Hands-On Lab

Authoring and Running Automated GUI Tests using Microsoft Test Manager 2012 and froglogic Squish

Lab version: 1.0.5

Last updated: 27/03/2013

Overview

This hands-on lab is part two out of a series of two labs showing how to use Microsoft Visual Studio 2012 and Microsoft Test Manager 2012 to run an automated Squish GUI tests.

Squish is a cross-platform, cross-device GUI test automation tool allowing to automate GUIs on a variety of platforms based on several different GUI technologies. The Squish for MS ALM plugin allows to centrally manage Squish GUI tests in TFS and to run them from Visual Studio, via TFS scheduled builds and continuous integration and via Microsoft Test Manager.

In this lab, you will learn how to author and run an automated GUI test using the Squish IDE. You will run the same test from Microsoft Visual Studio 2012 and Microsoft Test Manager 2012 and review the test results. For the setup and configuration that has to be completed before, please refer to the first lab of this series: Setup Microsoft Test Manager 2012 and froglogic Squish for Automated GUI Testing.

This hands-on lab is one out of a number of labs that deal with Microsoft Test Manager 2012. It focuses on running automated tests with Squish GUI Tester. To learn more about Microsoft Test Manager 2012 features that are unrelated to Squish GUI Tester, please refer to this this blog post.

Prerequisites

In order to complete this lab you will need the Visual Studio 2012 virtual machine provided by Microsoft. For more information on acquiring and using this virtual machine, please see this blog post.
Besides, you need to complete the exercises in the previous lab, Setup Microsoft Test Manager 2012 and froglogic Squish for Automated GUI Testing, before being able to continue with this lab.

Exercises

This Hands-On Lab comprises the following exercises:

1. Authoring a Manual Test
2. Authoring and Running an Automated Test using Squish GUI Tester
3. Running a Squish GUI Test from Microsoft Visual Studio 2012
4. Running a Squish GUI Test as Part of a Microsoft Test Manager 2012 Test Plan

---

Estimated time to complete this lab: **60 minutes**.

Exercise 1: Authoring a Manual Test

In this exercise, you will learn how to create a manual test plan and populate it with steps. The plan can later be run to confirm the expected behavior of your software.

1. Start the virtual machine and log in as **Julia**. All user passwords are **`P2ssw0rd`**.
2. Open Microsoft Test Manager from **Start** | **All Programs** | **Microsoft Visual Studio 2012** | Microsoft Test Manager.
3. Switch to **Testing Center** by clicking on **Lab Center** and selecting **Testing Center.**
4. If you are not in the test plan view already, click on **Plan** from the main menu. Microsoft Test Manager will connect to the most recently loaded test plan, which in this case is the test plan **Default** that we used in the previous lab. To be able to create a new test plan for the manual test, click on the test plan name **Default** in the upper right-corner.

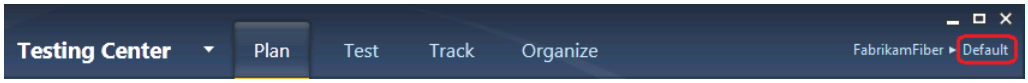

Figure 1
Currently opened test plan

5. Click on the **Add** button, enter the name **AddressbookTestPlan** into the field **Plan name** and click on **Add** to create the plan. Finally, click on **Select plan** to open it in **Testing Center.**

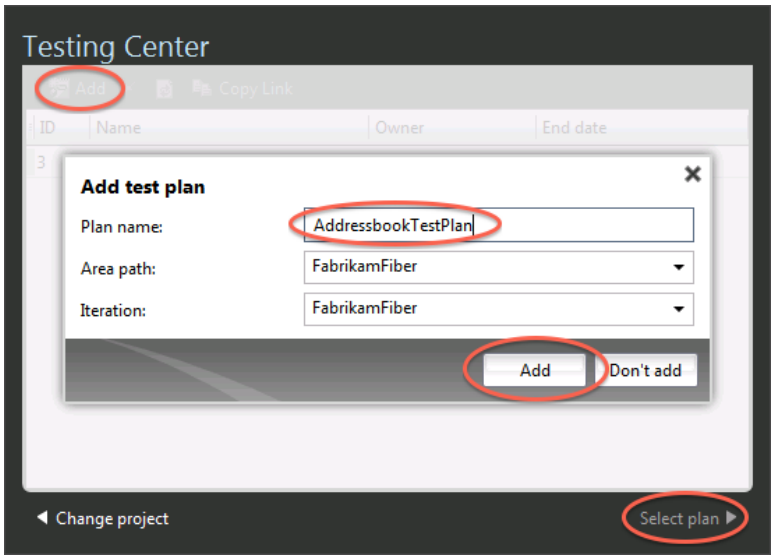

Figure 2
Creating a new test plan

6. In the left pane, select **AddressbookTestPlan** and click on the **New** button in the right pane.

Figure 3

Creating a new manual test

7. This creates a new Microsoft Test Manager test case and displays the configuration view for it. In the **Title** box, enter **AddressbookTest**.

