* 1. 

Demo Script

Microsoft SharePoint Developers Tools 2010 - Web Parts Templates

Lab version: 1.1.0

Last updated: 1/3/2011

* 1. 

1. Contents

[Overview 3](#_Toc281840472)

[Key Messages 3](#_Toc281840473)

[Key Technologies 3](#_Toc281840474)

[Time Estimates 3](#_Toc281840475)

[Setup and Configuration 4](#_Toc281840476)

[Demo Flow 4](#_Toc281840477)

[Opening Statement 5](#_Toc281840478)

[Step-by-Step Walkthrough 6](#_Toc281840479)

[Segment #1 – Adding a SharePoint WebPart 6](#_Toc281840480)

[Segment #2 – Adding a SharePoint Visual WebPart 10](#_Toc281840481)

[Segment #3 – Test SharePoint Feature 12](#_Toc281840482)

[Summary 16](#_Toc281840483)



Overview

* 1. This document provides setup documentation, systematic instructions, and a written script for showing the tasks required to create SharePoint Project Items like Web parts and Visual Web Parts using the SharePoint Tools templates. This document can also serve as a tutorial or walkthrough of the exposed technologies.
  2. One of the most exciting aspects of Visual Studio 2010 is the tight integration it will have with SharePoint. The Visual Studio team has worked hard to make the SharePoint developer experience much more streamlined and less dependent upon manual configuration files and custom scripts. They have also made a great investment into extensibility, allowing developers to move beyond the out-of-the box features and create their own custom extensions to fit their individual or team development environment.

### Key Messages

* 1. Web Parts are server-side controls that run inside the context of special pages (that is, Web Part Pages) within an ASP.NET application or a Windows SharePoint Services site.
  2. Visual Web Parts enable developers to visually design a Web Part that can be deployed to SharePoint.

### Key Technologies

This demo uses the following technologies:

* 1. Microsoft Visual Studio 2010
  2. Microsoft SharePoint Server 2010

### Time Estimates

* + Estimated time for setting up and configuring the demo: 3 min
  + Estimated time to complete the demo: 45 min

Setup and Configuration

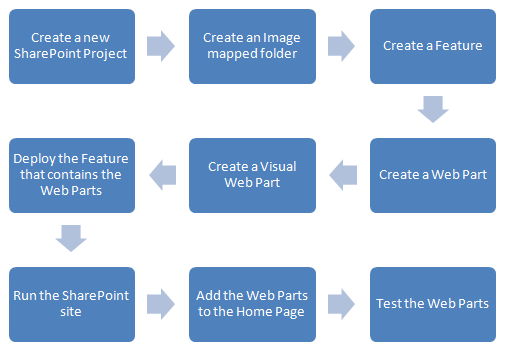
## System Requirements

* + Microsoft Visual Studio 2010
  + Internet Information Server 7.0
  + Microsoft SharePoint Server 2010
  + Microsoft SharePoint Server 2010 Developer Tools
    1. Note: You can follow the [Set up the Development Environment for SharePoint Server](http://msdn.microsoft.com/en-us/library/ee554869%28office.14%29.aspx) article in order to get instruction to install SharePoint Server appropiatelly.

# Setup

* 1. The setup of this demo is performed by running the setup.bat script that will create the required SharePoint site in for this demo.
  2. Run the setup.bat script located in the Source\Setup folder. You will be notified when the tasks are completed.
     1. **Note:** The Configuration Wizard will setup a new blank site collection at the location http://localhost/sites/SharePointToolsDemo. This is the site you will use to test and debug the code you are going to write with the Visual Studio 2010 SharePoint Tools.

Demo Flow

1. The following diagram illustrates the high-level flow for this demo and the steps involved:
   1. 

Opening Statement

Today I would like to walk you through a demo application built to show WebParts in Microsoft SharePoint Server 2010.

Web Parts are server-side controls that run inside the context of special pages (that is, Web Part Pages) within an ASP.NET application or a Windows SharePoint Services site. They are the building blocks of pages in Windows SharePoint Services. Windows SharePoint Services includes built-in Web Parts that you can use as soon as you have installed the product. In addition, you can build your own Web Parts and deploy them to the server.

