Test Case Management Using Visual Studio Test Professional 2010

Accentient™
ALM Experts

ACCENTIENT EDUCATION SERIES

Committed to training success

+1 (877) 710-0841 ■ www.accentient.com

Test Case Management Using Visual Studio Test Professional 2010

Course Number:	**TCM**
Version:	**1.1**
Software version:	**2010**

Copyright © 2012 Accentient, Inc. All rights reserved.

No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without the express written permission of Accentient, Inc.

Images and Artwork
Images and artwork in this book were obtained through Flickr and are licensed under a Creative Commons 3.0 license. Visit http://creativecommons.org/licenses/by/3.0 for more information.

All trademarks referenced are the property of their respective owners

---

Disclaimer

While Accentient takes great care to ensure the accuracy and quality of these materials, all material is provided without any warranty whatsoever, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

Accentient™

Test Case Management Using Visual Studio Test Professional 2010

Module 2
Test Planning

Agenda

- Microsoft Test Manager
- Test Plans
- Test Suites
- Test Settings
- Test Environments
- Test Cases
- Shared Steps

Sample

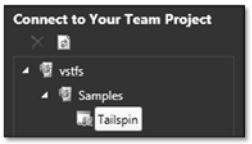
Microsoft Test Manager 2010 (MTM)

- Microsoft's new, standalone tool for testers
  - It's WPF-based and definitely not Visual Studio
  - Available with Visual Studio Ultimate and Test Professional editions

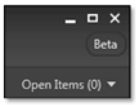
- It enables testers to …
  - Create and manage test plans
  - Create and manage test suites
  - Create and manage test case
  - Create and manage test configurations
  - Run manual tests
  - Record manual tests for playback
  - Playback manual tests
  - Create and manage work items

Using Microsoft Test Manager

- To use MTM, you must connect to a team project
  - This should be the same team project the rest of the team is using to manage user stories (requirements), bugs, source code, and builds



  - Your testing artifacts are created and managed in this same team project
- You can select a different team project
  - Click your test plan in the upper-right corner
  - Select a new plan, possibly on a new team project, collection, and server



Sample

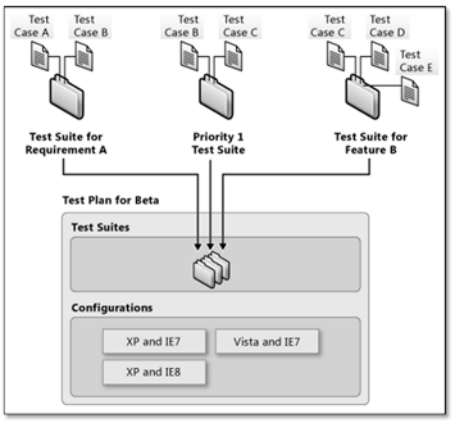
Testing Center – Plan, Test, Track

- Microsoft Test Manager has a Testing Center and a Lab Center
  - Testing efforts are planned and managed in the Testing Center
  - Virtual lab environments are created and managed in the Lab Center

- Lab Center has four modes
  - Plan
  - Test
  - Track
  - Organize

Test Case Management Workflow

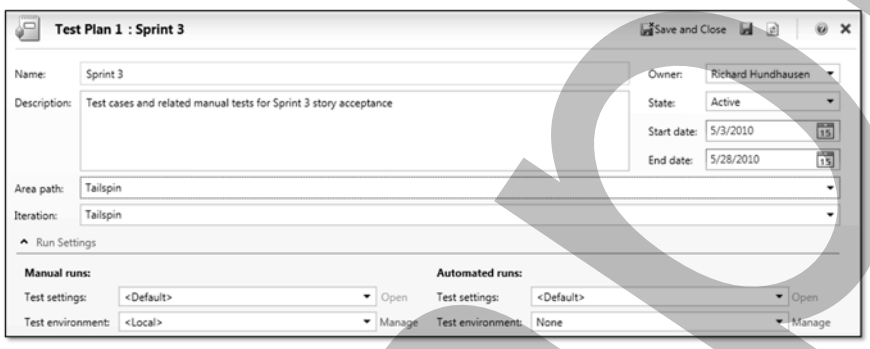
- Create a Test Plan
  - Alpha, Beta, Sprint 3, etc.
- Create one or more Test Suite
  - Requirements, features, etc.
- Create one or more Test Case
  - The manual/UI tests
- Manage the settings and configurations
  - Used when you run the test(s)
- Track your results, progress, and quality



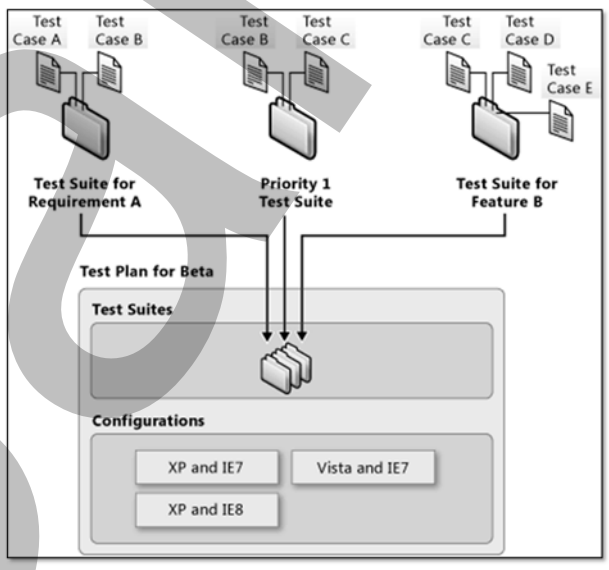
Sample

Test Plans

- A test plan lets you specify what you want to test and how to run those tests
  - You can associate a plan to dates, area, iteration
  - You can specify settings and environments (lab VMs)



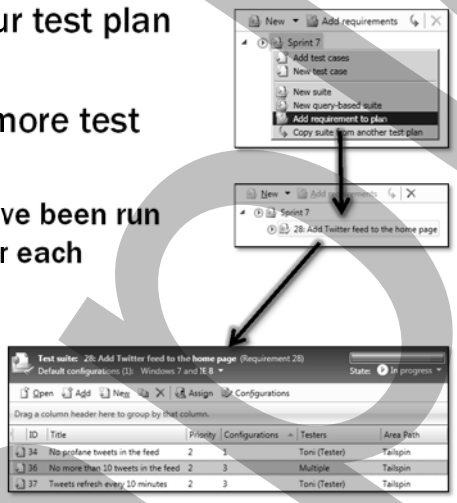
Key Components of a Test Plan



Sample

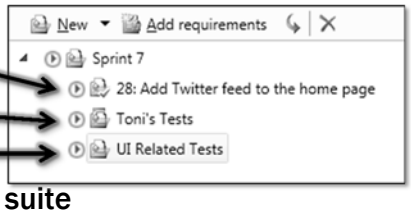
Adding Requirements to Test Plans

- You can add existing requirements (user stories, requirements, PBIs) to your test plan
  - This will create a test suite
- You can then add one or more test cases to this requirement
  - You can see which tests have been run and what the results are for each



Test Suites

- Suites allow you to logically group your test cases
  - This allows you to run all of your tests in a suite
  - The root node is a suite, with the same name as the plan
  - Test suites can be copied from other test plans
- Test suites can be
  - Based on requirements
  - Based on a query
  - Created manually
  - Icons identify the type of test suite

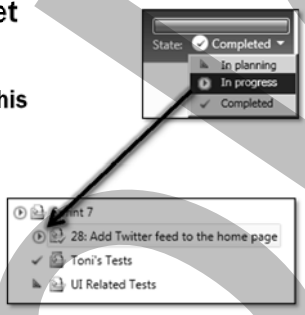


- Note: manually-created suites can contain other suites

Sample

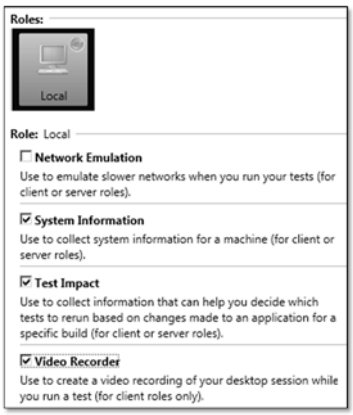
Test Suite States

- Test suites have a state that you can set
- In Planning
  - If you are not yet ready to run the tests in this test suite
  - No tests will be displayed in the Run Tests View for suites in this state
  - This is the default state
- In Progress
  - When you are ready to run the tests in this test suite
  - Note: Only tests in this state may be run
- Completed
  - When you have finished running the tests in this test suite and the quality level reached meets your goals



Test Settings

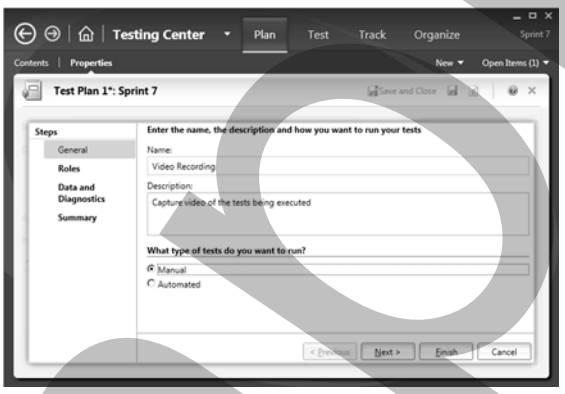
- Test plans can be configured with different settings
  - Type of tests (manual vs. automated)
  - Roles (lab VMs)
  - Data to collect for each role
- Available data collectors
  - Actions
  - ASP.NET IntelliTrace/Test Impact
  - Event Log
  - IntelliTrace
  - Network Emulation
  - System Information
  - Test Impact
  - Video Recorder (requires Windows Media Encoder)



