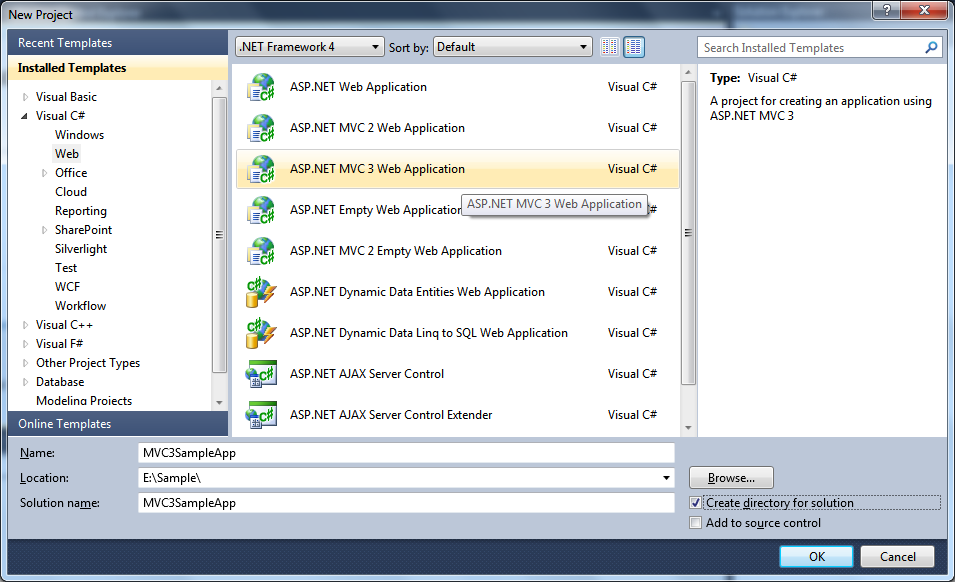
How to: Create Web Client UI Test using Coded UI Test

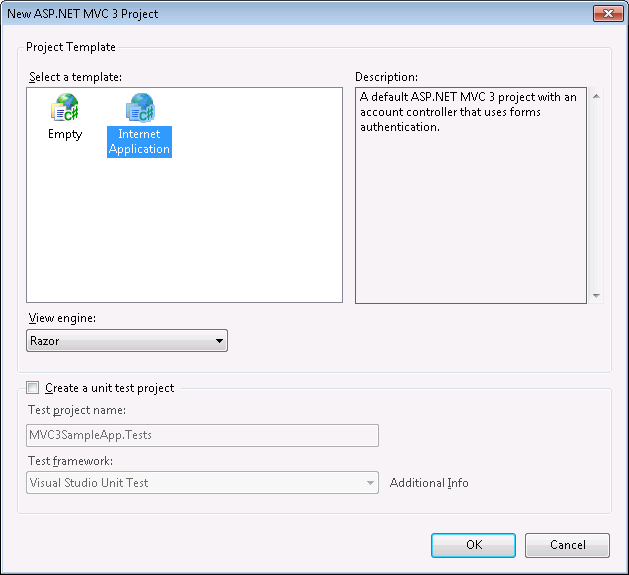
* 1. The following How-to topic will walk you through the creation of an automated test for your web application by creating a coded UI test using Visual Studio Premium or Visual Studio Ultimate. The coded UI test performs actions on the user interface (UI) controls and verifies that the correct controls are displayed with the correct values. For this topic, a default ASP.NET MVC 3 web application will be the targeted application used for testing.