In this demo you will specifically see these things:

* 1. How to Create a SharePoint project and add a Web Part
  2. How to Create and add a Visual Web Part
  3. The SharePoint Web Parts in action

Step-by-Step Walkthrough

1. This demo is composed of the following segments:
   1. Adding a SharePoint WebPart
   2. Adding a SharePoint Visual WebPart
   3. Test SharePoint Feature

### Segment #1 – Adding a SharePoint WebPart

|  |  |  |
| --- | --- | --- |
| Action | Script | Screenshot |
| * 1. Open **Microsoft Visual Studio 2010** from Start | All Programs.   2. Create a new Empty SharePoint project, name it **MetroWebParts**.   3. Set the Site Url to a site already created. For example, http://localhost/sites/SharePointToolsDemo.   4. Select the Deploy as a farm solution option in the SharePoint Customization Wizard.   5. Click Finish. | * + First, we will create a new Empty Sharepoint project. To do this, in the New Project dialog, we will choose the **Empty SharePoint Project** template located in the **SharePoint** category and use the default values just for simplicity. |  |
| * 1. Add the Images SharePoint mapped folder to the project by right-clicking on the project node, pointing to **Add**, and selecting **SharePoint “Images” Mapped Folder**.   2. Right-click on the **MetroWebParts** folder and select **Add Existing Item**. Then browse to the **Assets** folder of this demo, and select the FeatureIcon.gif and WebPartIcon.gif image files. | * + We will add an Image mapped folder to our SharePoint project. The files located in this folder will be deployed inside the SharePoint Images folder.   + This will automatically create the **MetroWebParts** folder.   + SharePoint Tools helps to ensure best practices with **SharePoint** development. When adding image files to your project, you should not add them directly inside the Images folder. Instead, add them to the inner directory in this case named **MetroWebParts** to avoid file name conflicts with the image deployed by **SharePoint** in the Images directory.   + In this step, we will add a couple of images to this folder. |  |
| * 1. Open the feature included in the Features folder by right-clicking on it and opening the Feature designer.   2. Change the scope to **Site**.   3. In the Properties window, edit the **Image Url** property and set it to **MetroWebParts\FeatureIcon.gif** to use the recently added image file. | * + Now we will create a feature, by right-clicking the Features folder and selecting **Add Feature**.   + We will change the feature scope to **Site.** A Scope setting of Site results in activation at the site collection level. The change of the feature's Scope setting to Site is required because Web Parts deployment requires a feature that activates at site collection scope in order to deploy Web Part description files properly.   + Let’s associate the icon we added previously to this feature. To do this, we have to go to the **Properties** window and set the **Image Url** property.   + The Image URL is relative to the **Images** mappedfolder. |  |
| * 1. Add a new **SharePoint Web Part** by right-clicking on the project node and using the **Add New Item** context menu. Name it **SimpleWebPart.**   2. Edit the elements.xml and change the **Group** property of the **File** element to **MetroWebParts**.   3. Edit the .webpart file of the Web Part by double-clicking it. Modify the **Title** property to **Simple DemoWeb Part** and the **Description** property to **A simple SharePoint 2010 Web Part**.   4. Include the **ChromeType**, **CatalogIconImageUrl** and **TitleIconImageUrl** properties by pasting the following code.   5. <property name="ChromeType" type="chrometype">TitleAndBorder</property>   6. <property name="CatalogIconImageUrl" type="string">\_layouts/images/MetroWebParts/WebPartIcon.gif</property>   7. <property name="TitleIconImageUrl" type="string">\_layouts/images/MetroWebParts/WebPartIcon.gif</property> | * + Now, we will create a new Web Part, called **SimpleWebPart**. This is done through the **Add New Item** window.   + Then, we will edit the elements.xml and change the **Group** property of the **File** element to something more meaningful like “**MetroWebParts**”.   + Then in the .webpart file of the WebPart we will change the Title and Description properties.   + Additionally, we will add the **ChromeType**, the **CatalogIconImageUrl** and the **TitleIconImageUrl** properties.   + After performing these steps we have our first Web Part created. |  |

