

IBM WebSphere Commerce V7 Feature Pack 1– Lab exercise

FVT test harness lab

What this exercise is about	1
Lab requirements	1
What you should be able to do	1
Introduction	2
Exercise instructions	2
Part 1: Download required software used by Storefront Automation Harness	3
Part 2: Installing the Storefront Automation Harness.....	4
Part 3: Importing Madisons starter store test bucket project	10
Part 4: Update config.properties file.....	12
Part 5: Run a test scenario	14
Part 6: Verifying the users are registered	18

What this exercise is about

The objective of this lab is to provide you with step-by-step instructions on how to get started with the storefront automation harness from setting up your environment to running a sample Madisons starter store test against Firefox browser.

This lab is provided **AS-IS**, with no formal IBM support.

Lab requirements

Before you start this lab, ensure your system meets following requirements:

- Installed WebSphere Commerce Developer V7
- Downloaded the test assets package **StorefrontTestAssets.zip** from the following web site and extract the file into directory <TEMP>/StorefrontTestAssets

https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-stae

What you should be able to do

At the end of this lab you should be able to install the Storefront test harness package and the Madisons starter store test bucket. You will be able to run a simple test.

Introduction

Storefront automation harness is software which allows you to automate the storefront functional verification test. Storefront automation harness uses JUnit to run the test scripts and uses the Selenium tool to perform the actions on browsers.

Storefront automation harness consists of four key assets: JUnit, test bucket project, Storefront Test Automation Engine and Selenium. The test assets package you downloaded only contains the starter store test bucket projects and the Storefront Test Automation Engine. To setup your own test environment, you need to download JUnit and the Selenium code.

In this lab, you will run the functional verification test on the same machine where you have WebSphere Commerce Developer installed.

Exercise instructions

The instructions in this lab are Windows® operating-system specific. The directory locations and variables are specified in the lab instructions using symbolic references. You will see some screen captures containing the specific values for these variables. When you do this lab, you should replace them with your own values.

Reference variable	Description	Value
<TEMP>	Temp directory	
<WCDE_installdir>	WebSphere Commerce Developer install directory	
<RAD_workspace>	Workspace directory of the Rational Application Developer	
<SiteAdmin_ID>	Site administrator ID	
<SiteAdmin_PWD>	Site administrator password	

Part 1: Download required software used by Storefront Automation Harness

In this section, you will need to download some third party's software which will be used by the Storefront Automation Harness

- ____ 1. Go to following web site

<http://sourceforge.net/projects/junit/files/junit/>

Download **junit3.8.2.zip** file. Extract the files into directory <TEMP>/

Note: The Storefront Test Automation Engine was tested with JUnit 3.8.2.

- ____ 2. Go to the web site

<http://seleniumhq.org/download/>

Download the **Selenium RC** project file selenium-remote-control-1.0.3.zip. Extract the file into directory <TEMP>/selenium-remote-control

Note: The Storefront Test Automation Engine was tested with Selenium RC 1.0.1. Since version 1.0.1 was removed from the download site, you need to use the newer version 1.0.3 for this lab.

- ____ 3. Go to the web site

<http://hc.apache.org/downloads.cgi>

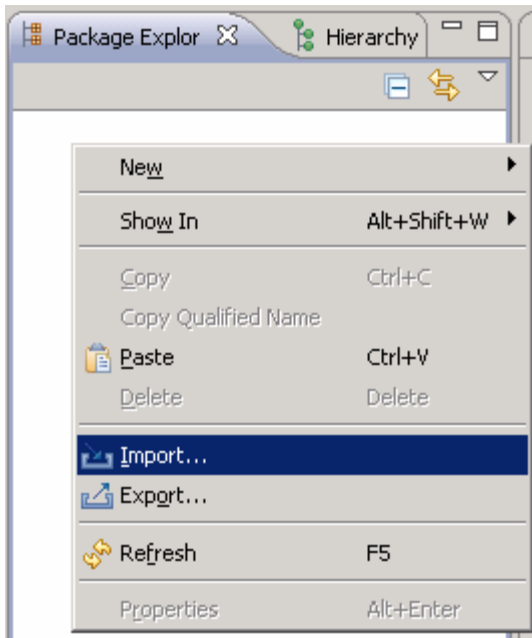
Download the **HttpClient Binary with dependencies** file httpcomponents-client-4.0.1-bin-with-dependencies.zip. Extract the files into directory <TEMP>/httpcomponents-client/

Note: The Storefront Test Automation Engine was tested with HttpClient Binary 4.0.1.zip.

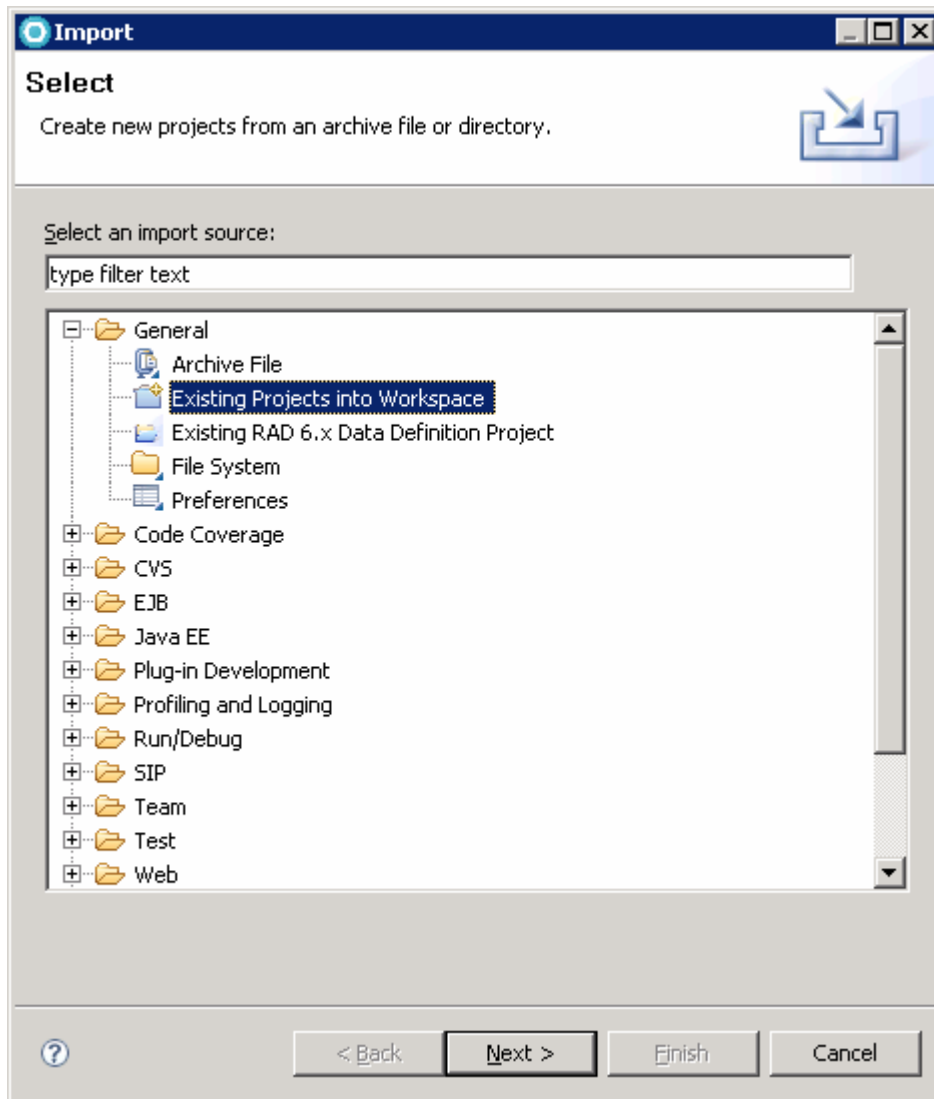
Part 2: Installing the Storefront Automation Harness

