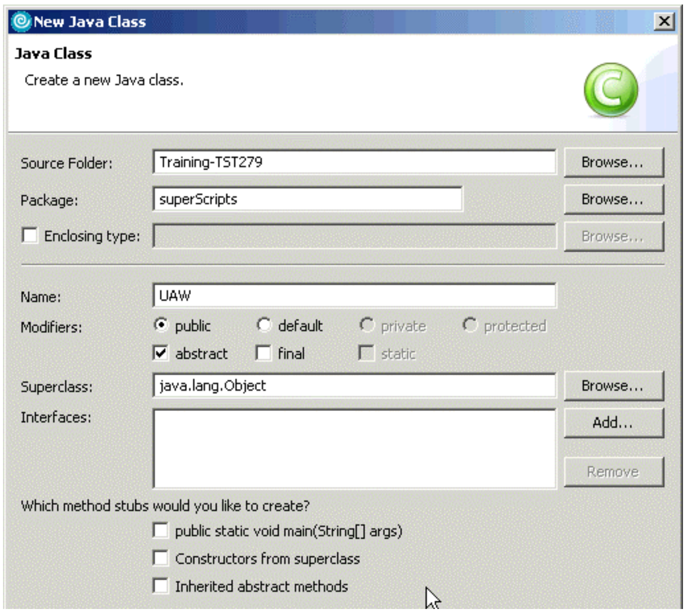
Steps	Comments
1. In the Functional Test Projects view, right-click the **Training-TST279** project.	
2. Click **Add Test Folder**.	
3. In the Test folder name box, type **superScripts** and then click **Finish**.	
4. Click the **New** button.	
5. Expand the Java category. Click **Class** and then click **Next**.	The Java Class dialog box opens.



Steps	Comments
6. In the **Name** box, type **UAW**.	
7. Select the **abstract** option.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
8. Clear the **Inherited abstract methods** check box.	



New Java Class

Java Class

Create a new Java class.

Source Folder: Training-TST279

Package: superScripts

Enclosing type:

Name: UAW

Modifiers: (● public) ○ default ○ private ○ protected

☑ abstract ☐ final ☐ static

Superclass: java.lang.Object

Interfaces:

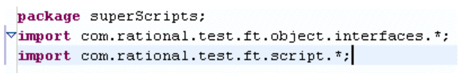
Which method stubs would you like to create?

☐ public static void main(String[] args)

☐ Constructors from superclass

☐ Inherited abstract methods

Steps	Comments
9. Click **Finish**.	The new Java helper class, UAW.java, opens in the Script view.

10. Type the following text lines in the UAW helper class immediately after the package line:	 `package superScripts;`
	`import com.rational.test.ft.object.interfaces.*;`
	`import com.rational.test.ft.script.*;`

```
import com.rational.test.ft.object.interfaces.*;
import com.rational.test.ft.script.*;
```

Steps	Comments
11. Position the cursor immediately before the left brace at the end of the line that reads **`public abstract class UAW {`** .	
12. Type **`extends RationalTestScript`**. (Do not type the period.)	
13. Press **Enter**. The result should look like the following code:	

```
public abstract class UAW extends RationalTestScript
{
```

Steps	Comments
14. Move the cursor to the blank line immediately after the **{** (left brace).	

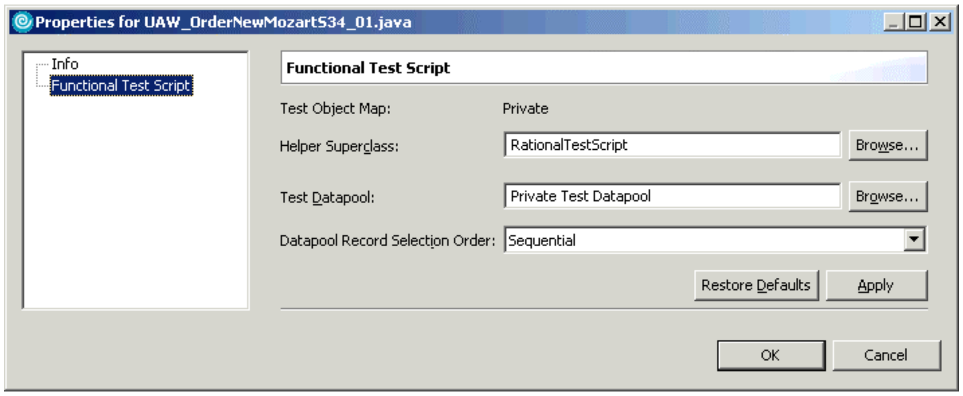
© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
15. Open the **UAW_OrderNewMozartS34_01** script and copy all the code that you wrote to handle unexpected active windows.	Select the code that begins with `public void onTestObjectMethodException` and ends with the second-to-last line of the script, which contains a single right brace **}**.
16. Paste the copied code into the UAW helper class (at the current cursor position).	The UAW helper class text should be similar to the following:

```
package superScripts
import com.rational.test.ft.object.interfaces.*;
import com.rational.test.ft.script.*;

public abstract class UAW extends RationalTestScript
{
  public void
onTestObjectMethodException(ITestObjectMethodState
  testObjectMethodState, TestObject foundObject)
  {
    if (
testObjectMethodState.getThrowableClassName().equals(

  "com.rational.test.ft.WindowActivateFailedException") )
    {
      IWindow activeWindow =
getScreen().getActiveWindow();
      if (activeWindow != null)
      {
        System.out.println("Unexpected active window
caption =
          "+activeWindow.getText());
        activeWindow.inputKeys("Enter{Enter}");

        testObjectMethodState.findObjectAgain();
      }
      else

  super.onTestObjectMethodException(testObjectMethodState,
          foundObject);
    }
    else

  super.onTestObjectMethodException(testObjectMethodState,
        foundObject);
  }
}
```

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
17. In the Projects view, right-click the **UAW_OrderNewMozartS34_01** script and then click **Properties**.	
18. Click **Functional Test Script**.	
19. In the Helper superclass box, delete the existing text and type **superScripts.UAW** and then click **OK**.	
 Properties for UAW_OrderNewMozartS34_01.java — Functional Test Script — Test Object Map: Private — Helper Superclass: RationalTestScript — Test Datapool: Private Test Datapool — Datapool Record Selection Order: Sequential	
20. Close UAW.java in the Script view, saving your changes.	
21. In the UAW_OrderNewMozartS34_01 script, with the text you copied still selected, click **Source** > **Toggle Comment**. (If Comment is not enabled in the Source menu, click the script title tab in the Script view to make it active and then try again.)	This code is no longer necessary in the script because it exists in one of the script's helper classes, UAW.java.
22. Run the UAW_OrderNewMozartS34_01 script again. (Accept default log information.)	The script should play back the same way that it did before you moved a portion of its code to the new helper class.
23. Close the test log.	
24. Close the UAW_OrderNewMozartS34_01 script.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 5
Using Test Object Maps

The Functional Tester test object map is a collection of test object descriptions for the application under test. A test object map can be private (associated with only one script) or shared by more than one script.

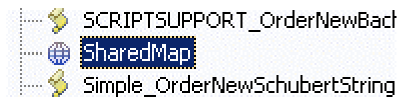
Objectives

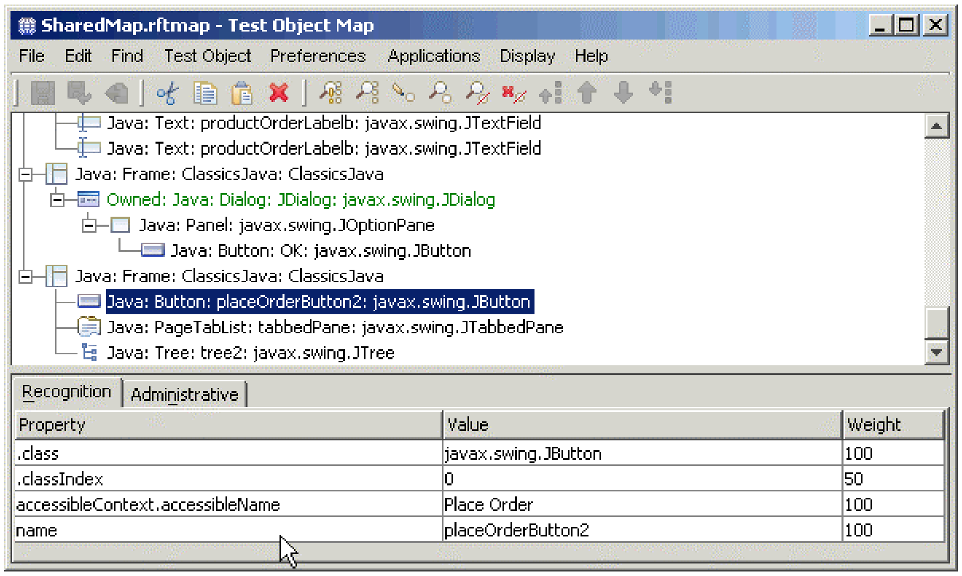
In this lab, you will perform the following tasks:

- Display a test object map.
- Create and use a shared test object map.
- Modify a test object map.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

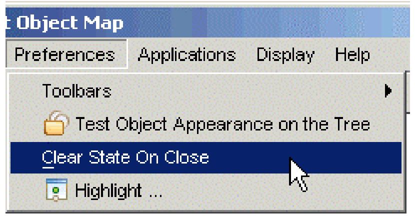
5.1 Display a Test Object Map

Steps	Comments
1. In the Functional Test Projects View, double-click the **SharedMap** test object map.	This opens the test object map.  SCRIPTSUPPORT_OrderNewBact SharedMap Simple_OrderNewSchubertString
2. In the Test Object Hierarchy pane, expand all of the test objects by clicking the **+** signs.	These are the test objects you accessed during recording.