### Segment #2 – Adding a SharePoint Visual WebPart

|  |  |  |
| --- | --- | --- |
| Action | Script | Screenshot |
| * 1. Add a new **SharePoint Visual Web Part** by right-clicking on the project node and using the **Add New Item** context menu. Name it **SiteExplorer.**   2. Edit the elements.xml and change the **Group** property of the **File** element to **MetroWebParts**.   3. Edit the .webpart file of the Web Part by double-clicking it. Modify the **Title** property to **Site Explorer** and the **Description** property to **A SharePoint 2010 Web Part, which displays the hierarchical virtual file system for the current site**.   4. Include the **ChromeType**, **CatalogIconImageUrl** and **TitleIconImageUrl** properties by pasting the following code.   5. <property name="ChromeType" type="chrometype">TitleAndBorder</property>   6. <property name="CatalogIconImageUrl" type="string">\_layouts/images/MetroWebParts/WebPartIcon.gif</property>   7. <property name="TitleIconImageUrl" type="string">\_layouts/images/MetroWebParts/WebPartIcon.gif</property> | * + Now we will create a SharePoint Visual Web Part, called **SiteExplorer**.   + One of the new project templates in Visual Studio 2010 is the **Visual Web Part** project template, which enables developers to visually design a Web Part that can be deployed to SharePoint.   + We will edit the same properties as we did it in the simple Web Part, that is, changing the **Group** property of the **File** element in the elements.xml file; changing the **Title** and **Description** properties, and adding the **ChromeType**, **CatalogIconImageUrl** and **TitleIconImageUrl** properties. |  |
| * 1. Open the **SiteExplorerUserControl.ascx** file of the Template, by double-clicking it.   2. Add a **SharePoint TreeView** to the user control by dragging it from the **Toolbox**. Change the **ID** property of it to **treeSiteFiles**.   3. Open a Windows Explorer window, and browse to the project folder.   4. Replace the **SiteExplorerUserControl.ascx.cs** file, located in \Source\{Language}\MetroWebParts\SiteExplorer folder, with the one located in the \Source\Assets folder.   5. Show the **OnPreRender** method.   6. Show the using **Microsoft.SharePoint** using statement. | * + Let’s add a **SharePoint TreeView** to the user control included in the Web Part, by dragging it from the **Toolbox**.   + Now we will add the code necessary to make this work, by replacing the empty code file, with a complete one. So let’s open it, and see its content.   + Notice that the **OnPreRender** method is overridden. This method loads the **TreeView** with the hierarchical virtual file system for the current site.   + Note that the **Microsoft.SharePoint** namespace is referenced. This namespace provides types and members for working with a top-level site and its subsites or lists. |  |

### Segment #3 – Test SharePoint Feature

|  |  |  |
| --- | --- | --- |
| Action | Script | Screenshot |
| * 1. Change the **Active Deployment Configuration** to **No Activation** using the **SharePoint** tab in the **Project** properties page. | * + Now, we will see the specific configuration for this SharePoint project.   + In the **Active Deployment Configuration** combo box we will change the selected item from **Default** to **No Activation**.   + Each configuration consists of one or more Deployment Steps, and each step may execute one or more actions and/or SharePoint Commands. |  |
| * 1. Right-click the project node and click **Deploy**. Review the Output window to see the details. | * + By right-clicking the SharePoint project node you will notice three new actions can be performed: Deploy, Package and Retract.   + Let’s deploy the site, and we will look at the **Output** window to see the details.   + At this point, the solution package for the project has been deployed on the local SharePoint. Let's test out our work by trying to activate the Features we defined inside the project. |  |
| * 1. Open I**nternet Explorer** and browse to the configured SharePoint site’s home page: http://localhost/sites/SharePointToolsDemo.   2. Click **Site Actions** - **Site Settings** to navigate to the **Site Settings** page.   3. Inside the **Site Collection Administration** section of the **Site Settings** page, click the **Site collection features** link to navigate to the **Site Collection Administration** > **Features** page.   4. Locate the feature you have been working on with a title of **MetroWebParts** and click the **Activate** button.   5. Click the **Home** button to return to the Home page. | * + Let’s open the site in Internet Explorer and activate the feature we created.   + To activate the feature that contains the Web Parts, we have to go to **Site Actions – Site Settings**.   + In the **Site Settings** section, click the **Site Collection Features** in the **Site Collection Administration** section.   + Let’s look for the feature we created, and activate it.   + Finally, if you return to the Home Page to create a new page that contains the Web Parts created recently. |  |
| * 1. Using the **Page** ribbon in the site’s home page, select the **Edit Page** command to edit the home page and add the created web parts.   2. Click the **Add Web Part** button located in the **Right** frame.   3. Select the **MetroWebParts** category and check the two created Web Parts are present.   4. Add a **SimpleWebPart** element by clicking **Add**.   5. Click the **Add Web Part** button located in the **Left** frame. This time add a **Site Explorer** web part.   6. Click the **Stop Editing** command to check the new site’s home page look and feel.   7. Click and use the Site Explorer Visual Web Part. | * + Let’s edit the home page to include the Web Parts we have just created.   + To do this, we are going to edit the home page, by using the Edit Page command located in the .   + Notice that there are two frames, where you can put content: the **Left** and the **Right** one. There are **Add Web Part** buttons in each of these frames. Every time you click one, you can add a Web Part. We will add the Site Explorer Visual Web Part to the left frame and the **SimpleWebPart** to the right frame.   + Once this is done, we will stop editing the page to see it in action.   + Finally, we will try the Site Explorer web part to see its functionality by exploring the web server hierarchical structure. |  |

Summary

1. In this demo, you explored SharePoint Web Parts. You saw how to create a standard Web Part and a Visual Web Part. Additionally, you learnt deploy a SharePoint project and activate the feature that includes the created Web Parts. Finally, you added the Web Parts to a SharePoint page to test their functionality.