Creating SharePoint Farm Solutions

Lab Time: 45 minutes

Lab Folder: C:\Student\Modules\02_FullTrustSolutions\Lab

Lab Overview: In this lab you will learn how to create a SharePoint farm solution. While Microsoft recommends the SharePoint Add-in model as the preferred way to extend SharePoint deployments both on-premises as well as in the cloud with Office 365, there are still a great many things that cannot be accomplished using add-ins. This is why farm solutions still play an important role in the product.

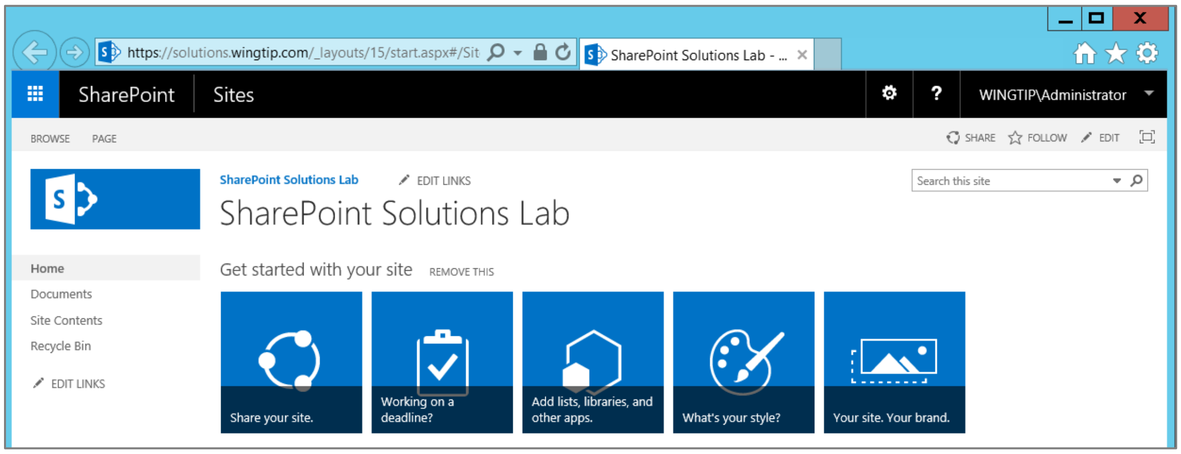
Exercise 1: Setup Lab Environment

In this exercise you will run a PowerShell script to create a new site collection at the URL of https://solutions.wingtip.com. Once you have created the new site collection, you ill then use it as a test site as you create and debug as farm solution.

1. Setup a new site collection for this lab:
   a) Ensure you are logged into the **WingtipServer** server as **WINGTIP\Administrator**.
   b) Run a PowerShell script, found in the root lab folder for this module:
      i) Right-click **SetupLab.ps1** and select **Run with PowerShell**. This file can be found in the files associated with this lab:

```
C:\Student\Modules\02_FullTrustSolutions\Lab
```

   c) When the script completes, it will launch a new browser and navigate to the lab site collection.



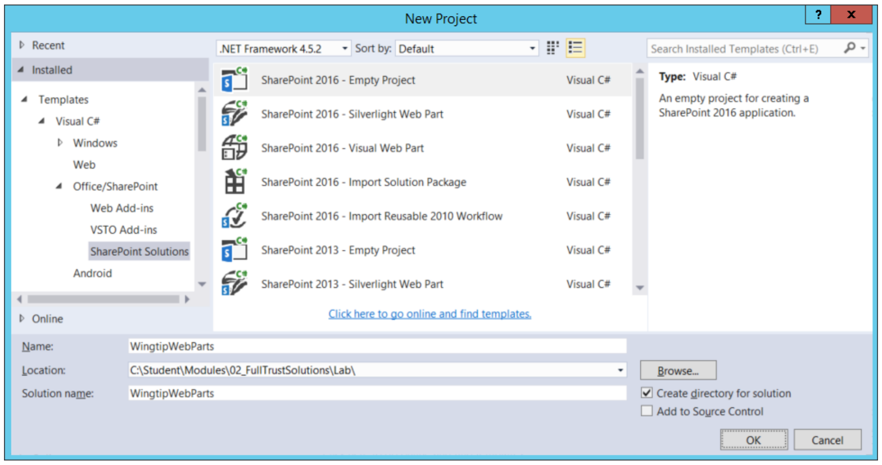
   d) Close the PowerShell console window.

Exercise 2: Creating Web Parts using Farm Solutions

In this exercise you will create a simple Web Part and add it to a custom Web Part group with a custom icon. This will be done using a farm solution.

1. Create a new project in Visual Studio:
   a) Launch **Visual Studio** as administrator:
      i) Alternatively, after pressing the **Windows Keyboard Key** you can simply start typing the name of the program you are looking for (e.g. Visual Studio); this will filter the results to those that match the letters typed on the keyboard)
   b) In Visual Studio select **File → New → Project**.
   c) In the **New Project** dialog:
      i) Find the **SharePoint 2016 - Empty Project** template under the following node:
         (1) **Templates → Visual C# → Office / SharePoint → SharePoint Solutions**
      ii) **Name**: WingtipWebParts
      iii) **Location:** C:\Student\Modules\02_FulltrustSolutions\Lab
      iv) **Uncheck** the **Create directory for solution** checkbox

v) Click **OK** to create the project


New Project
Recent
Installed
Templates
Visual C#
Windows
Web
Office/SharePoint
Web Add-ins
VSTO Add-ins
SharePoint Solutions
Android
Online
.NET Framework 4.5.2
Sort by: Default
Search Installed Templates (Ctrl+E)
SharePoint 2016 - Empty Project Visual C#
SharePoint 2016 - Silverlight Web Part Visual C#
SharePoint 2016 - Visual Web Part Visual C#
SharePoint 2016 - Import Solution Package Visual C#
SharePoint 2016 - Import Reusable 2010 Workflow Visual C#
SharePoint 2013 - Empty Project Visual C#
SharePoint 2013 - Silverlight Web Part Visual C#
Click here to go online and find templates.
Browse...
OK
Cancel

Type: Visual C#

An empty project for creating a SharePoint 2016 application.

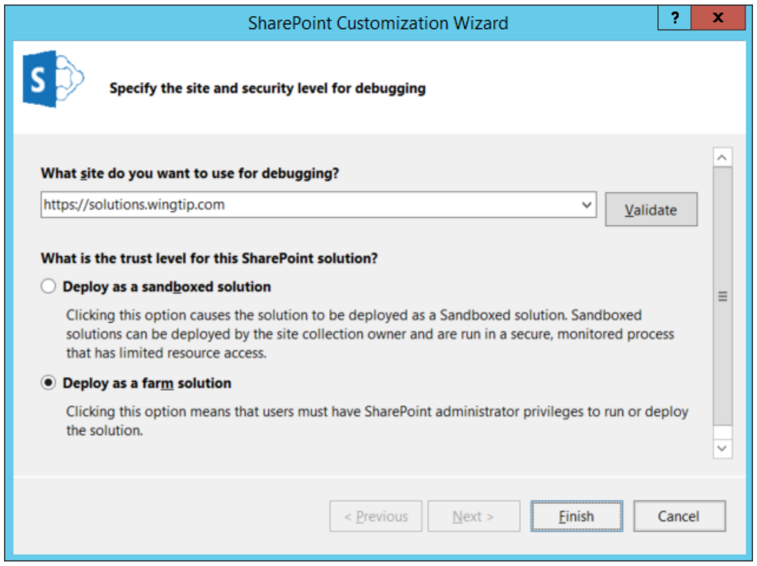
Name: WingtipWebParts
Location: C:\Student\Modules\02_FullTrustSolutions\Lab\
Solution name: WingtipWebParts
☑ Create directory for solution
☐ Add to Source Control

d) In the **SharePoint Customization Wizard**, use the following values to complete the wizard.

i) **What site do you want to use for debugging?** https://solutions.wingtip.com

ii) **What is the trust level for this SharePoint solution?** Deploy as farm solution

e) Click **Finish**



Specify the site and security level for debugging

What site do you want to use for debugging?

https://solutions.wingtip.com

What is the trust level for this SharePoint solution?