Sample

Test Settings

- Test settings define:
  - The type of tests that you will run (manual or automated)
  - The set of roles that are required for your test run
  - The role to use to run your tests
  - The diagnostic data adapters

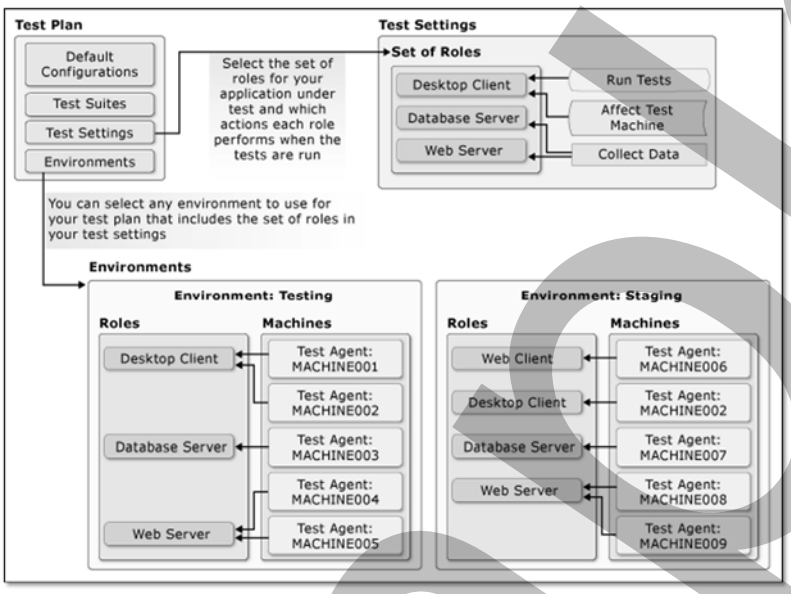


Environments

- MTM can create physical or virtual environments
  - A physical environment might use physical computers or virtual machines
  - A virtual environment uses only virtual machines
- An environment consists of a set of roles
  - A role specifies the purpose of one or more computers in the environment
  - You can run tests, collect data, or otherwise control a test machine for each specific role in your environment
  - Example: "Web Server"

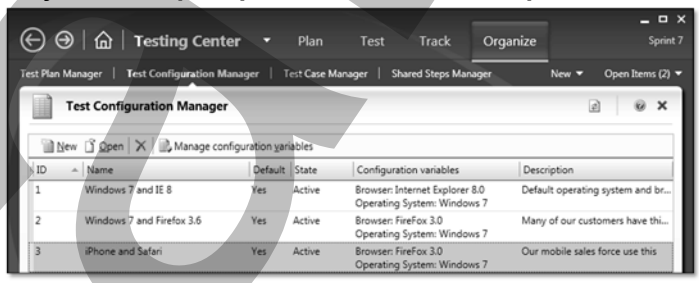
Sample

Environments – *Example*



Test Configurations

- A test configuration is a set of configuration variables that specify the setup requirements for a test plan

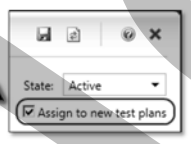


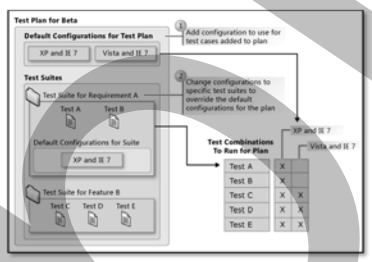
- <u>Note</u>: you cannot delete configurations referenced in a test result or that is a default configuration for a test plan/suite
  - Instead, you should change the state of the configuration to inactive so that it can no longer be selected as a default configuration

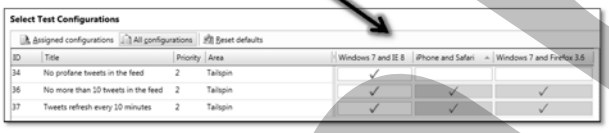
Sample

Default Test Configurations

- Test configurations can be set to "default"
- When you add test cases to the test plan, a pairing of each default configuration with the test case is added to the test plan
  - These pairings are called *test points*
  - If you add a new configuration, the test points are not created for existing test cases
  - The defaults can be overridden for the individual test cases

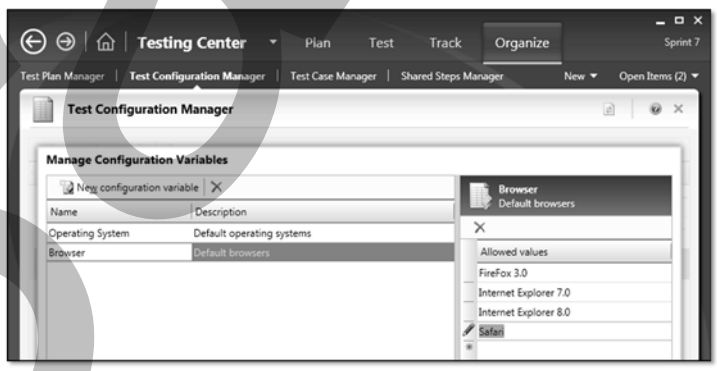






Configuration Variables

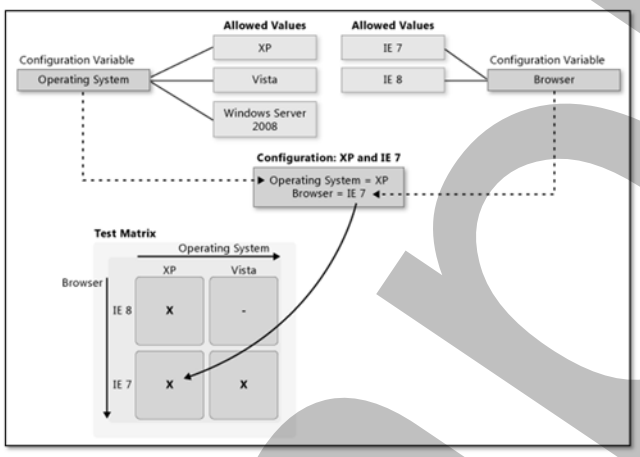
- You can add new variables and allowed values
  - Examples: service pack, drivers, video settings, sound capability, motherboard type, memory, CPU speed, network connectivity, network speed, etc.



Sample

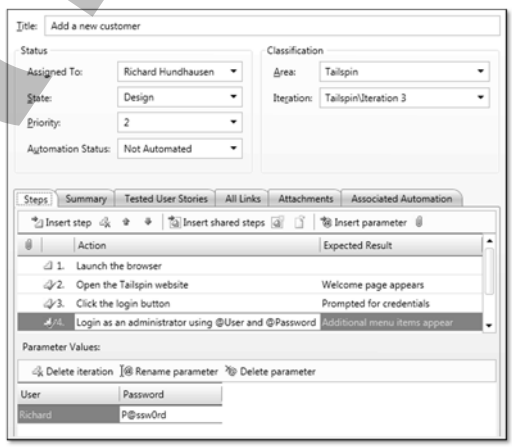
Defining your Test Matrix

- Each test configuration can represent an entry in your test matrix



Test Cases

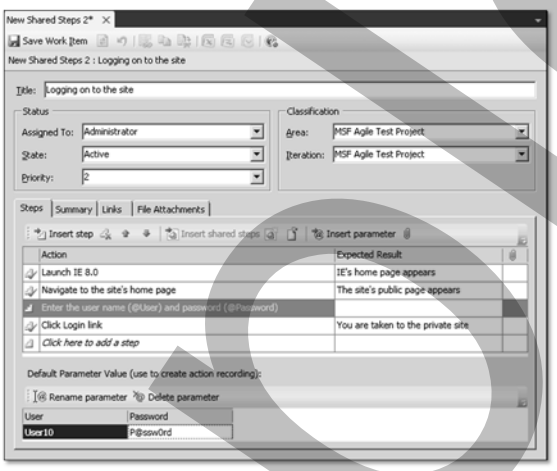
- Test Cases are just TFS work items
- Use MTM to specify the manual test steps
  - Steps are read-only from other clients
- Use Shared Steps to simplify test case management
- Use parameters to allow a test case to run multiple times with different data



Sample

Shared Steps

- Shared steps allow you to define a sequence of steps once and reuse it from many test cases
- "Shared steps" is just another type of work item



Benefits of using Shared Steps

- Shared Steps reduce test maintenance by sharing test steps across test cases
- You can add parameters to your shared steps to use them in test cases where you want to run the same test multiple times with different data
- To reduce the time impact of testing, you can create an action recording that is specific to shared steps
  - This will be automatically *inherited* by those test cases using those steps

Sample

Test Case Migrator Plus

- A wizard-based tool that migrates test case, user story, and bug work items into TFS
  - Maintains links between work items
  - Provides ability to create Test suites for a test case work item type
  - Allows you to save your selections/configurations into a settings/mappings file to be reused later
  - Command line support to run the tool in a batch mode
- Can migrate from two different data sources
  - Excel (3rd party)
  - MHT/doc (Visual Studio 2005/2008)
- Free tool available on CodePlex
  - http://tcmimport.codeplex.com

Summary

- Use Microsoft Test Manager to create and manage test plans, suites, cases, and test runs
  - It is available as part of Visual Studio 2010 Ultimate edition or Visual Studio 2010 Test Professional Edition
  - MTM can run both manual and automated tests
- Test settings, configurations, and environments allow you to setup the training environment to directly map to your environment and its requirements

Sample

Lab

This lab introduces you to the Microsoft Test Manager features and activities relating to test planning.

- Create a test plan
- Create test suites
- Create test configurations
- Create test cases
- Create shared steps
- Use Test Scribe (optional)
- Use Test Case Migrator Plus (optional)

Sample

Lab 2: Test Planning

Test Case Management Using Visual Studio Test Professional 2010

Sample

Estimated time to complete this lab: 90 minutes

There are many steps to test planning. Besides the collaboration that you and your team will do, there are also a number of tasks inside Microsoft Test Manager that have to be performed before you can start executing tests and tracking results.