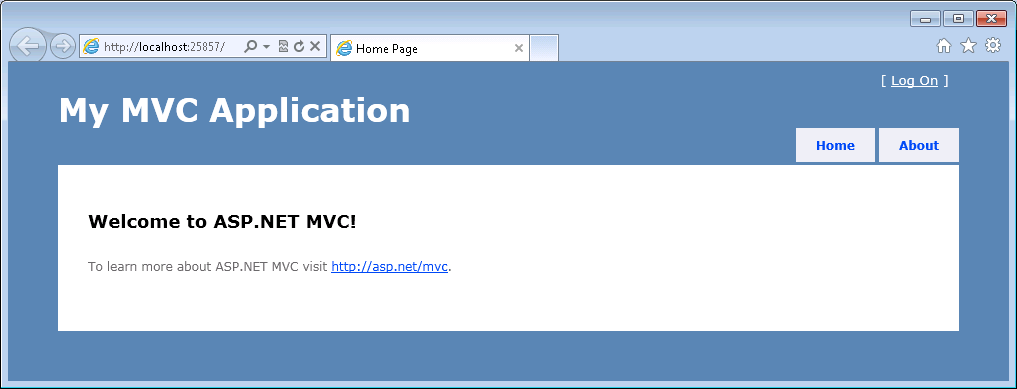
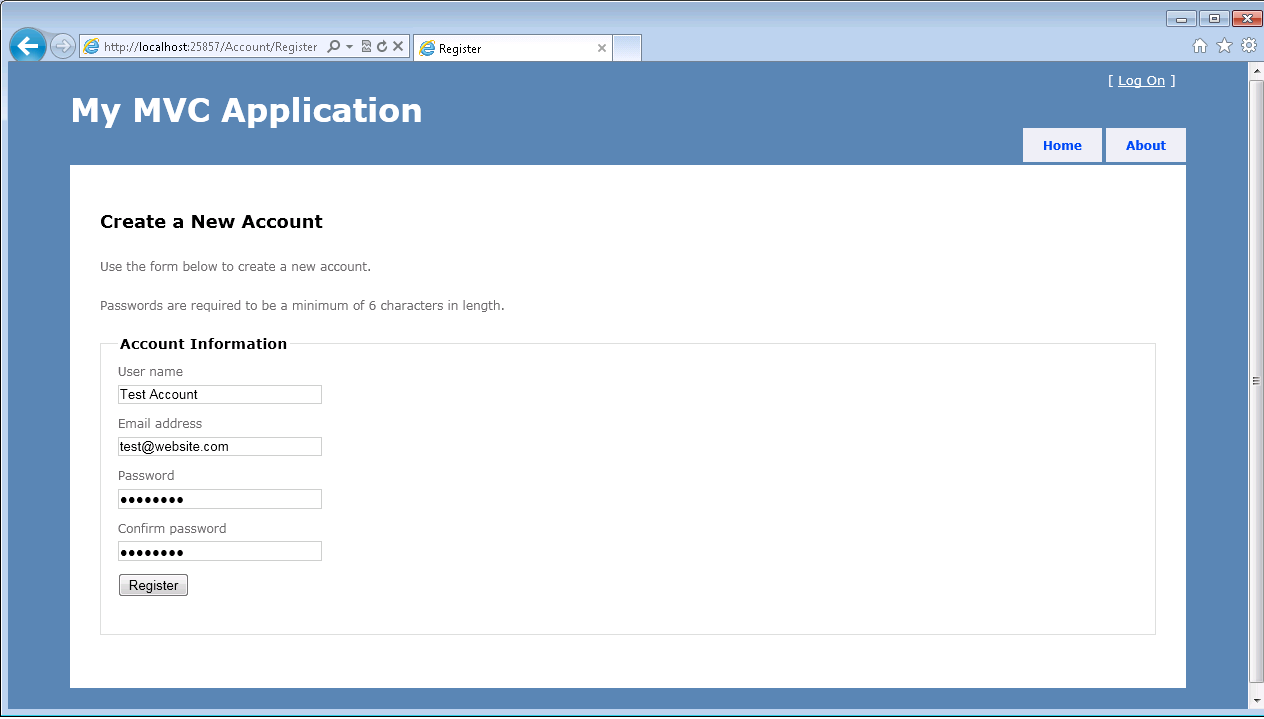
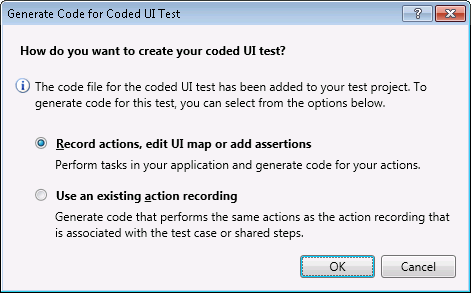