SharedMap.rftmap - Test Object Map
File Edit Find Test Object Preferences Applications Display Help
Recognition
Administrative

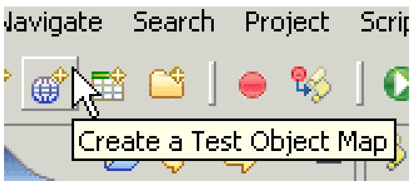
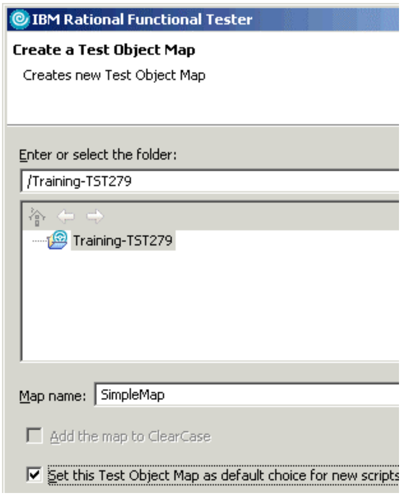
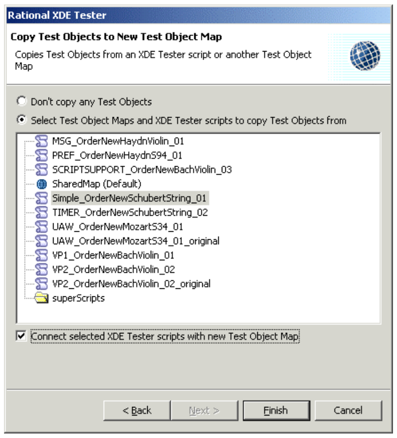
```
    Java: Text: productOrderLabelb: javax.swing.JTextField
    Java: Text: productOrderLabelb: javax.swing.JTextField
Java: Frame: ClassicsJava: ClassicsJava
  Owned: Java: Dialog: JDialog: javax.swing.JDialog
    Java: Panel: javax.swing.JOptionPane
      Java: Button: OK: javax.swing.JButton
Java: Frame: ClassicsJava: ClassicsJava
  Java: Button: placeOrderButton2: javax.swing.JButton
  Java: PageTabList: tabbedPane: javax.swing.JTabbedPane
  Java: Tree: tree2: javax.swing.JTree
```

Property	Value	Weight
.class	javax.swing.JButton	100
.classIndex	0	50
accessibleContext.accessibleName	Place Order	100
name	placeOrderButton2	100

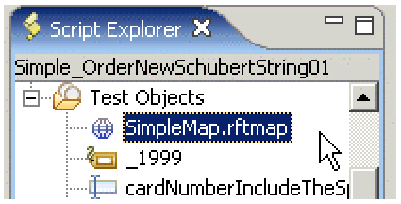
3. Click **Preferences.** If **Clear State On Close** is checked, click to clear it.	 Object Map Preferences Applications Display Help Toolbars Test Object Appearance on the Tree Clear State On Close Highlight ...
4. Close the Test Object Map window.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

5.2 Create and Use a Shared Test Object Map

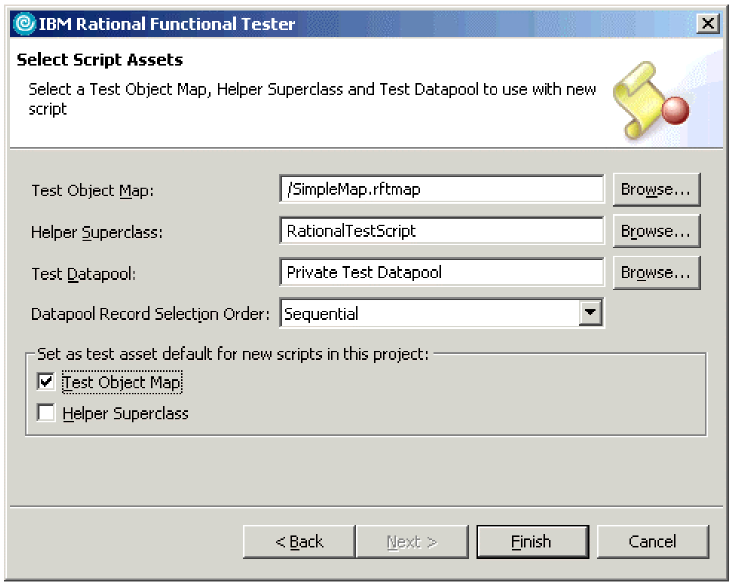
Steps	Comments
1. Create a new test object map by doing one of the following: On the menu bar, click **File** > **New** > **Test Object Map**. Click the **Create a Test Object Map** button on the toolbar.	 Navigate Search Project Scrip Create a Test Object Map
2. In the Create new Test Object Map dialog box: a. Select folder: /**Training-TST279** b. Type **SimpleMap** for the map name. c. If it is displayed, make sure the **Add the map to ClearCase** option is not selected d. Select the **Set this Test Object Map as default choice for new scripts** option. e. Click **Next**.	IBM Rational Functional Tester **Create a Test Object Map** Creates new Test Object Map Enter or select the folder: /Training-TST279 Training-TST279 Map name: SimpleMap ☐ Add the map to ClearCase ☑ Set this Test Object Map as default choice for new scripts
3. In the Copy Test Objects to New Test Object Map dialog box: a. Click the **Select Test Object Maps and scripts to copy Test Objects from** option. b. Click the **Simple_OrderNewSchubertString_01** script. c. Select the **Connect selected scripts with new Test Object Map** check box. d. Click **Finish**.	Rational XDE Tester **Copy Test Objects to New Test Object Map** Copies Test Objects from an XDE Tester script or another Test Object Map ○ Don't copy any Test Objects ● Select Test Object Maps and XDE Tester scripts to copy Test Objects from MSG_OrderNewHaydnViolin_01 PREF_OrderNewHaydnS94_01 SCRIPTSUPPORT_OrderNewBachViolin_03 SharedMap (Default) Simple_OrderNewSchubertString_01 TIMER_OrderNewSchubertString_02 UAW_OrderNewMozartS34_01 UAW_OrderNewMozartS34_01_original VP1_OrderNewBachViolin_01 VP2_OrderNewBachViolin_02 VP2_OrderNewBachViolin_02_original superScripts ☑ Connect selected XDE Tester scripts with new Test Object Map You have created a new shared map that contains the objects from the private map associated with the Simple_ OrderNewSchubertString_01 script.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

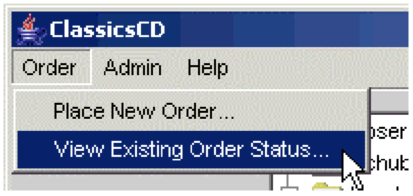
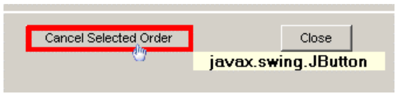
Steps	Comments
4. On the Object Map: Help Page, clear the checkbox at the bottom of the page and click **Finish**.	
5. Close the Test Object Map window.	
6. In the Projects View, find your new test object map.	
7. Open the Simple_OrderNewSchubertString_01 script. Notice (in the Script Explorer) that it is now associated with the SimpleMap test object map instead of a private test object map.	 Script Explorer; Simple_OrderNewSchubertString01; Test Objects; SimpleMap.rftmap; _1999; cardNumberIncludeTheS
7. Close the Simple_OrderNewSchubertString_01 script.	
8. Record a new script: a. Name the script **Simple_OrderNewSchubertS5_01**. b. Do not check the **Add the script to Source Control** box if it is displayed. c. Click **Next**.	Clicking **Next** opens the Select Test Object Map dialog box, in which you can specify the object map with which you want to associate the script.
9. In the Select Test Object Map dialog box: a. If not already selected, browse to select /**SimpleMap.rftmap** Test Object Map. b. In the **Set as test asset default for new scripts in this project** area, check the **Test Object Map** option. c. Click **Finish**.	Your new test object map will be associated with this script. We also ensured that SimpleMap remains the default test object map for new scripts.

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
 IBM Rational Functional Tester **Select Script Assets** Select a Test Object Map, Helper Superclass and Test Datapool to use with new script Test Object Map: /SimpleMap.rftmap Helper Superclass: RationalTestScript Test Datapool: Private Test Datapool Datapool Record Selection Order: Sequential Set as test asset default for new scripts in this project: ☑ Test Object Map ☐ Helper Superclass	
10. In the Recording Monitor, start the ClassicsJavaA application.	
11. Perform the following user actions: a. Expand the **Schubert** list. b. Click **Symphonies Nos. 5 & 9**. c. Click **Place Order**. d. Accept the default login user and click **OK**. e. Type 1234 1234 1234 1234 in the **Card Number** field. f. Type 12/07 in the **Expiration Date** field.	
12. Create a Data verification point to check the order total.	
13. Place the order, close the application, and stop recording.	