In this part, you will install the Storefront Automation Harness project in Rational Application Developer.

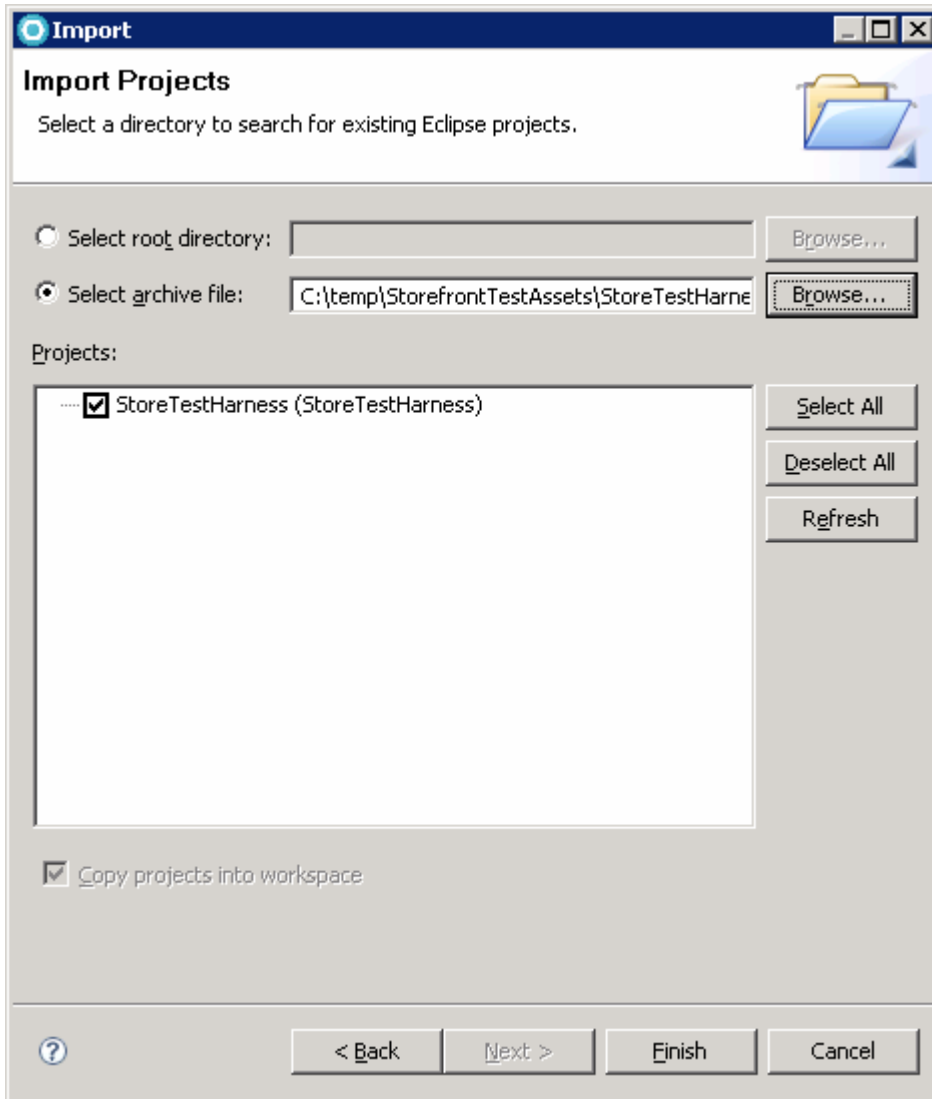
- ___ 1. Launch Rational Application Developer from Windows Start menu Start > Programs > IBM Software Delivery Platform > IBM Rational Application Developer 7.5 > IBM Rational Application Developer
- ___ 2. Select a workspace directory <RAD_workspace>
- ___ 3. Ensure you are in the Java perspective. If you are not in the Java perspective, follow these steps:
 - ___ a. In the menu bar, select Window > Open Perspective > Other ...
 - ___ b. Select Java
 - ___ c. Click OK
- ___ 4. Right click in the Package Explorer view, select **Import**



- ___ 5. Select **General > Existing Projects into Workspace**. Click **Next**



- ___ 6. Select the **Select archive file** option and click **Browse**.
- ___ 7. Navigate to the directory <TEMP>/StorefrontTestAssets, select the **StoreTestHarness.zip** file. Click **Finish**.