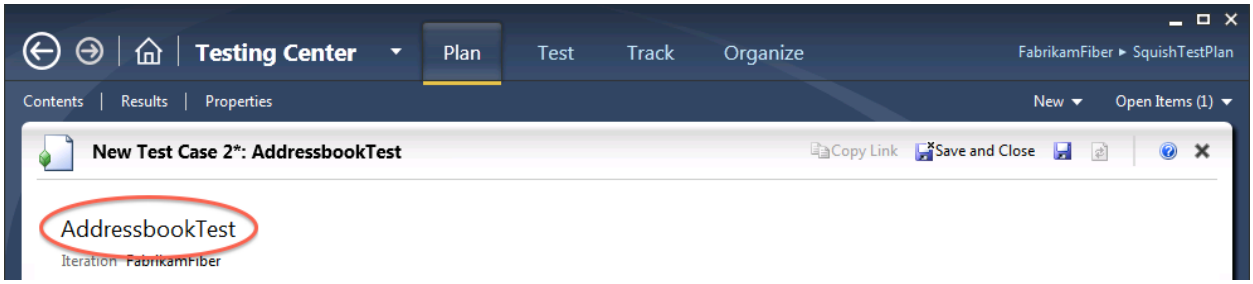

Figure 4

Entering the name of the test

8. At this point, we're ready to add steps to this manual test. Each step includes an **Action**, which describes the action the tester needs to perform. Optionally, a step can include an **Expected Result**, which describes the expected result of the given action.
9. In the **Steps** panel, create a step for each of the following **Actions**, only one of which has an Expected Result. The steps are shown in the table below.

Action	Expected Result
Start Address Book application	
Create a new address book	
Add an address book entry	
Verify data entry	Verify that the address book entry shows up in the list of addresses

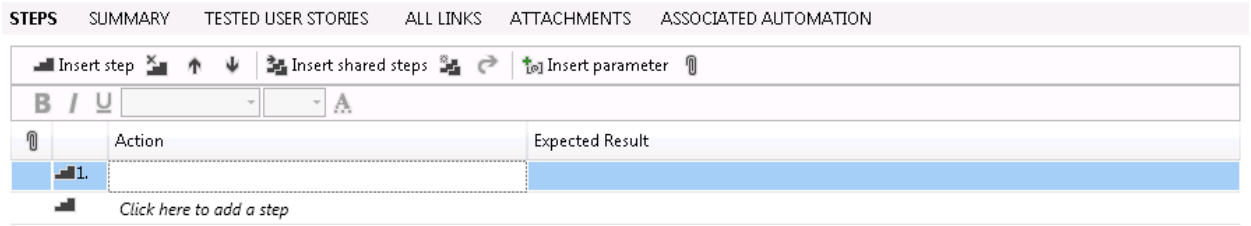

Figure 5

Insert the steps of the manual test

10. Save test case by clicking on the **Save** icon in the upper-right corner.

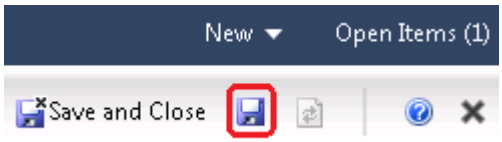

Figure 6

Location of the save button

Note: The test case is saved as a project work item.

11. Now you are ready to run the test manually as it is described in the Lab Authoring and Running Manual Tests using Microsoft Test Manager 2012.

    Once the manual test has passed on a regular basis, it makes sense to automate this test allowing us to e.g. re-execute this test as part of our nightly regression test plan. Therefor, in this Lab, we want to execute the same test as an automated test. Thus we are skipping the manual execution. As the next step, we are going to prepare the automated test execution.

Exercise 2: Authoring and Running an Automated Test using Squish GUI Tester

In this exercise, you will learn how to use the Squish GUI Tester to author and run an automated GUI test. The example test that you are going to execute automates an address book application that has been developed in the programming language Java. The application under test, the Java Runtime Environment and the Squish test suite were already installed as part of the Squish installation in the previous lab.

This exercise focuses on running the test cases that are contained in the example test suite. If you are interested in learning more about recording and authoring tests with Squish GUI Tester, please refer to the online manual at http://doc.froglogic.com/squish/latest/tutorial-getting-started-java.html.

1. Start the Squish IDE by the clicking on the **Squish for Java** icon on the desktop. The example test for this Lab is called **suite_js** and it will already be opened in the **Test Suites** view. The test suite contains three test cases, which you can open by double-clicking. This will open a script editor with the JavaScript source code of the test. Please note that Squish also supports many more scripting languages when authoring tests.

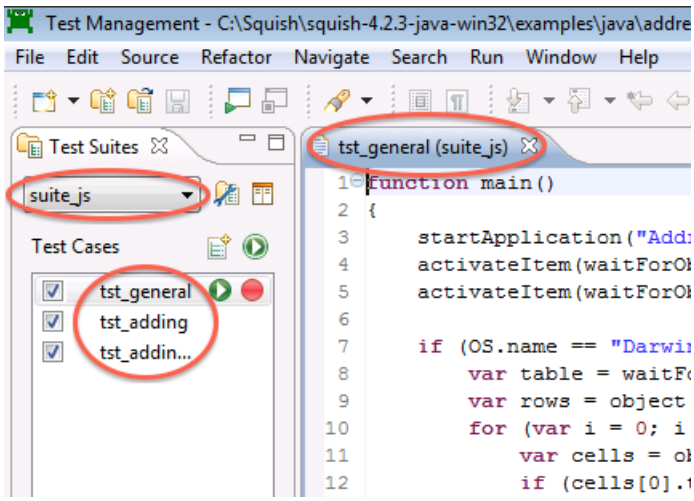

Figure 7
Squish IDE and location of the test suite and test cases of the example test

2. As the next step, you are going to execute the complete test suite. This will automatically start the application under test and perform all the interaction that is defined in the three test cases.
3. To start execution of the test suite, click on the **Play** button in the **Test Suites** view. To execute a single test case instead of the complete test suite, you can click on the **Play** button right beside

the test cases in the **Test Cases** list.

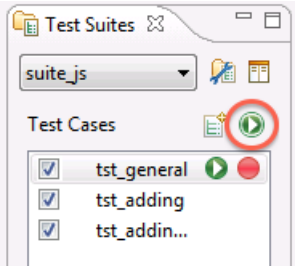

Figure 8
Location of the play button in the Test Suites view

4. The Squish IDE will be hidden and the **Control Bar** will show up while the test runs. Squish will automatically start the address book application and automate it.

