

A SharePoint Developer Introduction

Hands-On Lab

Lab Manual

SPCHOL200 – Building Visual Web Parts – C#

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2010 Microsoft. All rights reserved.

**Contents**

[SPCHOL200 – Building Visual Web Parts 2](#_Toc259797198)

[Lab Objective 2](#_Toc259797199)

[Additional Resources 2](#_Toc259797200)

[Getting Started 3](#_Toc259797201)

[Logging into the Virtual Machine 3](#_Toc259797202)

[Locations 3](#_Toc259797203)

[Lab Pre-requisites 3](#_Toc259797204)

[Copying code samples from Word document 4](#_Toc259797205)

[Code Snippets 4](#_Toc259797206)

[Exercise 1 – Web Part and Linq Walkthrough 6](#_Toc259797207)

[Task 1 - Create a new SharePoint Project 6](#_Toc259797208)

[Task 2 - Generate LINQ-to-SharePoint proxy class to access list data 9](#_Toc259797209)

[Task 3 - Access the SharePoint list data in Visual Web Part 11](#_Toc259797210)

[Task 4 – Build and Deploy the Visual Web Part 13](#_Toc259797211)

[Exercise 2 – Connecting Web Parts 18](#_Toc259797212)

[Task 1 – Create a new Empty SharePoint Project 18](#_Toc259797213)

[Task 2 – Create the Web Part Connection Interface 21](#_Toc259797214)

[Task 3 – Create the Provider Web Part 24](#_Toc259797215)

[Task 4 – Create the Consumer Web Part 28](#_Toc259797216)

[Task 5 – Build and Deploy the Web Parts 31](#_Toc259797217)

[Task 6 – Connect the Web Parts 37](#_Toc259797218)

[Task 7 – Verify the Web Part Connection 39](#_Toc259797219)

[Lab Summary 40](#_Toc259797220)

# SPCHOL200 – Building Visual Web Parts

Estimated time to complete this lab: **30 minutes**

Visual Studio 2010 and SharePoint Foundation 2010 are required for these exercises. These are installed on the Virtual Machine used in this lab.

## Lab Objective

Web Parts are an essential component of ASP.NET technologies used by SharePoint to present dynamic information to users. Web parts are the most common customization created for SharePoint. A Web Part is a reusable component that exists on a Web Part Page and can present any type of web-based information.

The objective of this lab is to learn about how to utilize enhancements available in SharePoint 2010 to build Visual Web Parts and connect web parts for use in the SharePoint system.

* Work with existing Web Parts and Linq.
* Connect two web parts.

## Additional Resources

This lab includes the following additional resources:

|  |  |  |
| --- | --- | --- |
| This Lab Manual | SPCHOL200\_Manual\_CS.docx | This document. |
| Source Code | Completed\CS\Ex1  Completed\CS\Ex2 | Completed lab source code in C#. |
| Resources | *Resources\CS* | Various other resources used throughout this lab. |

## Getting Started

### Logging into the Virtual Machine

Please log into the virtual machine as the following user:

**Username**: Administrator

**Password**: pass@word1

### Locations

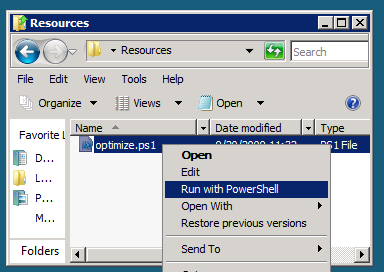
This Hands-On Lab contains a number of additional resources in fixed locations. By default, it is assumed that the base HOL Resources directory is ***C:\Content Packs\Packs\SharePoint 2010 Developer Labs 1.0\SUPPORTING FILES\SPCHOL200\Resources****.*

The default working folder for this lab is **C:\*SPHOLS\SPCHOL200***

### Lab Pre-requisites

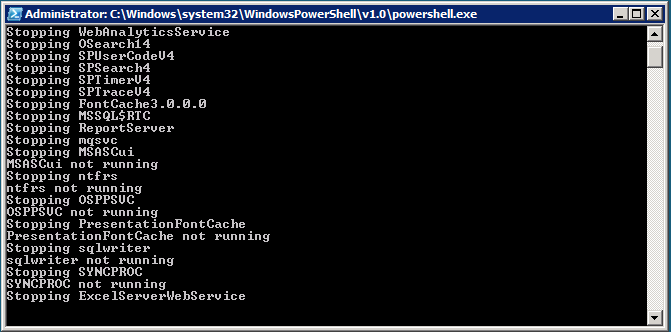
Browse to the base HOL directory ***Supporting Files\SPCHOL200\Resources***and execute the **optimize.ps1** PowerShell script:

1. Right click on **optimize.ps1** and select **Run with PowerShell**:



*Figure 1 - Execute the PowerShell script*

1. This will open a PowerShell window to execute the script. Click “Open” if you get a security window. Please wait until the PowerShell script completes executing the script and closes the PowerShell window (this may take a few moments):



*Figure 2 - PowerShell Window executing the script*

### Copying code samples from Word document

Copying and pasting code from this Word document to Visual Studio is only safe for the sections of formatted code, e.g.:

Console.WriteLine("This is safe code!");

Code not in these sections may contain Unicode or other invisible characters that are not valid XML or C#/VB code, *e.g*.:

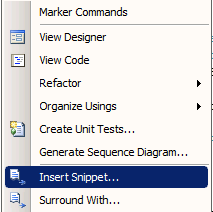
**Console.WriteLine(“This is NOT safe code !!”);**

### Code Snippets

You can also use Code Snippets to insert the appropriate code in the lab.

To use the required code snippet for this lab:

* Right-click on the code file where you want to insert the code snippet.
* Select **Insert Snippet:**

****

*Figure 3 - Insert Code Snippet*

* Select the required code snippet(s) from **My Code Snippets** gallery.

## Exercise 1 – Web Part and Linq Walkthrough

Estimated time to complete this exercise: **10 minutes**

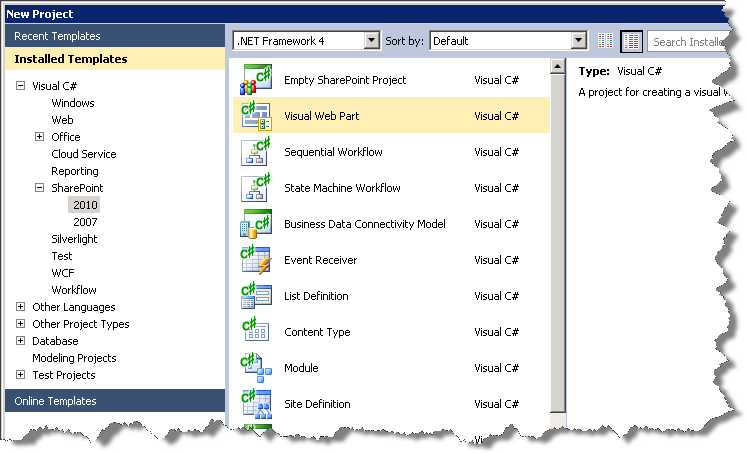
In this exercise, you will develop and deploy a Visual Web Part that reads data from a list and displays in a DataGrid. In this exercise, you will:

1. Create a Visual Web Part.
2. Generate Linq proxy code.
3. Use a Linq provider to read data from a SharePoint list.
4. Render the data using the SPDataGrid web control.

### Task 1 - Create a new SharePoint Project

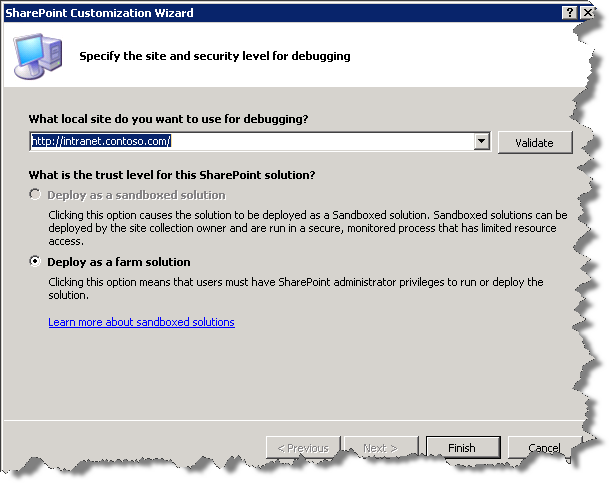
In this task, a solution and project will be created. It will contain the rest of the development work in Exercise 1 of this lab.

1. Open **Visual Studio 2010** by going to **Start Menu | All Programs | Microsoft Visual Studio 2010 | Microsoft Visual Studio 2010**.
2. From the menu, select **File | New | Project**.
3. In the New Project dialog window, choose **Visual C# | SharePoint | 2010** from the Installed Templates.
4. Select **Visual Web Part** from the Project Items.