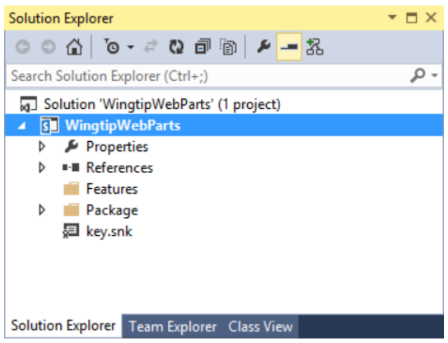
○ **Deploy as a sandboxed solution**

Clicking this option causes the solution to be deployed as a Sandboxed solution. Sandboxed solutions can be deployed by the site collection owner and are run in a secure, monitored process that has limited resource access.

● **Deploy as a farm solution**

Clicking this option means that users must have SharePoint administrator privileges to run or deploy the solution.

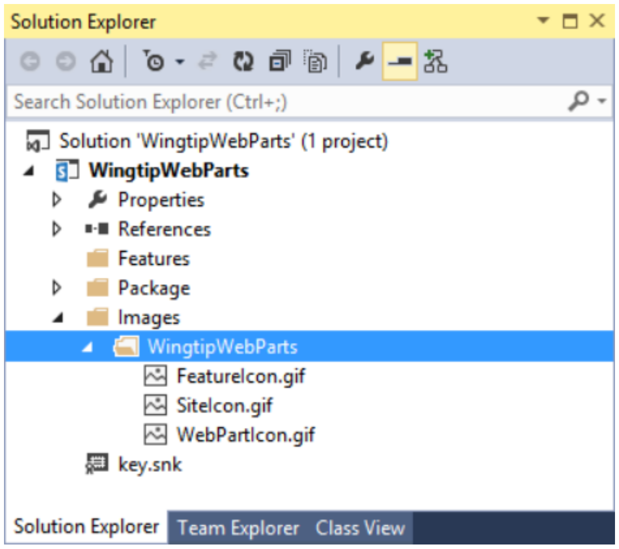
f) After the project has been created, examine its structure in the Solution Explorer.



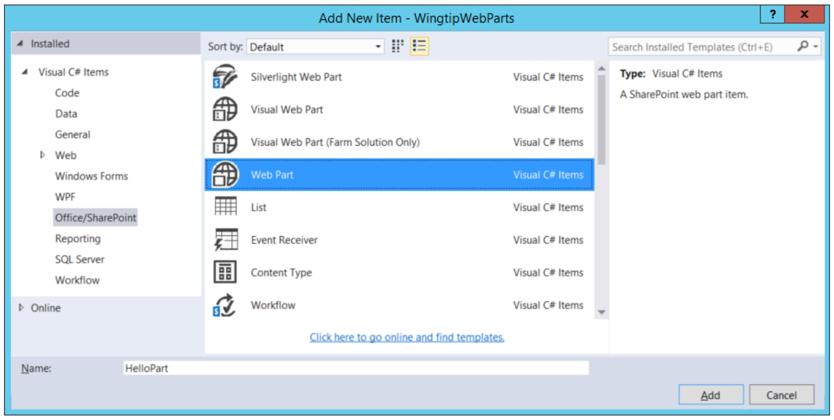
2. Add a few images into your project so they are deployed inside the SharePoint **Images** folder.
   a) Using the **Solution Explorer** tool window, right-click the **WingtipWebParts** project and select **Add → SharePoint "Images" Mapped Folder**.
   b) Right-click the **Images\WingtipWebParts** folder you just created and select **Add → Existing Item**.
   c) In the **Add Existing Item** dialog, navigate to the folder associated with this exercise:

```
C:\Student\Modules\02_FulltrustSolutions\Lab\StarterFiles
```

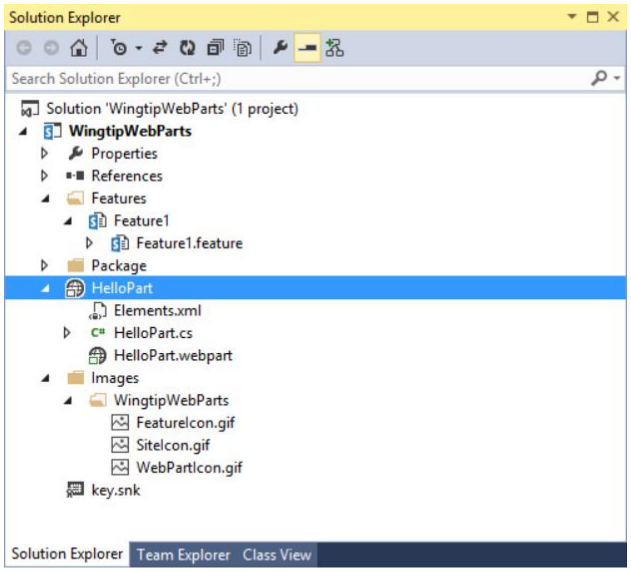
   d) Select the following files:
      i) FeatureIcon.gif
      ii) SiteIcon.gif
      iii) WebPartIcon.gif
   e) Click **Add**



3. Add a Web Part to the project:
   a) Using the **Solution Explorer** tool window, right-click the **WingtipWebParts** project and select **Add → New Item**.
   b) In the **Add New Item** dialog, select the **Web Part** template from the **Visual C# Items → Office/SharePoint** category.
      i) **Name:** HelloPart
   c) Click **Add**.


Add New Item - WingtipWebParts
Installed
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
Office/SharePoint
Reporting
SQL Server
Workflow
Online
Sort by: Default
Silverlight Web Part Visual C# Items
Visual Web Part Visual C# Items
Visual Web Part (Farm Solution Only) Visual C# Items
Web Part Visual C# Items
List Visual C# Items
Event Receiver Visual C# Items
Content Type Visual C# Items
Workflow Visual C# Items
Click here to go online and find templates.
Search Installed Templates (Ctrl+E)
Type: Visual C# Items
A SharePoint web part item.
Name: HelloPart
Add
Cancel

4. Inspect the SharePoint Project Item node for the Web Part named **HelloPart**. Notice it contains three files named **Elements.xml**, **HelloPart.cs** and **HelloPart.webpart**.



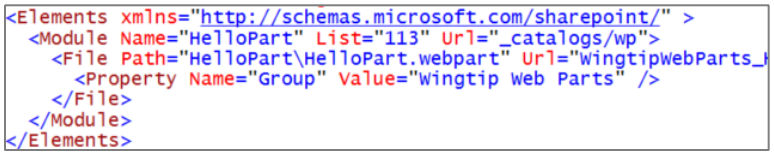
5. Right-click the **HelloPart \ HelloPart.webpart** file and select **Open**.
   a) Update the **<property>** node that has the attribute **name="Title"** and set the title to **The "Hello" Web Part**.
   b) Update the **<property>** node that has the attribute **name="Description"** and set the title to **A most compelling Web Part**.
6. The **HelloPart.webpart** file should look like the following code sample:

```
<?xml version="1.0" encoding="utf-8"?>
<webParts>
  <webPart xmlns="http://schemas.microsoft.com/WebPart/v3">
    <metaData>
      <type name="WingtipWebParts.HelloPart.HelloPart,
                  $SharePoint.Project.AssemblyFullName$" />
      <importErrorMessage>$Resources:core,ImportErrorMessage;</importErrorMessage>
    </metaData>
    <data>
      <properties>
        <property name="Title" type="string">The "Hello" Web Part</property>
        <property name="Description" type="string">A most compelling Web Part</property>
      </properties>
    </data>
  </webPart>
</webParts>
```

7. Add three more properties to the **HelloPart.webpart** file, just after the last property by adding the following XML:

```
<property name="ChromeType" type="string">TitleAndBorder</property>
<property name="CatalogIconImageUrl" type="string">_layouts/15/images/WingtipWebParts/WebPartIcon.gif</property>
<property name="TitleIconImageUrl" type="string">_layouts/15/images/WingtipWebParts/WebPartIcon.gif</property>
```