5. Once the test run is finished, the Squish IDE will show up again. You can find the results of the test run in the **Test Results** view of the IDE.

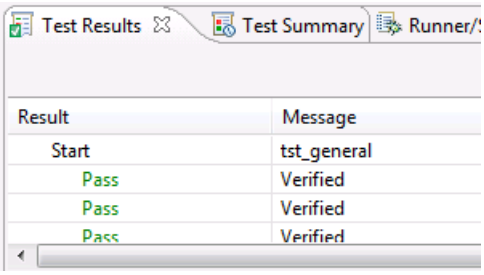

Figure 9
Test Results view displaying the results of the test run

Exercise 3: Running a Squish GUI Test from Microsoft Visual Studio 2012

After having created and executed the GUI tests in Squish, the next step would be to integrate this test into a Visual Studio project. This would, for example, allow developers to run the Squish tests directly from Visual Studio while working on code changes of the application. Or, this can be used to put the Squish tests into TFS version controls.

In this exercise, you will learn how to create a Microsoft Visual Studio 2012 project that contains the Squish GUI test that we already executed in the last exercise. You will execute the same test in Microsoft Visual Studio 2012 and without the Squish IDE.

1. Open **Microsoft Visual Studio** from **Start** | **All Programs** | **Microsoft Visual Studio 2012** | **Microsoft Visual Studio**.

2. First, we need to create a Visual Studio project that we will use to import the Squish GUI test to. Select menu **File | New | Project…** and open **Installed | Visual C# | Test | Unit Test Project**.

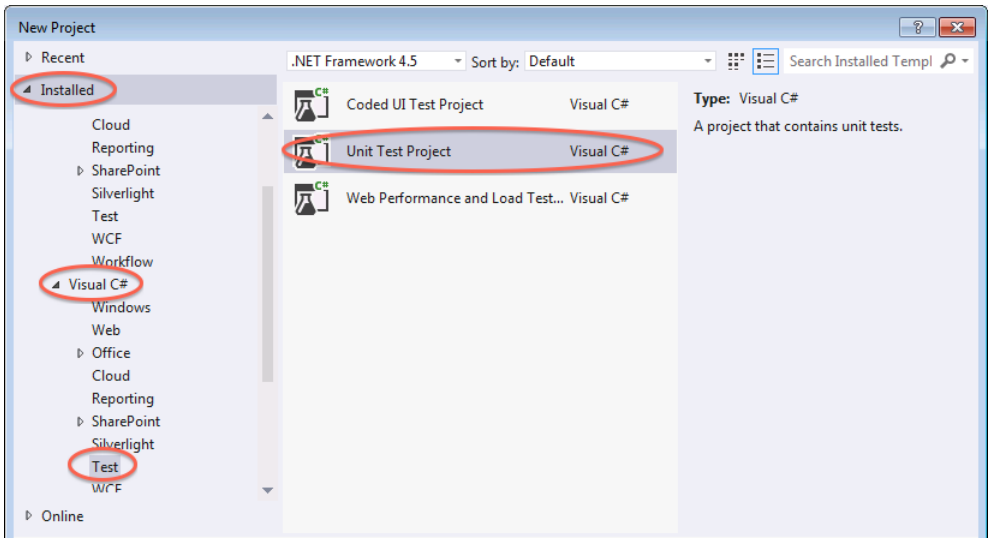

Figure 10

Create a new Unit Test Project

3. Enter the name **SquishTestProject** and click on **OK** to create the project.

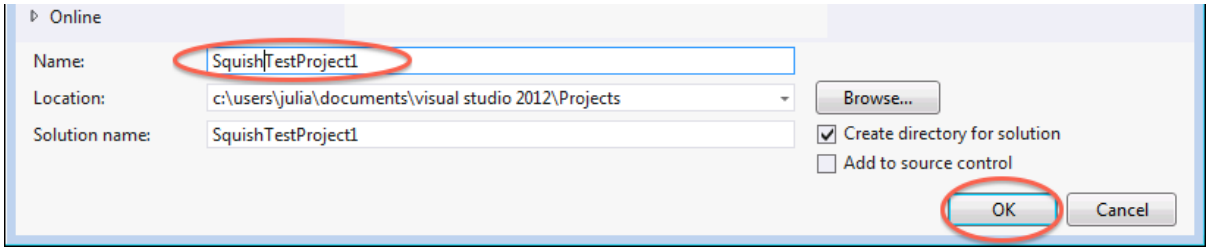

Figure 11

Entering the name and creating the project

4. Once the project has been created, a Unit Test file will show up. As we do not want to create a Unit Test, but run a Squish GUI test instead, we can safely delete the file. To do so, open **Solution Explorer** and select the file **UnitTest1.cs** in the solution. Right-click the file and select **Delete** from the context menu. You will be asked to prompt the deletion.

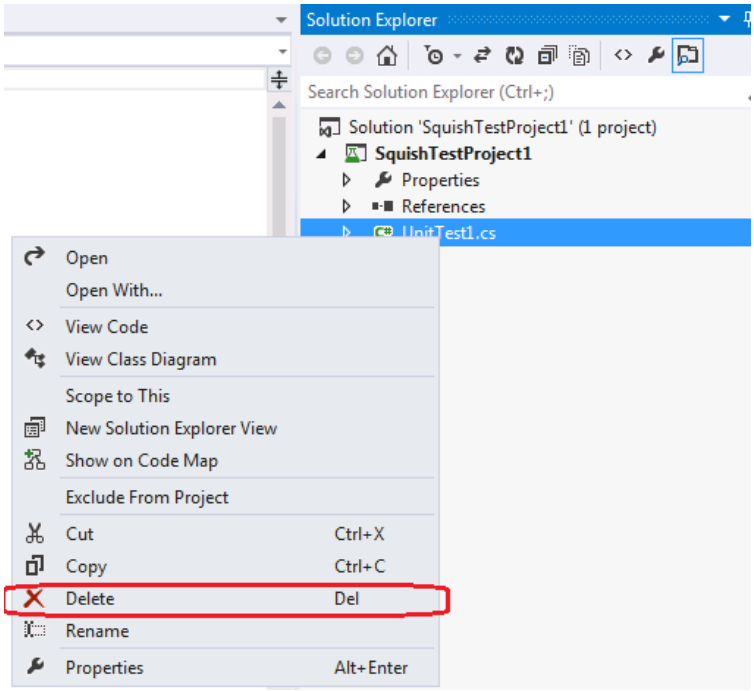

Figure 12

Deleting the Unit Test file

5. Next, we need to add a new item to our project that will contain the Squish GUI test. Right-click the **SquishTestProject** in Solution Explorer and select **Add | New Item…**.