In this lab you will create a test plan and test suites. You will create and associate a test configuration and then create a number of different test cases. Time and interest-permitting, you can install and get experience with some aftermarket utilities that help import and document your tests cases.

Sample

EXERCISE 1 – CREATE TEST PLAN

TASK – CONFIGURE INTERNET EXPLORER

In this task you will logon to Windows as Toni and configure Internet Explorer for first-time use and to trust the localhost server for testing.

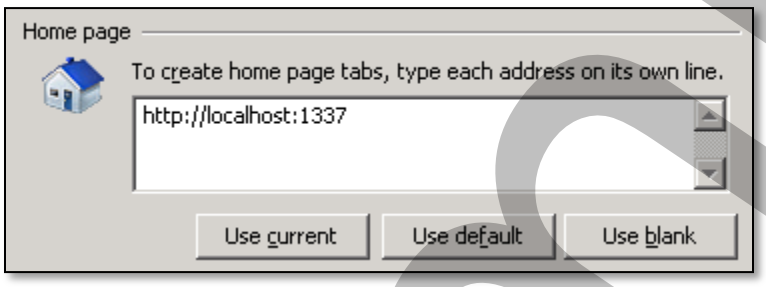
1. Log on to Windows using **Toni** for the user name and **password** for the password.

2. Launch **Internet Explorer**.

   The first time a user brings up Internet Explorer, it wants to the user to set it up. You can either go through the process or click the Ask me Later button.

3. From the toolbar select **Tools** > **Internet Options**.

4. Set the home page to **http://localhost:1337**.



5. Click the **Security** tab.

6. Select the **Local intranet** zone and click **Sites**.

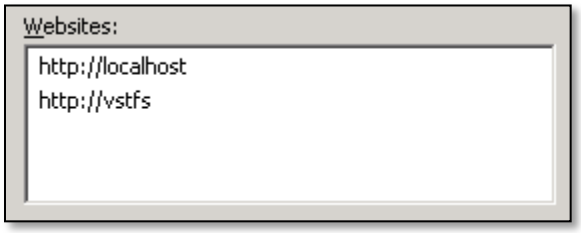
7. Click **Advanced**.

8. Enter **http://vstfs** and click **Add**.

Sample

9. Enter **http://localhost** and click **Add**.

   You should now have two Websites listed:



10. Click **Close**

11. Click **OK** twice.

12. Exit **Internet Explorer**.

    You may be prompted to close all tabs. This is fine.

13. Start **Internet Explorer** again.

    Did the Tailspin Toys Web site come up? _____

    If it did not, double-check your homepage settings and the ask your teammates or the instructor for help.

14. Exit **Internet Explorer**.

Sample

TASK – CREATE A TEST PLAN

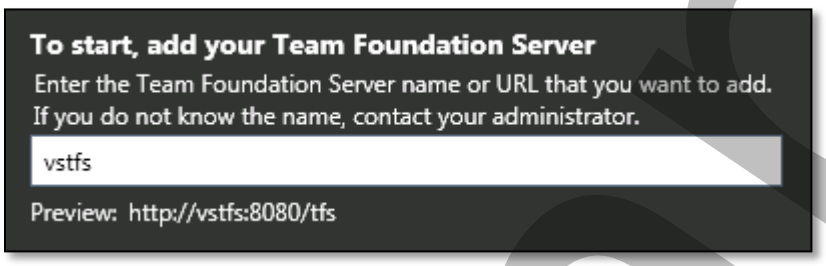
In this task you will launch Microsoft Test Manager and create a test plan.

A test plan lets you specify what you want to test and how to run those tests. A test plan can be applied to a specific iteration of your project.

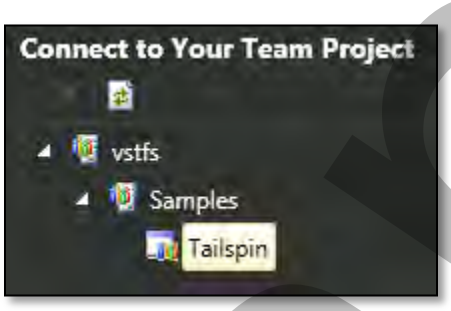
1. From the **Start** menu select **All Programs** > **Microsoft Visual Studio 2010** > **Microsoft Test Manager 2010**.

   Feel free to create a shortcut on your Start menu or desktop if you'd like.

2. Enter **vstfs** for Team Foundation Server and click **Add**.

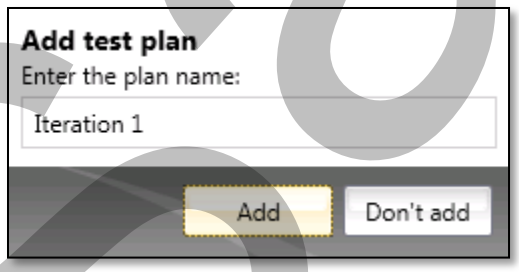


3. Expand the **Samples** collection, highlight the **Tailspin** project, and click **Connect Now**.



4. Click **Add**.

5. Name the plan **Iteration 1** and click **Add**.



Tip: if you are going to have multiple test plans being used concurrently, you should give them descriptive names to identify them.

Sample

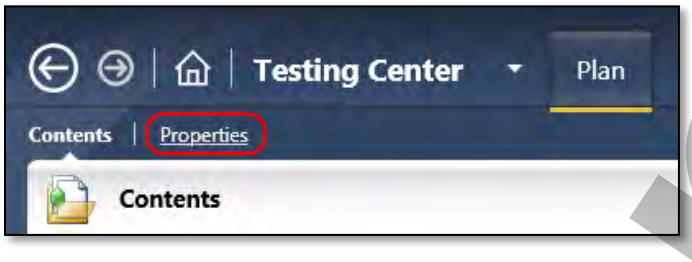
6. Click **Select plan**.

   This takes you to the Testing Center in the Plan mode.

   What other modes are listed at the top? _____

7. Maximize the **Microsoft Test Manager** window.

8. Click the **Properties** link in the above left.



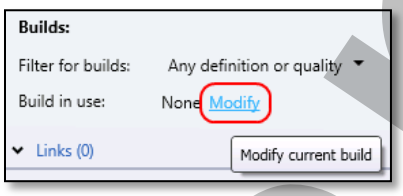
9. Enter **Iteration 1 acceptance and bug testing** in the **Description** field.

10. Select **Tailspin\Iteration 1** as the **Iteration**.

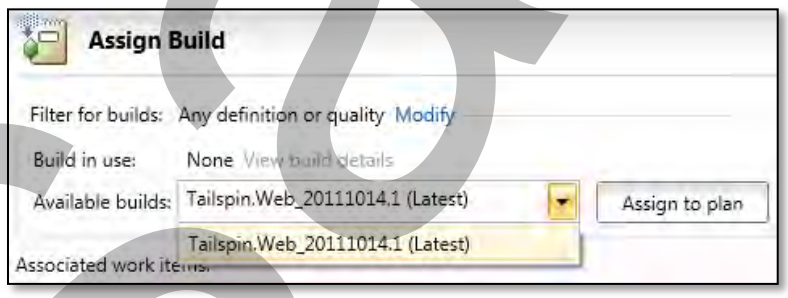
11. Click the **Save** ( ) button in the upper right.

12. Click the **Modify** link next to **Builds in use**.

    This in the lower left area of the properties window.



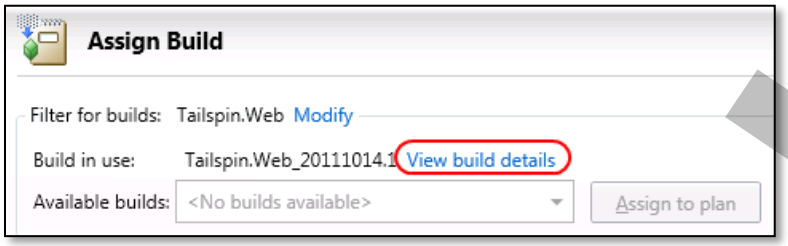
13. Select the latest build from the **Available builds** dropdown and click **Assign to plan**.



Note: If you don't have any builds listed, you may have to rerun the respective tasks in the previous lab, or you can ignore this step. It could be too that you need to change the filter.

Sample

14. Click the **View build details** hyperlink.



This will open Internet Explorer and display a Web page showing the details of the build, including its success/failure, summary, log, and drop folder links. Make sure the build you selected completed successfully. You don't want to plan your testing efforts against a failed build!

15. Close **Internet Explorer**.

16. Close the **Assign Build** window by clicking the (✖).

    You should see the build number listed next to "Build in use" in the properties window now.

17. Click **Save and Close** (Save and Close) in the upper right.

    You will configure the Test settings and configurations in a later exercise.

TASK – CREATE TEST SUITES

In this task you will create a few test suites, including one that map's to this iteration's user stories.

1. Right-click on **Iteration 1** test plan in the tree and select **New Suite**.

   There is also a toolbar button for this.

2. Name the test suite **User Stories**.

3. Right-click on the **User Stories** test suite and select **Add requirement to plan**.

   There is also a toolbar button for this.