8. Now, modify the element manifest that will provision the Web Part definition:
   a) Using the **Solution Explorer** tool window, right-click the **HelloPart\Elements.xml** file and select **Open**.
   b) Modify the **<Property>** element with the Name="Group" attribute by setting the **Value** attribute to **Wingtip Web Parts**.



```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/" >
  <Module Name="HelloPart" List="113" Url="_catalogs/wp">
    <File Path="HelloPart\HelloPart.webpart" Url="WingtipWebParts_
      <Property Name="Group" Value="Wingtip Web Parts" />
    </File>
  </Module>
</Elements>
```

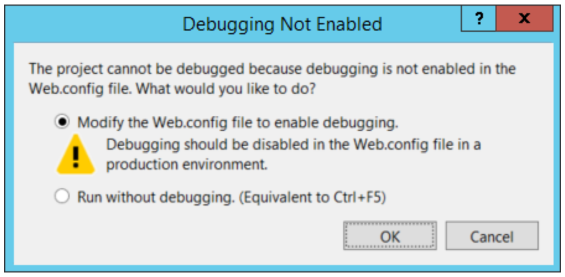
9. Finally, add some logic to the Web Part:
    a) Using the **Solution Explorer** tool window, right-click the **HelloPart \ HelloPart.cs** file and select **Open**.
    b) You should see that the the **CreateChildControls** method is currently empty.
    c) Add the following code to the body of the **CreateChildControls** method:

```
namespace WingtipWebParts.HelloPart {
  [ToolboxItemAttribute(false)]
  public class HelloPart : WebPart {
    protected override void CreateChildControls() {
      var label = new Label() { Text = "Hello Web Part" };
      this.Controls.Add(label);
    }
  }
}
```

10. Save all changes: **File → Save All**.

Build and Test the Project

11. Build and test your application by pressing **[F5]** or **Debug → Start Debugging**.
12. If you are prompted with the **Debugging Not Enabled** dialog, click **OK** to continue.

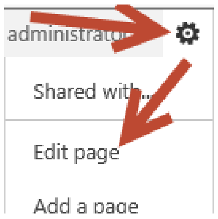

Debugging Not Enabled

The project cannot be debugged because debugging is not enabled in the Web.config file. What would you like to do?

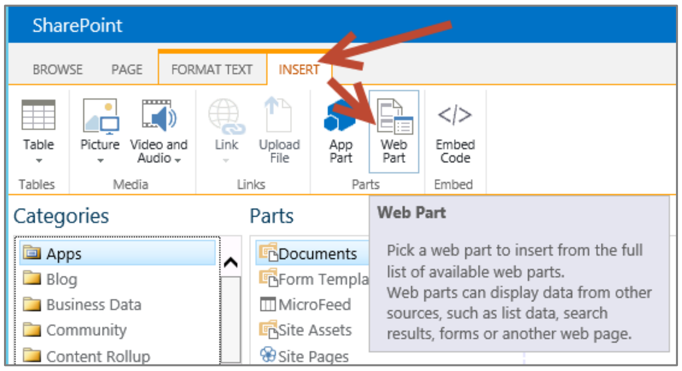
[✓ Modify the Web.config file to enable debugging.]
Debugging should be disabled in the Web.config file in a production environment.

Run without debugging. (Equivalent to Ctrl+F5)

13. Once the solution has been deployed, Internet Explorer will launch and navigate to the https://solutions.wingtip.com site.
14. Add the Web Part to the page:
    a) Using the **Site Actions** "gear" icon in the top-right corner, select **Edit Page**.



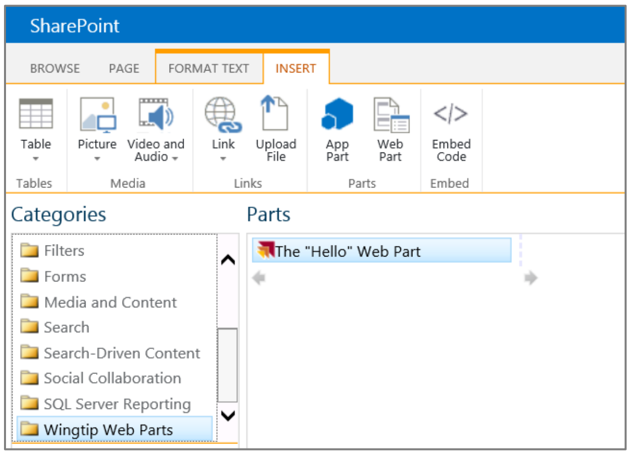
    b) Before you add your new web part to the page, remove the other three existing web parts from the page.
    c) Using the ribbon, select the **Insert** tab and click the **Web Part** button.



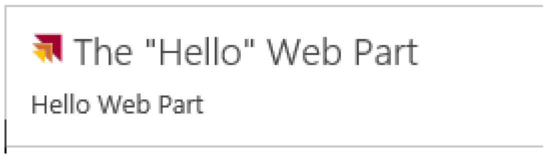
Web Part

Pick a web part to insert from the full list of available web parts.
Web parts can display data from other sources, such as list data, search results, forms or another web page.

d) Select the **The "Hello" Web Part** from the **Wingtip Web Parts** category and click the **Add** button.



e) At this point, you should see your new web part displayed on the page.



15. Close the browser to stop the debugger and go back to Visual Studio.

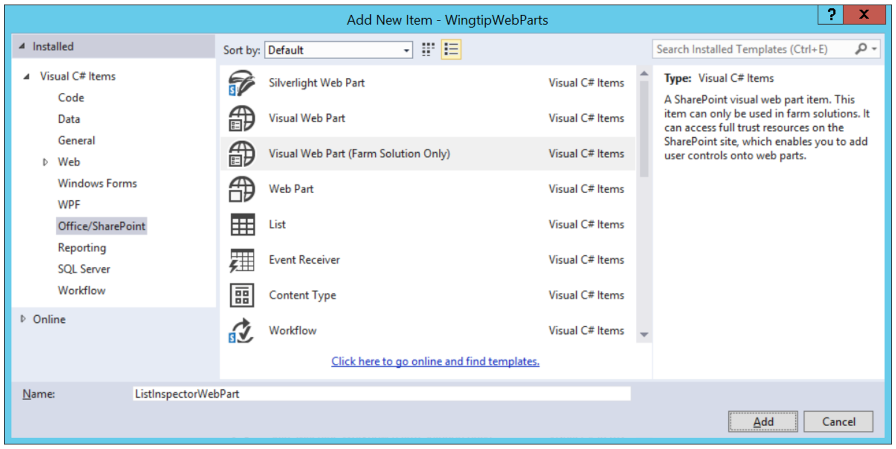
In this exercise you created a Web Part, added it to your project and added it to the sample SharePoint site.

Exercise 3: Creating a Visual Web Part with AJAX Behavior

Now you will add a second Web Part using the Visual Web Part template. This makes it possible to create the UI for a Web Part using an ASP.NET User Control and the Visual Studio User Control Designer. You will also use the `UpdatePanel` control from ASP.NET AJAX to give your Web Part a Web 2.0 user experience eliminating postbacks.

1. In this exercise, you should continue working with the **WingtipWebParts** project you created in the previous exercise.
2. Add a new Visual Web Part to the **WingtipWebParts** project named **ListInspectorWebPart**:
   a) Using the **Solution Explorer** tool window, right-click the **WingtipWebParts** project and select **Add → New Item**.
   b) In the **Add New Item** dialog, select the **Visual Web Part (Farm Solution Only)** template from the **Visual C# Items → Office/SharePoint** category.
      i) **Name:** ListInspectorWebPart

c) Click **Add**.