8. Open **Windows Explorer** and navigate to the directory **<RAD_workspace>/StoreTestHarness/lib**. Copy following files into **lib** directory

<TEMP>/junit3.8.2/junit.jar

<TEMP>/selenium-remote-control/selenium-java-client-driver-1.0.1/selenium-java-client-driver.jar

<TEMP>/selenium-remote-control/selenium-server-1.0.3/selenium-server.jar

<TEMP>/httpcomponents-client/httpcomponents-client-4.0.1/lib/apache-mime4j-0.6.jar

<TEMP>/httpcomponents-client/httpcomponents-client-4.0.1/lib/commons-codec-1.3.jar

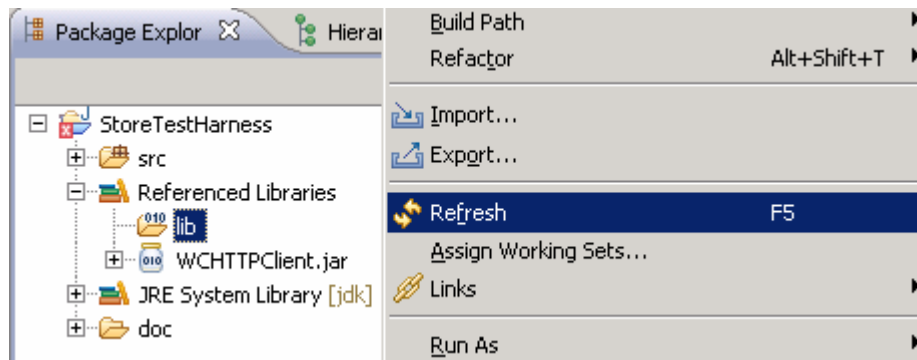
<TEMP>/httpcomponents-client/httpcomponents-client-4.0.1/lib/commons-logging-1.1.1.jar

<TEMP>/httpcomponents-client/httpcomponents-client-4.0.1/lib/httpclient-4.0.1.jar

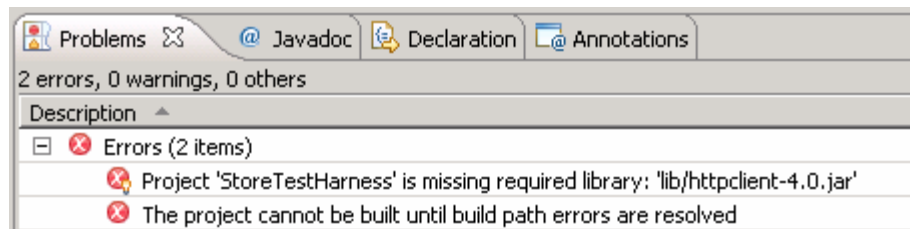
<TEMP>/httpcomponents-client/httpcomponents-client-4.0.1/lib/httpcore-4.0.1.jar

<TEMP>/httpcomponents-client/httpcomponents-client-4.0.1/lib/httpmime-4.0.1.jar

9. In the Package Explorer view, expand StoreTestHarness > Referenced Libraries. Right click the **lib** folder and select **Refresh**