   What work item type *category* is going to be queried? _____

4. Click the **Run** (Run) button in the middle of the page.

   How many "requirements" were returned? _____

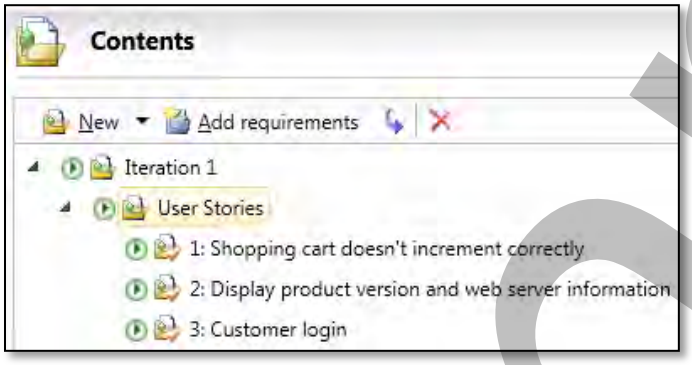
   How many are in iteration 1? _____

Sample

5. Highlight the Tailspin\Iteration 1 work items and click **Add requirements to plan** button.

ID	Title	State	Assigned To	Iteration Path	Area Path	Stack Rank
1	Shopping cart doesn't increment correctly	Active	Paula (Product Owner)	Tailspin\Iteration 1	Tailspin\Orders	1
2	Display product version and web server information	Active	Paula (Product Owner)	Tailspin\Iteration 1	Tailspin\Administration	2
3	Customer login	Active	Paula (Product Owner)	Tailspin\Iteration 1	Tailspin\Customers	3
4	Check order status	Active	Paula (Product Owner)	Tailspin	Tailspin\Orders	4
5	Returns & exchanges	Active	Paula (Product Owner)	Tailspin	Tailspin\Orders	5
6	Shipping options & charges	Active	Paula (Product Owner)	Tailspin	Tailspin\Orders	6
7	Store locations	Active	Paula (Product Owner)	Tailspin	Tailspin\Marketing	7
8	About us	Active	Paula (Product Owner)	Tailspin	Tailspin\Administration	8

You should now see three requirements listed under the User Stories test suite.



6. Right-click on the **Shopping cart doesn't increment correctly** requirement and select **Open requirement**.

   This will open a Team Explorer style work item edit window. It has all the same functionality as that inside Visual Studio.

   How many Story Points was estimated to implement this story? _____

7. Click on the **Implementation** tab.

   How many tasks are listed? _____

8. Click (✖) to close the user story.

9. Right-click on **Iteration 1** test plan in the tree and select **New suite**.

10. Name the suite **Exploratory Tests**.

    This is where you will create and save your exploratory test cases.

Sample

11. Right-click on **Iteration 1** test plan in the tree and select **New query-based suite**.

    What work item type category is going to be queried? _____

    You can click the Run button, but you don't have any tests cases created yet. You'll do that in the next exercise.

12. Name the test suite **Bug verification tests**.

13. Click the **Click here to add a clause** area.

	And/Or	Field	Operator	Value
▶		Team Project	=	@Project
	And	Work Item Type	In Group	Test Case Category
*	Click here to add a clause			

14. Choose **Title** for the Field, **Contains** for the Operator, and **bug** for the **Value**.

And	Title	Contains	bug

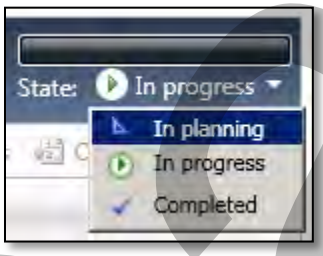
15. Click **Create test suite**.

    You should now have three test suites under the Iteration 1 test plan.

    Note: there won't be any bug test cases listed in this new suite.

16. Highlight the **Exploratory Tests** test suite.

17. Change the **State** to **In planning** in the right-hand side.



    Test suites that are "in planning" state won't have any tests displayed in the Run Tests View.

Sample

TASK – CREATE A TEST CONFIGURATION

In this task you will review the default test configurations, create some new configuration variables, and then create two new test configurations.

1. Click the large **Organize** link at the top.

   Organize is where you can manage your test plans, configurations, cases and shared steps.

   How many test plans are listed? _____

2. Double-click the **Iteration 1** test plan.

   This is another way to set the properties of a test plan.

3. Click (✖) to close the test plan.

4. Click **Test Configuration Manager**.

   How many configurations are listed? _____

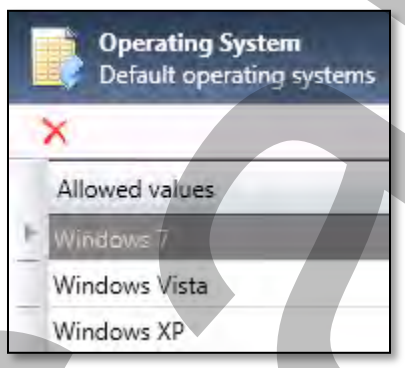
   What is the name of that configuration? _____

5. Click **Manage configuration variables** (Manage configuration variables).

6. Highlight the **Operating System** variable.

   What are the allowed values? _____

   You should have the following allowed values by default:



7. Remove the **Windows Vista** and **Windows XP** values.

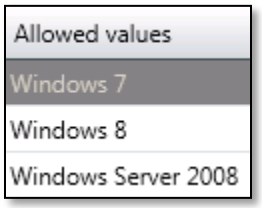
   You can do this by highlighting them and then clicking the red X button.

Sample

8. Add **Windows 8** and **Windows Server 2008** values.

   You can do this by typing in the blank row with the asterisk and then clicking on another row to commit your change.

   You should now have the following allowed values:



9. Highlight the **Browser** variable.

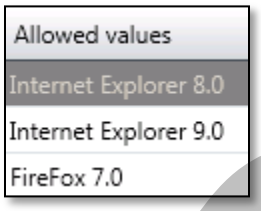
   What are the allowed values? _____

   Note: support for Mozilla Firefox playback of UI actions and coded UI tests is not available in the RTM version of Visual Studio 2010. To download a power tool that provides this support, please visit: http://bit.ly/q1oLsu. For a blog post showing the current levels of support for other UI frameworks visit: http://bit.ly/9h0iEv.

10. Remove the **FireFox 3.0** and **Internet Explorer 7.0** values.

11. Add **Internet Explorer 9.0** and **Firefox 7.0** values.

    You should now have the following allowed values:

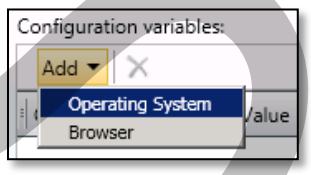


12. Click the **Save configuration variables** button in the lower right.

13. Click the **New** button to create a new Test Configuration.

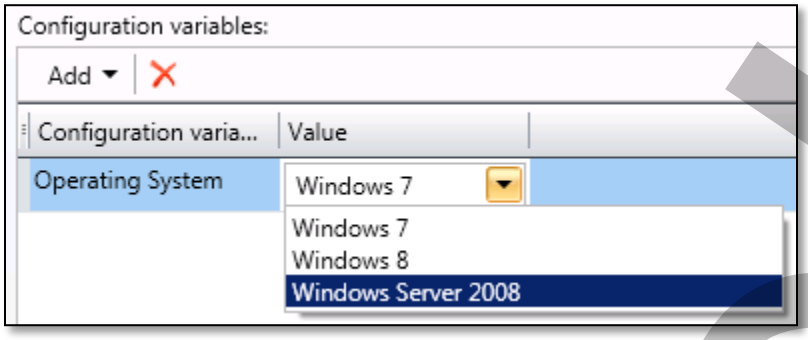
14. Name the new configuration **W2K8 and IE 8** with the description **Windows Server 2008 and Internet Explorer 8.0**.

15. In the **Configuration variables** section select **Add** > **Operating System**.



Sample

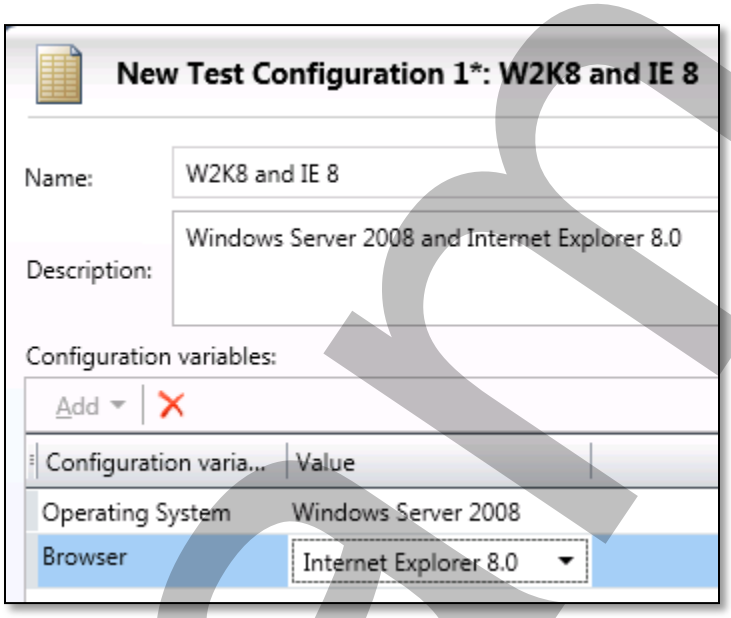
16. Ensure the value is **Windows Server 2008**.



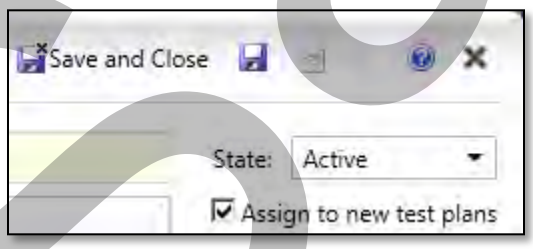
17. Select **Add** > **Browser**.

18. Ensure the value is **Internet Explorer 8.0**.

    Your new configuration should look like this now:



19. Check the **Assign to new test plans** checkbox in the upper right corner.



20. Click **Save and Close**.

Sample

21. Click the **New** button to create a new Test Configuration.

22. Name the new configuration **W2K8 and Firefox 7** with the description **Windows Server 2008 and Firefox 7.0**.

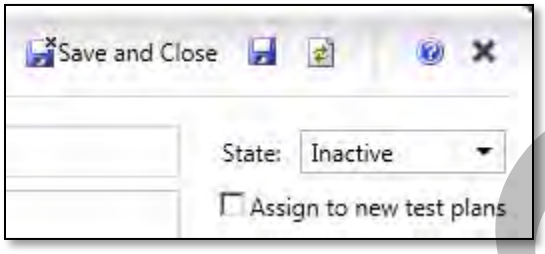
23. Select **Windows Server 2008** for the Operating System and **Firefox 7.0** for the Browser.