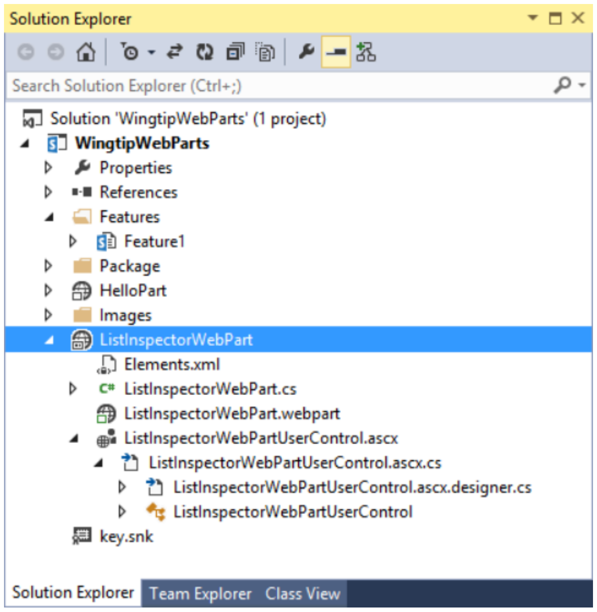

Add New Item - WingtipWebParts
Installed
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
Office/SharePoint
Reporting
SQL Server
Workflow
Online
Sort by: Default
Silverlight Web Part Visual C# Items
Visual Web Part Visual C# Items
Visual Web Part (Farm Solution Only) Visual C# Items
Web Part Visual C# Items
List Visual C# Items
Event Receiver Visual C# Items
Content Type Visual C# Items
Workflow Visual C# Items
Click here to go online and find templates.
Search Installed Templates (Ctrl+E)
Add
Cancel

Type: Visual C# Items

A SharePoint visual web part item. This item can only be used in farm solutions. It can access full trust resources on the SharePoint site, which enables you to add user controls onto web parts.

Name: ListInspectorWebPart

3. Inspect the SharePoint Project Item (SPI) node for the Web Part named **ListInspectorWebPart**. Notice it contains multiple files:


Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'WingtipWebParts' (1 project)
WingtipWebParts
Properties
References
Features
Feature1
Package
HelloPart
Images
ListInspectorWebPart
Elements.xml
ListInspectorWebPart.cs
ListInspectorWebPart.webpart
ListInspectorWebPartUserControl.ascx
ListInspectorWebPartUserControl.ascx.cs
ListInspectorWebPartUserControl.ascx.designer.cs
ListInspectorWebPartUserControl
key.snk
Solution Explorer
Team Explorer
Class View

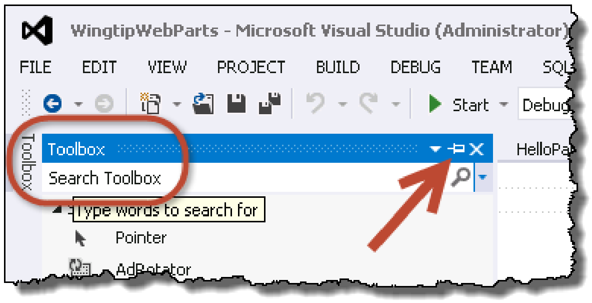
4. Right-click the **ListInspectorWebPart \ ListInspectorWebPart.webpart** file and select **Open**.
5. Update the two existing properties (**Title** & **Description**) and add additional properties, as listed below, to the **ListInspectorWebPart.webpart** file. The properties should look like this:

```
<property name="Title" type="string">List Inspector Web Part</property>
<property name="Description" type="string">
  A Web Part which shows all the lists in the current site and allows you to get several of its property values.
</property>
<property name="ChromeType" type="string">TitleAndBorder</property>
<property name="CatalogIconImageUrl" type="string">_layouts/16/images/WingtipWebParts/WebPartIcon.gif</property>
<property name="TitleIconImageUrl" type="string">_layouts/16/images/WingtipWebParts/WebPartIcon.gif</property>
```

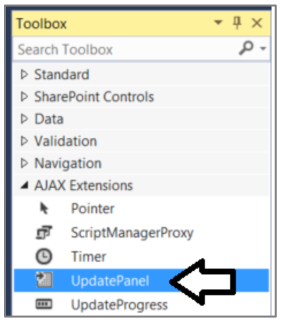
6. Now, modify the element manifest that will provision the Web Part definition:
   a) Using the **Solution Explorer** tool window, right-click the **ListInspectorWebPart \ Elements.xml** file and select **Open**.
   b) Modify the **<Property>** element with the Name=”Group” attribute by setting the **Value** attribute to **Wingtip Web Parts**.
7. Update the user interface portion of the Web Part:

a) Using the **Solution E**xplorer tool window, right-click the **ListInspectorWebPart \ ListInspectorWebPart.ascx** file and select **View Designer**.

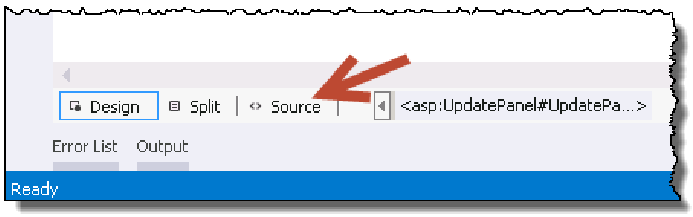
b) If the **Toolbox** tool window is not open, click it and then click the pushpin to pin it open:


WingtipWebParts - Microsoft Visual Studio (Administrator)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL
Start Debug
Toolbox
Toolbox
Search Toolbox
HelloPa
Type words to search for
Pointer
AdRotator

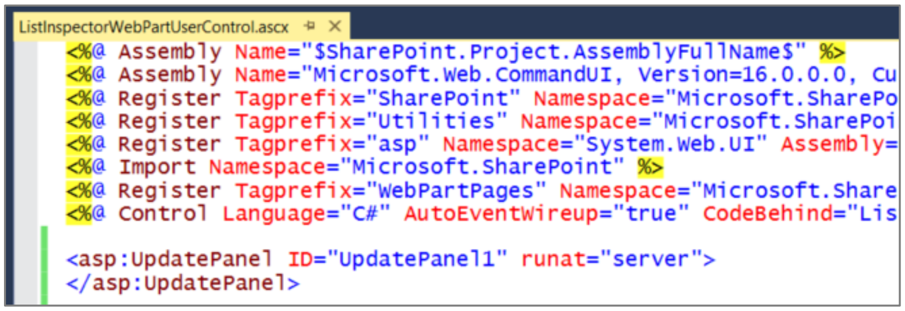
c) Using the **Toolbox** tool window, find the **UpdatePanel** control in the **AJAX Extensions** grouping. Drag the **UpdatePanel** onto the design surface.



d) After adding the UpdatePanel control, use the buttons at the bottom of the designer, click **Source** to switch to the **Source** view:



e) Once you switch over to **Source** view, you should be able to see the **<asp:UpdatePanel>** tag.

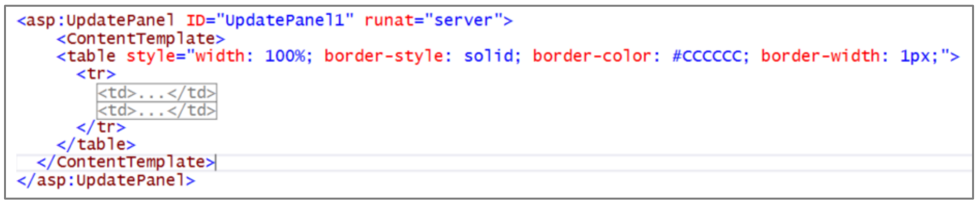


```
<%@ Assembly Name="$SharePoint.Project.AssemblyFullName$" %>
<%@ Assembly Name="Microsoft.Web.CommandUI, Version=16.0.0.0, Cu
<%@ Register Tagprefix="SharePoint" Namespace="Microsoft.SharePo
<%@ Register Tagprefix="Utilities" Namespace="Microsoft.SharePoi
<%@ Register Tagprefix="asp" Namespace="System.Web.UI" Assembly=
<%@ Import Namespace="Microsoft.SharePoint" %>
<%@ Register Tagprefix="WebPartPages" Namespace="Microsoft.Share
<%@ Control Language="C#" AutoEventWireup="true" CodeBehind="Lis

<asp:UpdatePanel ID="UpdatePanel1" runat="server">
</asp:UpdatePanel>
```

f) Add a new HTML table inside the **<asp:UpdatePanel>** element on the page. You can find the HTML content to create this table in the following file in the folder associated with this exercise:

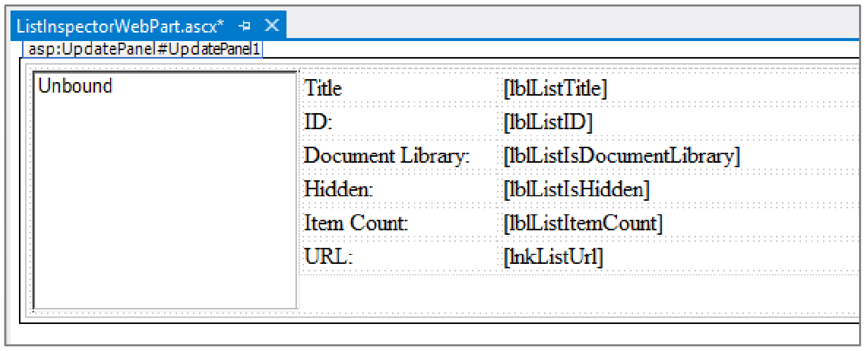
```
C:\Student\Modules\02_FulltrustSolutions\Lab\StarterFiles\ListInspectorTable.txt
```

g) When you are done, the code in your screen should match the code in the following screenshot.



```
<asp:UpdatePanel ID="UpdatePanel1" runat="server">
    <ContentTemplate>
    <table style="width: 100%; border-style: solid; border-color: #CCCCCC; border-width: 1px;">
      <tr>
        <td>...</td>
        <td>...</td>
      </tr>
    </table>
  </ContentTemplate>
</asp:UpdatePanel>
```

8. Using the buttons at the bottom of the designer, click **Design** to switch to the **Design** view: (the ListInspectorWebPart.ascx design view should appear as below:



Title	[lblListTitle]
ID:	[lblListID]
Document Library:	[lblListIsDocumentLibrary]
Hidden:	[lblListIsHidden]
Item Count:	[lblListItemCount]
URL:	[lnkListUrl]

9. Now code the Visual Web Part to make it do something:

a) Using the **Solution Explorer** tool window, right-click the **ListInspectorWebPart \ ListInspectorWebPart.ascx \ ListInspectorWebPart.ascx.cs** file and select **Open**.

b) Add the following to the top of the file:

```
using Microsoft.SharePoint;
```

c) Create two protected fields by adding the following code just inside the class:

```
protected Guid SelectedListId = Guid.Empty;
protected bool UpdateListProperties = false;
```

d) Add a method named **lstLists_SelectedIndexChanged()** by adding the following to the class. This event handler is already referenced in the markup you added to the **ListInspectorWebPart.ascx** file:

```
protected void lstLists_SelectedIndexChanged(object sender, EventArgs e)
{
  SelectedListId = new Guid(lstLists.SelectedValue);
  UpdateListProperties = true;
}
```

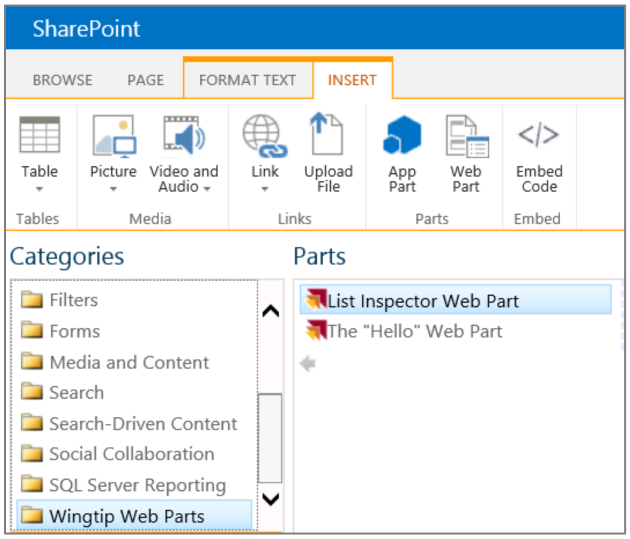
e) Add an overridden implementation of the **OnPreRender()** method using the parameter list shown in the following code block.

```
protected override void OnPreRender(EventArgs e) {

  if ((lstLists.SelectedIndex > -1) & (!UpdateListProperties)) {
    SelectedListId = new Guid(lstLists.SelectedValue);
  }

  lstLists.Items.Clear();
  SPWeb site = SPContext.Current.Web;
  foreach (SPList list in site.Lists)   {
    ListItem listItem = new ListItem(list.Title, list.ID.ToString());
    lstLists.Items.Add(listItem);
  }

  // when the page reloads, default the selected item to the current list
  if (SelectedListId != Guid.Empty) {
    lstLists.Items.FindByValue(SelectedListId.ToString()).Selected = true;
  }

  if (UpdateListProperties) {
    SPList list = SPContext.Current.Web.Lists[SelectedListId];
    lblListTitle.Text = list.Title;
    lblListID.Text = list.ID.ToString().ToUpper();
    lblListIsDocumentLibrary.Text = (list is SPDocumentLibrary).ToString();
    lblListIsHidden.Text = list.Hidden.ToString();
    lblListItemCount.Text = list.ItemCount.ToString();
    lnkListUrl.Text = list.DefaultViewUrl;
    lnkListUrl.NavigateUrl = list.DefaultViewUrl;
  }

}
```

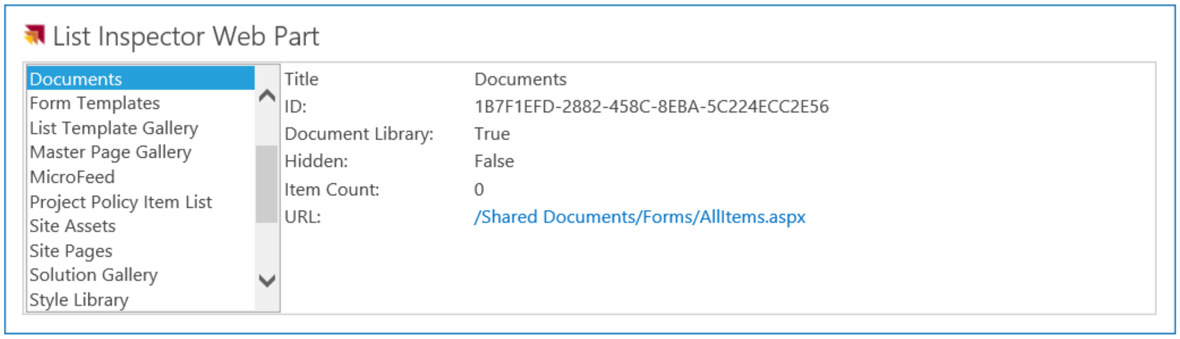
10. Save all changes: **File → Save All**.

Build and Test the Project

11. Build and test your application by pressing **[F5]** or **Debug → Start Debugging**.
12. Once the solution has been deployed, Internet Explorer will launch and navigate to the https://solutions.wingtip.com site.
13. Add the Web Part to the page:
    a) Using the **Site Actions** "gear" icon in the top-right corner, select **Edit Page**.
    b) Using the ribbon, select the **Insert** tab and click the **Web Part** button.
    c) Select the **List Inspector Web Part** from the **Wingtip Web Parts** category and click the **Add** button.



14. Click different lists to see the values refresh to show the respective properties.



List Inspector Web Part

Documents
Form Templates
List Template Gallery
Master Page Gallery
MicroFeed
Project Policy Item List
Site Assets
Site Pages
Solution Gallery
Style Library

Title	Documents
ID:	1B7F1EFD-2882-458C-8EBA-5C224ECC2E56
Document Library:	True
Hidden:	False
Item Count:	0
URL:	/Shared Documents/Forms/AllItems.aspx

15. Close the browser to stop the debugger and go back to Visual Studio.

In this exercise you created a new Visual Web Part and deployed it as a farm solution.

Exercise 4: Add Site Pages to a SharePoint Solution

In this exercise you will add some site pages to an existing solution and create links to these new pages.

