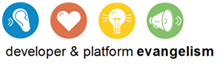
* 1. 

Demo Script

SharePoint Tools Projects

Lab version: 1.1.0

Last updated: 1/4/2011



1. Contents

[Overview 3](#_Toc281917266)

[Key Messages 3](#_Toc281917267)

[Key Technologies 3](#_Toc281917268)

[Time Estimates 3](#_Toc281917269)

[Setup and Configuration 4](#_Toc281917270)

[Opening Statement 4](#_Toc281917271)

[Step-by-Step Walkthrough 5](#_Toc281917272)

[Segment #1 – Creating a SharePoint Project 5](#_Toc281917273)

[Segment #2 – Feature Event Receiver 9](#_Toc281917274)

[Summary 13](#_Toc281917275)



Overview

* 1. Microsoft SharePoint is a Content Management System with integrated search functionality that allows users to work in a web-based collaborative environment. Microsoft provides certain built-in functionality, and third party developers can develop custom modifications to extend functionality.
  2. Microsoft Visual Studio 2010 supports the creation of SharePoint features, through the Microsoft SharePoint Server 2010 Developer Tools, that extend its functionality.
  3. In this demo script, you are walked through the creation of SharePoint project and its components, the creation of a feature and its purposes. Additionally, the Feature event Receiver is shown, which let you run custom code at certain point in the lifetime of a Feature, such as activation or deactivation. Finally, it demonstrates the debug and deployment of a SharePoint feature.

### Key Messages

* 1. SharePoint let you create Intranet sites easily. Visual Studio 2010 provides native support for developing SharePoint Features.
  2. Feature Event Receivers react to SharePoint feature activation and deactivation, and allow you to run code during these events.

### Key Technologies

This demo uses the following technologies:

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

### Time Estimates

* + Estimated time to complete the demo: 30 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.

Opening Statement

* 1. In this demo, we will start using the Visual Studio 2010 SharePoint Developers Tools. This will give you a chance to become familiar with the standard project structure used by SharePoint Developers Tools. We will go through the steps of creating and testing a project that contains a Feature, and a Feature Receiver. Along the way, you will learn to configure the SharePoint project’s deployment options, as well as debug a SharePoint project using Visual Studio 2010 by single stepping through the code in your solution.
  2. The following is a brief description of the topics that will be shown during this demo:
  3. Creating a SharePoint Project
  4. Adding a Feature Event Receiver

Step-by-Step Walkthrough

1. This demo is composed of the following segments:
   1. Creating a SharePoint project
   2. Feature Event Receiver

### Segment #1 – Creating a SharePoint Project

|  |  |  |
| --- | --- | --- |
| Action | Script | Screenshot |
| * 1. Open Microsoft Visual Studio 2010 from **Start | All Programs**.   2. Create a new Empty SharePoint project.   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.   + The Customization Wizard will appear. Here, you can specify the site that will be used for debugging and the trust level for the Sharepoint solution. We will choose the farm solution deployment option. Farm Solutions are hosted in the IIS worker process, and run code that can affect the whole farm. |  |
| * 1. In **Solution Explorer**, show SharePoint special folders: **Features** and **Package**. | * + Once the new project has been created, you should be able to observe it has four nodes: Properties, References, Features and Package.   + While the first two nodes are standard on all Visual Studio projects, the Features node and the Package node are unique to projects created with the Visual Studio 2010 SharePoint Tools.   + These nodes are for exclusive use of the SharePoint Project template and cannot be renamed, deleted, cut, copied, or dragged off from the project. |  |
| * 1. Add a new feature by right-clicking the **Features** node and selecting **Add Feature**.   2. Rename the recently created feature to **Main**.   3. Open the Main feature in the feature designer. Use the feature designer to change the feature’s Title, Description and Scope using the following values:      + **Title**: Custom Web Parts      + **Description**: My very first project using the Visual Studio 2010 Tools      + **Scope**: Site   4. Save the modifications. | * + The Features node contains one or more SharePoint project features. A feature is a container of extensions for SharePoint. After a feature is deployed to SharePoint server, it can be included in site definitions or activated individually by SharePoint administrators on SharePoint sites.   + Let’s add a new feature. The feature designer will open and we can configure some feature attributes in it.   + A Scope setting of **Web** results in site-level activation of the feature while 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. |  |
| * 1. Review the package node and the Package designer.   2. Click the **Edit** feature link in the Package designer to open the **Features Designer**.   3. Right-click the project name and click Package. | * + Now, let’s review the **Package** node. To open it, double-click it.   + The Package node contains a single file that serves as the distribution mechanism for the SharePoint project. This file, known as a solution package, is .CAB-based with a .WSP extension. A solution package is a deployable, reusable file that contains a set of features, site definitions, and assemblies that apply to SharePoint sites, and that you can enable or disable individually. The Package node also always contains a file that is named Package.wspdef, an XML definition file for the package.   + Once a package is deployed to the SharePoint server, the SharePoint administrator can install it and activate its features.   + By using the Package Designer, you can create and customize deployment packages. For example, you can add SharePoint project items and Features, reset the IIS server, set Feature activation scopes, and identify Feature dependencies. The designer also generates a manifest, an XML file that describes each package.   + If you look at the Items in Package feature, you can see the feature created previously. Clicking the **Edit** link, will open the **Feature Designer**.   + Finally, to generate this package, you just have to right-click the project name, and select **Package**. |  |

