## Creating SharePoint Farm Solutions

**Lab Time**: 60 minutes

**Lab Folder**: [[StudentFolder]]\DevSolutions

**Lab Overview**: In this lab you will learn how to create SharePoint farm solutions. While Microsoft is pushing their new SharePoint App 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 apps. This is why farm solutions still play an important role in the product.

### Exercise 1: Setup Lab Environment

In this exercise you will setup your environment.

All exercises in this lab assume you will work in a new site collection, http://solutions.wingtip.com.

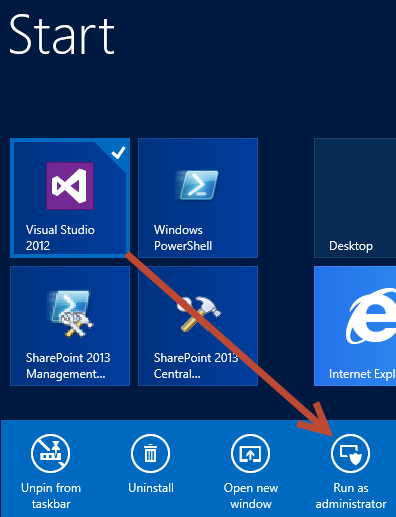
1. Setup a new site collection for this lab:
   1. Ensure you are logged into the **WingtipServer** server as **WINGTIP\Administrator**.
   2. Run a PowerShell script, found in the root lab folder for this module:
      1. Right-click **SetupModule.ps1** and select **Run with PowerShell**. This file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

[..]\DevSolutions

* 1. When the script completes, it will launch a new browser and navigate to the lab site collection.
  2. Close the PowerShell console window.

### Exercise 2: Creating Web Parts using Farm Solutions

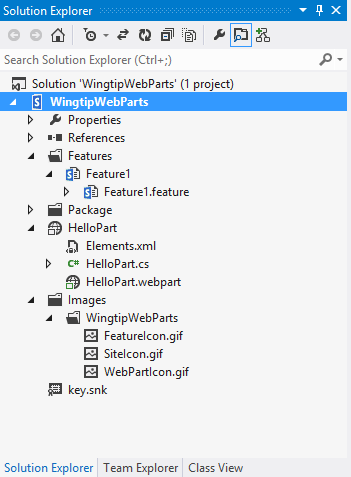
In this exercise you will create a simple Web Part and add it to a custom Web Part group with a custom icon. This will be done using a farm solution.

1. Create a new project in Visual Studio 2012:
   1. Launch **Visual Studio 2012** as administrator:
      1. **Windows Keyboard Key** 🡪 **Right click** on the **Visual Studio 2012** tile and select **Run as administrator** **(Note:** 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)
   2. In Visual Studio select **File 🡪 New 🡪 Project**.
   3. In the **New Project** dialog:
      1. Find the **SharePoint 2013 - Empty Project** template under the following node:
         1. **Templates** 🡪 **Visual C#** 🡪 **Office / SharePoint** 🡪 **SharePoint Solutions**
      2. **Name**: WingtipWebParts
      3. **Location:** [..]\DevSolutions\Exercises\Ex2  
         (Where [..] represents the location of the student files (e.g. c:\student\..)
      4. **Uncheck** the **Create directory for solution** checkbox
      5. Click **OK** to create the project
   4. In the **SharePoint Customization Wizard**, use the following values to complete the wizard.
      1. **What site do you want to use for debugging?** <http://solutions.wingtip.com>
      2. **What is the trust level for this SharePoint solution?** Deploy as farm solution
   5. Click **Finish**
2. Add a few images into your project so they are deployed inside the SharePoint **Images** folder.
   1. Using the **Solution Explorer** tool window, right-click the **WingtipWebParts** project and select **Add 🡪 SharePoint “Images” Mapped Folder**.
   2. Right-click the **Images\WingtipWebParts** folder you just created and select **Add 🡪 Existing Item**.
   3. In the **Add Existing Item** dialog, navigate to the folder associated with this exercise:

[..]\DevSolutions\Exercises\Ex2

* 1. Select the following files:
     1. FeatureIcon.gif
     2. SiteIcon.gif
     3. WebPartIcon.gif
  2. Click **Add**

1. Add a Web Part to the project:
   1. Using the **Solution Explorer** tool window, right-click the **WingtipWebParts** project and select **Add 🡪 New Item**.
   2. In the **Add New Item** dialog, select the **Web Part** template from the **Visual C# Items 🡪 Office/SharePoint** category.
      1. **Name:** HelloPart
   3. Click **Add**.
2. Inspect the SharePoint Project Item (SPI) node for the Web Part named **HelloPart**. Notice it contains three SPI files named **Elements.xml**, **HelloPart.cs** and **HelloPart.webpart**.