1. Open an existing starter project in Visual Studio:
   a) Launch **Visual Studio** as administrator.
   b) Select **File → Open → Project/Solution**.
   c) In the **Open Project** dialog, select the following project provided in the files associated with this lab:

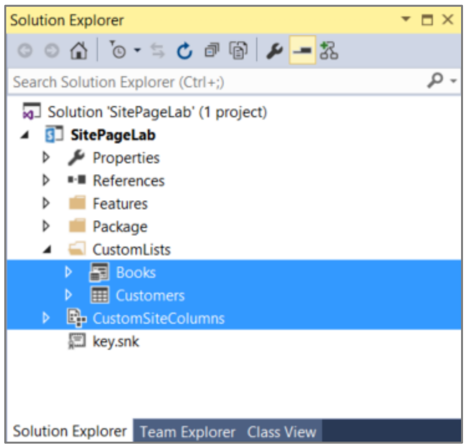
```
C:\Student\Modules\02_FullTrustSolutions\Lab\StarterProjects\SitePageLab\SitePageLab.sln
```

Note: You may get an error from Visual Studio about not having a site URL or debugging site specified. Ignore it for now.

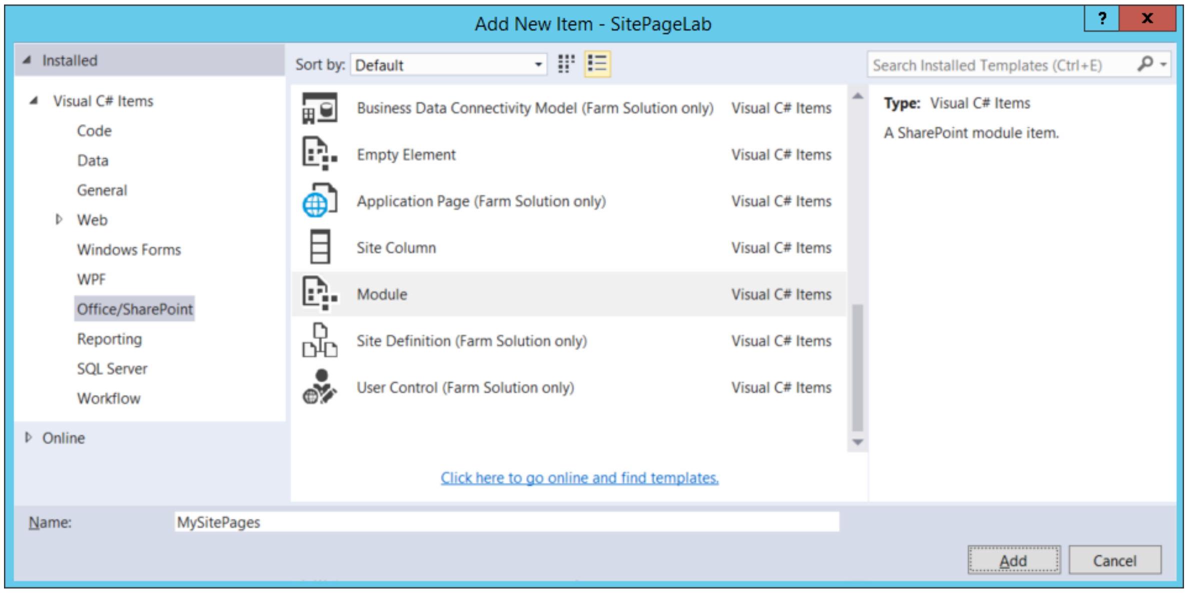
2. Ensure the project is using the lab's site collection for testing:
   a) Select the project **SitePageLab** in the **Solution Explorer** tool window.
   b) Look in the **Properties** tool window and ensure the **Site URL** property is set to https://solutions.wingtip.com.
   c) Make sure the **Sandbox Solution** property of the **SitePageLab** project is set to a value of **False**.

Remember that the VM you are using to complete lab exercises does not support server-side code from a sandboxed solution.

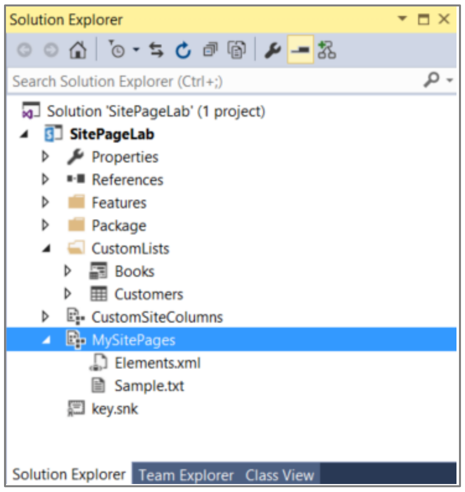
3. The **SitePageLab** project already contains SharePoint Project Items to create a few site columns and two lists. These two lists, **Books** and **Customers**, will be used throughout the remainder of this lab.



4. Create a new module to provision some site pages to the site:
   a) Right-click the project **SitePageLab** in the **Solution Explorer** tool window and select **Add → New Item**.
      i) In the **Add New Item** dialog, select the **Module** template from the **Visual C# Items → Office / SharePoint** category.
      ii) **Name:** MySitePages
      iii) Click **Add.**


Add New Item - SitePageLab
Installed
Visual C# Items
Code
Data
General
Web
Windows Forms
WPF
Office/SharePoint
Reporting
SQL Server
Workflow
Online
Sort by: Default
Search Installed Templates (Ctrl+E)
Business Data Connectivity Model (Farm Solution only) Visual C# Items
Empty Element Visual C# Items
Application Page (Farm Solution only) Visual C# Items
Site Column Visual C# Items
Module Visual C# Items
Site Definition (Farm Solution only) Visual C# Items
User Control (Farm Solution only) Visual C# Items
Type: Visual C# Items
A SharePoint module item.
Click here to go online and find templates.
Name: MySitePages
Add
Cancel

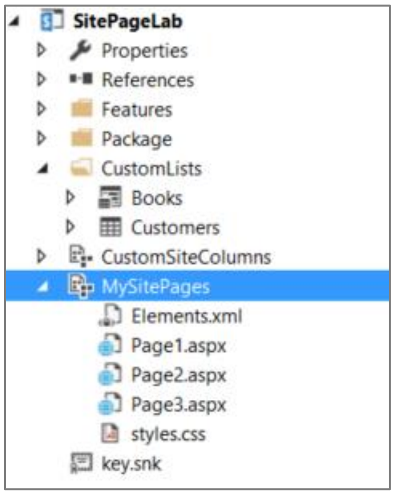
   b) Your project should look like the one in the following screenshot.



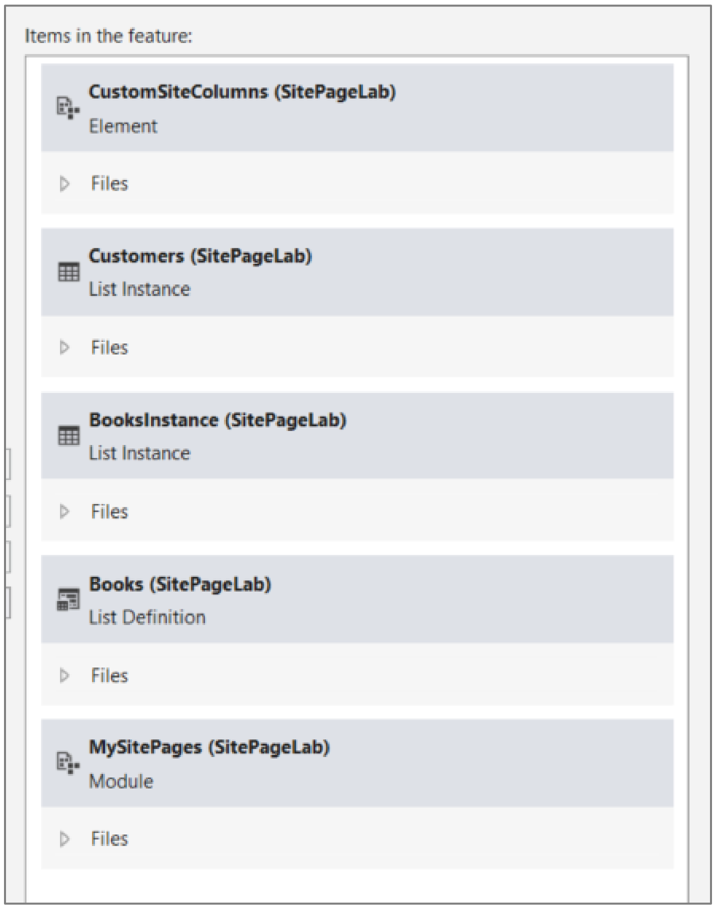
5. Add the provided site pages to the **MySitePages** module:
   a) Within the **MySitePages** module, right click **Sample.txt** and select **Delete**.
   b) Right-click the **MySitePages** module in the **Solution Explorer** tool window and select **Add → Existing Item**.
   c) In the **Add Existing Item** dialog, add the files **Page1.aspx**, **Page2.aspx**, **Page3.aspx** and **styles.css** from the following folder:

```
C:\Student\Modules\02_FullTrustSolutions\StarterFiles
```