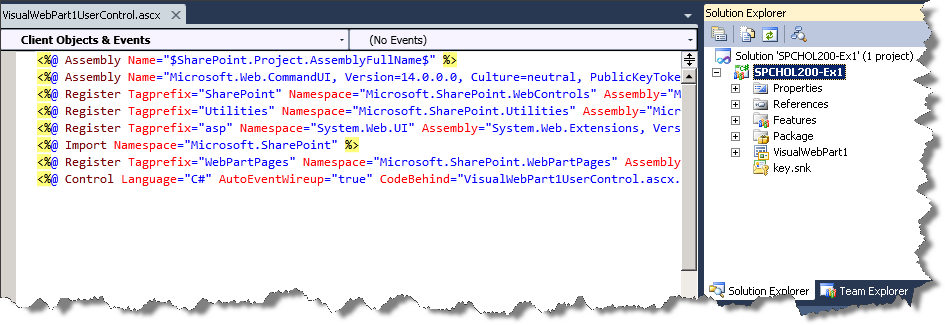
*Figure 4 - SharePoint 2010 Visual Web Part Project*

1. Enter **SPCHOL200-Ex1**in the Name textbox
2. Enter**C:\SPHOLS\SPCHOL200\CS\Ex1**in the Location textbox.
3. Uncheck Create directory for solution.
4. Click **OK**.
5. In the SharePoint Customization Wizard:
   * Enter [**http://intranet.contoso.com**](http://intranet.contoso.com)**/** for the local site.
   * Set the trust level to **Deploy as a farm solution**.
   * Click **Finish** button.



*Figure 5 - SharePoint Customization Wizard*

1. Visual Studio will create the new **SPCHOL200-Ex1** project and add the necessary files.



*Figure 6 - SPCHOL200-Ex1 Project*

1. Notice that Visual Studio also creates a Visual Web Part named **VisualWebPart1**. Within the Solution Explorer, expand **VisualWebPart1** and open **VisualWebPart1.webpart**.

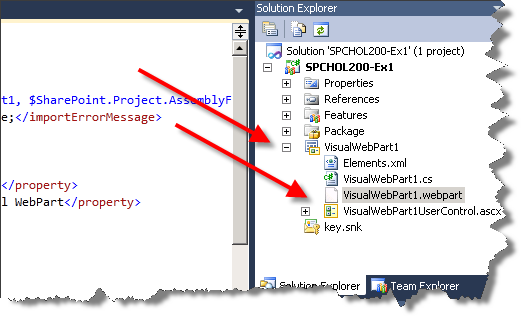


Figure 7 - Visual Web Part

1. Change the value of the property element with the name attribute value of Title to *SPLinqDemoTitle* and the value of the property element with the name attribute value of Description to *SPLinqDemoPart Description*. This will change the Title and Description property of the Visual Web Part once it is deployed. Save the file.

<properties>

<property name="Title" type="string">SPLinqDemoTitle</property>

<property name="Description" type="string">SPLinqDemoPart Description</property>

</properties>

### Task 2 - Generate LINQ-to-SharePoint proxy class to access list data

In this task, you will use the new **spmetal.exe** code generation utility and generate the Linq-to-SharePoint proxy code.

1. In the Solution Explorer, right-click on **SPCHOL200-Ex1** and select **Open Folder in Windows Explorer.**
2. Hold **Shift key** and **right click** anywhere in the **Explorer Window** and select **Open Command Window Here** to open the command prompt window in the current project directory:

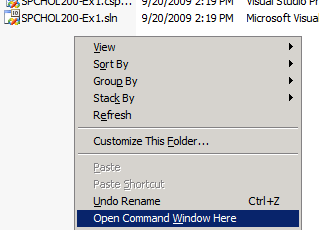


Figure 8 - Open Command Window Here

1. Type the following command in the command prompt and press **Enter** to set the path to the SharePoint 2010 folder:

**set path=%path%;c:\program files\common files\microsoft shared\web server extensions\14\bin**

1. Type the following command in the command prompt and press **Enter** to generate the Linq-to-SharePoint proxy code.

spmetal.exe /web:http://intranet.contoso.com /namespace:SPCHOL200\_Ex1.VisualWebPart1 /code:SPLinq.cs

**Note – you may get warnings about content types for list Form Templates. You can safely ignore this warning and continue**

1. Close the command window and switch back to Visual Studio.
2. In Visual Studio, right click on **SPCHOL200-Ex1** project and select **Add | Existing Item**.