### Segment #2 – Feature Event Receiver

|  |  |  |
| --- | --- | --- |
| Action | Script | Screenshot |
| * 1. In the **Solution Explorer**, right-click on the Main feature node, and choose **Add Event Receiver**, to add an event receiver to the selected feature.   2. Show the code of the **Event Receiver** class and the commented methods for the key events. | * + Now, let’s add a **Feature Event Receiver** to the **Main** feature.   + An Event Receiver specifies a server-side code routine that is called as part of four key events in the lifetime of a Feature: installation, activation, deactivation, and removal.   + This action will create the Event Receiver class associated to that feature. Notice that there are commented stub methods for every key event. Therefore, you can uncomment the desired method and write your own code. |  |
| * 1. Open the completed solution, located in **Source\{Language}**, where **{Language}** can be the language of your preference.   2. Open the **Main.EventReceiver** class located in the **Features\Main** node.   3. Show the **GUID** attribute.   4. Show the **FeatureActivated** and **FeatureDeactivating** methods that implement code to update the site’s title. | * + In order to speed up this demo, we will open another project with the same structure but with code written in some of these events.   + In addition, a GUID attribute has been applied to the receiver class to give a unique identifier. Do not remove the GUID from the class because it will be used behind the scenes by the SPT during the packaging process.   + Let’s look at the **FeatureActivated** method. It first obtains a **SPWeb** reference to the top-level from the **SPSite** to get the current value of the **Title** property. It stores the value as a name/value pair inside the SPWeb.Properties collection and then modifies it to the new value “VS 2010 SPT Rocks”.   + Finally, in the **FeatureDeactivation** method, the original title is restored. |  |
| * 1. Change the **Active Deployment Configuration** to **No Activation** using the **SharePoint** tab in the **Project** properties page.   2. Save the configuration. | * + Now, we will see the specific configuration for SharePoint projects.   + Note that there are two text boxes that allow you to add command-line instructions, which will execute either just before or directly after the SPT deploy steps, are processed. You are not going to add anything to either of the two top text boxes.   + In the Active Deployment Configuration combo we will change the selected item from **Default** to **No Activation**.   + If you select any configuration in the **Edit Configuration** list, you will see its description and you can edit the configuration. |  |
| * 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. We already comment around the Package action, which generates the .WSP file. You can also Deploy and Retract (remove) your Package from the configured SharePoint Server directly from the contextual menu.   + 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 feature we defined inside the project. |  |
| * 1. Add a breakpoint to the **FeatureActivated** method in the Event Receiver class created in a previous step.   2. Click the **Start Debugging (F5)** button in Visual Studio 2010.   3. Click Site Actions - Site Settings to navigate to the Site Settings page.   4. 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.   5. Locate the feature you have been working on with a title of **CustomWebParts** and click the Activate button.   6. When Visual Studio gets the focus due to the breakpoint, press **F5** to continue.   7. Click the **Home** button to return to the Home page. | * + We will also add a breakpoint in the **FeatureActivated** method and run the solution using Visual Studio Debug button.   + Let’s run the site and activate the feature we created. To do this, 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.   + At this point, the focus will return to Visual Studio, because the breakpoint was caught. You can press F5 to continue or F11 to debug the method line by line.   + Finally, if you return to the Home Page, you will see that the feature changed the Title to “VS 2010 SPT Rocks”.   + If you deactivate the feature, the home title will return to the original value. |  |



Summary

1. In this demo, you explored SharePoint Projects. You saw how to create a SharePoint project and its components. Additionally, you learnt how to create Features and Event Receivers for them. Finally, you deployed a SharePoint project and activated the created feature.