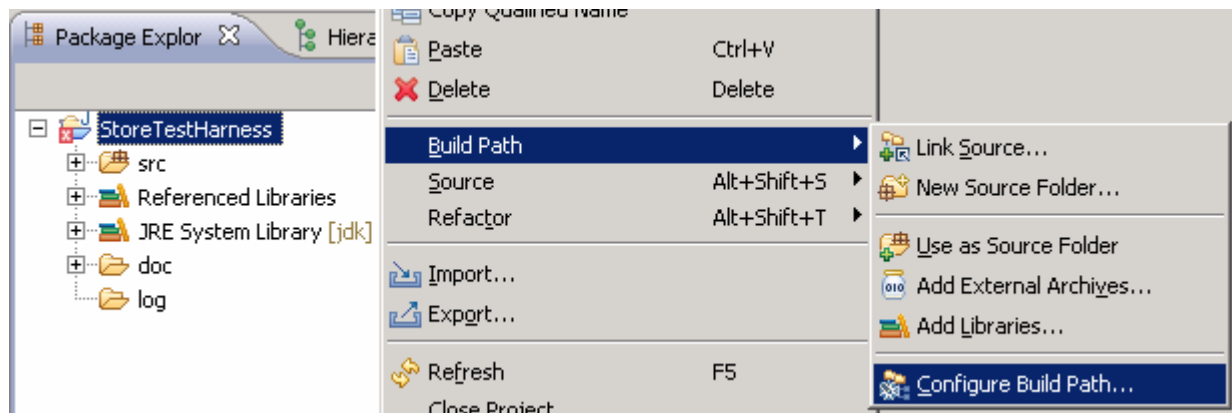


10. In the **Problems** view, you may see following two errors related with the project building

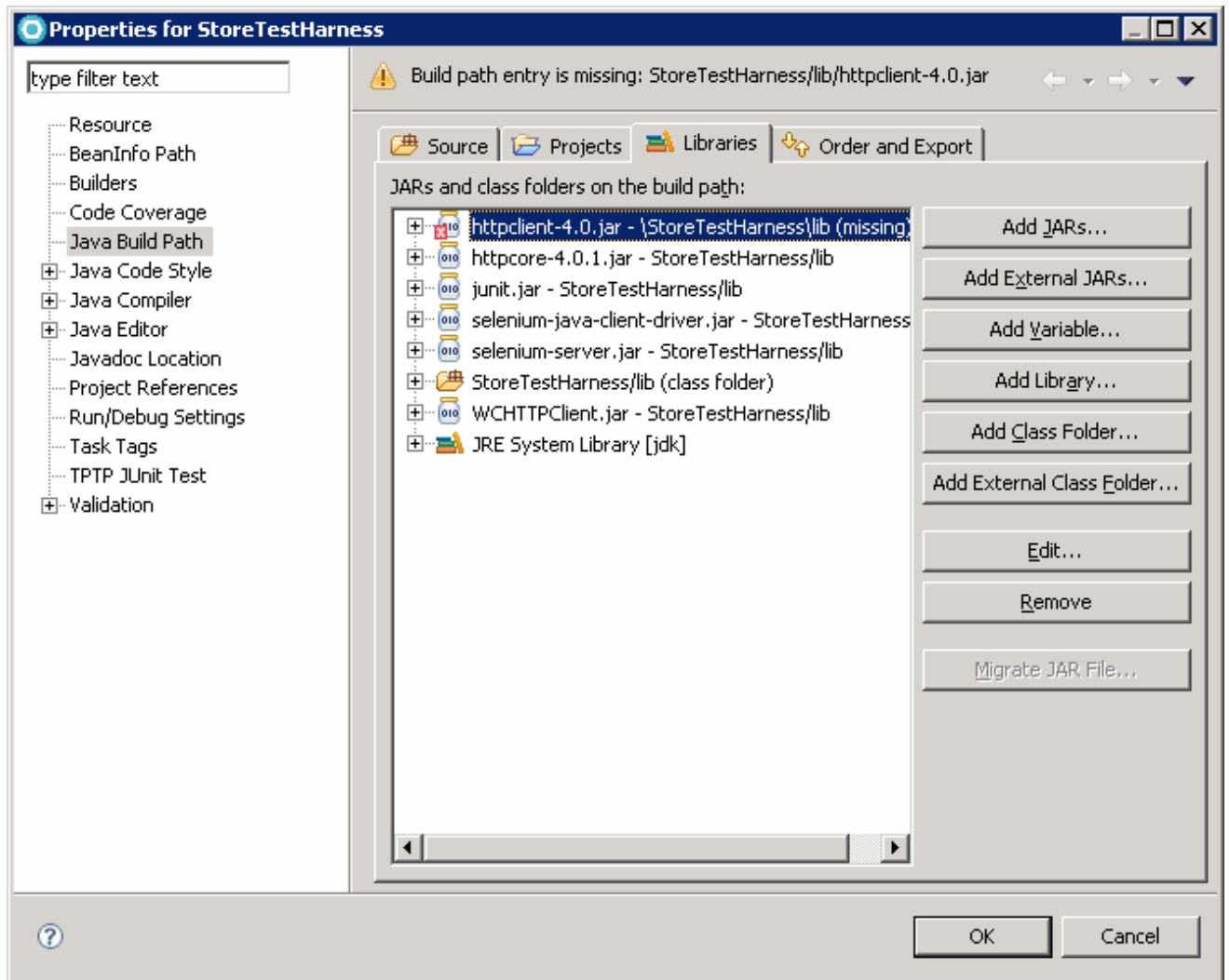


11. You can fix the build errors by doing the following steps.

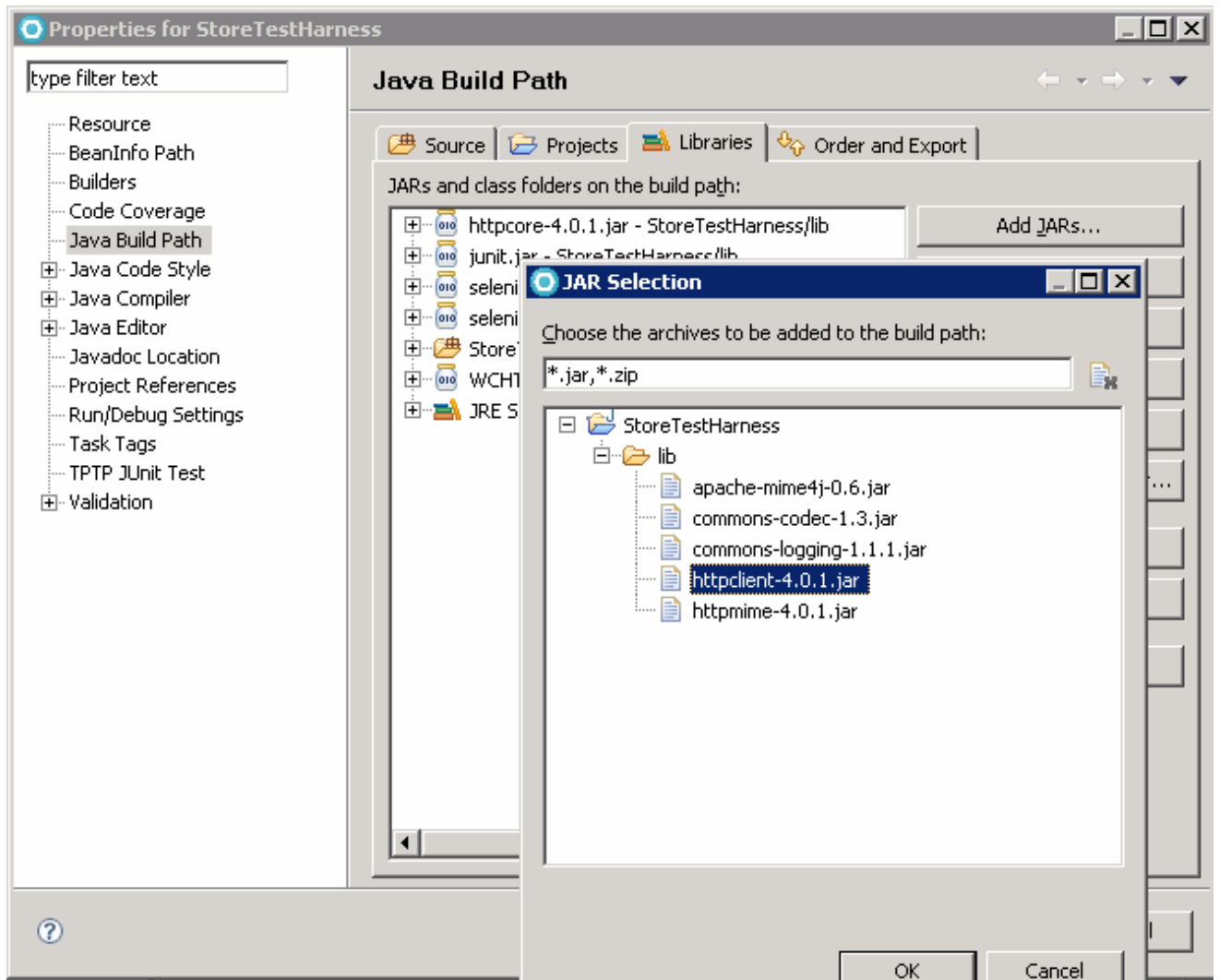
- a. Right click project **StoreTestHarness**, select **Build Path > Configure Build Path**.