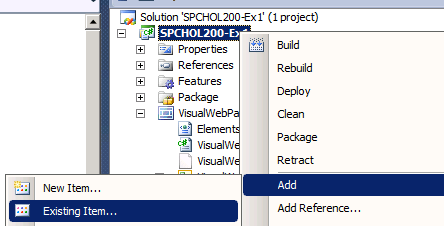


Figure 9 - Add Existing Item

1. Select **SPLinq.cs** from the **Add Existing Item** dialog window and click **Add:**

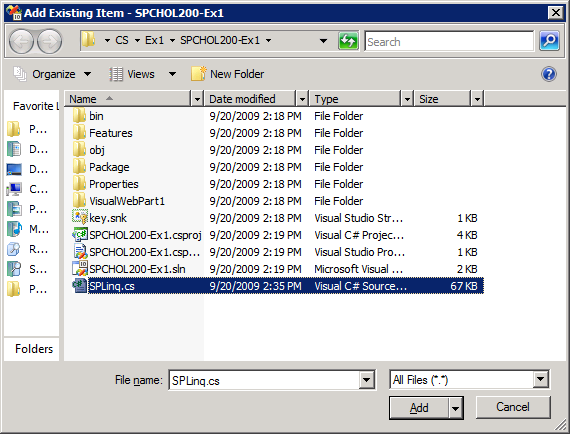
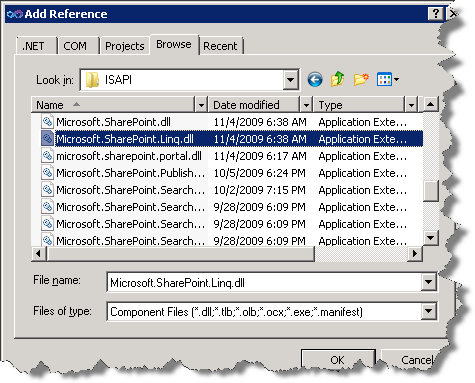


Figure 10 - Add SPLinq.cs File

1. In the Solution Explorer, right click on **References** and select **Add Reference.**
2. Switch to **Browse** tab and enter **C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\ISAPI** in the **File Name** text box. Press **Enter** to change directories. Your Add Reference window should now look like Fig. 11.
3. Select **Microsoft.SharePoint.Linq.dll.**



*Figure 11 - Add Reference*

1. Click **OK** to add the reference to your project.

### Task 3 - Access the SharePoint list data in Visual Web Part

In this task, you will add code to your solution that will allow the Visual Web Part to retrieve SharePoint list data.

1. In Solution Explorer, expand **VisualWebPart1** and double-click on **VisualWebPart1UserControl.ascx**.
2. Visual Studio will open the Visual Web Part User Control.
3. Add the following code to the user control to construct your grid view.

<%@ Import Namespace="Microsoft.SharePoint.WebControls" %>

<SharePoint:SPGridView id="spGridView" runat="server" AutoGenerateColumns="false">

<HeaderStyle HorizontalAlign="Left" ForeColor="Navy" Font-Bold="true" />

<Columns>

<SharePoint:SPBoundField DataField="Title" HeaderText="Title"></SharePoint:SPBoundField>

<SharePoint:SPBoundField DataField="JobTitle" HeaderText="JobTitle"></SharePoint:SPBoundField>

<SharePoint:SPBoundField DataField="ProjectTitle" HeaderText="ProjectTitle"></SharePoint:SPBoundField>

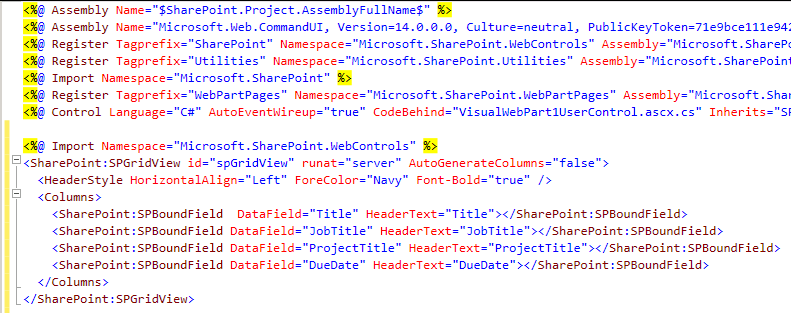
<SharePoint:SPBoundField DataField="DueDate" HeaderText="DueDate"></SharePoint:SPBoundField>