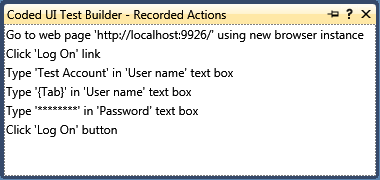
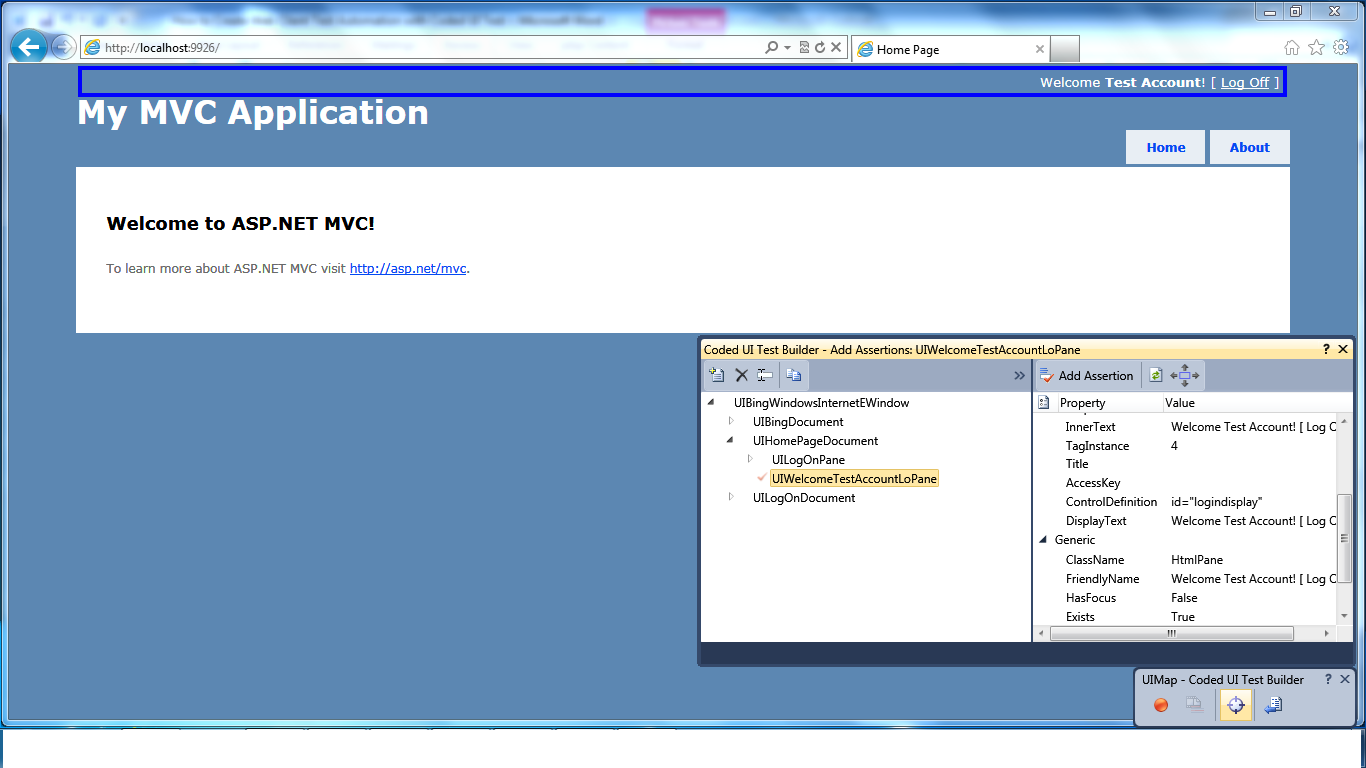
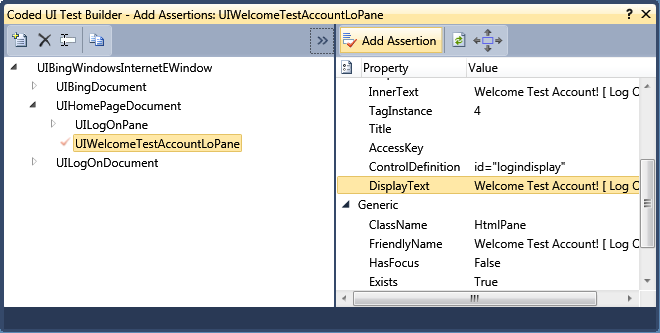
# Prerequisites