- b. Under the Libraries tab, remove the missing library **httpclient-4.0.jar**. You can do this by selecting **httpclient-4.0.jar** and clicking **Remove**



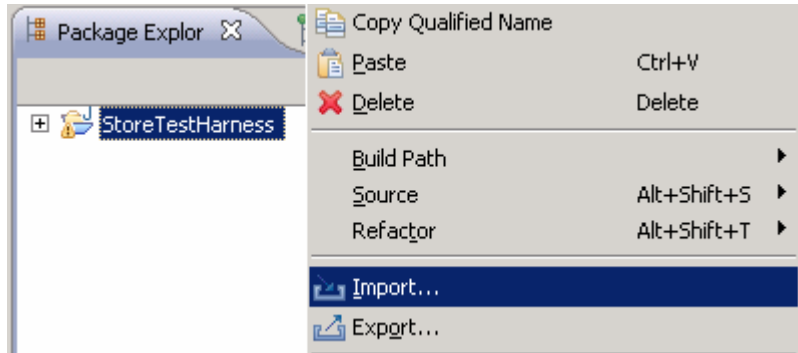
- c. In the same panel, click **Add JARs**. In the **JARs selection** window, expand **StoreTestHarness** > **lib**. Select **httpClient-4.0.1.jar**. Select **OK**. Select **OK** again in **Java Build Path** window. At this time, the errors in the **Problems** view should be gone



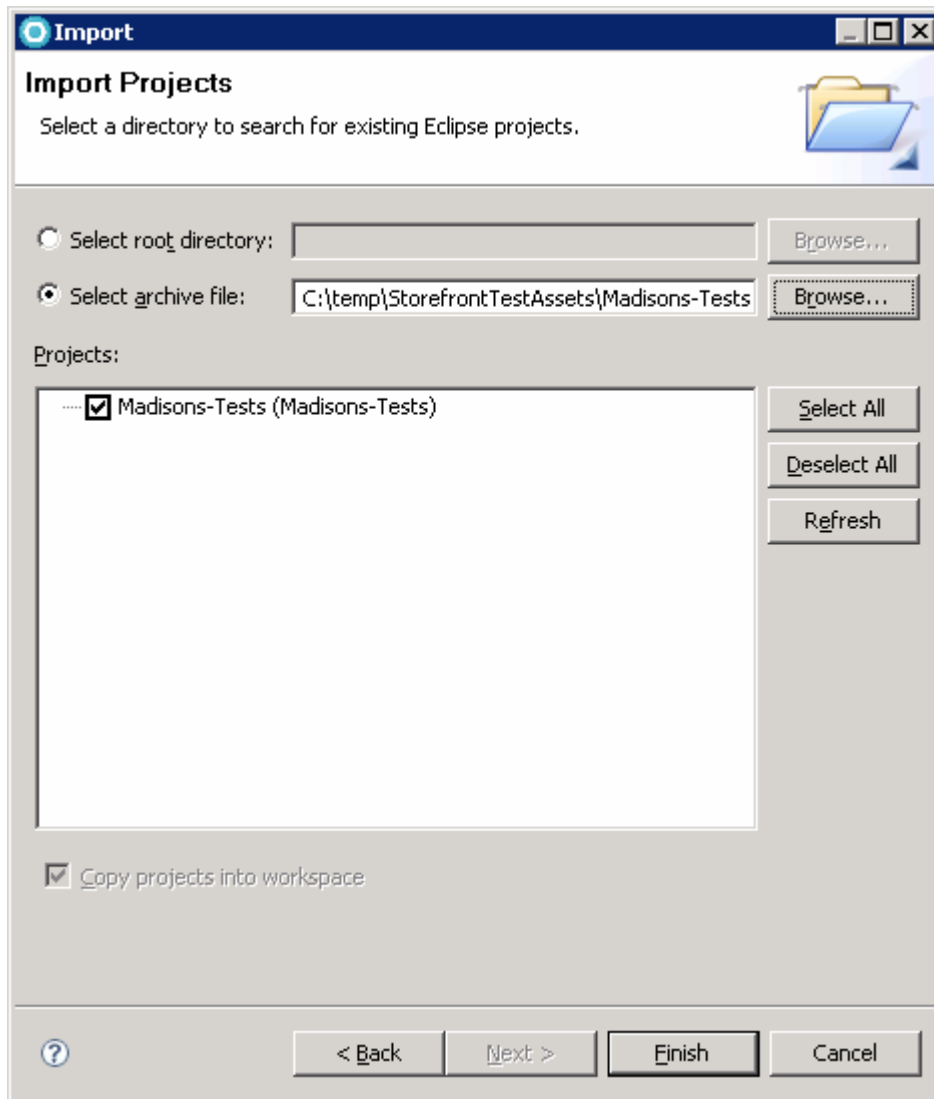
Part 3: Importing Madisons starter store test bucket project

In this section, you will need to import the Madisons starter store test bucket project into the Rational Application Developer.

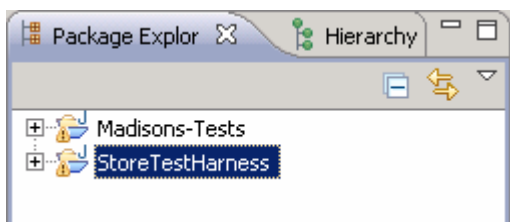
1. Right click in the Package Explorer view, select **Import**.



2. Select General > Existing Projects into Workspace. Click **Next**.
3. Select the **Select archive file** option and click **Browse**.
4. Navigate to the directory <TEMP>/StorefrontTestAssets, select the **Madisons-Tests.zip** file. Click **Finish**.



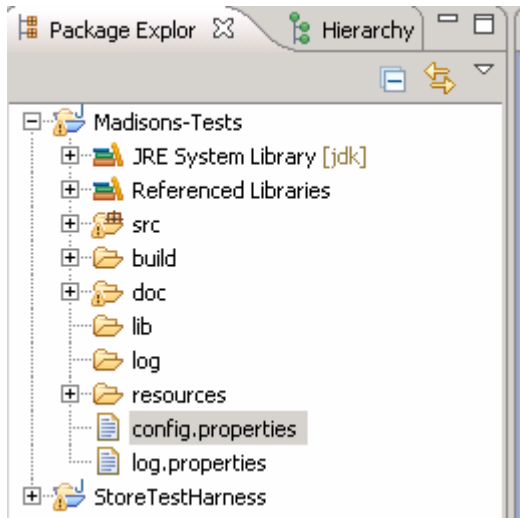
___ 5. You should be able to see Madisons-Tests project is imported.



Part 4: Update config.properties file

In this section, you will need to update several parameter values in the config.properties file.

1. In the Package Explorer view, expand the Madisons-Test project. Open the **config.properties** file by double clicking.