24. Click **Save and Close**.

25. Double-click the original **Windows 7 and IE 8** configuration.

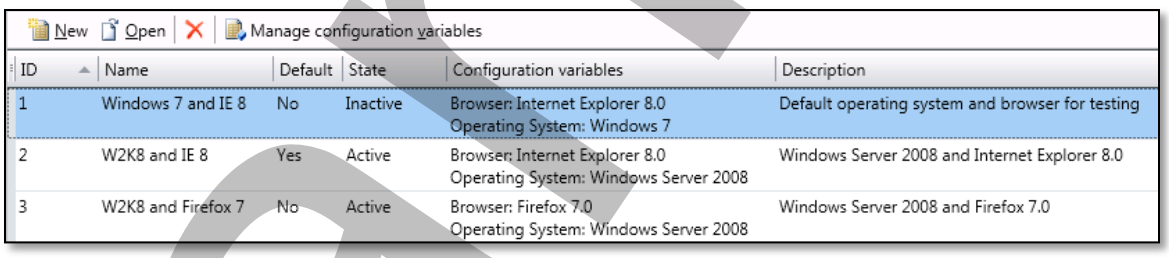
26. Change its **State** to **Inactive**.

27. Clear the **Assign to new test plans** checkbox if it's checked.



28. Click **Save and Close**.

    You should now have two new, active configurations and an older, inactive one:



ID	Name	Default	State	Configuration variables	Description
1	Windows 7 and IE 8	No	Inactive	Browser: Internet Explorer 8.0 Operating System: Windows 7	Default operating system and browser for testing
2	W2K8 and IE 8	Yes	Active	Browser: Internet Explorer 8.0 Operating System: Windows Server 2008	Windows Server 2008 and Internet Explorer 8.0
3	W2K8 and Firefox 7	No	Active	Browser: Firefox 7.0 Operating System: Windows Server 2008	Windows Server 2008 and Firefox 7.0

    Which test configuration is marked as Default? _____

    If you add a test to a plan, it will be assumed that the test is required to be run on all default configurations for the testing to be completed.

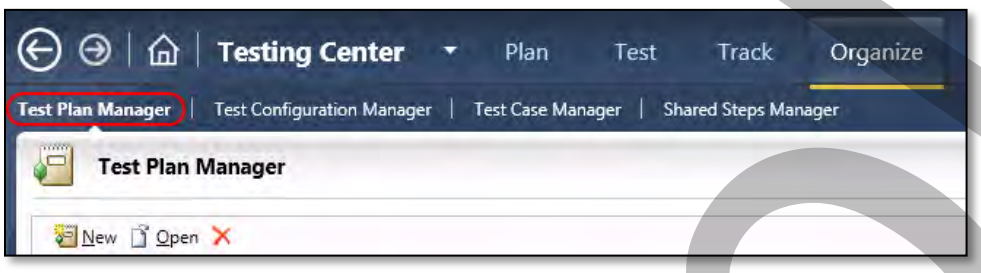
29. Close the **Test Configuration Manager**.

Sample

TASK – ASSOCIATE TEST CONFIGURATIONS

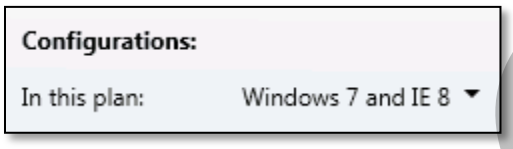
In this task you will associate the new test configurations with the test plan and one of the existing test cases.

1. Return to **Test Plan Manager**.



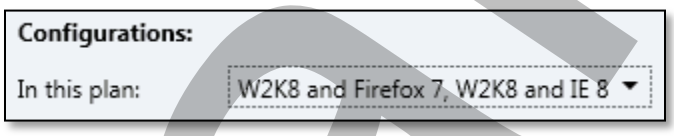
2. Double-click the **Iteration 1** test plan.

   Notice the Configurations that are currently enabled for this test plan:



3. Click the dropdown (▾) control next to **Windows 7 and IE 8**.
4. Uncheck the **Windows 7 and IE 8** configuration.
5. Check the other, new configurations and click **Apply**.

   You should now have these configurations selected for this test plan:



6. Click **Save and Close**.

   At this point, any new test cases which are added to the Iteration 1 test plan will have these two configurations enabled.

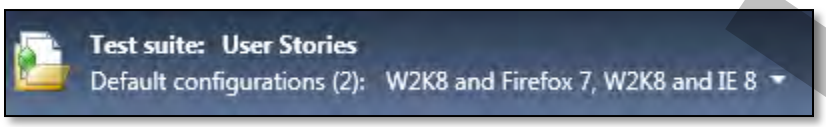
7. Return to the **Plan** tab.

   This is the large *Plan* link at the top of the window.

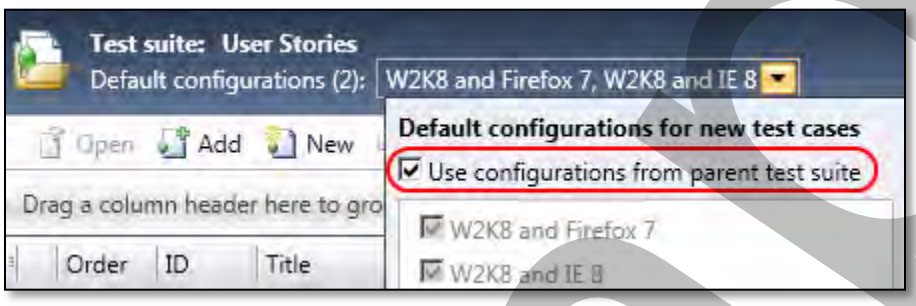
Sample

8. Highlight the **User Stories** test suite.

   Notice the new configurations are visible in the right-hand side of the window. Each test suite will automatically use the configurations set at the test plan level.



   Note: A specific test suite may break this inheritance chain and use a different set of configurations. You won't be doing that in this lab.

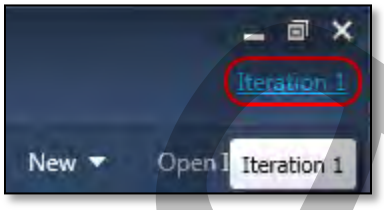


TASK – CREATE ANOTHER TEST PLAN (OPTIONAL)

In this task you will create an Iteration 2 test plan and copy the test suites from the Iteration 1 test plan to gain experience doing this.

Note: This task is optional and has no impact on the rest of the lab exercises.

1. Click the **Iteration 1** link in the upper-right corner of the screen.



2. Click **Add**.

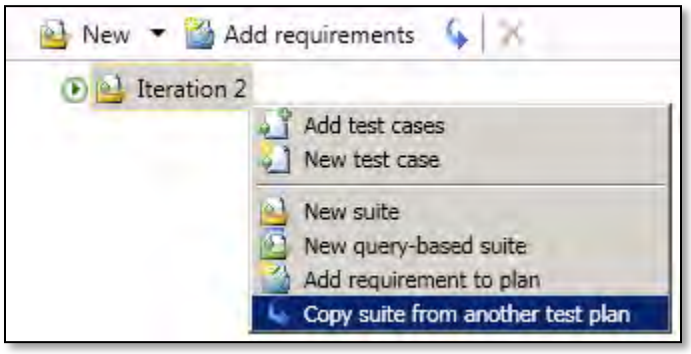
3. Name the plan **Iteration 2** and click **Add**.

4. With **Iteration 2** selected, click **Select plan**.

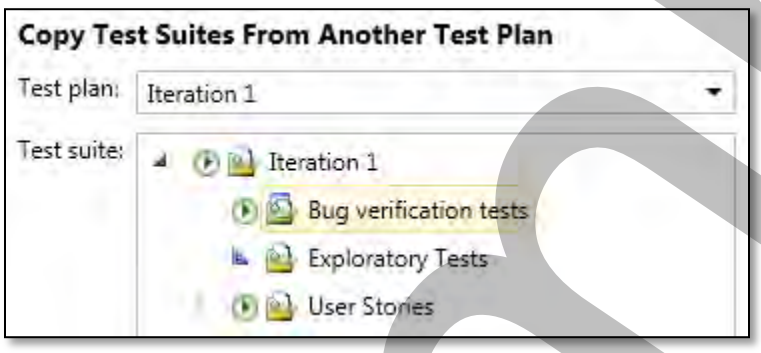
   This will take you to a list of test suites for the Iteration 2 plan. There should only be the root (default) test suite.

Sample

5. Right-click on the **Iteration 2** (root) node and select **Copy suite from another test plan**.



6. Select **Bug verification tests** and click **Copy suite into plan**.



7. Right-click on the copied **Bug verification tests** suite and select **Edit query**.

   Is this the same query that you wrote earlier in this exercise? _____

8. Click the **Don't save** button.

9. Right-click on the **Iteration 2** (root) node and select **Copy suite from another test plan**.

10. Select **Bug verification tests** and click **Exploratory Tests**.

    Notice that the copy of the test suite maintained it's *in planning* state.

11. Click the **Iteration 2** link in the upper-right corner of the screen.

12. Select **Iteration 1** and click **Select plan**.

    You should now be returned to the original test plan.