* 1. This topic requires you to have the following prerequisites:
  2. Microsoft Visual Studio 2010 Premium or Ultimate edition
  3. [Microsoft Visual Studio 2010 SP1](http://www.microsoft.com/downloads/en/details.aspx?FamilyID=75568aa6-8107-475d-948a-ef22627e57a5&displaylang=en)
  4. Microsoft Internet Explorer 9
  5. [ASP.NET MVC 3](http://www.asp.net/mvc/mvc3)

# Steps

* 1. In Visual Studio, create a new project. To do this, point to **New** on the File menu, and then click **Project**. In the **New Project** dialog, select **Visual C#** under **Installed Templates**, and then select **ASP.NET MVC 3 Web Application**. You can create applications using either Visual Basic or Visual C#. Name your project **MVC3SampleApp,** then click **OK.**
     1. 
     2. In the **New ASP.NET MVC 3 Project** dialog box, select **Internet Application**. Select **Razor** as the view engine. Click **OK**.



* + 1. Select **Start Debugging** from the **Debug** menu. Visual Studio will launch a browser and open the application's home page. Notice that the address bar of the browser says localhost. That's because localhost points to your own local computer, which is running the application you just built. When Visual Studio runs a web project, a random port is used for the web server. In the image below, the random port number is 25857. When you run the application, you'll probably see a different port number.
       1. 
    2. Click the **Log On** link on the Home page.
    3. Click the **Register** link and register an account by entering a **User name** and **Password**. Click the **Register** button.
       1. 
  1. Add a test project to the solution using the Test Project template. Right-click the **MVC3SampleApp** solution in Solution Explorer, point to **Add**, and select **New Project**. In the **New Project** dialog, select **Test Documents** under **Test Projects**. Set the project's name to **TestProject**, specify a valid location, and then click **OK**.
  2. Add a coded UI test. To do this, in Solution Explorer, right-click the **TestProject** project, point to **Add**, and select **New Test**. In the **Add New Test** dialog, select the **Coded UI Test**. Name the coded UI test **MyCodedUITest** and click **OK**.
  3. In the **Generate Code for Coded UI Test** dialog, select **Record actions, edit UI map or add assertions**, and click **OK**.
     1. 
     2. The **Coded UI Test Builder** dialog box appears. You can use the **Coded UI Test Builder** to add a UI control to the [UIMap](http://msdn.microsoft.com/en-us/library/microsoft.visualstudio.testtools.uitest.common.uimap.uimap.aspx) for your test, or to generate code for a validation method that uses an assertion for a UI control. Click the red button to start recording steps.
     3. C:\Users\Public\Pictures\Sample Pictures\How To\VerifyUIElements\Coded UI Test Builder.bmp
  4. Record the UI Test as follows:
     1. Navigate to the home page
     2. Click the **Log on** link on the top-right corner.
     3. Enter the **user name** and **password**.
     4. Click the **Log on** button.
  5. Click the record button to stop recording. Click the **Show Recorded Steps** button to check if the steps are recorded correctly. The result should look like the screenshot below:
     1. 
  6. Click the **Generate Code** button in UIMap - Coded UI Test Builder. In the **Coded UI Test Builder-Generate Code** dialog box, change the method name to **LogOn** and click the **Add and Generate** button.
  7. The code, which will be generated to the UIMap.Designer.cs file, is shown in the following code snippet.
     1. **Note:** If you record a new method, the generated code will override the code in UIMap.Designer.cs.
     2. C#
     3. [GeneratedCode("Coded UITest Builder", "10.0.40219.1")]
     4. public partial class UIMap
     5. {
     7. /// <summary>
     8. /// LogOn - Use 'LogOnParams' to pass parameters into this method.
     9. /// </summary>
     10. public void LogOn()
     11. {
     12. #region Variable Declarations
     13. HtmlHyperlink uILogOnHyperlink = this.UIBingWindowsInternetEWindow.UIHomePageDocument.UILogOnPane.UILogOnHyperlink;
     14. HtmlEdit uIUsernameEdit = this.UIBingWindowsInternetEWindow.UILogOnDocument.UIUsernameEdit;
     15. HtmlEdit uIPasswordEdit = this.UIBingWindowsInternetEWindow.UILogOnDocument.UIPasswordEdit;
     16. HtmlInputButton uILogOnButton = this.UIBingWindowsInternetEWindow.UILogOnDocument.UIMainPane.UILogOnButton;
     17. #endregion