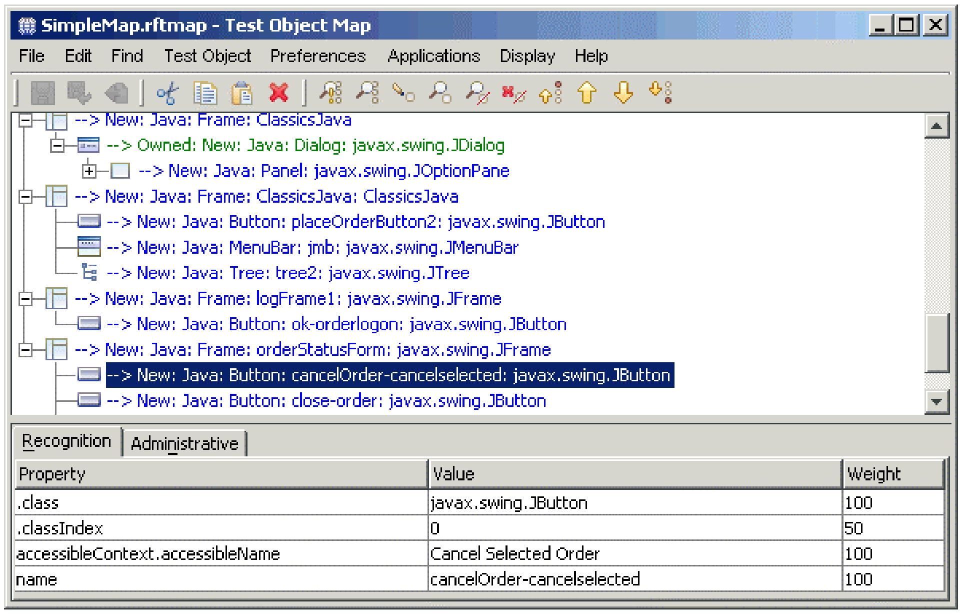
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
14. Note that the SimpleMap test object map is listed in both the Projects view and the Script Explorer.	SimpleMap appears in the Projects view because it is a shared map. (It is a shared map because it was created independently, not as part of the process of creating a script.) SimpleMap also appears in the Script Explorer because it is associated with the active script in the Script view.
15. Close the Simple_OrderNewSchubertS5_01 script.	
16. Create another simple script, named **Simple_TCViewOrder_01**, using the SimpleMap test object map.	We will use this script to demonstrate sharing of the map.
17. In the Recording Monitor, start the ClassicsJavaA application.	
18. Perform the following user actions: a. On the menu, click **Order**. b. Click **View Existing Order Status**. c. Click **OK** in the login dialog box.	 ClassicsCD Order Admin Help Place New Order... View Existing Order Status...
19. Create two Data verification points to check the text on the **Cancel Selected Order** and **Close** buttons.	
20. Click **Close** in the View Existing Orders dialog box, close the application, and stop recording.	You may need to close the Object Map: Help Page and the Test Object Map.
21. Close the Test Object Map window.	
22. Notice (in the Script Explorer) that the Simple_TCViewOrder_01 script is associated with the SimpleMap test object map.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
23. Open the SimpleMap test object map and expand all of the objects. (Do not close the map after reviewing it.)	Notice the new objects that are listed now. The Simple_TCViewOrder_01 script added test objects to the map.


SimpleMap.rftmap - Test Object Map

File Edit Find Test Object Preferences Applications Display Help

```
--> New: Java: Frame: ClassicsJava
    --> Owned: New: Java: Dialog: javax.swing.JDialog
        --> New: Java: Panel: javax.swing.JOptionPane
--> New: Java: Frame: ClassicsJava: ClassicsJava
    --> New: Java: Button: placeOrderButton2: javax.swing.JButton
    --> New: Java: MenuBar: jmb: javax.swing.JMenuBar
    --> New: Java: Tree: tree2: javax.swing.JTree
--> New: Java: Frame: logFrame1: javax.swing.JFrame
    --> New: Java: Button: ok-orderlogon: javax.swing.JButton
--> New: Java: Frame: orderStatusForm: javax.swing.JFrame
    --> New: Java: Button: cancelOrder-cancelselected: javax.swing.JButton
    --> New: Java: Button: close-order: javax.swing.JButton
```

Recognition | Administrative

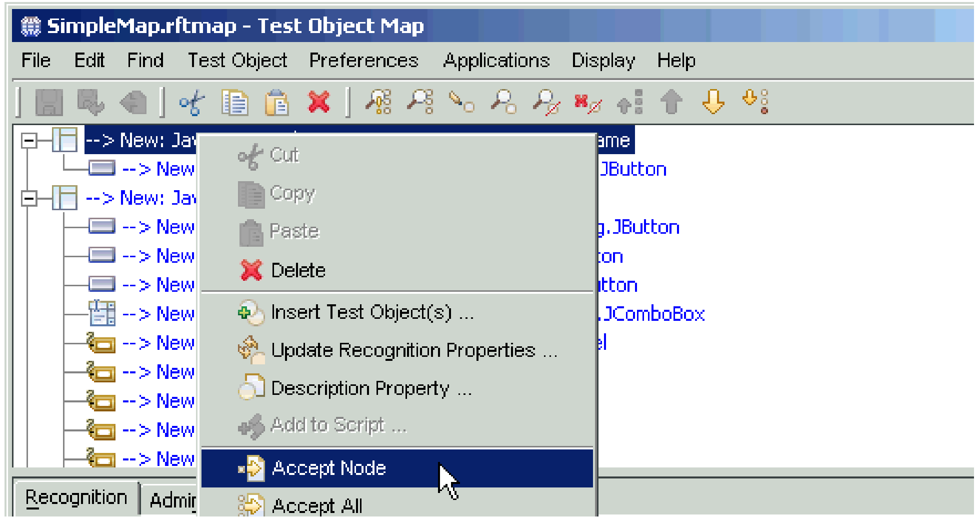
Property	Value	Weight
.class	javax.swing.JButton	100
.classIndex	0	50
accessibleContext.accessibleName	Cancel Selected Order	100
name	cancelOrder-cancelselected	100

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

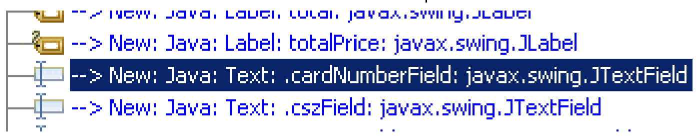
5.3 Modify a Test Object Map

In this exercise, you will modify your shared test object map in various ways and demonstrate that both scripts that use this map can "see" the changes.

Steps	Comments
1. In the Test Object Map window (SimpleMap), note that all test objects are marked **New** and listed in blue type. Right-click the first top-level object in the hierarchy and click **Accept Node** in the shortcut menu.	The test object is now listed in black type and is no longer marked as **New**.

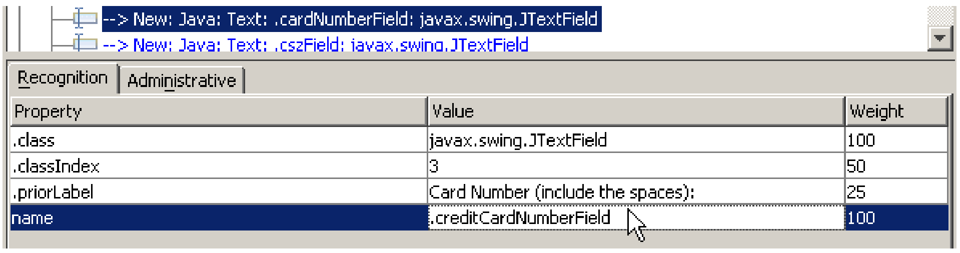


Steps	Comments
2. Expand all objects by clicking the + signs.	
3. Under the **Frame: orderForm: javax.swing.Jframe** top-level object, find and click the **.cardNumberField** text object to select it.	


--> New: Java: Label: total: javax.swing.JLabel
--> New: Java: Label: totalPrice: javax.swing.JLabel
--> New: Java: Text: .cardNumberField: javax.swing.JTextField
--> New: Java: Text: .cszField: javax.swing.JTextField

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

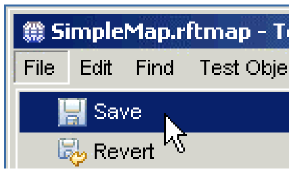
Steps	Comments
4. In the **Recognition** tab, find the **name** property for this test object. Double-click the associated value field and change the value to **.creditCardNumberField**.	

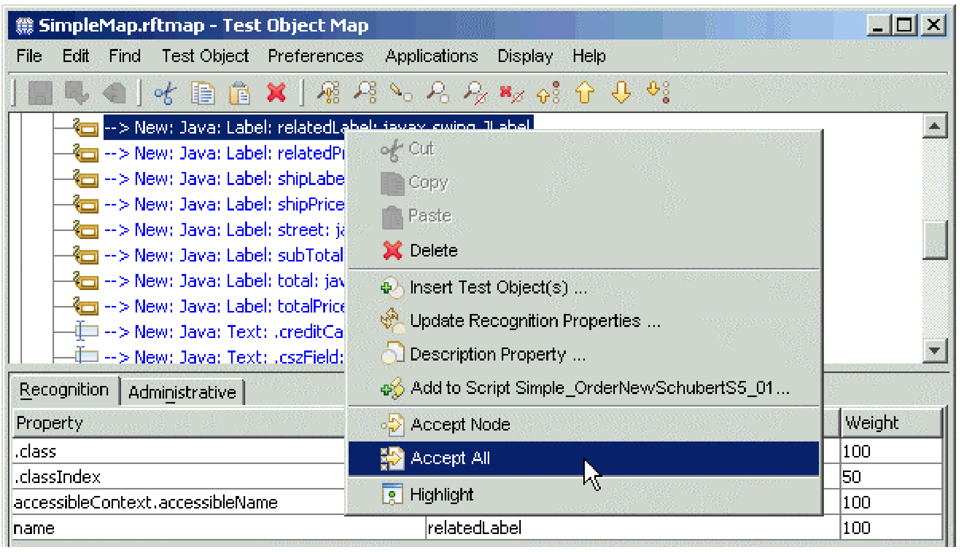


--> New: Java: Text: .cardNumberField: javax.swing.JTextField
--> New: Java: Text: .cszField: javax.swing.JTextField

Recognition | Administrative

Property	Value	Weight
.class	javax.swing.JTextField	100
.classIndex	3	50
.priorLabel	Card Number (include the spaces):	25
name	.creditCardNumberField	100

Steps	Comments
5. Select **File > Save** to save the changes you have made.	 SimpleMap.rftmap - T File Edit Find Test Obje Save Revert
6. Right-click any object and then click **Accept All** in the shortcut menu.	Note that all of the test objects are now listed in black type, and they are no longer listed as **New**.