Sample

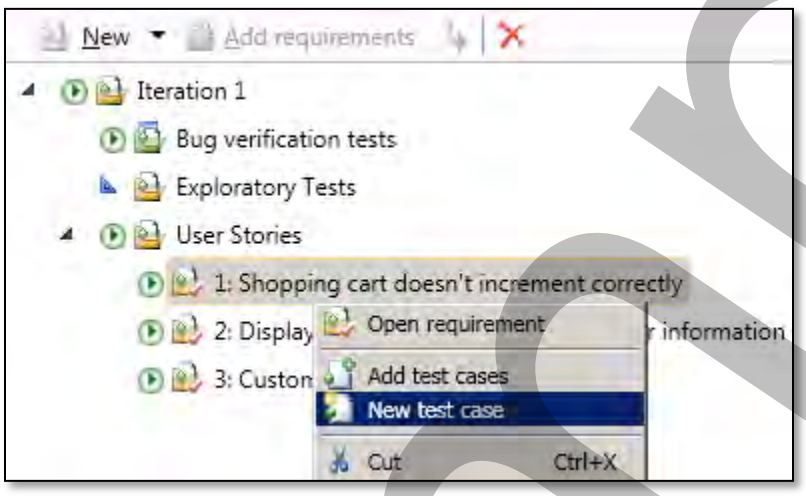
EXERCISE 2 – CREATE TEST CASES

TASK – CREATE A TEST CASE TO VERIFY THE EXISTENCE OF THE BUG

In this task you will create a test case associated with the "shopping cart" user story to verify the existence of the bug user story created in the last lab.

1. Expand the **User Stories** test suite.

2. Right-click on the **Shopping cart doesn't increment correctly** requirement and select **New test case**.



3. Specify a **Title** of **Verify shopping cart item count bug**.

   What is the default Area for this new test case work item? _____

   What is the default Iteration for this new test case work item? _____

   These default values are inherited from the requirement (user story).

4. Click the **Summary** tab.

5. Enter the following **Description**:

   Add an item to the cart, then go back and add the same item again and notice that the item quantity didn't change.

6. Click **Save and Close**.

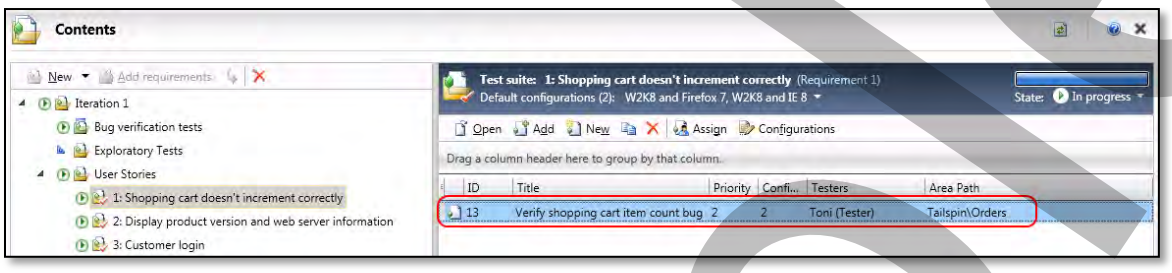
   At this point, the test case has enough information that a competent tester might be able to go about verifying the existence of the bug. You're going to make it more air-tight and provide that actual test steps in the next task.

Sample

TASK – SPECIFY THE TEST CASE STEPS

In this task you will specify the actual test steps for the test case.

1. Select the new **Verify shopping cart item count bug** test case in the right-hand side.



What color is the bar above the State: In progress area? _____

What does the blue color mean (hover mouse over it)? _____

The "2 tests" is really just the 1 test case x 2 test configurations (Firefox and IE8).

2. Double-click the **Verify shopping cart item count bug** test case.

This will open the test case for editing. You will enter the test steps now.

3. In the **Steps** tab, enter the following actions and expected results:

Tip: press enter after entering the *Expected Result* to automatically go to the next step.

Action	Expected Result
Launch the browser	Tailspin Toys home page appears
Click the MODEL AIRPLANES link	Model airplanes home page appears
Click the Fourth Coffee Flyer's View Plane link	Fourth Coffee Flyer details appear
Click ADD TO CART	Shopping cart appears with message "1 items in your cart"
Click the MODEL AIRPLANES link	Model airplanes home page appears
Click the Fourth Coffee Flyer's View Plane link	Fourth Coffee Flyer details appear
Click ADD TO CART	Shopping cart appears with message "2 items in your cart"
Close the browser	

4. Click **Save and Close**.

Note: You'll execute this test case in the next lab.

Sample

TASK – CREATE A PARAMETERIZED TEST CASE

If you want to run a manual test case multiple times with different data, you do not have to create multiple copies of the test case. You can add parameters to the actions or expected results for any test step in your test case.

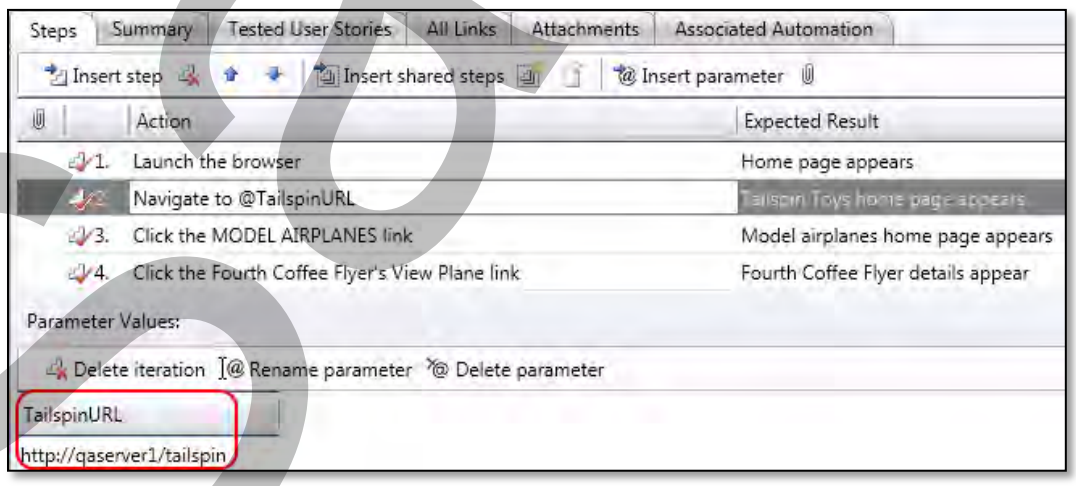
In this task you will create a copy of the test case you just created, edit it, and add a parameter in order to simplify testing across a diverse testing environment.

1. Right-click on **Iteration 1** test plan in the tree and select **New Suite**.
2. Name the test suite **Parameterized Tests**.
3. Right-click on the **Verify shopping cart item count bug** test case and select **Create Copy and add to suite**.

   This test case is found in the *Shopping cart doesn't increment correctly* test suite.
4. Change the title to **Verify shopping cart item count bug (Parameterized)**.
5. Change the first step's **Expected Result** to: **Home page appears**.
6. Right-click on the second step and select **Insert step**.
7. Add this step:

Action	Expected Result
Navigate to @TailspinURL	Tailspin Toys home page appears

8. Click on another row to commit your changes.
9. Enter **http://qaserver1/tailspin** for the value of the new TailspinURL parameter.



Sample

10. Add another value below the first one: **http://stageserver1/tailspin**.

    You can add multiple values for the parameters that you have defined. Each set of values for the parameters is run as an individual iteration of the test.

11. Click **Save and Close**.

12. Drag the new **Verify shopping cart item count bug (Parameterized)** test case to the **Parameterized Tests** suite.

    Note: You won't be executing this test.

TASK – CREATE AN EXPLORATORY TEST CASE

In this task you will create another test case, one that is exploratory in nature.

Exploratory testing is a type of ad-hoc testing where the tester doesn't have fully-specified test steps. He or she may not have any test steps at all – just a title and a description that summarizes what part of the application to be looking at and, possibly, for what behaviors.

1. Right-click on the **Exploratory Tests** suite and select **New test case**.

2. Specify the following information for the new test case work item:

   - Title: **Explore the checkout process**
   - Area: **Tailspin\Orders**
   - Steps:

Action	Expected Result
Add some items to the cart and explore the checkout process	There are no surprises
Close the browser	

3. Click **Save and Close**.

   You should see the new test case listed in the Exploratory Tests suite.

   What test case(s) do you see in the Bug verification tests suite? _____

   _____

   Note: You'll execute this test case in the next lab.

Sample

EXERCISE 3 – CREATE SHARED STEPS

TASK – CREATE SHARED STEPS

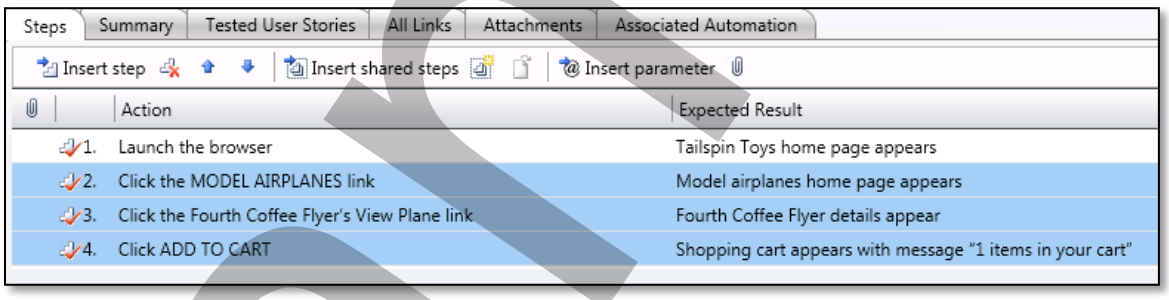
You can share steps between manual test cases by creating shared steps. For example, the test steps to log on to your application under test might have to be run in several test cases.

In this task you will replace some repetitive steps in your test case with a shared sets.