2. The **config.properties** file defines several environment variables. For this lab, you will use the Firefox browser to test the Madisons starter store, make sure **BROWSER** value is *firefox.
3. The **HOSTNAME** is where your WebSphere Commerce server is running from. For this lab, you will use WebSphere Commerce Developer test environment as WebSphere Commerce server, and the test harness and **WebSphere Commerce Developer** are on the same machine. Make sure **HOSTNAME** value is **localhost**.

```
#The host name of the server you are testing
HOSTNAME=localhost

#Possible values:
#IE: *iexplore/*iehta
#Firefox: *chrome/*firefox
#Safari: *safari
BROWSER=*firefox
```

4. The **DEFAULTTIMEOUT** is the default amount of time in seconds to wait for an element in test case methods. **TESTTIMEOUT** is the number of seconds to wait before timing out a test case. You should tune these two values according to your test environment. For this lab, change **DEFAULTTIMEOUT** value to be 600. Change **TESTTIMEOUT** to 3000.

```
#The default amount of time in seconds to wait for an element
DEFAULTTIMEOUT=600

#The number of seconds to wait before timing out a test case
TESTTIMEOUT=3000
```

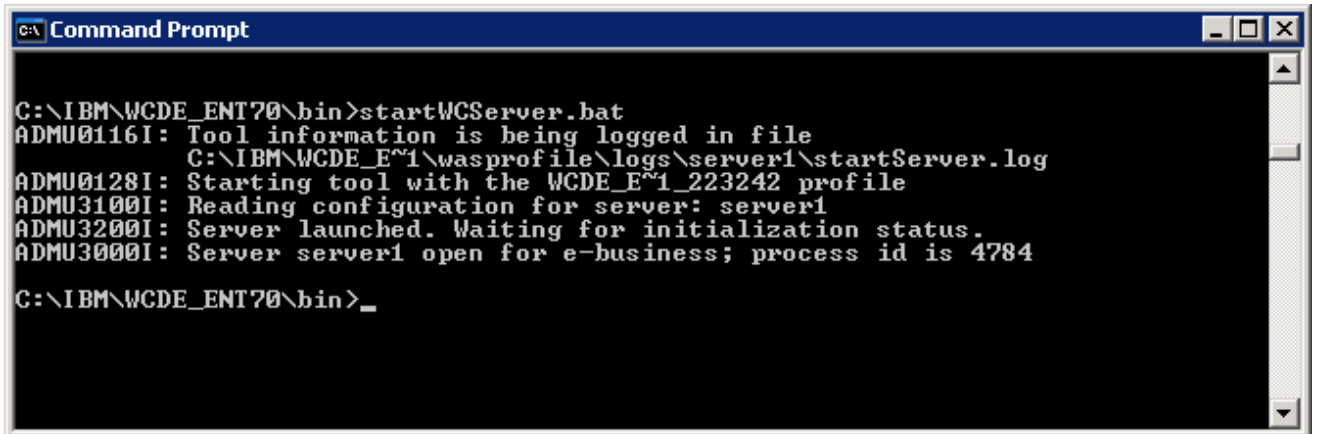
- ____ 5. Find ADMIN_USER_NAME and ADMIN_PASSWORD lines. They are the site administrator's user ID and password. Update the values if your system uses different site administrator ID and password.

```
#Site administrator user name  
ADMIN_USER_NAME=wcsadmin  
  
#Site administrator password  
ADMIN_PASSWORD=wcs1admin
```

Part 5: Run a test scenario

In this section, you will run a test scenario associated with user registration.

- ___ 1. Start the WebSphere Commerce Developer test server
 - ___ a. Open a Windows Command Prompt window,
 - ___ b. Navigate to the directory < WCDE_installdir>/bin. Run startWCServer.bat to start the test server.

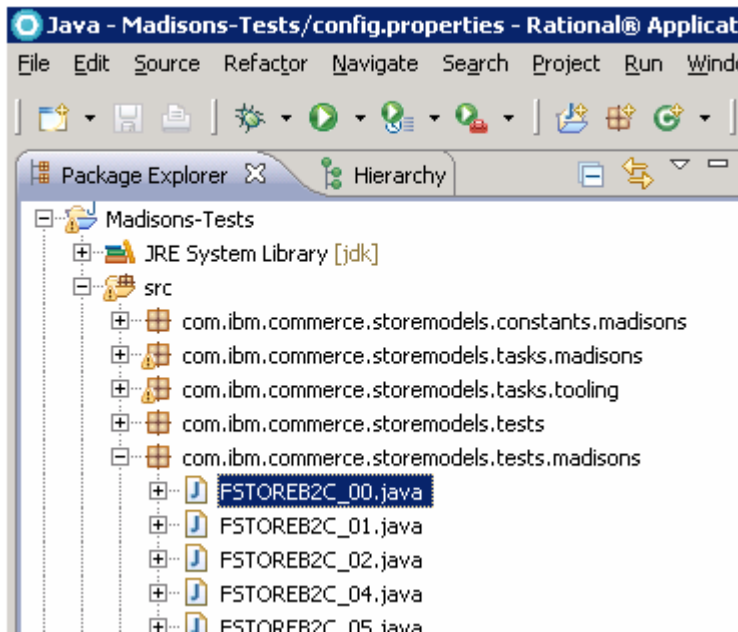


```

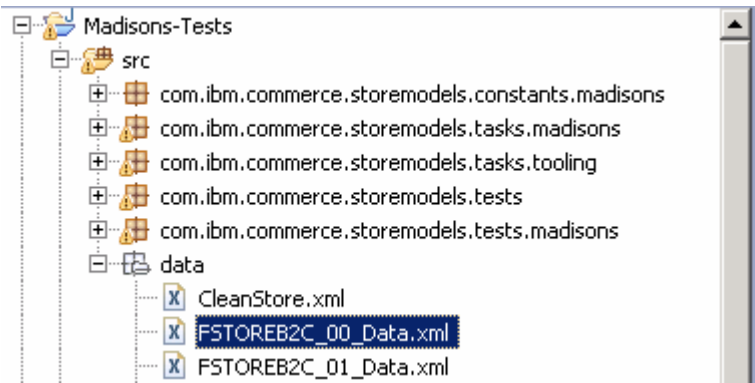
C:\IBM\WCDE_ENT70\bin>startWCServer.bat
ADMU0116I: Tool information is being logged in file
           C:\IBM\WCDE_E~1\wasprofile\logs\server1\startServer.log
ADMU0128I: Starting tool with the WCDE_E~1_223242 profile
ADMU3100I: Reading configuration for server: server1
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server server1 open for e-business; process id is 4784

C:\IBM\WCDE_ENT70\bin>_
  
```

- ___ 2. Go to the Madisons-Tests project **Package Explorer** view, and expand Madisons-Tests until you see the test script **FSTOREB2C_00.java**. This test script creates reusable data, such as a shopper profile. You should run this script before running others