SimpleMap.rftmap - Test Object Map
File
Edit
Find
Test Object
Preferences
Applications
Display
Help
Cut
Copy
Paste
Delete
Insert Test Object(s) ...
Update Recognition Properties ...
Description Property ...
Add to Script Simple_OrderNewSchubertS5_01...
Accept Node
Accept All
Highlight

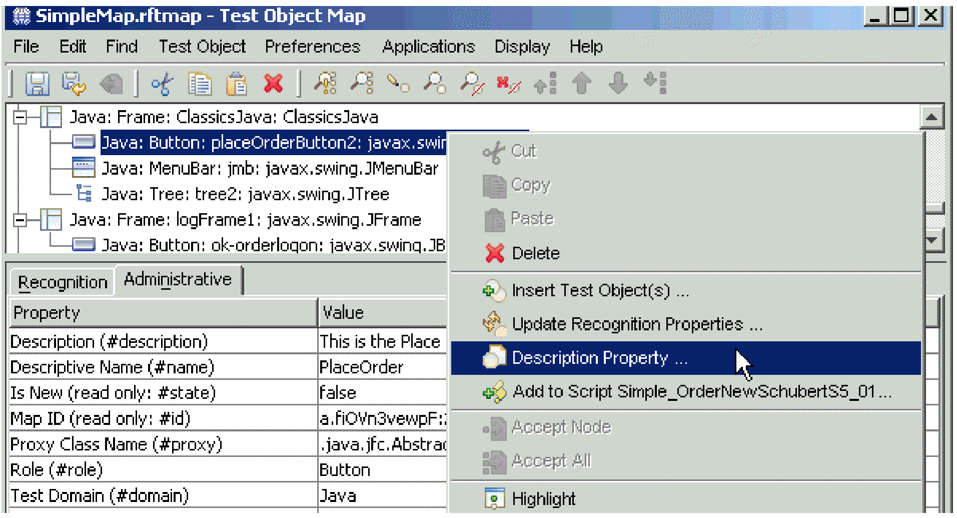
--> New: Java: Label: relatedLabel: javax.swing.JLabel
--> New: Java: Label: relatedPr
--> New: Java: Label: shipLabe
--> New: Java: Label: shipPrice
--> New: Java: Label: street: ja
--> New: Java: Label: subTotal
--> New: Java: Label: total: jav
--> New: Java: Label: totalPrice
--> New: Java: Text: .creditCa
--> New: Java: Text: .cszField:

Recognition | Administrative

Property	Value	Weight
.class		100
.classIndex		50
accessibleContext.accessibleName		100
name	relatedLabel	100

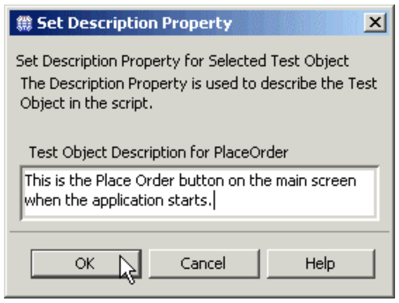
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

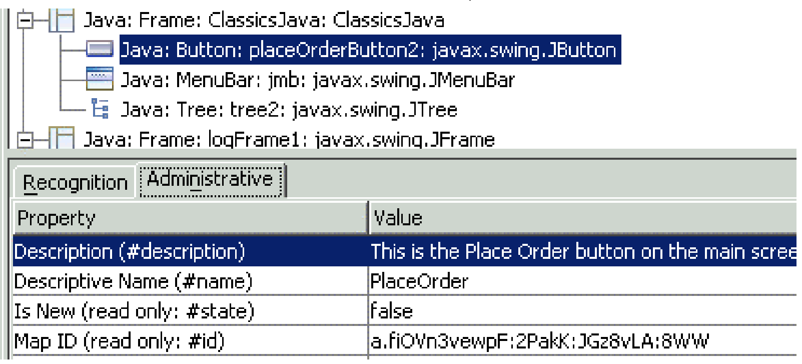
Steps	Comments
7. Scroll to the **ClassicsJava: ClassicsJava** frame. Right-click the **placeOrderButton2** button under it and then click **Description Property** in the shortcut menu.	You will enter some descriptive information about the object.


SimpleMap.rftmap - Test Object Map
File Edit Find Test Object Preferences Applications Display Help
Cut
Copy
Paste
Delete
Insert Test Object(s) ...
Update Recognition Properties ...
Description Property ...
Add to Script Simple_OrderNewSchubertS5_01 ...
Accept Node
Accept All
Highlight
Recognition
Administrative

Java: Frame: ClassicsJava: ClassicsJava
Java: Button: placeOrderButton2: javax.swin
Java: MenuBar: jmb: javax.swing.JMenuBar
Java: Tree: tree2: javax.swing.JTree
Java: Frame: logFrame1: javax.swing.JFrame
Java: Button: ok-orderlogon: javax.swing.JB

Property	Value
Description (#description)	This is the Place
Descriptive Name (#name)	PlaceOrder
Is New (read only: #state)	false
Map ID (read only: #id)	a.fiOVn3vewpF:
Proxy Class Name (#proxy)	.java.jfc.Abstrac
Role (#role)	Button
Test Domain (#domain)	Java

Steps	Comments
8. In the Set Description Property dialog, enter some text, such as "This is the Place Order button on the main screen when the application starts." and then click **OK**.	 Set Description Property Set Description Property for Selected Test Object The Description Property is used to describe the Test Object in the script. Test Object Description for PlaceOrder This is the Place Order button on the main screen when the application starts. OK Cancel Help
9. Click the **Administrative** tab and verify that the description was attached to the object.	

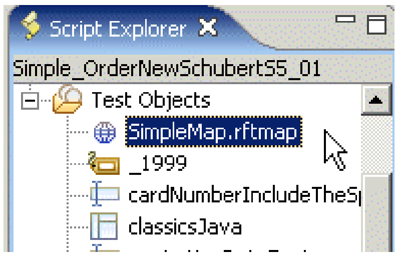


Java: Frame: ClassicsJava: ClassicsJava
Java: Button: placeOrderButton2: javax.swing.JButton
Java: MenuBar: jmb: javax.swing.JMenuBar
Java: Tree: tree2: javax.swing.JTree
Java: Frame: logFrame1: javax.swing.JFrame

Property	Value
Description (#description)	This is the Place Order button on the main scree
Descriptive Name (#name)	PlaceOrder
Is New (read only: #state)	false
Map ID (read only: #id)	a.fiOVn3vewpF:2PakK:JGz8vLA:8WW

Steps	Comments
10. Save the changes you have made and close the test object map.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
11. Close the Simple_TCViewOrder_01 script.	
12. Open the Simple_OrderNewSchubertS5_01 script.	
13. In the Script Explorer, double-click /**SimpleMap.rftmap** to open the SimpleMap test object map.	 Script Explorer Simple_OrderNewSchubertS5_01 Test Objects SimpleMap.rftmap _1999 cardNumberIncludeTheS classicsJava
14. Expand all objects and find the objects that you modified. Verify that this script can "see" the modifications. For example, are all the objects now accepted? There should be no "new" designations. Does the description for the placeOrderButton2 that you entered appear in the **Administrative** tab?	All of the test objects reflect your modifications. The shared test object map really is shared!
15. Close the Test Object Map window.	
16. Close the Simple_OrderNewSchubertS5_01 script.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 6
Managing Object Recognition

Managing object recognition during playback enables you to successfully play back scripts even when the application under test has been updated.

Objectives

In this lab, you will perform the following tasks:

► Set recognition score thresholds.

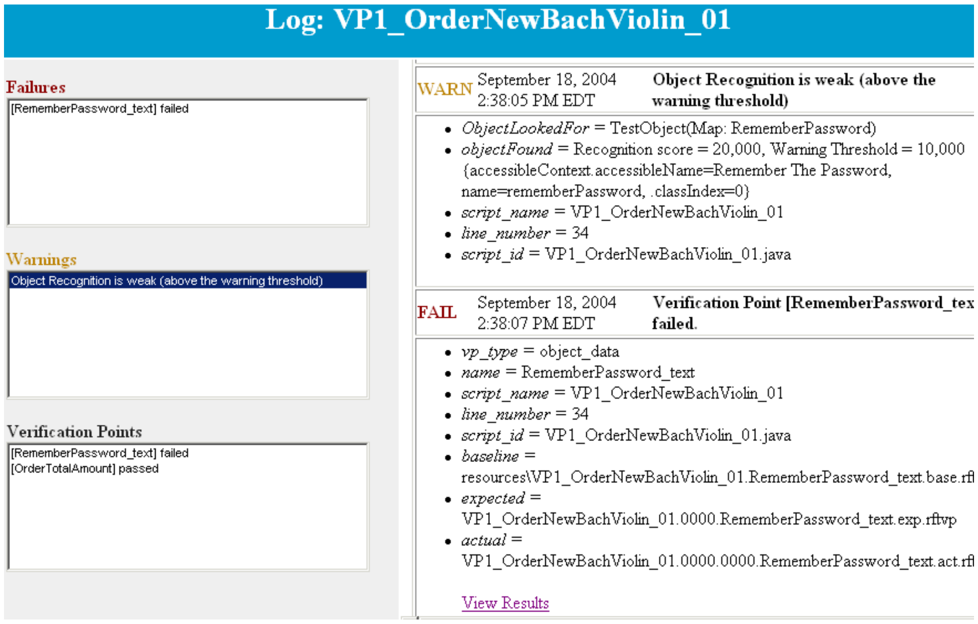
► Set up pattern-based recognition.

► Update the baseline for a verification point.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

6.1 Set Recognition Score Thresholds

Steps	Comments
1. Play back the VP1_OrderNewBachViolin_01 script. Name the log **DefaultScores**.	
2. When playback finishes, view the test log.	Note that the script did play back all of the user actions against the application.
3. Notice the left Failures and Warnings panes and the details in the right panes.	In the Warn details, the objectFound data indicates that there may be a problem with the RememberPassword object.