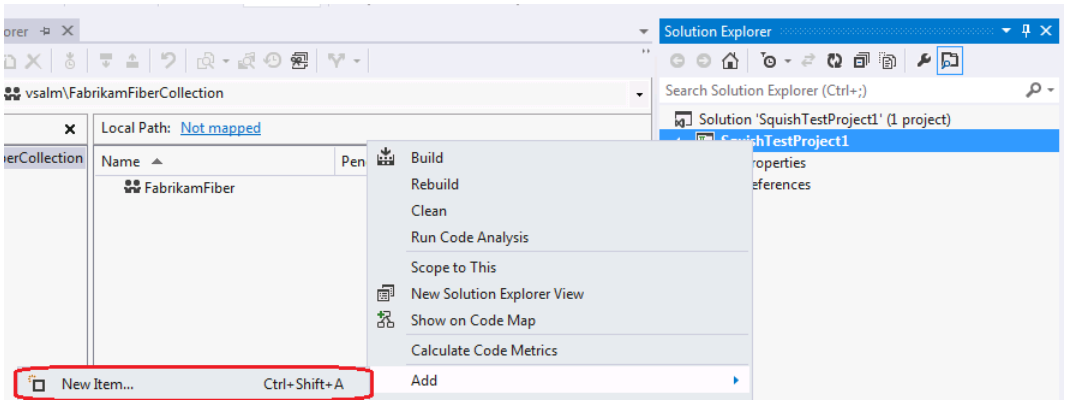

Figure 13

Adding a new item

6. Select **Installed | Visual C# Items | Squish Test**. Just leave the name its default of **SquishTest1.st** and click on **Add**.

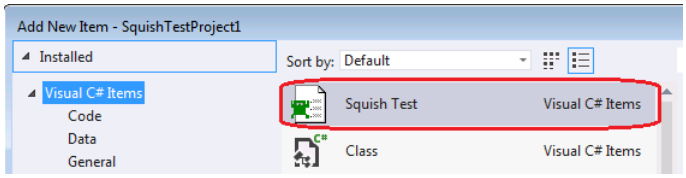

Figure 14

Squish Test type selection

7. The Squish Test will be created and opened as shown in the screenshot below.

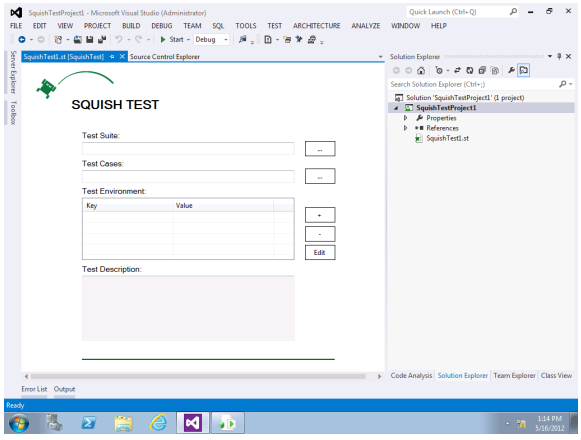

Figure 15

Squish Test in Visual Studio

8. So far we have not selected a Squish test suite. Click on the **Browse** button right beside the **Test Suite** field. In the file dialog that is opened, navigate to the folder **C:\Squish\squish-4.2.3-java-win32\examples\java\addressbook\suite_js** and click on **OK**.

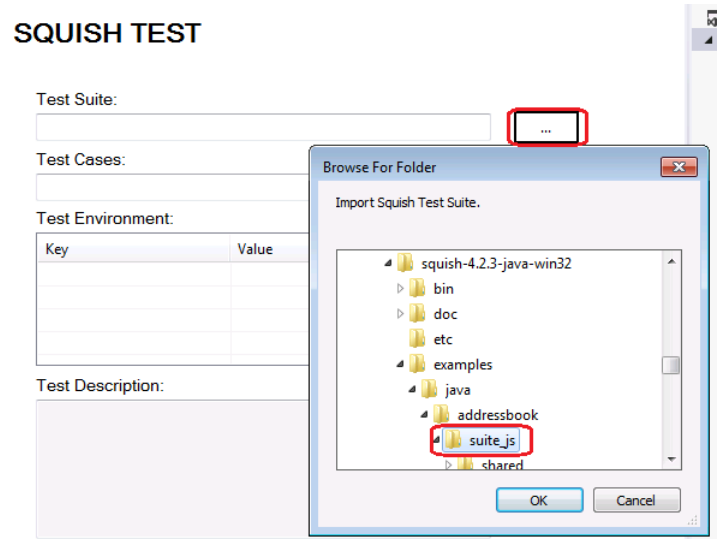

Figure 16

Select Squish test suite for import

9. Select **File | Save all** to save the Squish test.
10. Finally, to execute the test in Visual Studio, open **Test | Windows | Test Explorer**. It might take a short while unit the test **SquishTest1** shows up under **Not Run Tests**. Once it is visible, select **Run All** to start the test run.