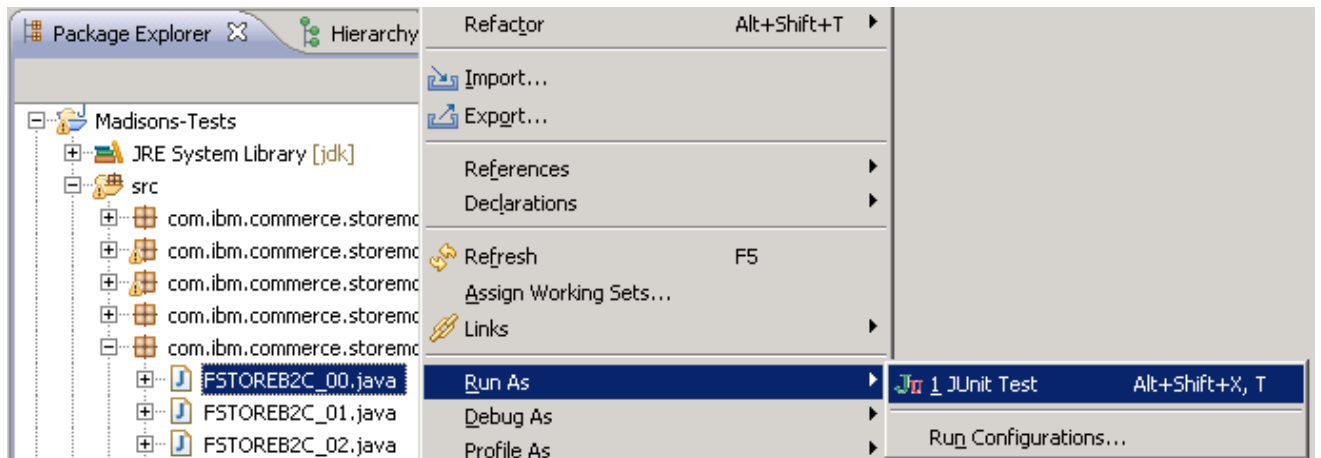
- ___ 3. Before you run the test script **FSTOREB2C_00.java**, open the data file **FSTOREB2C_00_data.xml**



Find the data block “setupAdmin” in the file. This data block contains the site administrator ID and password. You need to update the values for LOGONID and PASSWORD if your site administrator ID and password are different from the default ones.

```
<Test name="setupAdmin">
  <Datablock name="setupAdmin">
    <Input>
      <Parameter name="LOGONID" value="wcs admin"/>
      <Parameter name="PASSWORD" value="wcs1admin"/>
    </Input>
  </Datablock>
</Test>
```

- ___ 4. Right click **FSTOREB2C_00.java**, select **Run As > JUnit Test**



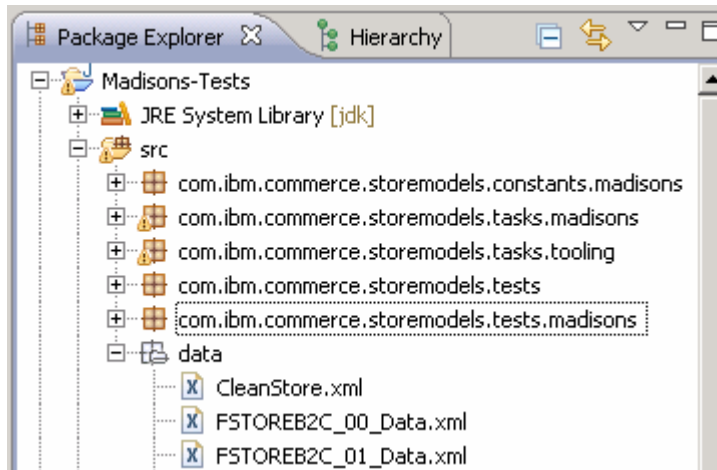
- ___ 5. Open **FSTOREB2C_01.java** and read the code. You can see that this test scenario contains several test cases related with user registration. It uses the data file **data/FSTOREB2C_01_Data.xml** as input data.

```
public class FSTOREB2C_01 extends StoreModelTestCase {

    /**
     * The internal copyright field.
     */
    public static final String COPYRIGHT = com.ibm.commerce.storemodels.utils.IBMCopyright.SHORT_

    //A variable to hold the name of the data file where input parameters can be found.
    //$ANALYSIS-IGNORE
    protected final String dataFileName = "data/FSTOREB2C_01_Data.xml";
```

- ___ 6. Open the data file **FSTOREB2C_01_Data.xml** as shown in the screen capture below.



- ___ 7. Read this data file and look at the first data block shown below. The test case **testFSTOREB2C_0101** in test script **FSTOREB2C_01.java** will use the ID **shawn** and password **Shopper1101** to register a user in the Madisons store.

```
<Scenario name="FSTOREB2C_01">
  <Env>
    <Parameter name="ACCELERATOR_LOGON_ID" value="Seller"/>
    <Parameter name="ACCELERATOR_PASSWORD" value="Seller01"/>
  </Env>

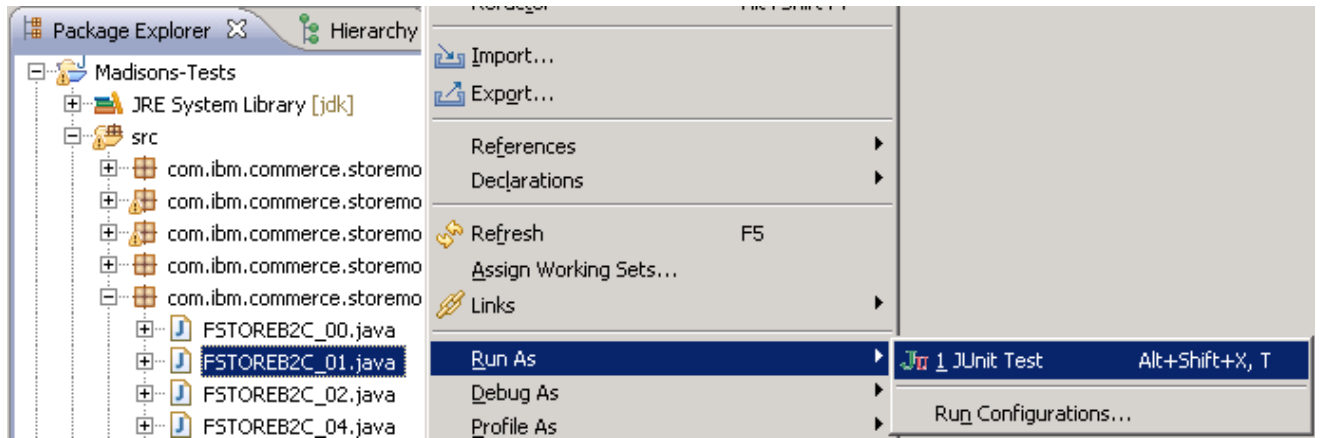
  <Test name="testFSTOREB2C_0101">
    <Datablock name="testFSTOREB2C_0101">
      <Input>
        <Parameter name="LOGONID" value="shawn"/>
        <Parameter name="FIRST_NAME" value="John"/>
        <Parameter name="LAST_NAME" value="Clark"/>
        <Parameter name="PASSWORD" value="Shopper1101"/>
        <Parameter name="PASSWORD_VERIFY" value="Shopper1101"/>
      </Input>
    </Datablock>
  </Test>
</Scenario>
```

