Hands-On Lab

Introduction to Test Case Management with Microsoft Test Manager 2010

Lab version: 1.0.0
Last updated: 12/10/2010
CONTENTS

OVERVIEW .................................................................................................................................................. 3

EXERCISE 1: MANAGING TEST SUITES AND TEST CASES ....................................................................... 4

EXERCISE 2: ANALYZING TEST RUNS ..................................................................................................... 13

EXERCISE 3: SELECTING BUILDS TO TEST AGAINST ........................................................................... 17
Overview

In this lab you will be introduced to working with Microsoft Test Manager. Microsoft Test Manager can be used to organize your test plans, author and manage your test cases, and run manual tests. This lab will provide you with a basic understanding of how Microsoft Test Manager can be used to perform these activities. Also be sure to see “Authoring and Running Manual Tests with Microsoft Test Manager” if you are interested in these aspects of Microsoft Test Manager.

System Requirements

In order to complete this lab you will need the Visual Studio 2010 RC virtual machine provided by Microsoft. For more information on acquiring and using this virtual machine, please see “Working with the Visual Studio 2010 Virtual Machine”.

Exercises

This Hands-On Lab comprises the following exercises:

1. Managing Test Suites and Test Cases
2. Analyzing Test Runs
3. Selecting Builds to Test Against

Estimated time to complete this lab: 60 minutes.

Next Step

Exercise 1: Managing Test Suites and Test Cases
Exercise 1: Managing Test Suites and Test Cases

In this exercise, you will learn how to use the Microsoft Test Manager to manage test suites and test cases.

1. Log in as Abu Obeida Bakhach (Dev) if you have not already done so. The password is P2ssw0rd (capital letter P, the number two, the letter s, the letter s, the letter w, the number zero, the letter r, and the letter d). Please see “Working with the Visual Studio 2010 Virtual Machine” for instructions on how to log into the VM.

2. Open Microsoft Test Manager from Start | All Programs | Microsoft Visual Studio 2010 | Microsoft Test Manager 2010. This is a new tool dedicated to manual software testing that has been built from the ground up for Visual Studio 2010.

   **Note:** Microsoft Test Manager allows testers to work with test plans, author and organize manual test cases, execute test cases, file bugs, and post results back to Team Foundation Server.

![Microsoft Test Manager starting up](image)

3. Select the activity center drop down from the top menu that currently shows Testing Center. The **Testing Center** is used to work with manual test cases and other general test case management. **Lab Center** is used to work with physical or virtual testing labs. For this lab, we will focus on the Testing Center.
4. Select the Plan tab from the Testing Center.

5. In the Contents panel under the Iteration 2 node, locate the nodes labeled 7, 8, and 9. These nodes are requirements-based test suites which contain test cases linked to requirements. Select test suite 7 to see the list of linked test cases.

6. Open the test case with ID = 41 by right-clicking on it and selecting Open Test Case from the context menu that appears.
7. Select the **Tested User Stories** tab and note that this test case links back to a user story.

8. Close the test case by selecting the X in the top-right corner of the test case window. Be careful not to close the Test Manager application.

9. Locate and select the test suite labeled **End-to-End Tests**. This **static test suite** contains arbitrary test cases and other test suites.

   **Note:** There may not be any items in the End-to-end tests suite.

10. The third and final test suite type is the **query-based suite**. Although not represented in this lab, query-based test suites allow work item queries to be constructed in order to gather test cases.
For example, imagine a scenario where you want to create a test suite that includes all priority 1 test cases from iteration 1’s test plan.

11. Locate and select **test suite 9** and note that its state is set to **In Planning**. The remaining two states are **In Progress** and **Completed**. Only test suites that are set to the In Progress state are shown on the Test activity tab.

12. Locate and select **test suite 7**. Note that you can open, add, create, and remove test cases from the right-hand side of the window.

13. Select the test case with **ID = 41**.

**Figure 8**
*Query-based suite*

**Figure 9**
*Test suite state selection*

**Figure 10**
*Working with test cases [your screen may look different]*

**Note:** To learn more about working with manual test cases, see the lab titled Authoring and Running Manual Tests using Microsoft Test Manager 2010.
14. Select the **Configurations** button to load the Select Test Configurations window.

15. Select the **All Configurations** button to show all configuration options that are available and select the **Vista and IE7** checkbox.

![Select Test Configurations](image)

**Figure 11**
Working with test configurations

16. Select the **Apply Changes** button to continue.

17. Select the **Properties** link to load the test plan properties window. The first section of the test plan properties window allows you to edit the basic properties like name, description, area path, iteration, state, and so on. The **Run Settings** section allows you to specify how manual and automated test runs should be setup, what the test environment to use, and even the specific build to test.

![Test Plan Properties Window](image)

**Figure 12**
Test plan properties window
Note: Test settings are particularly important as they specify how and what data will be collected during test runs. If bugs are found during a test run, this data will be used by developers to reproduce and better understand the problem.

18. Open the test settings for **Manual Runs** by selecting on the Open link.

![Figure 13](image1)

*Location of Open link*

19. Select the **Data and Diagnostics** step from the left-hand side of the test settings window. This allows you to select which **Data Diagnostic Adapters** you want to utilize. For example, the Video Recorder will record the screen as seen by the tester during test runs.

![Figure 14](image2)

*Data and Diagnostics options*
20. Enable the **Event Log** diagnostic adapter and select the **Configure** button.

![Configure Event Log diagnostic adapter](image1)

**Figure 15**
**Event Log diagnostic adapter**

21. In the **Configure Diagnostic Data Adapter — Event Log** window, note that you can specify which event logs and event types to collect from. Select the **Save** button to continue.