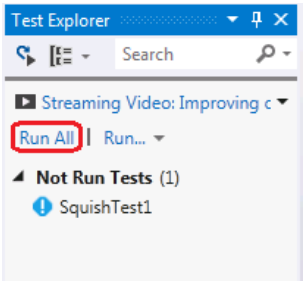

Figure 17

Start test run in Test Explorer

11. Visual Studio will run the test without the Squish IDE being involved. You will notice that the address book application is started and automated.

12. To open the result log of the test run, select **SquishTest1** in Test Explorer. At the bottom **of Test Explorer**, click on the **Output** link to open the **Test Output** view. Click on the attachment with the file extension **.trx** and the **Test Results** view will show up.

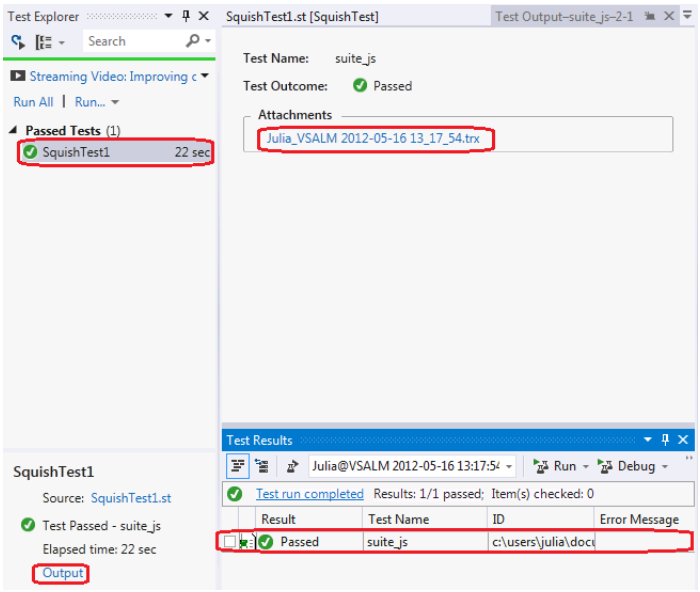

Figure 18

Opening the test result

13. To see more details of the result including the results of each verification, double-click on the result item in the **Test Result** list.

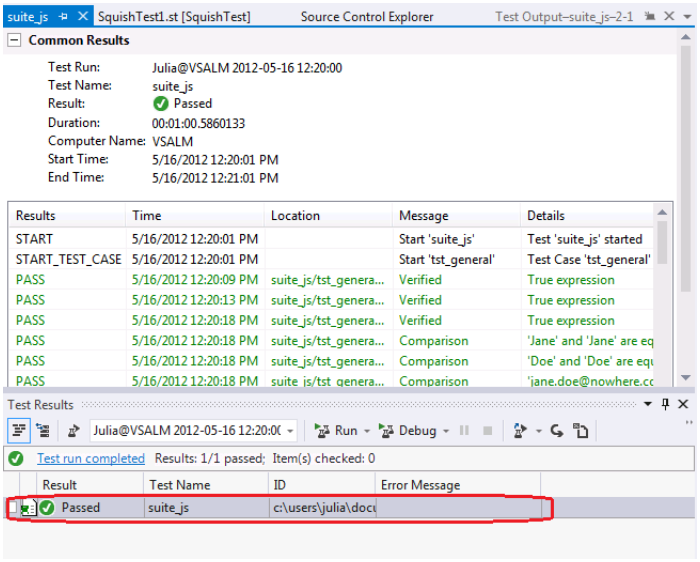

Figure 19

Test result details

Exercise 4: Running a Squish GUI Test as Part of Microsoft Test Manager 2012 Test Plan

A tester will primarily work in MTM to manage tests and run them. One main feature of the Squish for MS ALM integration is to allow associating Squish GUI tests with MTM test cases. This allows running a MTM test case which will execute the automated Squish GUI test on a given test agent and report back the results.

In this exercise, you will learn how to run a Squish GUI test as part of a test plan in Microsoft Test Manager 2012. You will use the test plan that you created in exercise 1 and the Visual Studio project that you create in exercise 3.

1. As the first step, you need to add the solution from the previous exercise to Team Foundation Server source control. Right-click the solution in **Solution Explorer** and select **Add solution to Source Control**.