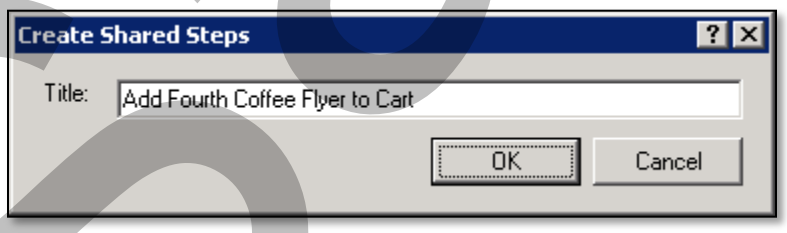
1. Right-click on **Iteration 1** test plan in the tree and select **New Suite**.
2. Name the test suite **Shared Step Tests**.
3. Right-click on the **Verify shopping cart item count bug** test case and select **Create Copy and add to suite**.

   This test case is found in the *Shopping cart doesn't increment correctly* test suite.
4. Change the title to **Verify shopping cart item count bug (Shared Steps)**.
5. Select steps 2 through 4.

   You can do this by clicking on the second step and then holding down the shift key and clicking the fourth step.



6. Right-click on the selected rows and select **Create shared steps**.
7. Name it **Add Fourth Coffee Flyer to Cart** and click **OK**.



What happened to those selected rows? _____

Sample

8. Select steps 3 through 5.

9. Right-click on the selected rows and select **Delete**.

10. Right-click on step 3 (Close the browser) and select **Insert shared steps**.

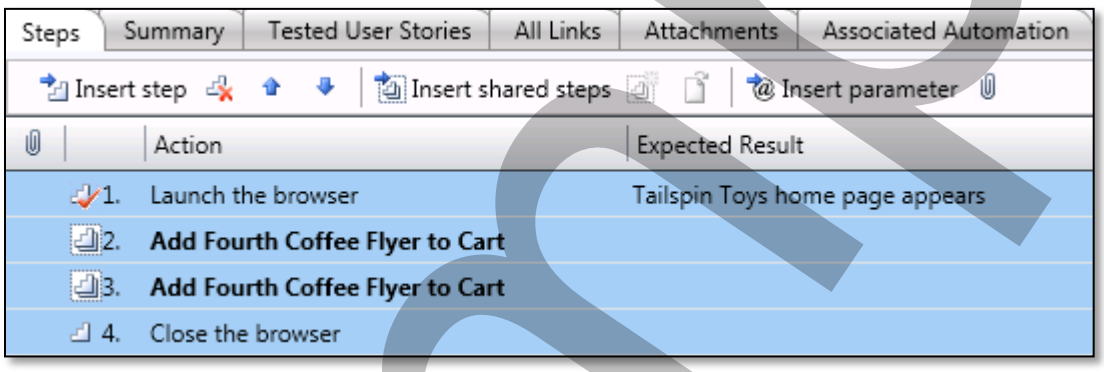
    This will bring up a work item query. By default it will look for all work items in the "shared step" category for the current team project.

11. Click **Run** (Run).

    This should only return the one shared step defined.

12. Select the **Add Fourth Coffee Flyer to Cart** shared step and click **Add shared steps**.

    You should now have four simple steps in your test case:



13. Click **Save and Close**.

14. Drag the new **Verify shopping cart item count bug (Shared Steps)** test case to the **Shared Step Tests** suite.

TASK – MANAGE SHARED STEPS

In this task you will use the Shared Steps Manager to edit the shared steps work item and make the expected result more generic.

1. Go to the **Organize** tab by clicking the large link at the top.

2. Select the **Shared Steps Manager** link.

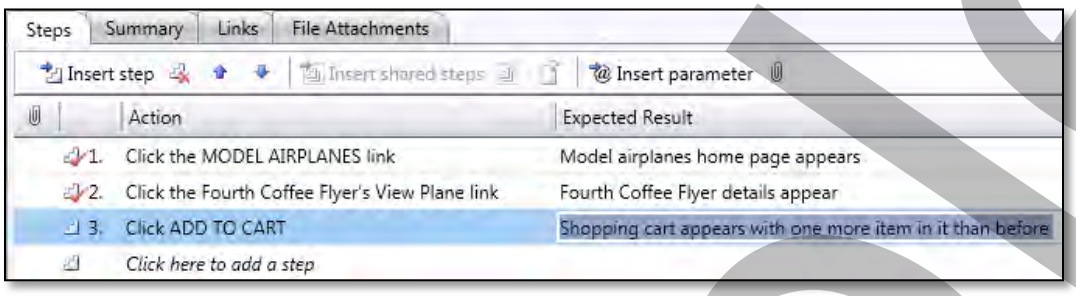
3. Double-click the **Add Fourth Coffee Flyer to Cart** shared step.

   This will open the shared steps work item for editing.

Sample

4. Change the **Expected Result** for step 3 to:

   Shopping cart appears with one more item in it than before.



5. Click **Save and Close**.

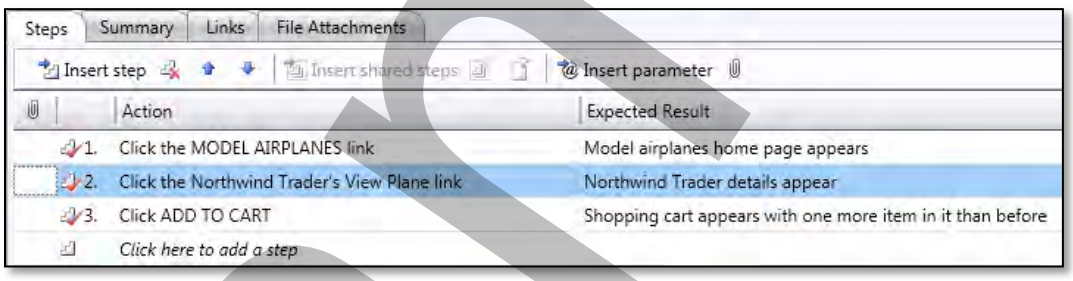
6. Click the **Create Copy** (Create copy) button.

7. Change the **Title** to **Add Northwind Trader to Cart**.

8. Change step 2's **Action** to **Click the Northwind Trader's View Plane link**.

9. Change step 2's **Expected Result** to **Northwind Trader details appear**.

   Step 2 should now look like this:



10. Click **Save and Close**.

    You now have a couple of shared steps, to give your test cases some variety.

11. Exit **Microsoft Test Manager**.

12. Log off of **Windows**.

Sample

EXERCISE 4 – INSTALL AND USE TEST SCRIBE (OPTIONAL)

TASK – INSTALL AND USE TEST SCRIBE

Test Scribe is a documentation power tool designed to construct documents directly from the TFS for test plan and test run artifacts for the purpose of discussion, reporting etc. Test Scribe requires either the Test Professional or Visual Studio Ultimate edition to be installed. It is available for free from Visual Studio Gallery: http://bit.ly/hiqgtR.

In this task you will logon to Windows as Andy, install the Test Scribe utility, and use it to document your team's test planning effort.

1. Log on to Windows using **Andy** for the user name and **password** for the password.

2. Using **Windows Explorer**, navigate to **C:\Course\Software\Test Scribe**.

3. Double-click **OpenXMLSDKv2.msi** and install the software.

   This SDK is a prerequisite for the Test Scribe tool. You can accept the defaults when installing this SDK.

4. Double-click **TestScribe_x86.exe** and install the software.

5. From the **Start** menu select **All Programs** > **Microsoft Visual Studio 2010** > **Microsoft Test Manager 2010**.

6. Connect to the **Tailspin** team project in the **Tailspin** collection on the **vstfs** server.

7. Select the **Iteration 1** plan.

8. In the main dropdown at the top of the screen, next to **Plan**, change from **Testing Center** to **Tools**.

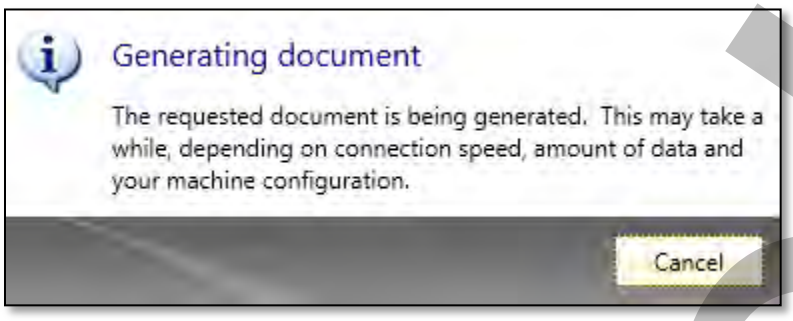
   The Test Scribe utility added the Tools center to Microsoft Test Manager.

9. Click the **Test Plan Summary** link.

Sample

10. Select **Iteration 1** and click **Generate** (Generate).

    It may take a moment to generate the documentation.



    When finished, Microsoft Word will launch and open the Test plan summary.docx.

    Take a moment and review this document.

    Do you see the two configurations listed? _____

    How many test cases are listed in the Bug Verification Tests suite? _____

    Are the shared steps resolved and displayed in the test case? _____

    Are the parameter values listed in the document? _____

11. Close **Microsoft Word**.

12. Return to **Microsoft Test Manager** and click the **Test Run Summary** link.

    From this section, you can select a particular test run in the connected test plan, right click and select "Generate" to create a document for the selected test run. The output document will contain the test run details with pie charts displaying test outcomes and test failure analysis.

    We don't have any test runs yet, but after the next lab you may return to this tool and generate a test run summary document.

13. Return to the **Testing Center**.

Sample

EXERCISE 5 – USE TEST CASE MIGRATOR PLUS (OPTIONAL)

TASK – USE TEST CASE MIGRATOR PLUS

Among other things, the Test Case Migrator Plus can import test cases from existing Excel spreadsheets and .mht formatted files. It is available for free from CodePlex: http://tcmimport.codeplex.com.

In this task you will use Test Case Migrator Plus to import several test cases currently residing in an Excel spreadsheet.