Log: VP1_OrderNewBachViolin_01

Failures
[RememberPassword_text] failed

Warnings
Object Recognition is weak (above the warning threshold)

Verification Points
[RememberPassword_text] failed
[OrderTotalAmount] passed

WARN September 18, 2004 2:38:05 PM EDT **Object Recognition is weak (above the warning threshold)**

- *ObjectLookedFor* = TestObject(Map: RememberPassword)
- *objectFound* = Recognition score = 20,000, Warning Threshold = 10,000 {accessibleContext.accessibleName=Remember The Password, name=rememberPassword, .classIndex=0}
- *script_name* = VP1_OrderNewBachViolin_01
- *line_number* = 34
- *script_id* = VP1_OrderNewBachViolin_01.java

FAIL September 18, 2004 2:38:07 PM EDT **Verification Point [RememberPassword_tex failed.**

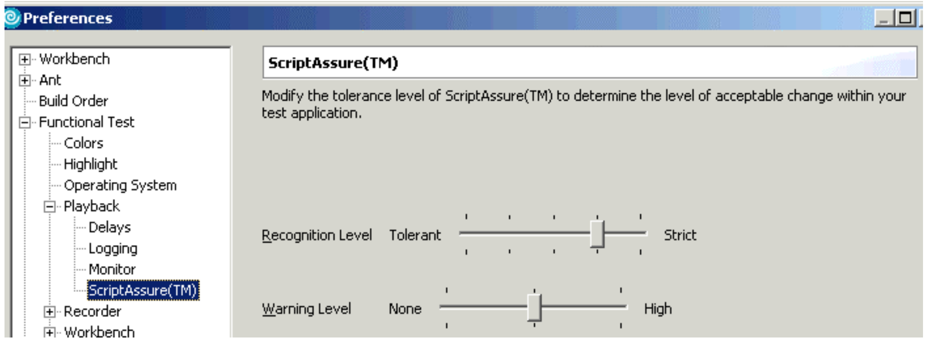
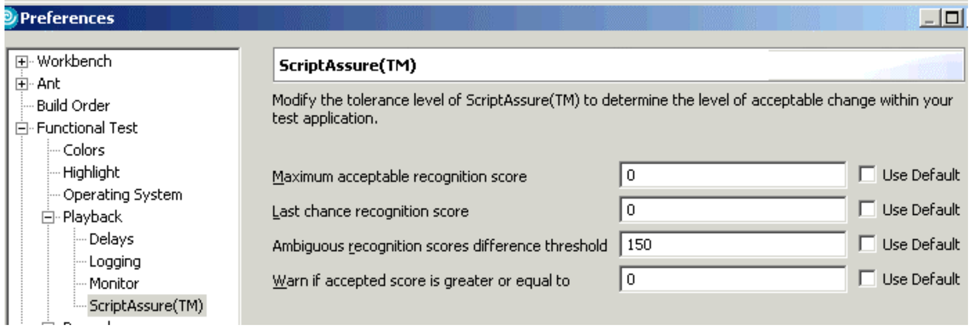
- *vp_type* = object_data
- *name* = RememberPassword_text
- *script_name* = VP1_OrderNewBachViolin_01
- *line_number* = 34
- *script_id* = VP1_OrderNewBachViolin_01.java
- *baseline* = resources\VP1_OrderNewBachViolin_01.RememberPassword_text.base.rfl
- *expected* = VP1_OrderNewBachViolin_01.0000.RememberPassword_text.exp.rftvp
- *actual* = VP1_OrderNewBachViolin_01.0000.0000.RememberPassword_text.act.rfl

View Results

Steps	Comments
4. Click the **View Results** hyperlink in the Fail details pane.	The VP Comparator opens.
5. Close the Comparator and the log.	
6. In Functional Tester, click **Window** > **Preferences**.	
7. Expand **Functional Test** and then expand **Playback**.	
8. Click **ScriptAssure(TM)**.	The standard ScriptAssure settings are displayed.

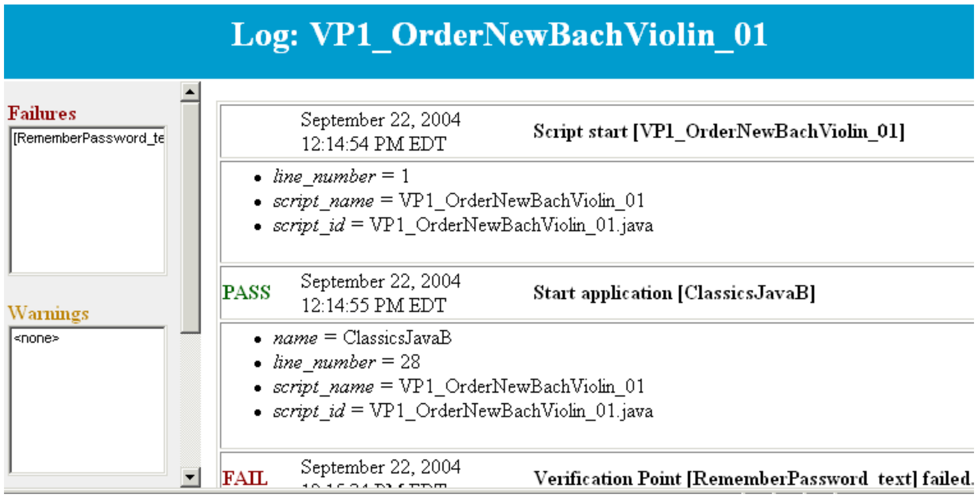
© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
 Preferences — ScriptAssure(TM): Modify the tolerance level of ScriptAssure(TM) to determine the level of acceptable change within your test application. Recognition Level: Tolerant – Strict. Warning Level: None – High.	
9. Click **Advanced**.	The advanced ScriptAssure settings are displayed.
10. Deselect the **Use Default** checkboxes. Set the **Ambiguous recognition score difference threshold** to **150**, set all other scores to **0**, and click **OK**.	
 Preferences — ScriptAssure(TM): Modify the tolerance level of ScriptAssure(TM) to determine the level of acceptable change within your test application. Maximum acceptable recognition score: 0 [ ] Use Default; Last chance recognition score: 0 [ ] Use Default; Ambiguous recognition scores difference threshold: 150 [ ] Use Default; Warn if accepted score is greater or equal to: 0 [ ] Use Default	
11. Play back the VP1_OrderNewBachViolin_01 script. Name the log **ExactScores**.	
12. When playback finishes, view the test log.	Note that the script did not play back all of the actions against the application.
13. Examine the details in the Fail pane, which indicates that the playback had an unhandled exception.	Functional Test didn't recognize "Remember the Password" the RememberPassword object.
14. Close the test log.	
15. Cancel the member logon and close ClassicsCD.	
16. In Functional Tester, click **Window** > **Preferences**.	The standard ScriptAssure settings are displayed.
17. Click **Advanced**.	The advanced ScriptAssure settings are displayed.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
18. Click the **Restore Defaults** button.	
19. Clear the **Use Default** check box next to **Warn if accepted score is greater than** and set the value to **20000**. Click **OK**.	Our DefaultScores log indicates that the RememberPassword object recognition score was 20000 while the warning threshold was 10000, the default.
20. Play back the VP1_OrderNewBachViolin_01 script. Name the log **WarnScores**	
21. When playback finishes, view the test log.	Note that there is no warning.



Log: VP1_OrderNewBachViolin_01

Failures
[RememberPassword_te

Warnings
<none>

	September 22, 2004 12:14:54 PM EDT	Script start [VP1_OrderNewBachViolin_01]

- *line_number* = 1
- *script_name* = VP1_OrderNewBachViolin_01
- *script_id* = VP1_OrderNewBachViolin_01.java

PASS	September 22, 2004 12:14:55 PM EDT	Start application [ClassicsJavaB]

- *name* = ClassicsJavaB
- *line_number* = 28
- *script_name* = VP1_OrderNewBachViolin_01
- *script_id* = VP1_OrderNewBachViolin_01.java

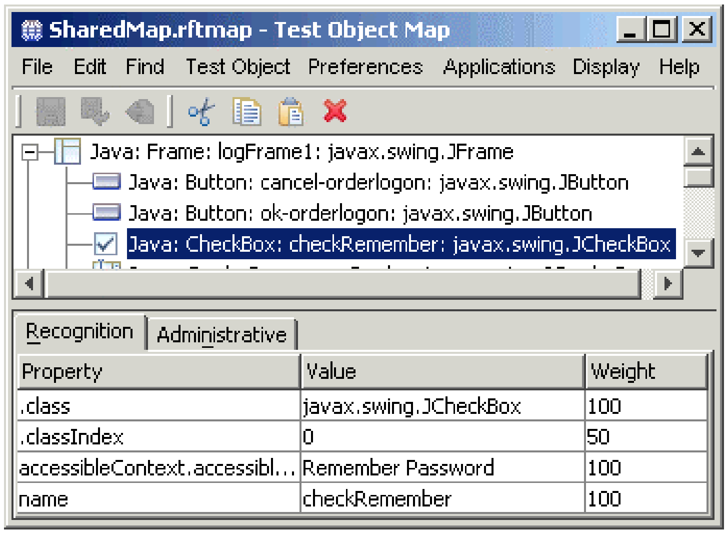
FAIL	September 22, 2004	Verification Point [RememberPassword_text] failed.

22. Close the test log.	
23. Restore all recognition score default values.	Typically, the default scores should be used.

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

6.2 Set Up Pattern-Based Recognition

Steps	Comments
1. Double-click the **SharedMap**.	SharedMap
2. Expand the Java:Frame:logFrame1:javax.swing.Jframe object.	
3. Click the check box named **checkRemember**.	**Remember Password** appears below in the **Recognition** tab **Value** column.