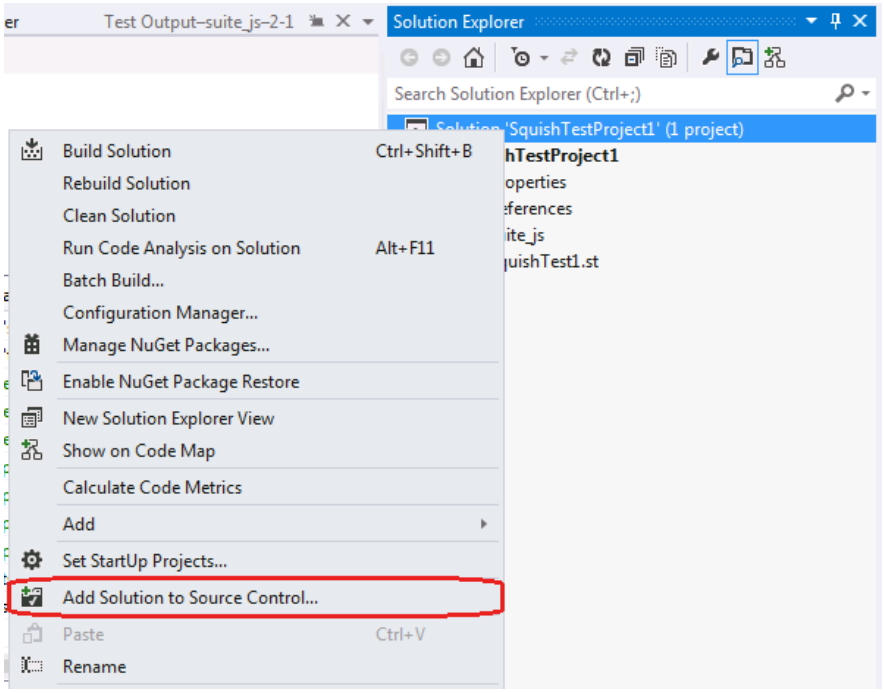

Figure 20

Adding the solution to source control

2. Now select the **FabricamFiber** team project and click on **OK**.

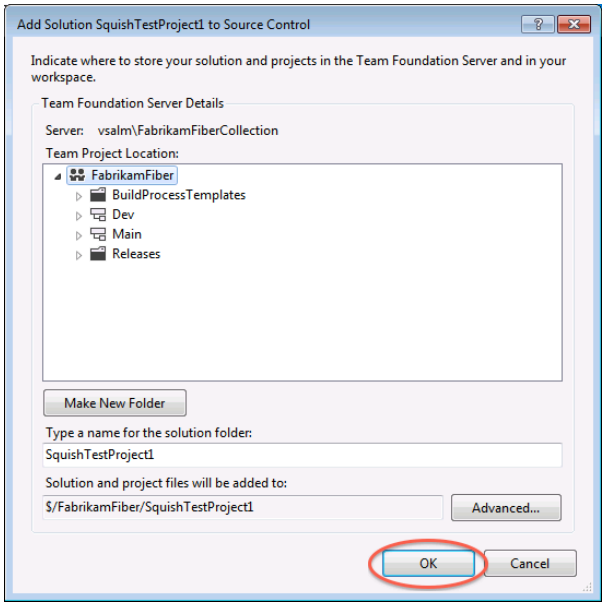

Figure 21

Selecting the team project

3. In **Solution Explorer**, right-click **Solution 'SquishTestProject'** and select **Check In…**. **Team Explorer** will display a dialog for the check-in. Click on **Check-In** to confirm.

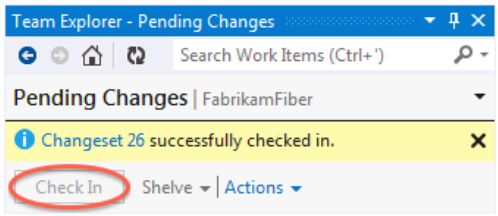

Figure 22

Confirming the source control check-in

4. Now you need to start a Team Foundation Server build to make the test available through Microsoft Test Manager. In **Team Explorer**, click on the **Home** button and select **Builds**.

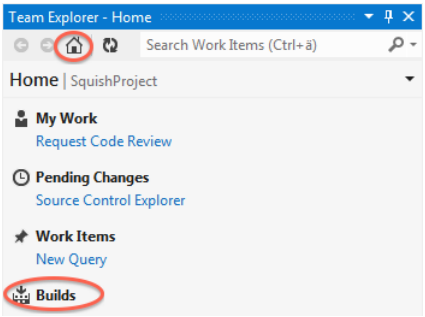

Figure 23

Opening the builds section in Team Explorer

5. Select **New Build Definition** to create a definition for the test.

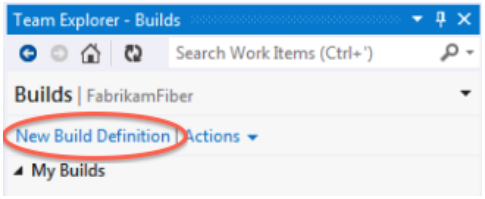

Figure 24

Create a new build definition

6. Visual Studio will open the configuration view for the build definition. In the **Build Defaults** section you need to specify a folder where the Team Foundation Server will save the build output to. Enable the checkbox in front of **Copy build output to the following drop folder** and enter **\\VSALM\ffdrops** into the text field. Finally, select **File | Save SquishTestProject**.

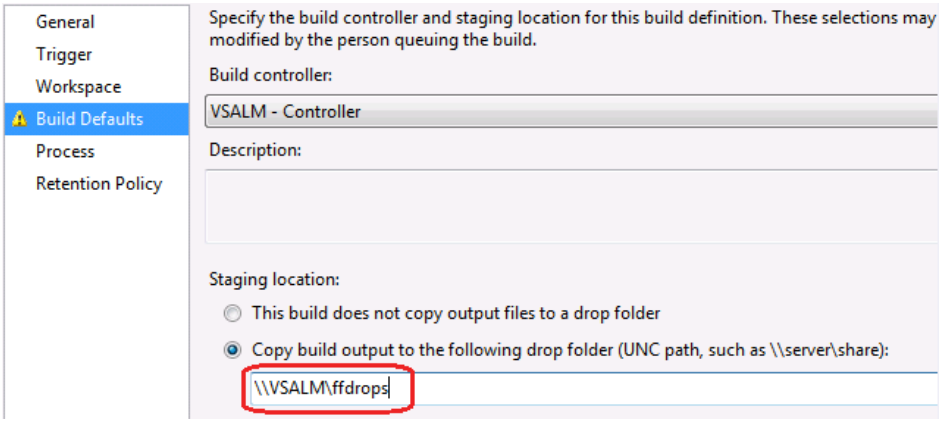

Figure 25

Specify drop folder for build output

7. The next step is to queue a build to let Team Foundation Server produce the build artifacts that are required by Microsoft Test Manager. Right-click the **SquishTestProject** build definition in **Team Explorer** and select **Queue New Build.** When asked to confirm, click on **Queue**.