1. Right-click on **Iteration 1** test plan in the tree and select **New Suite**.

2. Name the test suite **Migrated Tests**.

3. Minimize **Microsoft Test Manager**.

4. Using **Windows Explorer**, navigate to **C:\Course\Software\Test Case Migrator Plus**.

5. Double-click **SampleTestCasesWorkbookWithRelationships.xlsx**.

   Review the contents of this spreadsheet. Notice that test cases with more than one test step will use multiple rows. Some teams like to put multiple test steps into a single cell, separated by new line characters. This utility supports that format as well.

   How many test cases are contained in this spreadsheet? _____

6. Exit **Microsoft Excel**.

7. Double-click **TestCaseMigratorPlus.exe**.

   This will launch Test Case Migrator Plus.

8. Click **Next**.

   What data source types are supported? _____

9. Select **Excel Workbook**.

10. Click the ellipses button ( ) and select **C:\Course\Software\Test Case Migrator Plus\SampleTestCases.xlsx**.

    The utility will open the spreadsheet and give you a summary of the number of worksheets found and the fields found on each worksheet, including the field names:

11. Click **Next**.

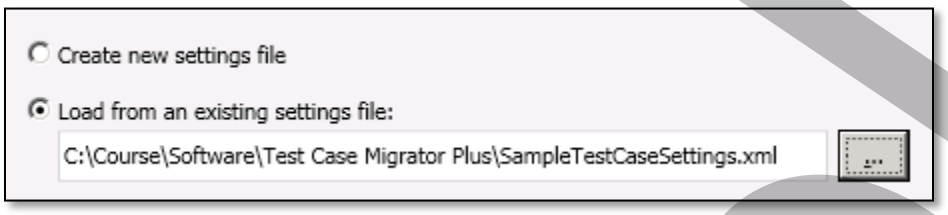
12. Click the ellipses button ( ).

Sample

13. Select the **vstfs** server, **Tailspin** collection, **Tailspin** team project and click **Select**.

14. Click **Next**.

15. Load an existing settings file: **SampleTestCaseSettings.xml**.



16. Click **Next**.

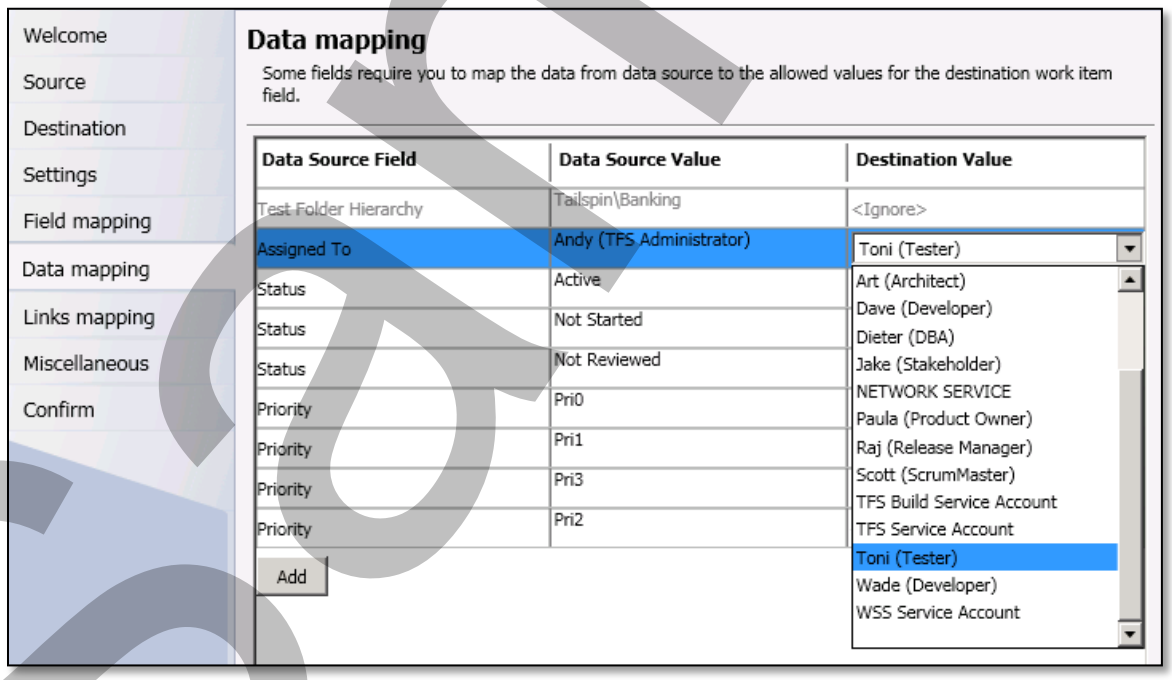
    At this point, the utility will parse through the test cases in the spreadsheet.

    Review the field mappings, but don't change any values. The settings file is matched to the sample data for this exercise.

17. Click **Next**.

    Review the data mappings.

18. Change the **Destination Value** of the **Assigned To** field to **Toni (Tester)**.

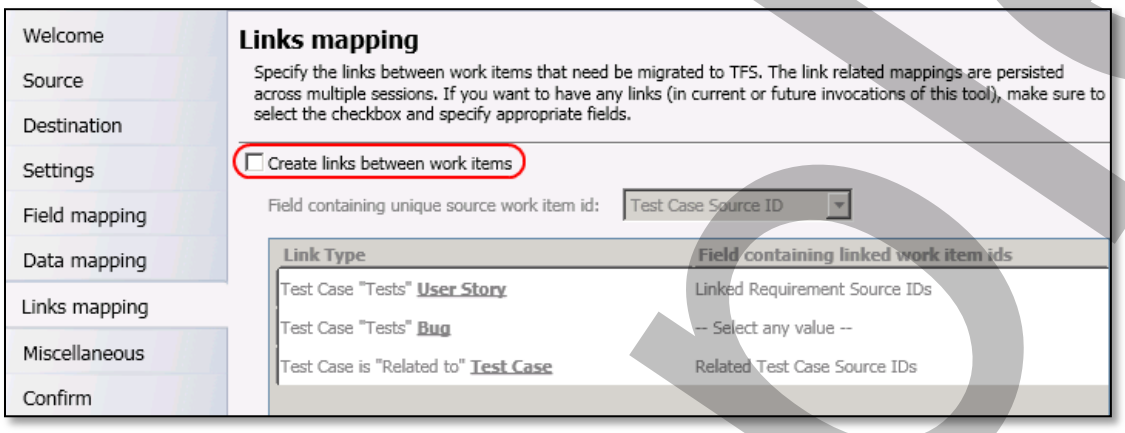


Sample

19. Click **Next**.

    Review the links mappings.

20. Clear the **Create links between work items** option.



21. Click **Next**.

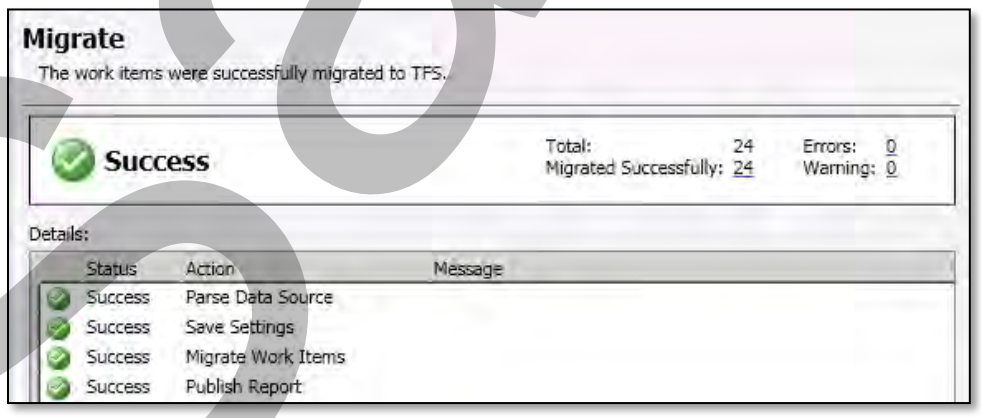
    Review the miscellaneous settings, but don't change any settings.

22. Click **Next**.

    This page shows a summary of all the choices made thus far. At this point you can go back and make changes, save and exit, save and migrate, copy the commandline to the clipboard, or cancel. Copying the command line to the clipboard is very useful when you need to script the migration of many different files of test cases.

23. Click **Save and migrate**.

    It can take a few moments to migrate the test cases into Team Foundation Server. When you are finished, you should see a summary similar to this. Notice the count of successful, errors, and warnings are hyperlinks. Clicking on a number takes you to a spreadsheet of information to help troubleshoot.



Sample

24. When the process finishes, click **Close**.

25. Launch **Microsoft Test Manager 2010**.

26. Select the **Migrated Tests** suite.

    You should see 24 new test cases in here.

    Note: These particular test cases don't map to the Tailspin Toys Web application. In other words, don't try to run the tests!

27. Double-click the **Account Details screen displays credit card information properly** test case.

    Who is the test case work item assigned to? _____

    What is the Area? _____

    Note: this area was automatically added to the team project during migration.

    How many test steps were migrated? _____

    Are there parameters defined for the test steps? _____

28. Click the **Summary** tab.

    Was a description migrated? _____

29. Exit **Microsoft Test Manager**.

30. Log off of **Windows**.

Sample

SUMMARY

Microsoft Test Manager helps you plan your testing effort, based on your specific approach. Microsoft Test Manager can create a test plan that is associated with your team project. In your test plan, you can define which tests you plan to run for a specific iteration. Also, you can create test configurations that specify the test set up that you want to use to run your tests. By defining what tests you want to run on which test configurations, you can now use the test plan to measure your progress as soon as you begin to run your tests.

You can create test cases for your test plan that can be used for both manual and automated tests. You can add all the information that you need in order to run a test manually by adding test steps with both actions and expected results. In addition, you can share common test steps with other tests to reduce the overhead of maintaining your manual test steps.

Sample