   d) When you are done, your project should match the following screenshot.



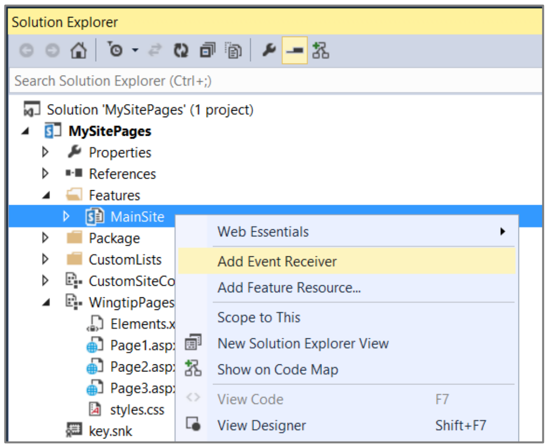
6. Verify the **MySitePages** module is included in the list of items in the Feature:
   a) In the **Solution Explorer** tool window, expand the **SitePageLab** project to the **Features\MainSite** node.
   b) Right-click the **MainSite** Feature and select **View Designer**.
   c) Verify the **MySitePages** module is listed on the right-hand column **Items in the Feature**. If it isn't select it and use the arrows between the two columns to add it from the left column: **Items in the Solution**.



Add Navigation Elements to the Site to Surface the New Site Pages

At this point your project will simply create a few pages in the development site. Users would have to know the URL and manually type it into the browser to navigate to these files. To make this more user friendly, write some custom code to add some navigation elements to the site to point to these three new files.

7. In the **Solution Explorer** tool window, expand the **SitePageLab** project to the **Features\MainSite** node.
8. Right-click the **MainSite** Feature and select **Add Event Receiver**.



9. Within the **MainSite.EventReceiver.cs** code file, add the following line to the top of the file to add a reference to the SharePoint navigation namespace:

```
using Microsoft.SharePoint.Navigation;
```

10. Find the commented out method **FeatureActivated()** and replace it with the following code to create three new links in the top navigation link bar:

```
public override void FeatureActivated(SPFeatureReceiverProperties properties) {

  SPSite siteCollection = properties.Feature.Parent as SPSite;
  if (siteCollection != null) {
    SPWeb site = siteCollection.RootWeb;
    // create menu items on top link bar for custom site pages
    SPNavigationNodeCollection topNav = site.Navigation.TopNavigationBar;
    topNav.AddAsLast(new SPNavigationNode("Page 1", "MySitePages/Page1.aspx"));
    topNav.AddAsLast(new SPNavigationNode("Page 2", "MySitePages/Page2.aspx"));
    topNav.AddAsLast(new SPNavigationNode("Page 3", "MySitePages/Page3.aspx"));
  }
}
```

11. While the previous code will create three new links when the Feature is activated, these links are not automatically removed when you deactivate the Feature. In addition, the provisioned files are not automatically removed. To clean up everything done by the Feature activation, find the commented out method **FeatureDeactivating()** and replace it with the following code to delete the links and provisioned pages:

```
public override void FeatureDeactivating(SPFeatureReceiverProperties properties) {
  SPSite siteCollection = properties.Feature.Parent as SPSite;
  if (siteCollection != null) {
    SPWeb site = siteCollection.RootWeb;

    try {
      // delete folder of site pages provisioned during activation
      SPFolder sitePagesFolder = site.GetFolder("MySitePages");
      sitePagesFolder.Delete();
    }
    catch { }

    SPNavigationNodeCollection topNav = site.Navigation.TopNavigationBar;
    for (int i = topNav.Count - 1; i >= 0; i--) {
      if (topNav[i].Url.Contains("MySitePages")) {
        // delete node
        topNav[i].Delete();
      }
    }
  }
}
```

Update the User Interface of the Provisioned Site Page Page1.aspx

Page1.aspx has references to CSS classes to create a more specialized user interface. In order to get this styling, you need to link to a custom CSS file.

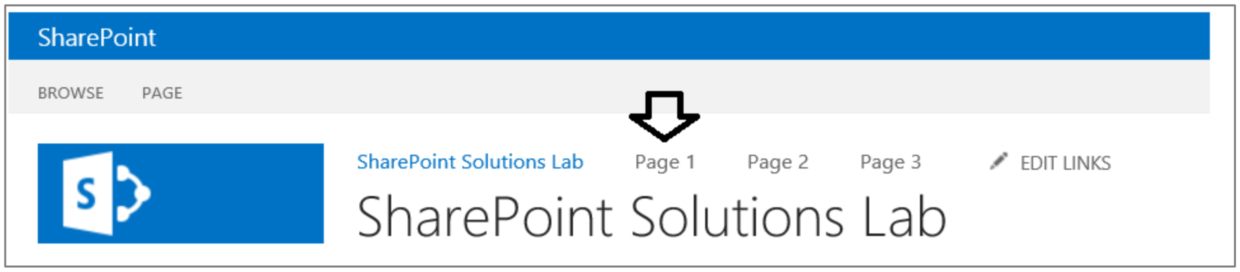
12. Add a reference to the CSS file in **Page1.aspx**:
    a) In the **Solution Explorer** tool window, find the **MySitePages\Page1.aspx** file.
    b) Right-click **Page1.aspx** and select **Open**.
    c) Find the ASP.NET content placeholder **PlaceHolderAdditionalPageHead** and add the following markup to the content of the content placeholder:

```
<link href="styles.css" rel="stylesheet" type="text/css" />
```

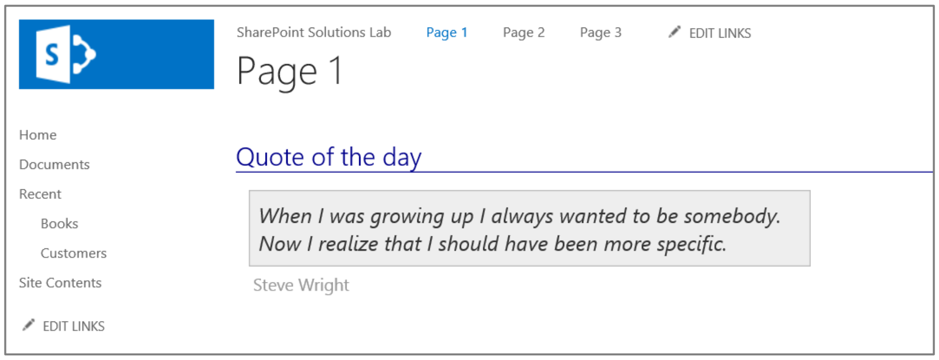
13. Save all changes: **File → Save All**.
14. Build and test your application by pressing **[F5]** or **Debug → Start Debugging**.

Note: If prompted with an **Attach Security Warning** dialog, click **Attach**.

15. Once the solution has been deployed, Internet Explorer will launch and navigate to the default page for the site.
16. In the top navigation, select **Page 1** to navigate to the **Page1.aspx** site page you provisioned.