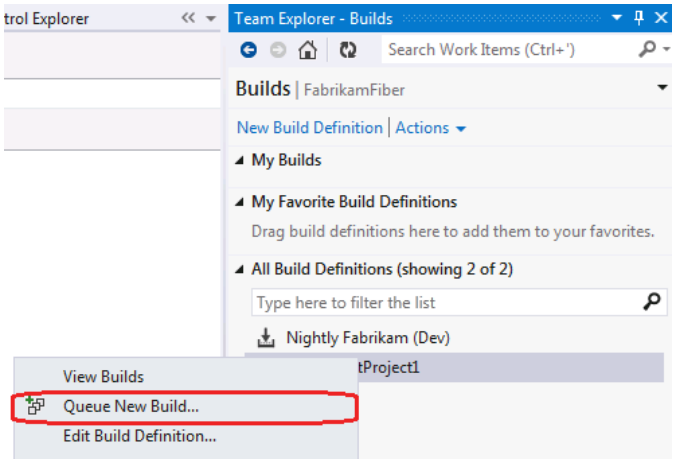

Figure 26

Queue a build

8. The build will take a while to finish. Once it is ready, it will show up in **Team Explorer**.

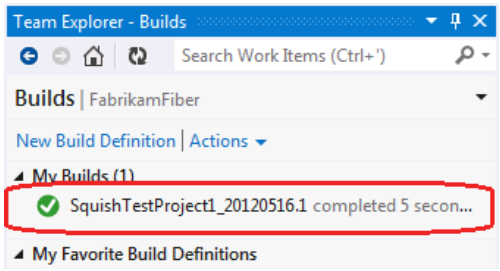

Figure 27

Finished build

9. Now you need to associate the Squish GUI test with test work item. To lookup the work item, enter **AddressbookTest** into the search field of **Team Explorer** and press the return key. Once the **AddressbookTest** item shows up in the **Search Results** list, double-click on it.

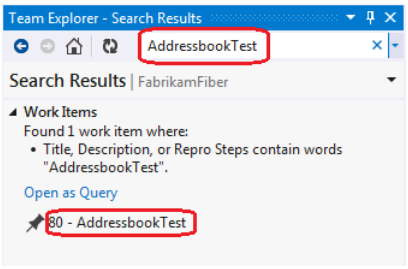

Figure 28

Searching and open the test work item

10. The **AddressbookTest** will be opened. Select the **Associated Automation** tab and click on the **Browse** button.

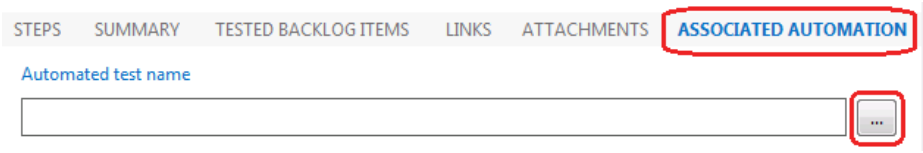

Figure 29

Open the dialog to browse for the Squish GUI test

11. In the following dialog, select the Squish GUI test **suite_js** and click on **OK**.

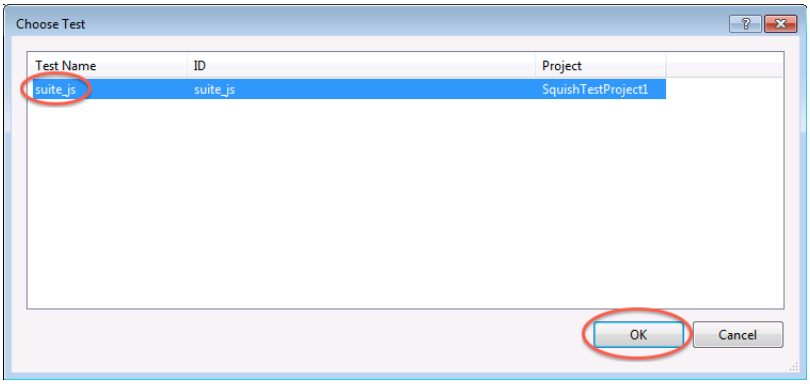

Figure 30

Select the Squish GUI test to associate it with the Microsoft Test Manager test

12. Finally, save the work item by clicking on the **Save Work Item** button.

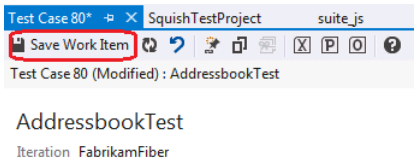

Figure 31
Save Work Item button

13. Now it is time to execute the test through Microsoft Test Manager. If Microsoft Test Manager is not running any more, open it by selecting **Start** | **All Programs** | **Microsoft Visual Studio 2012** | **Microsoft Test Manager.** Open the **SquishTestPlan** that we have created in exercise 1.
14. Open **Testing Center | Test** and select the **SquishTestPlan** in the left pane. In the right pane, right-click **AddressbookTest** and select **Run with options**.