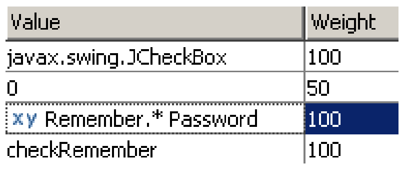

SharedMap.rftmap - Test Object Map

File Edit Find Test Object Preferences Applications Display Help

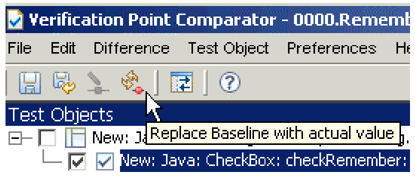
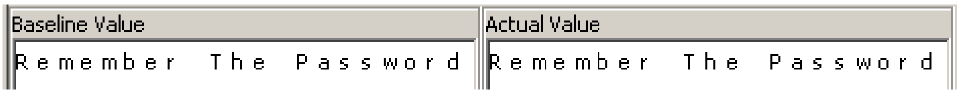
Java: Frame: logFrame1: javax.swing.JFrame
Java: Button: cancel-orderlogon: javax.swing.JButton
Java: Button: ok-orderlogon: javax.swing.JButton
Java: CheckBox: checkRemember: javax.swing.JCheckBox

Recognition | Administrative

Property	Value	Weight
.class	javax.swing.JCheckBox	100
.classIndex	0	50
accessibleContext.accessibl...	Remember Password	100
name	checkRemember	100

Steps	Comments
4. Right-click **Remember Password** and then click **Convert Value to Regular Expression** in the shortcut menu.	Note that the value is now designated as a regular expression by the blue "xy."
5. Double-click the **Remember Password** value and type **.*** between Remember and Password. (Be sure to include the period.)	 Value / Weight: javax.swing.JCheckBox 100; 0 50; xy Remember.* Password 100; checkRemember 100
6. Close the Test Object Map window, saving the changes you made.	
7. Play back the VP1_OrderNewBachViolin_01 script. Name the log **RE**.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
8. When playback finishes, view the test log and expand all events.	Because we used a regular expression, there is no warning about finding the object with the default recognition scores. However, the data captured by the VP is different from the baseline data so the VP still fails.
9. Click the **View Results** hyperlink at the bottom of the Fail details pane.	The Verification Point Comparator opens.
10. Click the **Replace Baseline with actual value** button.	 Verification Point Comparator - 0000.Rememb File Edit Difference Test Object Preferences He Test Objects New: Ja Replace Baseline with actual value New: Java: CheckBox: checkRemember: Note that the Baseline and Actual Values are now the same.
 Baseline Value: R e m e m b e r T h e P a s s w o r d; Actual Value: R e m e m b e r T h e P a s s w o r d	
11. Close the Verification Point Comparator window and the test log.	
12. Play back the VP1_OrderNewBachViolin_01 script. Name the log **UpdatedVP**.	
13. When playback finishes, view the test log.	Note that the RememberPassword VP now passes.
14. Close the test log. Close the VP1_OrderNewBachViolin_01 script.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 7.1 Creating Data-Driven Tests in Functional Tester

Objectives

In this lab, you will perform the following tasks:

- ► Record a data-driven test script
- ► Change a verification point reference from a literal to a datapool variable
- ► Edit data in a datapool
- ► Run a data-driven test script and view the results

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.1.1 Record a Functional Test Script

Steps	Comments
1. Open the Training-TST279 project.	
2. Begin recording a functional test script to place an order in ClassicsCD: a. Name the script **OrderTotal** and then click **Next**. b. Accept all defaults and click **Finish**.	
3. On the Recording toolbar, click **Start Application**, select the **ClassicsJavaA – java** application, and then click **OK**.	ClassicsCD opens.
4. In ClassicsCD, select **Schubert String Quartets Nos. 4 & 14** and then click **Place Order**.	
5. Accept the defaults and click **OK** to close the Member Logon window.	The Place an Order window opens.
6. On the Recording toolbar, click **Insert Data Driven Commands**.	The test script recording pauses, and the Insert Data Driven Actions window opens.
7. In the ClassicsCD Place an Order window, type a credit card number and an expiration date.	The recording is paused, so these actions are not recorded. However, when you capture the object, these literal values are captured and can later be replaced by variables.
8. In the Insert Data Driven Actions window, drag the **Object Finder** to select the entire Place an Order window and then release the mouse.	A red outline indicates the object that is selected. Once you release the mouse, the Data Driven Actions window re-opens. Information about the objects you selected appears in the Data Driven Commands table.

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.1.2 Add Descriptive Variable Names to the Data

Steps	Comments
1. Resize the window as necessary. In the Data Driven Commands table, in the first row of the **Variable** column, double-click **ItemText** to select it.	Item text is the first descriptive name under the **Variable** header. The first row appears selected.
2. Type **Composer** in the cell.	
3. Double-click in the cell below Composer and type **Item**.	
4. Repeat step 3 for each value in the **Variable** column. Use the following descriptive names to complete the **Variable** column: **Variable** Composer Item Quantity CardNum CardType ExpDate Name Street CityStateZip Phone	You have already entered Composer and Item from this list. **Note:** Do not use spaces in variable names.
5. Click **OK**.	The Insert Data Driven Actions window closes and script recording resumes.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.1.3 Insert a Verification Point with a Datapool Reference

Steps	Comments
1. On the Recording toolbar, click **Insert Verification Point or Action Command**.	The Verification Point Action Wizard opens.
2. On the Select an Object page, click the mouse and drag the **Object Finder** over **$19.99** in the Place an Order window and then release the mouse button.	A red outline indicates that an object is selected. When the order Total object is selected, release the mouse button.
3. On the Select an Action page, if necessary, click **Perform Data Verification Point** and then click **Next**.	
4. Click **Next**.	
5. On the Verification Point Data page toolbar, click **Convert Value to Datapool Reference.**	The Datapool Reference Converter dialog box opens. You can convert the literal value that you recorded as the baseline to compare to a datapool variable.
6. In the **Datapool Variable** box, type **Total** as the new variable name in the datapool.	
7. If necessary, select the **Add value to new record in datapool** check box.	This action will add the Total to the existing datapool record you created.
8. Click **OK** to close the Datapool Reference Converter.	
9. Click **Finish**.	
10. In ClassicsCD, click **Place Order** and then click **OK** to close the message box.	
11. Close ClassicsCD.	
12. Stop recording.	The test script opens in the script editor. The new datapool is listed in the Script Explorer.
13. If necessary, close the Test Object Map window.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.1.4 Add Data to the Datapool

Steps	Comments
1. In the Script Explorer, double click the datapool to open it.	The datapool displays at the bottom of the screen below the script editor. Look at the data that has been collected.
2. Double-click the test datapool title bar to expand the datapool editor.	
3. To add an empty record to the datapool, right-click in the datapool editor under row 0 and then click **Add Record** on the shortcut menu.	
4. Be sure that **Add After** is selected and click **OK**.	
5. Add a second empty row.	
6. Position the mouse pointer in the first cell of row 0, right-click, and then click **Copy**.	
7. Position the mouse in the first cell of row 1, right-click, and then click **Paste.**	When prompted, overwrite existing values.
8. Position the mouse pointer in the first cell of row 2, right-click, and then click **Paste**.	When prompted, overwrite existing values.
9. In row 1, change the quantity to **2** and the total to **$38.98**.	You may need to scroll to the right in the datapool to see the total column.
10. In row 2, change the quantity to **3** and the total to **$57.97**.	
11. Double-click the test datapool title bar to restore the datapool editor to its docked view.	
12. Click **X** in the datapool title bar to close the datapool editor and save your changes.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.1.5 Run the Test Script and View the Results

Steps	Comments
1. Run the OrderTotal test script.	The Select Log dialog opens.
2. Name the test log **OrderTotal** and then click **Next**.	
3. For the Datapool Iteration Count, select **3** and then click **Finish**.	The script runs three times, each time drawing data from a different row, or record, in the datapool.
4. When the test script finishes, view the results in the test log.	
5. Close the test log.	
6. Close the OrderTotal test script.	

© Copyright IBM Corp. 2005
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 7.2
Importing a Datapool

Objectives

In this lab, you will perform the following tasks:

- Import and edit an external datapool
- Associate the datapool with a test script
- Change script literal values to variable references
- Play back script
- Troubleshoot an imported datapool

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

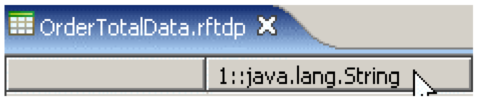
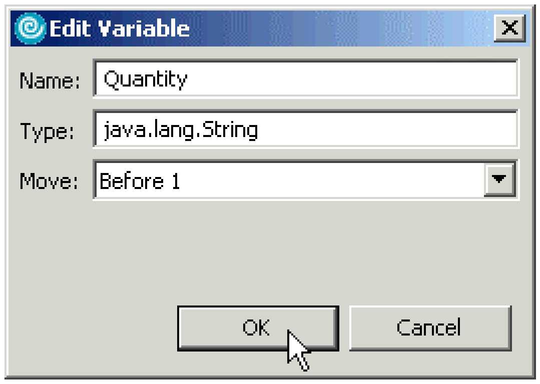
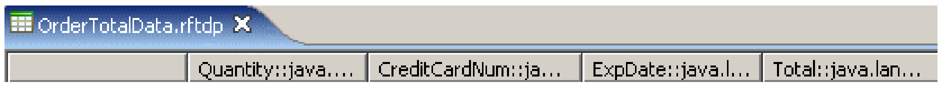
7.2.1 Import an External Datapool into a Functional Tester Project

Steps	Comments
1. Open the Training-TST279 project.	
2. On the main menu, click **File > New > Test Datapool**.	
3. In the Create Test Datapool dialog box: a. Accept the default location. **b.** Name the datapool **OrderTotalData.** c. Click **Next**.	The location should be within the Functional Tester project.
4. In the Import Datapool dialog box, browse to and select the following file: **C:\Training-TST279\ClassicsOrders.csv**.	
5. Accept other defaults and click **Finish**.	The datapool is imported into the project. Notice the datapool in the project directory.
6. Check the data to see that the datapool has imported correctly.	You should have seven rows of data (zero through six).