17. Notice how **Page1.aspx** has unique styling compared to just plain old text:



Quote of the day

When I was growing up I always wanted to be somebody. Now I realize that I should have been more specific.

Steve Wright

18. Close the browser to stop the debugger and go back to Visual Studio.

Update the User Interface of the Provisioned Site Pages Page2.aspx and Page3.aspx

Page2.aspx is a Web Part Page but does not include any Web Parts. In this section you will add a new instance of the XsltListViewWebPart to display the Customer list. For Page3.aspx, which is also a Web Part Page, you will also add a XsltListViewWebPart to show the contents of the Books list.

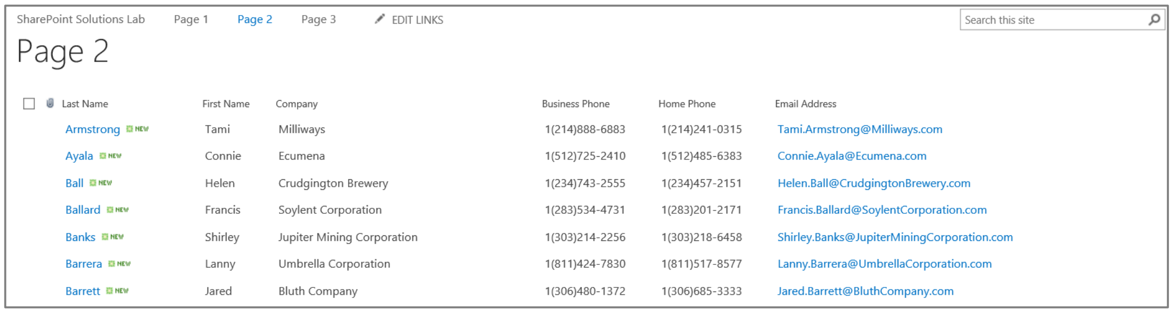
19. Add a Web Part to **Page2.aspx**:
    a) In the **Solution Explorer** tool window, find the **MySitePages\Page2.aspx** file.
    b) Right-click **Page2.aspx** and select **Open**.
    c) Find the ASP.NET content placeholder **PlaceHolderMain** and add the following markup to the content of the content placeholder:

```
<WebPartPages:WebPartZone ID="Main" Title="Main Web Part Zone"
                          FrameType="TitleBarOnly" runat="server" >
  <ZoneTemplate>
    <WebPartPages:XsltListViewWebPart
      runat="server"
      ID="CustomersWebPart"
      Title="Customers"
      ListUrl="Lists/Customers"
      ChromeType="None"
      InplaceSearchEnabled="false" >
    </WebPartPages:XsltListViewWebPart>
  </ZoneTemplate>
</WebPartPages:WebPartZone>
```

20. Add a Web Part to **Page3.aspx**:
    a) In the Solution Explorer tool window, find the **MySitePages\Page3.aspx** file.
    b) Right-click **Page3.aspx** and select **Open**.
    c) Find the ASP.NET content placeholder **PlaceHolderMain** and add the following markup to the content of the content placeholder:

```
<WebPartPages:WebPartZone ID="Main" Title="Main Web Part Zone"
                          FrameType="TitleBarOnly" runat="server" >
  <ZoneTemplate>
    <WebPartPages:XsltListViewWebPart
      runat="server"
      ID="BooksWebPart"
      Title="Books"
      ListUrl="Lists/Books"
      ChromeType="None"
      InplaceSearchEnabled="false" >
    </WebPartPages:XsltListViewWebPart>
  </ZoneTemplate>
</WebPartPages:WebPartZone>
```

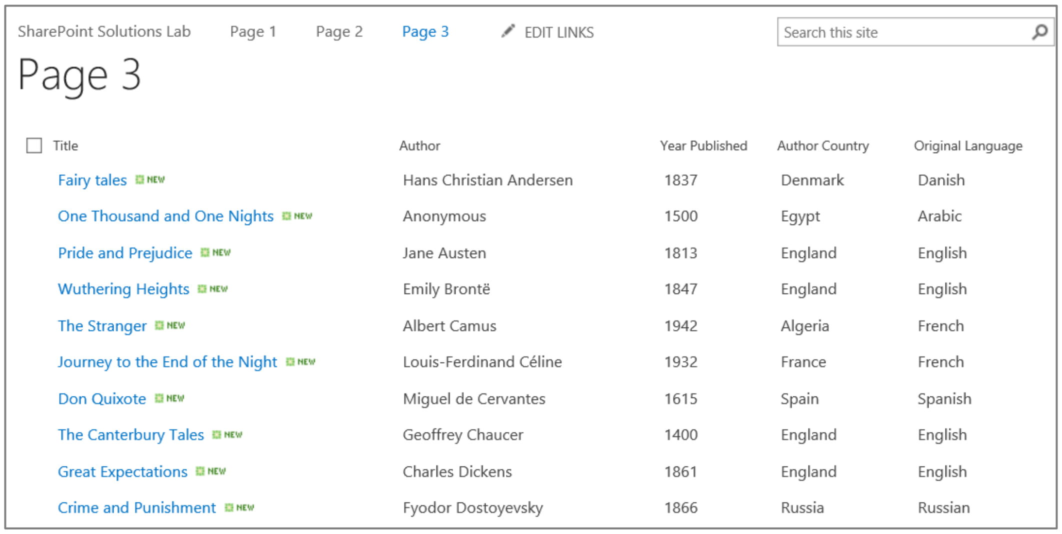
21. Save all changes: **File → Save All**.
22. Build and test your application by pressing **[F5]** or **Debug → Start Debugging**.
    a) If prompted with an **Attach Security Warning** dialog, click **Attach**.
23. When the site loads in Internet Explorer, select **Page 2** to navigate to the **Page2.aspx** site page you provisioned. Notice the **Customers** list is displayed in a typical format.



Page 2

Last Name	First Name	Company	Business Phone	Home Phone	Email Address
Armstrong NEW	Tami	Milliways	1(214)888-6883	1(214)241-0315	Tami.Armstrong@Milliways.com
Ayala NEW	Connie	Ecumena	1(512)725-2410	1(512)485-6383	Connie.Ayala@Ecumena.com
Ball NEW	Helen	Crudgington Brewery	1(234)743-2555	1(234)457-2151	Helen.Ball@CrudgingtonBrewery.com
Ballard NEW	Francis	Soylent Corporation	1(283)534-4731	1(283)201-2171	Francis.Ballard@SoylentCorporation.com
Banks NEW	Shirley	Jupiter Mining Corporation	1(303)214-2256	1(303)218-6458	Shirley.Banks@JupiterMiningCorporation.com
Barrera NEW	Lanny	Umbrella Corporation	1(811)424-7830	1(811)517-8577	Lanny.Barrera@UmbrellaCorporation.com
Barrett NEW	Jared	Bluth Company	1(306)480-1372	1(306)685-3333	Jared.Barrett@BluthCompany.com

24. Now select **Page 3** and notice it displays the **Books** list in the same format.



Page 3

Title	Author	Year Published	Author Country	Original Language
Fairy tales NEW	Hans Christian Andersen	1837	Denmark	Danish
One Thousand and One Nights NEW	Anonymous	1500	Egypt	Arabic
Pride and Prejudice NEW	Jane Austen	1813	England	English
Wuthering Heights NEW	Emily Brontë	1847	England	English
The Stranger NEW	Albert Camus	1942	Algeria	French
Journey to the End of the Night NEW	Louis-Ferdinand Céline	1932	France	French
Don Quixote NEW	Miguel de Cervantes	1615	Spain	Spanish
The Canterbury Tales NEW	Geoffrey Chaucer	1400	England	English
Great Expectations NEW	Charles Dickens	1861	England	English
Crime and Punishment NEW	Fyodor Dostoyevsky	1866	Russia	Russian

25. Close the browser to stop the debugger and go back to Visual Studio.

In this exercise you added a few site pages to a full trust solution and created links in the navigation pointing to these pages. You are now done with this lab.