     18. // Go to web page 'http://localhost:9926/' using new browser instance
     19. this.UIBingWindowsInternetEWindow.LaunchUrl(new System.Uri(this.LogOnParams.UIBingWindowsInternetEWindowUrl));
     20. // Click 'Log On' link
     21. Mouse.Click(uILogOnHyperlink, new Point(8, 11));
     22. // Type 'Test Account' in 'User name' text box
     23. uIUsernameEdit.Text = this.LogOnParams.UIUsernameEditText;
     24. // Type '{Tab}' in 'User name' text box
     25. Keyboard.SendKeys(uIUsernameEdit, this.LogOnParams.UIUsernameEditSendKeys, ModifierKeys.None);
     26. // Type '\*\*\*\*\*\*\*\*' in 'Password' text box
     27. uIPasswordEdit.Password = this.LogOnParams.UIPasswordEditPassword;
     28. // Click 'Log On' button
     29. Mouse.Click(uILogOnButton, new Point(17, 16));
     30. }
     31. ...
     32. ...
     34. }
  8. Use the **UIMap -** **Coded UI Test Builder** to create an assert method to validate properties of the UI control. In this case, you will verify if logging on to the default site was successful by following these steps:
     1. Add an assertion to the UI control. To do this, drag the crosshairs onto the UI control in your application that you want to test. When the box outlines your control, release the mouse. In this case, draw a box around the Logon portion of the home page.
        1. 
        2. The properties for this control are now listed in the **Coded UI Test Builder - Add Assertions** dialog box.
     2. Right-click the **Display Text** property and select **Add Assertion**. Keep all values as default and click **OK**.
        1. 
     3. Click the **Generate Code** button in the Coded UI Test Builder. Name the assertion method **VerifyLogOn**. This method will be auto generated and added to the **CodedUITestMethod1**  in the **MyCodedUITest**.cs file, as shown below:
     4. C#
     5. [TestMethod]
     6. public void CodedUITestMethod1()
     7. {
     8. // To generate code for this test, select "Generate Code for Coded UI   
         // Test" from the shortcut menu and select one of the menu items.
     9. // For more information on generated code, see   
         // http://go.microsoft.com/fwlink/?LinkId=179463
     10. this.UIMap.LogOn();
     11. this.UIMap.VerifyLogOn();
     12. }
  9. Modify the generated code as follows:
     1. Copy the code in UIMap.Designer.cs and paste it into UIMap.cs.
     2. In UIMap.cs, if it is not already present, add the following **using** statement:
        + 1. C#
          2. using Microsoft.VisualStudio.TestTools.UITesting.HtmlControls;
     3. If you want to close the browser window automatically after each test case runs, add a **CloseBrowserWindow** function in the UIMap.cs partial class, as follows:
        1. C#
        2. public partial class UIMap
        3. {
        5. ...
        6. ...
        7. public void CloseBrowserWindow()
        8. {
        9. #region Variable Declarations
        10. BrowserWindow currentBrowserWindow = this.mUIBlankPageWindowsInteWindow;
        11. #endregion
        12. currentBrowserWindow.Close();
        13. }
        15. ...
        17. }
     4. Add the following code snippet to the **MyCodedUITest** Class in **MyCodedUITest.cs** file. The **TestCleanup** attribute in this method marks this method to be executed every time a test method completes its run.
        1. C#
        2. //Use TestCleanup to run code after each test has run
        3. [TestCleanup()]
        4. public void MyTestCleanup()
        5. {
        6. // To generate code for this test, select "Generate Code for Coded UI Test" from the shortcut menu and select one of the menu items.
        7. // For more information on generated code, see http://go.microsoft.com/fwlink/?LinkId=179463
        8. this.UIMap.CloseBrowserWindow();
        9. }
  10. To run the test, close all browser windows. Right-click inside the **MyCodedUITest.cs** file and click **Run Tests**. The coded UI test will begin to execute; this will open a browser, will run the application programmatically based on the recorded steps, and will assert if the conditions are met. Once the test completes, the results will be shown in the Test Results window.

# Outcome

* 1. The Automation test project is created and can be used to automate UI testing of your web application.

# Further Reading

* + [Testing the User Interface with Automated UI Test](http://msdn.microsoft.com/en-us/library/dd286726.aspx) on MSDN.
  + [How to: Create a Coded UI Test](http://msdn.microsoft.com/en-us/library/dd286681.aspx) on MSDN.