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.2.2 Edit the Variable Names

Steps	Comments
1. In the Datapool Editor, in the variables row of the datapool, click in the column heading that contains the number 1.	The Edit Variable dialog box opens.
 OrderTotalData.rftdp — 1::java.lang.String	
2. In the Name field, delete the 1, type **Quantity**, and then click **OK**.	
 Edit Variable — Name: Quantity; Type: java.lang.String; Move: Before 1; OK; Cancel	
3. Edit the remaining variable names in the column headings: • Change 2 to CreditCardNum • Change 3 to ExpDate • Change 4 to Total	
 OrderTotalData.rftdp — Quantity::java.... — CreditCardNum::ja... — ExpDate::java.l... — Total::java.lan...	
4. Save the datapool.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.2.3 Record a Test Script

Steps	Comments
1. Start recording a new test script: a. Name the script **OrderTotal2**. b. Accept the default location and click **Next**. c. On the Select Script Assets page, accept all default settings and click **Finish**.	
2. Start ClassicsJavaA. a. In the Recording dialog box, click the **Start Application** button. b. In the Applications list, verify that ClassicsJavaA – java is selected and click **OK**.	Recording begins.
3. In ClassicsCD, select the **Beethoven Symphony No. 9** CD and then click **Place Order**.	The Member Logon window opens.
4. Log in as existing customer Trent Culpito (no password) and click **OK**.	
5. Click in the **Quantity** box. Press **[HOME] [SHIFT+END] [DELETE]** and then type 1.	
6. In the **Card Number** box type: 1234 1234 1234 1234 and in the **Expiration Date** box type: 12/08.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
7. Create a Total verification point for the total dollar amount due. a. On the Recording toolbar, click the **Insert Verification Point or Action Command** button. b. Click the **Object Finder** button, hold the mouse button down, and drag the **Object Finder** over the total dollar amount due ($16.99). c. Release the mouse button. d. In the Verification Point and Action Wizard dialog box, verify that **Perform Data Verification Point** is selected and then click **Next**. e. In the Verification Point Name box, type **Total**. f. Click **Next**. g. Click **Finish**.	
8. Complete the order and stop recording. a. Click **Place Order**. b. In the message box, click **OK**. c. Close the ClassicsCD application. d. In the Recording toolbar, click the **Stop Recording** button. e. If necessary, close the Test Object Map window.	
9. Play back the script. a. Click the **Run Functional Test Script** button. b. Accept the default log name and click **Finish**. c. View the log. d. Close the log.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.2.4 Associate the Datapool with the Test Script

Steps	Comments
1. Under the Projects view, right-click the **OrderTotalData** datapool and then click **Associate with Script** on the shortcut menu.	
2. In the Associate the datapool with scripts dialog box, if necessary, expand the Training-TST279 project node, check the box to select the **OrderTotal2** script, and then click **Finish**.	
3. If necessary, close the Datapool Editor.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.2.5 Change the Verification Point Reference

Steps	Comments
1. To change the verification point reference from a literal to a variable, in the Script Explorer, double-click the **Total** verification point to open the Verification Point Editor.	We are still working with the OrderTotal2 script.
2. Click the **Convert Value to Datapool Reference** button.	
3. In the Datapool Reference Converter dialog box, select **Total** from the Datapool Variable drop-down list and then click **OK**.	Once you change the verification point reference, the literal value is changed to the variable reference.
4. Save changes and close the Verification Point Editor.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.2.6 Replace the Literal Values in the Script with Variables

Steps	Comments
1. If necessary, open the **OrderTotal2** script.	
2. In the Script Editor, scroll to find the line in the script that sets the quantity: **placeAnOrder().InputKeys (“{ExtHome}+{ExtEnd} {ExtDelete}1”);**	When Functional Tester replaces the literal **“{ExtHome}+{ExtEnd}{ExtDelete}1”** with a datapool reference to the Quantity variable, the key sequence that clears the box before typing quantity will be lost unless you put it on its own line.
3. Copy the line and paste it into a blank line immediately below the line you copied.	
4. In the first of the two matching lines, delete the **1**.	
5. Click **Script > Find Literals and Replace with Datapool Reference**.	The Datapool Literal Substitution dialog box opens.
6. In the Datapool Literal Substitution dialog box, verify that **All** is selected under Literal Type.	
7. Click **Find** until the literal for the quantity, **“{ExtHome}+{ExtEnd}{ExtDelete}1”** displays in the Literal box.	Depending on the cursor position in the script, you may already be on the line.
8. In the **Datapool Variable** box, select **Quantity** from the drop-down list and then click **Replace**.	Notice in the Script Editor that the quantity literal value is replaced by the variable “Quantity.”
9. Click **Find** until the literal for the credit card number, 1234 1234 1234 1234, displays in the **Literal** box.	
10. In the **Datapool Variable** box, select **CreditCardNum** from the drop-down list and then click **Replace**.	Notice in the Script Editor that the credit card number literal value is replaced by the variable “CreditCardNum.”

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
11. In the Datapool Literal Substitution dialog box, click **Find** until the literal for the expiration date displays in the **Literal** box.	
12. In the **Datapool Variable** box, select **ExpDate** from the drop-down list and then click **Replace**.	Notice in the Script Editor that the expiration date literal value is replaced by the variable “ExpDate.”
13. Click **Close**.	The literals for quantity, credit card number, and expiration date are now replaced in the script with variables.

Sample script (Your script should look similar to the following in the Place an Order section. It is okay if you do not have "{ExtHome}+{ExtEnd}{ExtDelete}" keystrokes before the **Credit Card Number** and **Expiration Date** input fields.):

```
// Frame: Place an Order
quantityText().click(atPoint(23,12));
placeAnOrder().inputKeys("{ExtHome}+{ExtEnd}{ExtDelete}");
placeAnOrder().inputKeys(dpString("Quantity"));
cardNumberIncludeTheSpacesText().click(atPoint(18,8));
placeAnOrder().inputKeys("{ExtHome}+{ExtEnd}{ExtDelete}");
placeAnOrder().inputKeys(dpString("CreditCardNum"));
expirationDateText().click(atPoint(21,11));
placeAnOrder().inputKeys("{ExtHome}+{ExtEnd}{ExtDelete}");
placeAnOrder().inputChars(dpString("ExpDate"));
TotalVP().performTest();
placeOrder2().click();
```

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.2.7 Run the Test Script and View the Results

Steps	Comments
1. Run the test script.	The Select Log dialog opens.
2. Name the test log **OrderTotal2_run002** and click **Next**.	
3. For the Datapool Iterator Count, select **4** and then click **Finish**.	The script runs. It will run four times, each time drawing data from a different row in the datapool.
4. When the test script finishes, view the results in test log.	
5. In the test log, scroll down to the results of the first verification point, and click **View Results**.	This verification point should have failed. Can you tell why?
6. In the Verification Point Comparator, click the **Show Hidden Characters** button.	Notice that there is an extra space after the $16.99 in the expected value column. There is an extra space after that value in the datapool.
7. Close the Verification Point Editor and test log.	
8. In the Projects view, double-click the **OrderTotalData** datapool to open it in the Datapool Editor.	
9. Double-click the cell in the **Total** column that contains $16.99. Delete the extra space after $16.99.	Check the other cells in the column to verify that there are no extra spaces in the cells.
10. Save changes and close the Datapool Editor.	
11. Re-run the OrderTotal2 script and view the results in the test log.	
12. Close any open test logs, scripts, and datapools.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Lab 7.3
Exporting a Datapool

Objectives

In this lab, you will perform the following tasks:

- ► Create a datapool while recording a test script
- ► Edit a recorded verification point
- ► Edit a datapool
- ► Import a CSV file into a new datapool
- ► Associate a datapool with an existing test script
- ► Modify a recorded script to use datapool variables

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.3.1 Record the Script

Steps	Comments
1. Start recording a new functional test. a. Name the script **OrderTotal3_part1** and then click **Next**. b. On the Select Script Assets page, accept all default settings and click **Finish**.	The Functional Tester window is minimized, and the Recording dialog box opens.
2. Start ClassicsJavaA. a. In the Recording dialog box, click the **Start Application** button. b. In the Application Name list, verify that **ClassicsJavaA – java** is selected and click **OK**.	The Start Application dialog box opens. The ClassicsCD window opens.
3. Select the **Haydn Violin Concertos** CD and then click **Place Order**.	To select the Haydn Violin Concertos CD, expand the Haydn folder and then click **Violin Concertos**. The Member Logon dialog box opens.
4. Log in as Trent Culpito. a. Verify that **Trent Culpito** is displayed in the **Full Name** box. b. Click **OK**.	The Place an Order dialog box opens.
5. On the Recording toolbar, click the **Insert Data Driven Commands** button.	Recording is paused and the Insert Data Driven Actions dialog box opens.
6. In the Place an Order dialog box, **Card Number** box, type 1234 1234 1234 1234.	
7. In the **Expiration Date** box, type 12/08.	
8. In the Insert Data Driven Actions dialog box, use the object finder to select the entire Place an Order window.	The Insert Data Driven Actions dialog box disappears, and the mouse pointer is changed to a small blue hand. When the Place an Order dialog box is selected, a red border is displayed around the entire dialog box.

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
9. In the **Variable** column, rename the first six variables as follows:	Double-click the variable to edit its value.

Test Object	**Variable**
ItemText	**Composer**
_1499Text	**Item**
QuantityText	**Quantity**
CardNumberIncludeTheSpacesText	**CardNumber**
CreditCombo	**CardType**
ExpirationDateText	**ExpDate**

Steps	Comments
10. For each of the last four rows, select the row and then click the **Delete the selected row from the commands table** button.	
11. Click **OK**.	The Insert Data Driven Actions dialog box closes and recording resumes.