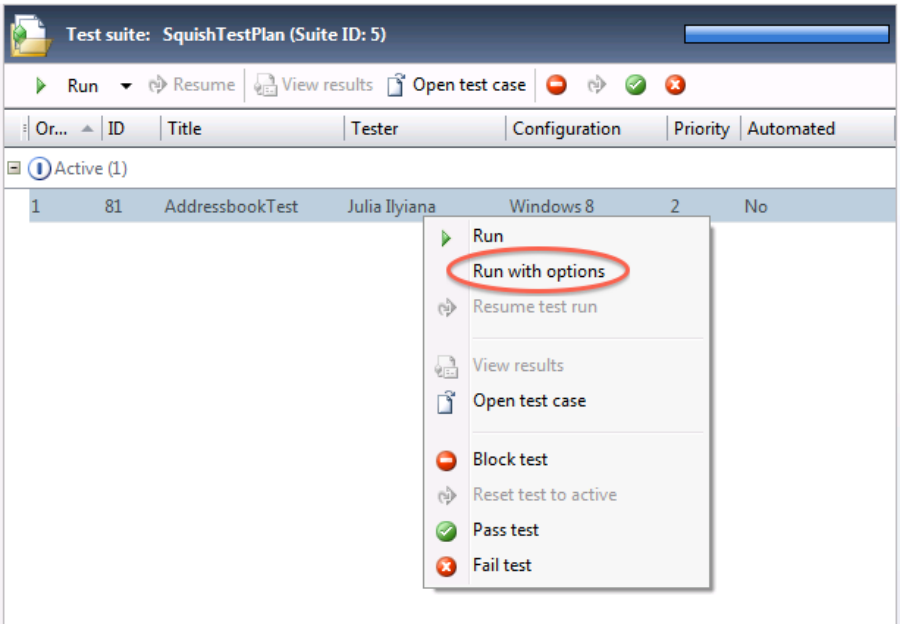

Figure 32
Location of the Run with option menu item

15. In the **Run Options** dialog, select the latest build in the **Build in use** combo box. Also make sure that **SquishEnvironment** is selected. Finally, click on **Run** to start the test run. The address book application will be launched and the Squish GUI test will be executed.

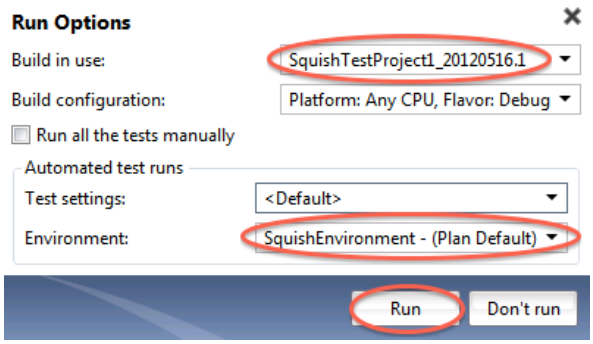

Figure 33

Run Options dialog

16. Wait for the test to be finished. To see the test result, click on the **Refresh** button. In the **Results Overview** section under **Current State of Tests**, one passed test should be shown.

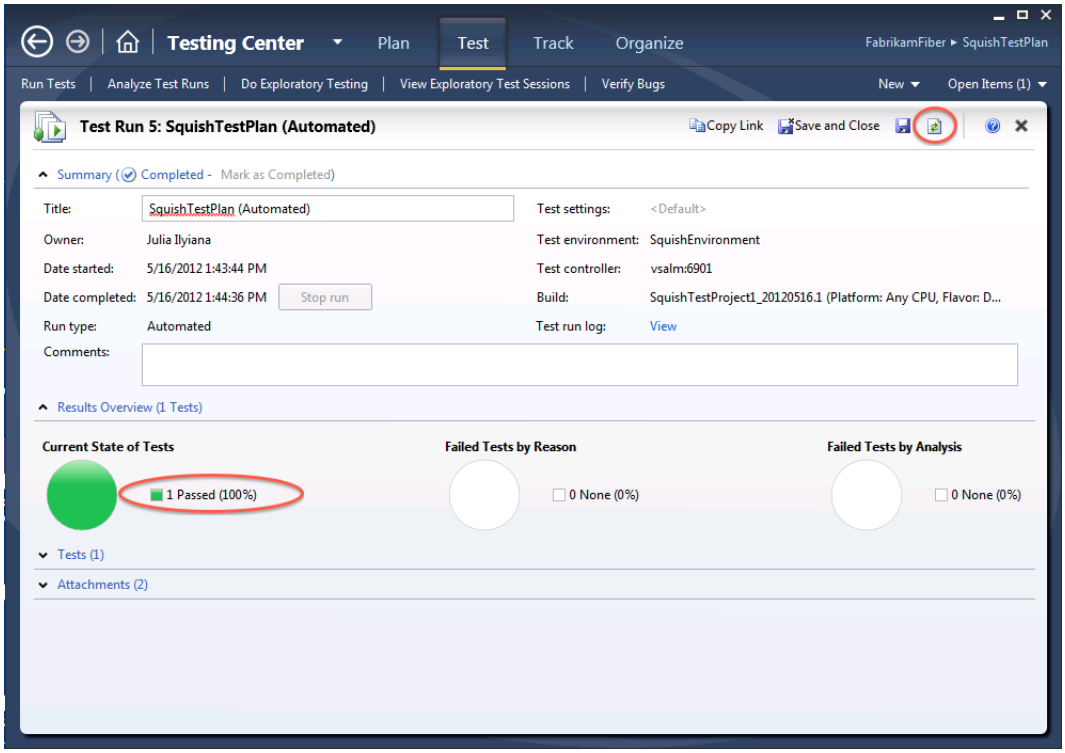

Figure 34

Test run showing the result of the automated GUI test

17. The test results are stored as an attachment. Expand **Attachments** and double-click on the file **tmiRun.trx**, which will open the result in Visual Studio.

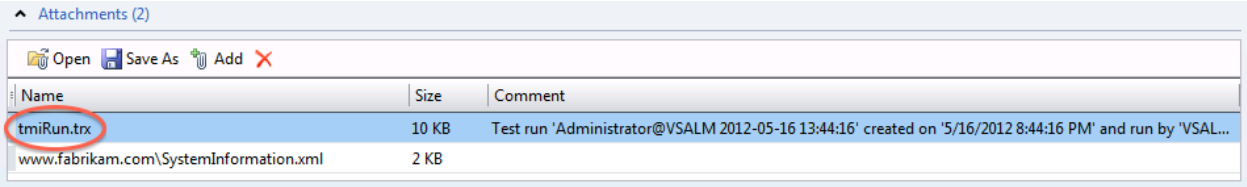

Figure 35

Location of the result file that can be opened in Visual Studio

18. In Visual Studio, double-click on the result in the **Test** Results view to open the details of the test run including the result of each verification.