1. Right-click the **HelloPart \ HelloPart.webpart** file and select **Open**.
   1. Update the **<property>** node that has the attribute **name=”Title”** and set the title to **The “Hello” Web Part**.
   2. Update the **<property>** node that has the attribute **name=”Description”** and set the title to **A most compelling Web Part**.
2. 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>

1. 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>

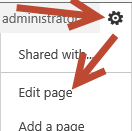
1. Now, modify the element manifest that will provision the Web Part definition:
   1. Using the **Solution Explorer** tool window, right-click the **HelloPart \ Elements.xml** file and select **Open**.
   2. Modify the **<Property>** element with the Name=”Group” attribute by setting the **Value** attribute to **Wingtip Web Parts**.
2. Finally, add some logic to the Web Part:
   1. Using the **Solution Explorer** tool window, right-click the **HelloPart \ HelloPart.cs** file and select **Open**.
   2. Add the following code to the body of the **CreateChildControls()** method:

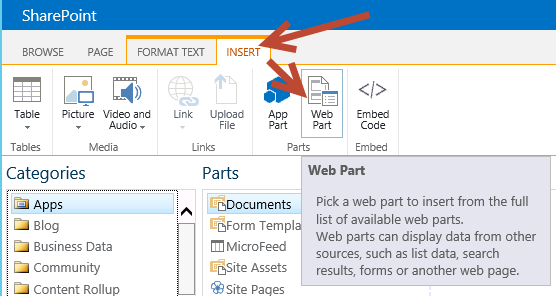
var label = new Label() { Text = "Hello Web Part" };

this.Controls.Add(label);

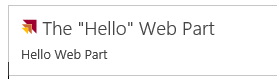
1. Save all changes: **File 🡪 Save All**.

#### Build and Test the Project

1. Build and test your application by pressing **[F5]** or **Debug 🡪 Start Debugging**.
2. Once the solution has been deployed, Internet Explorer will launch and navigate to the <http://solutions.wingtip.com> site.
3. Add the Web Part to the page:
   1. Using the **Site Actions** “gear” icon in the top-right corner, select **Edit Page**.  
      
   2. Using the ribbon, select the **Insert** tab and click the **Web Part** button.



* 1. Select the **The “Hello” Web Part** from the **Wingtip Web Parts** category and click the **Add** button.



1. 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

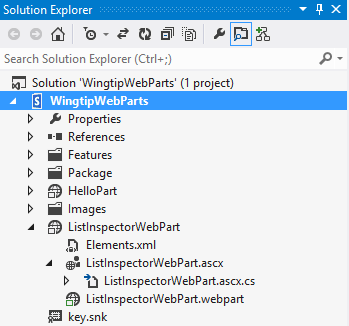
Now you will add a second Web Part using the Visual Web Part template. This makes it possible to create the UI for a Web Part using an ASP.NET User Control and the Visual Studio User Control Designer. You will also use the UpdatePanel control from ASP.NET AJAX to give your Web Part a Web 2.0 user experience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

1. **If you want to continue working with the solution from the previous example, you can skip this step**. Otherwise you can follow these steps to open a clean project which effectively the where you left off if you had completed the previous exercise as outlined:
   1. Open an existing starter project in Visual Studio 2012:
      1. Launch **Visual Studio 2012** as administrator: **Start 🡪 All Programs 🡪 Microsoft Visual Studio 2012 🡪 Visual Studio 2012**.
      2. Select **File 🡪 Open 🡪 Project/Solution**.
      3. In the **Open Project** dialog, select the following project provided in the files associated with this lab:

[..]\DevSolutions\Exercises\Ex3\StarterProject\WingtipWebParts.sln

* + 1. Once the project is open select the **WingtipWebParts** project and then click the **wrench** icon in the top menu bar of the Solution Explorer Window to open the properties window below  
       (Note: this is NOT the same as right clicking on the project and using the shortcut menu; this method opens a separate properties window and is NOT what we are looking for here).
    2. In the **Properties** window (Below the Solution Explorer window) find the site URL and make sure it is set to **http://solutions.wingtip.com**