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
12. Create a Total verification point for the total dollar amount due.	
a. On the Recording toolbar, click the **Insert Verification Point or Action Command** button.	Recording is paused, and the Verification Point and Action Wizard dialog box, Select an Object page opens.
b. Click the **Object Finder** button and hold the mouse button down.	The Verification Point and Action Wizard dialog box disappears, and the mouse pointer is changed to a small blue hand.
c. Drag the **Object Finder** over the total dollar amount due (**$15.99**).	When the total amount is selected, a red border is displayed around the text **$15.99**.
d. Release the mouse button.	The Verification Point and Action Wizard dialog box, Select an Action page opens.
e. In the Verification Point and Action Wizard dialog box, verify that **Perform Data Verification Point** is selected and then click **Next**.	The Insert Verification Point Data Command page opens.
f. In the Verification Point Name box, type Total.	
g. Click **Next**.	The Verification Point Data page opens, displaying the selected object and its recognition properties.
h. Click **Finish**.	The Verification Point and Action Wizard dialog box closes and recording resumes.
13. Complete the order, exit ClassicsCD, and stop recording.	
a. Click **Place Order**.	A message box opens indicating that your order has been received.
b. Click **OK**.	
c. Click the **Close** button to exit ClassicsCD.	
d. In the Recording toolbar, click the **Stop Recording** button.	The new script opens in the Script Editor.
e. If necessary, close the Test Object Map window.	
14. Play back the script.	
a. Click the **Run Functional Test Script** button.	
b. Accept the default log name and click **Finish**.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
15. View the results and close the test log.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.3.2 Edit the Verification Point to Reference the Datapool

Steps	Comments
1. In the Script Explorer, under Verification Points, double-click the **Total** VP.	The Verification Point Editor opens displaying the expected value for the Total VP.
2. On the Verification Point Editor toolbar, click the **Convert Value to Datapool Reference** button.	The Datapool Reference Converter dialog box opens.
3. In the **Datapool Variable** box, type Total.	
4. Verify that the **Add value to new record in datapool** option is selected.	
5. Click **OK**.	
6. Close the Verification Point Editor.	
7. In the Save Verification Point message box, click **Yes** to save your changes.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.3.3 Edit the Datapool and Play Back the Script

Steps	Comments
1. In the Script Explorer, double-click **Private Test Datapool**.	
2. In the second row of the datapool, change the quantity to **2** and the total to $**30.98**.	Double-click the quantity to edit its value.
3. Save your changes.	
4. Play back the script, assigning a new log name and playing through two iterations. a. Click the **Run Functional Test Script** button. b. Name the log **OrderTotal3_part1_run2**. c. Click **Next**. d. In the **Datapool Iteration Count** box, select **2**. e. Click **Finish**.	
5. View the results and then close the test log.	

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.3.4 Export and Edit the Datapool

Steps	Comments
1. In the Script Explorer, right-click **Private Test Datapool** and then click **Export**.	The Export page opens.
2. Click **Browse**.	
3. Navigate to **C:\Training-TST279** and name the file **OrderTotalData2.csv**.	
4. Click **Save**.	You are returned to the Export page.
5. Click **Finish**.	
6. Open Windows Explorer, navigate to **C:\Training-TST279**, and double-click **OrderTotalData2.csv**.	The datapool opens in Microsoft Excel. The datapool contains a row of column headings followed by two rows of data.
7. Copy row 3 and paste it into rows 4 through 6.	
8. Edit rows 4 through 6 as follows:	

Row	Quantity	Total
4	**5**	**$75.95**
5	**10**	**$150.90**
6	**50**	**$750.50**

Steps	Comments
9. Save the file and close Excel.	
10. Close Windows Explorer.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.3.5 Record Another Script

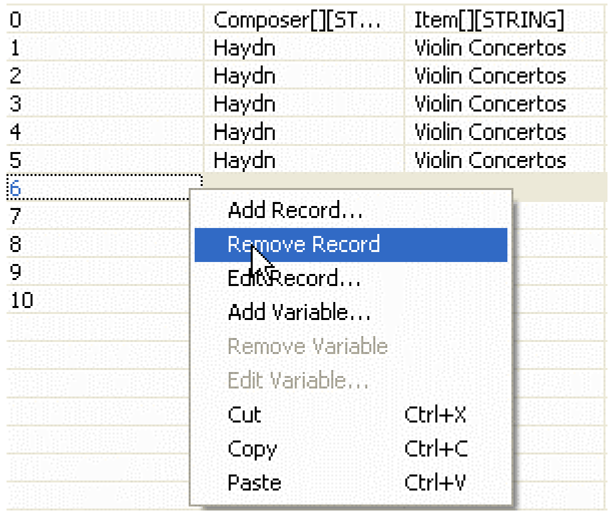
Steps	Comments
1. Start recording a new script named **OrderTotal3_part2**.	
2. Start ClassicsJavaA.	The ClassicsCD window opens.
3. Select the **Haydn Violin Concertos** CD and then click **Place Order**.	The Member Logon dialog box opens.
4. Log in as Trent Culpito.	The Place an Order dialog box opens.
5. Clear the **Quantity box** and then type 2.	To clear the box: 1. Press **Home**. 2. Press **Shift+End**. 3. Press **Delete**.
6. Clear the **Card Number** box and then type 2222 2222 2222 2222.	
7. Clear the **Expiration Date** box and then type 12/08.	
8. Create a verification point named **Total** for the total dollar amount due (**$30.98**).	
9. Complete the order, exit ClassicsCD, and stop recording.	The new script is opened in the Functional Tester main window.
10. If necessary, close the Test Object Map window.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.3.6 Import Datapool and Associate it with a Test Script

Steps	Comments
1. In the Functional Tester main window, click **File > New > Test Datapool**.	
2. Accept the default location, and in the Name box, type OrderTotal3.	
3. Click **Next**.	The Import Datapool page opens.
4. Click **Browse**.	
5. Navigate to **C:\Training-TST279** and double-click **OrderTotalData2.csv**.	
6. Select the **First Record is Variable Information** option.	With this option selected, Functional Tester interprets the first record in the CSV file as column headings rather than data.
7. Click **Finish**.	The new datapool is opened in the Functional Tester main window. The data from OrdersTotal2.csv is now contained in a new datapool (OrderTotal3.rftdp). For Functional Tester to use this data during test script playback, the datapool must be associated with the test script.
8. If you have rows that are empty, right-click the row number and select **Remove Record**.	



	Composer[][ST...	Item[][STRING]
0	Composer[][ST...	Item[][STRING]
1	Haydn	Violin Concertos
2	Haydn	Violin Concertos
3	Haydn	Violin Concertos
4	Haydn	Violin Concertos
5	Haydn	Violin Concertos
6		
7		
8		
9		
10		

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
9. Close the TestDatapool (OrderTotal3.rftdp) and save the changes, if necessary.	Click the X in the upper right corner of the view.
10. In the Script Explorer, right-click **Test Datapool**.	The OrderTotal3_part2 script should be open.
11. Click **Associate with Datapool**.	The Select Test Datapool dialog box opens displaying a list of datapools contained in the current project.
12. Click /**OrderTotal3.rftdp** and then click **OK**.	The datapool is listed under Test Datapool in the Script Explorer.

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

7.3.7 Edit Script to Use Datapool Variables

Steps	Comments
1. In the OrderTotal3_part2 script, find and copy the line that sets the quantity: **placeAnOrder().inputKeys (“{ExtHome}+{ExtEnd} {ExtDelete}2”);**	When Functional Tester replaces the literal **“{ExtHome}+{ExtEnd}{ExtDelete}2”** with a reference to the Quantity variable, the key sequence that clears the box before typing the quantity will be lost unless it is retained in a line of its own.
2. Paste the line into a blank line immediately after the line that you copied.	
3. In the first of the two matching lines, delete the value **2**.	
4. Click **Script > Find Literals and Replace with Datapool Reference**.	The Datapool Literal Substitution dialog box opens.
5. Click **Find** until you find the following literal: **”{ExtHome}+{ExtEnd}-{ExtDelete}2”**	
6. In the Datapool Variable list, select **Quantity**.	
7. Click **Replace**.	In your script, Functional Tester replaces the literal value **2** with a reference to the **Quantity** variable in the RoundTrip datapool.
8. Click **Close**.	
9. In your script, find the line that sets the quantity and verify that the literal has been replaced by a reference to the Quantity variable: **(dpString(“Quantity”))**	Now that the script will use quantities defined in the datapool, we must update the Total VP to use its corresponding values defined in the datapool. Otherwise the VP expected result will be $30.98 regardless of the quantity of CDs ordered.
10. Restart Functional Tester and open the Training-TST279 project.	If prompted, click **Yes** to Save Resources.
11. If necessary, open the **OrderTotal3_part2** script and close the Datapool editor.	
12. In the Script Explorer, double-click the **Total** VP.	The Verification Point Editor opens.

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Steps	Comments
13. In the toolbar above the right pane, click the **Convert Value to Datapool Reference** button.	The Datapool Reference Converter dialog box opens.
14. In the **Datapool Variable** box, select **Total**.	
15. If necessary, clear the **Add value to new record in datapool** check box.	
16. Click **OK**.	Verify that the right pane of the Verification Point Editor now displays **Total** instead of $30.98.
17. Close the Verification Point Editor.	
18. In the Save Verification Point message box, click **Yes** to save your changes.	
19. Play back the script, naming the log **OrderTotal3_part2_run3** and setting the iteration count to 5.	You may need to click the script view to make it active before you can play it back. When playback is complete, the log opens in a browser window.
20. Examine the test results.	Did any verification points fail? If so, how can you fix the problem? If you are unsure, ask your instructor for further assistance.
21. If necessary, fix problems and play back the script again.	

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

© Copyright IBM Corp. 2005

Course materials may not be reproduced in whole or in part without the prior written permission of IBM.