![Configure Diagnostic Data Adapter](image2)

**Figure 16**
**Configuring the Event Log diagnostic adapter**

22. Select the **Finish** button to return to the test plan properties window.

23. **Test environments** are also managed from the test plan properties window. Test environments are either physical or virtual environments used to run tests against or to collect data. For example, when testing a Web application, it may be necessary to collect data from both the client and the server at the same time as components of that system reside in different environments. For this test plan, there is just one local test environment.

24. **Test configurations** describe which platforms to test against during test runs. Select the drop down that currently has Windows Server 2008 and IE8 selected to see the options available. Select the **Don't Apply** button when finished.
25. The **Builds** section of the test plan properties window defines the build definition that test results and bugs will be filed against. Select the drop down labeled **Filter For Builds**.

26. Build quality can also be specified so that testers do not prematurely start the testing process. Select the **Build Quality** drop down to see the options available.
27. Press the Escape key to cancel the selection of a build quality and select the Cancel button to return to the test plan properties window.

28. Finally, note that a specific build is setup for test runs.

29. Close the test plan properties window by selecting the X in the top-right corner of the test plan properties window. Be careful not to close the Test Manager application.

Next Step

Exercise 2: Analyzing Test Runs
Exercise 2: Analyzing Test Runs

In this exercise, you will learn how to use the Test activity to analyze test runs.

1. In Microsoft Test Manager, select the Test tab to open the test activity used by testers. By default, the Run Tests window is loaded.

2. Select the test suite 7 node to view the test runs and their states. Note that the test case with ID = 41 is represented three times, one for each configuration that has been configured for it.

   ![Test suite 7: As a customer I should be able to remove... (Requirement 7)](image)

   Figure 21
   Viewing test runs for a test suite [your screen may look different]

3. Select the test suite 8 node to view its test runs. Note that two of the test cases are listed as being blocked. **Blocking** a test case is an action that can be performed by the tester to indicate that they are unable to perform the test case, perhaps due to something such as a user story not being implemented yet or a dependency requirement not working. For example, if the test case is to create an account on a Web application, and some other related component prevents the tester from even attempting the action, the tester can indicate that they are blocked.

   ![Test suite 8: As a customer I should have to enter a st... (Requirement 8)](image)
4. Select the **Analyze Test Runs** link and select the **Show Manual Runs** button to view the historical test runs for the current test plan.

![Analyze Test Runs window](image)

**Figure 23**

**Analyze Test Runs window**

5. Open the test run with **ID = 21** by double-clicking on it. A summary shows start and completion timestamps, type, build version, and so on. Expand the **Tests** area to expose the results of the test run for the associated test cases.
Figure 24  
*Results of a test run*

6. Open one of the test results for the test case with ID = 41. Note that there are a number of attachments that were collected during the test such as system information and a video of the manual test.
7. Close the test results window by selecting the X in the top-right corner of the test results window. Be careful not to close the Test Manager application.

Next Step

Exercise 3: Selecting Builds to Test Against
Exercise 3: Selecting Builds to Test Against

In this exercise, you will learn how to use the Track activity to navigate and select the builds to test against.

1. In Microsoft Test Manager, select the Track tab to open the track activity. By default, the Queries window is loaded.

2. Select the Assign Build link to view the available builds and associated work items.

3. In the Available Builds drop down, select the last build in the list (Tailspin Toys – Iteration 2_20100318.4).

Figure 26
Assigning builds for testing

Figure 27
Selecting a build
4. Look at the associated work items to see what work was done during the selected build. Note that a single work item to “Write regular expression for strong password” was completed.

5. Select the Tailspin Toys – Iteration 2_20100318.5 build from the Available Builds drop down. There are three associated work items for this build, one of which is the same work item as before.

   **Note:** The Track activity can be used to help determine when it will be worthwhile to perform a test pass.

6. Select the Toys – Iteration 2_20100318.6 build from the Available Builds drop down and select the Assign To Plan button. This will help provide an indication to the testers that this build warrants doing a full test run. Future test runs will use the new build by default.

7. When notified that some tests may need to be run again based on code changes, choose to view the recommended tests.
8. These recommendations are made possible by a new feature called **Test Impact Analysis**, which is able to determine when code changes impact previously executed tests. For example, a test that was successfully executed against an earlier build may be recommended to be executed again if it is determined that (a), code has changed and (b), that it is in the code path exercised by that test.

**Note:** Test Impact Analysis can be configured to run in the background when the application being tested is written in managed code (.NET Framework 2.0 or higher).

9. Select all recommended tests that are impacted by this latest build and select the **Reset To Active** button. This resets the state of test cases within the test plan to the Active state.

![Figure 31](image1.png)

*Figure 31*

*Resetting test case state to active for recommended tests*

10. Return to the **Test** activity window and select the **Run Tests** link.

![Figure 32](image2.png)

*Figure 32*

*Return to Test activity*

11. Select test suite 7. Note that the two test cases there were previously in the Passing state are now Active again.
Test cases were reset to Active state

12. Return to the Track activity window, select the Recommended Tests link, and select the Related Work Items button. Note that a number of work items that have been closed or resolved since build Tailspin Toys – Iteration 2_20100318.3, including a user story work item.

13. Return to the Test activity window and select test suite 8, which is associated with the impacted user story that we just looked at in the previous step.

14. Select the blocked test cases and then select the Reset button to unblock them. Their state goes from Blocked to Active.
15. Select the **Verify Bugs** link. Although we will not do so in this lab, this is where testers could select previously reported bugs and re-run the tests. If the tests do not pass, the testers would re-assign the bugs back to the developers.