1. Add a new Visual Web Part to the **WingtipWebParts** project named **ListInspectorWebPart**:
   1. Using the **Solution Explorer** tool window, right-click the **WingtipWebParts** project and select **Add 🡪 New Item**.
   2. In the **Add New Item** dialog, select the **Visual Web Part** template from the **Visual C# Items 🡪 Office/SharePoint** category.
      1. **Name:** ListInspectorWebPart
   3. Click **Add**.
2. Inspect the SharePoint Project Item (SPI) node for the Web Part named **ListInspectorWebPart**. Notice it contains multiple files:



1. Right-click the **ListInspectorWebPart \ ListInspectorWebPart.webpart** file and select **Open**.
2. 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>

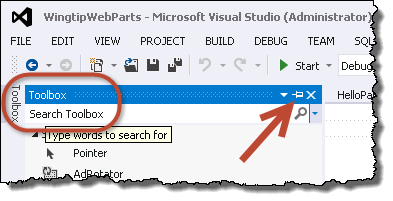
<property name="Description" type="string">A Web Part which shows all the lists in the current site and allows you to get several of its property values.</property>

<property name="ChromeType" type="string">TitleAndBorder</property>

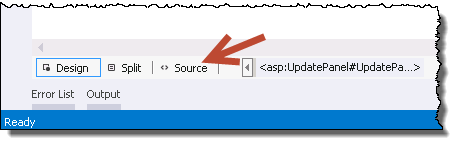
<property name="CatalogIconImageUrl" type="string">\_layouts/15/images/WingtipWebParts/WebPartIcon.gif</property>

<property name="TitleIconImageUrl" type="string">\_layouts/15/images/WingtipWebParts/WebPartIcon.gif</property>

1. Now, modify the element manifest that will provision the Web Part definition:
   1. Using the **Solution Explorer** tool window, right-click the **ListInspectorWebPart \ Elements.xml** file and select **Open**.
   2. Modify the **<Property>** element with the Name=”Group” attribute by setting the **Value** attribute to **Wingtip Web Parts**.
2. Update the user interface portion of the Web Part:
   1. Using the **Solution E**xplorer tool window, right-click the **ListInspectorWebPart \ ListInspectorWebPart.ascx** file and select **View Designer**.
   2. If the **Toolbox** tool window is not open, click it and then click the pushpin to pin it open:

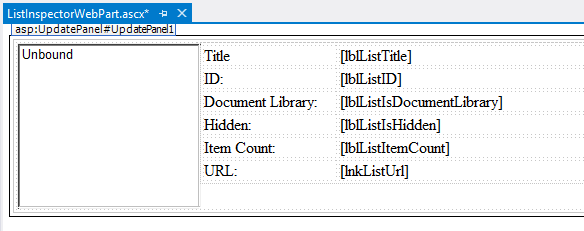


* 1. Using the **Toolbox** tool window, find the **UpdatePanel** control in the **AJAX Extensions** grouping. Drag the **UpdatePanel** onto the design surface.
  2. Using the buttons at the bottom of the designer, click **Source** to switch to the **Source** view:



* 1. Add an HTML table to the **<asp:UpdatePanel>** element on the page. You can find the contents of this file in the following file in the folder associated with this exercise:

[..]\DevSolutions\Exercises\Ex3\StarterFiles\ListInspectorTable.txt

1. 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:  
   
2. Now code the Visual Web Part to make it do something:
   1. Using the **Solution Explorer** tool window, right-click the **ListInspectorWebPart \ ListInspectorWebPart.ascx \ ListInspectorWebPart.ascx.cs** file and select **Open**.
   2. Add the following to the top of the file:

using Microsoft.SharePoint;

using System.Web.UI.WebControls;

* 1. Create two protected fields by adding the following code just inside the class:

protected Guid SelectedListId = Guid.Empty;

protected bool UpdateListProperties = false;

* 1. 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;

}

* 1. 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;

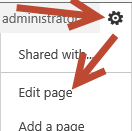
lnkListUrl.NavigateUrl = list.DefaultViewUrl;

}

}

1. Save all changes: **File 🡪 Save All**.

#### Build and Test the Project

1. Build and test your application by pressing **[F5]** or **Debug 🡪 Start Debugging**.
2. Once the solution has been deployed, Internet Explorer will launch and navigate to the <http://solutions.wingtip.com> site.
3. Add the Web Part to the page:
   1. Using the **Site Actions** “gear” icon in the top-right corner, select **Edit Page**.  
      
   2. Using the ribbon, select the **Insert** tab and click the **Web Part** button.
   3. Select the **List Inspector Web Part** from the **Wingtip Web Parts** category and click the **Add** button.
4. Click different lists to see the values refresh to show the respective properties.
5. 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.