Note: If you look at following code in the method **testFSTOREB2C_0101()** in the Java class **FSTOREB2C_01.java**:

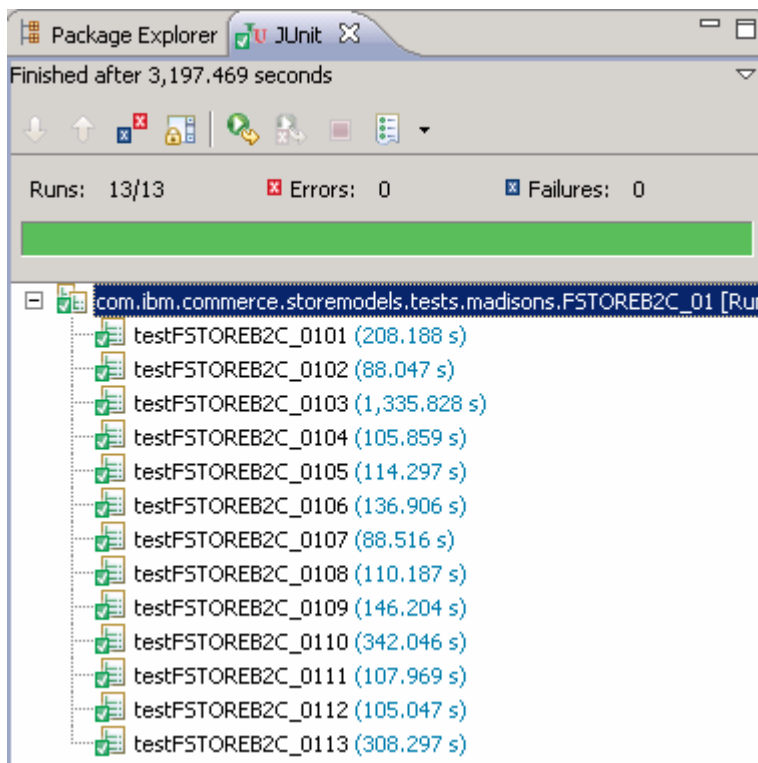
```
//click on the Register button and fill up the registration form with specified data
RegistrationPage.registerNewUser(getInputParameter("LOGONID")+test.getUniqueID(),
```

You can see that the logon ID used to register the user is not **"shawn"**. In fact it is **"shawn"** + a unique id. By doing this, you can run the test case multiple times without need cleaning up store's database.

- ___ 8. To run the test scenario, right click **FSTOREB2C_01.java**, select **Run As > JUnit Test**. A JUnit view will be opened and you can see the running progress from there.



- ___ 9. You should be able to see that the Firefox browser is launched automatically and all the test cases methods in the **FSTOREB2C_01.java** test script are ran.
- ___ 10. After the testing is done, go to the Rational Application Developer, you should be able to see the JUnit report of the testing. If you see any failure of a test case, you can click that test case and read error message in the Failure Trace view



Part 6: Verifying the users are registered

In this section you will use the Accelerator to view the users registered by running the test script.

1. Log on to the Accelerator, select **Madisons** from Store name field.

Select Store and Language

Store name: **Madisons** (dropdown menu showing: Madisons, MadisonsESite, MadisonsReseller, MadisonsResellerStorefrontAssetStore, MadisonsStorefrontAssetStore)

Find Store: []

Fulfillment center: -- Please specify -- (dropdown menu)

Language to work in: United States English (dropdown menu)

Buttons: OK, Cancel, Help

2. Select **Operations > Find Customers** from the menu bar.

Menu bar: Store, Marketing, Merchandise, Auctions, **Operations**, Payments, Help

Operations sub-menu: Create New Customer, **Find Customers**

3. In the **First name** field, type in **John**. Click **Find**.

Find Customers

To search for a customer, provide information below and click **Find**.

Customer logon ID: [] Match case, beginning with []

First name: John Match case, beginning with []

4. You can see the registered IDs with the first name John. Note that John register IDs has a unique ID appended.

Find Customers - Search Results

Page Number

Number of items: **10**

[« First](#) | [1 of 1](#) | [Last »](#)

<input type="checkbox"/>	Customer Logon ID ▲	First Name ▲	Last Name ▲	Phone Number ▲	City ▲	Zip/Postal Code ▲
<input type="checkbox"/>	Test1275450895140	John	Clark	905-555-2000	Markham	L3G 1H2
<input type="checkbox"/>	Test1275450985281	John	Clark	905-555-2000	Markham	L3G 1H2
<input type="checkbox"/>	Test1275451183359	John	Clark	905-555-2000	Markham	L3G 1H2
<input type="checkbox"/>	Test1275451285484	John	Clark	905-555-2000	Markham	L3G 1H2
<input type="checkbox"/>	Test1275451545796	John	Clark	905-555-2000	Markham	L3G 1H2
<input type="checkbox"/>	Test1275451884968	John	Clark	905-555-2000	Markham	L3G 1H2
<input type="checkbox"/>	Test1275451990093	John	Clark	905-555-2000	Markham	L3G 1H2
<input type="checkbox"/>	Test1275452099171	John	Clark	905-555-2000	Markham	L3G 1H2
<input type="checkbox"/>	TestUser	John	Clark	905-555-2000	Toronto	L3G 1H2
<input type="checkbox"/>	shawn1275449333890	John	Clark	905-555-2000	Markham	L3G 1H2

This page is left intentionally blank.