</Columns>

</SharePoint:SPGridView>

**Code Snippets:** *ASP.NET | spchol200\_ex1\_spgridview*

1. The Visual Web Part user control should look like this after adding the code above:



*Figure 12 - Visual Web Part User Control*

1. In the Solution Explorer, right click on **VisualWebPart1UserControl.ascx** and select **View Code**.
2. Add the following using statements to the code behind:

using Microsoft.SharePoint.Linq;

using Microsoft.SharePoint;

using System.Linq;

**Code Snippet**: *My Code Snippets | spchol200\_ex1\_namespaces*

1. Insert the following code inside the **Page\_Load** method:

var dc = new SPLinqDataContext(SPContext.Current.Web.Url);

var Employees = dc.GetList<EmployeesItem>("Employees");

var empQuery = from emp in Employees

where emp.Project.DueDate <

DateTime.Now.AddMonths(6)

select new

{

emp.Title,

emp.JobTitle,

ProjectTitle = emp.Project.Title,

DueDate =

emp.Project.DueDate.Value.ToShortDateString()

};

spGridView.DataSource = empQuery;

spGridView.DataBind();

**Code Snippet**: *My Code Snippets | spchol200\_ex1\_pageload*

### Task 4 – Build and Deploy the Visual Web Part

1. In the Solution Explorer, right click on **SPCHOL200-Ex1** and select **Deploy.** This will build and deploy the Visual Web Part to the local SharePoint site: <http://intranet.contoso.com>

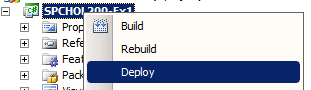


Figure 13 - Deploy VisualWebPart

1. Open Internet Explorer and browse to the following site:

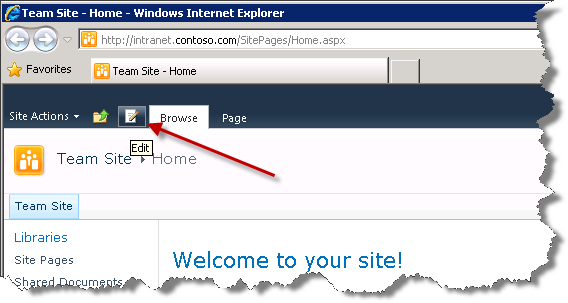
<http://intranet.contoso.com>

1. If prompted for authentication, enter the following details.

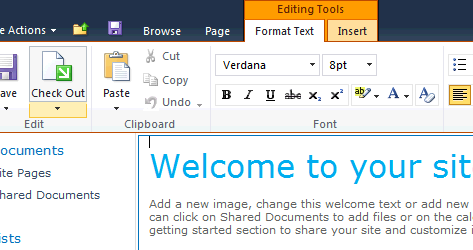
**Username**: Administrator

**Password**: pass@word1

1. Click the **Edit** icon in the top menu to open the SharePoint Ribbon to the Editing Tools.



*Figure 14 – SharePoint Edit Icon*



*Figure 15 - SharePoint Ribbon - Edit Page*

1. Switch to **Insert** tab in the Ribbon and click on **Web Part** to insert a Web Part to the page.

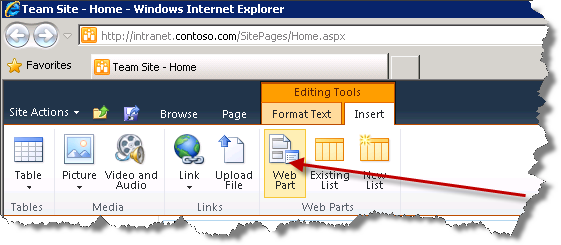


Figure 14 - SharePoint Ribbon - Insert WebPart

1. Under **Categories**, Select **Custom.**
2. Under **Web Parts**, select **SPLinqDemoTitle** web part.

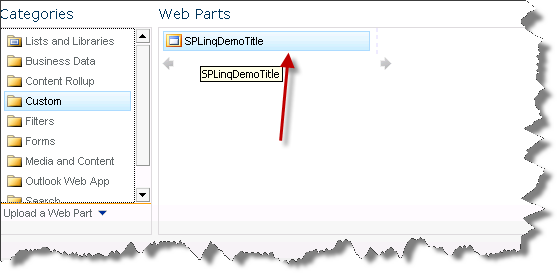
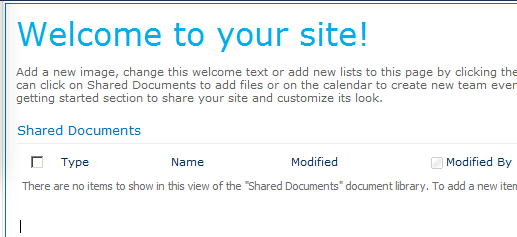
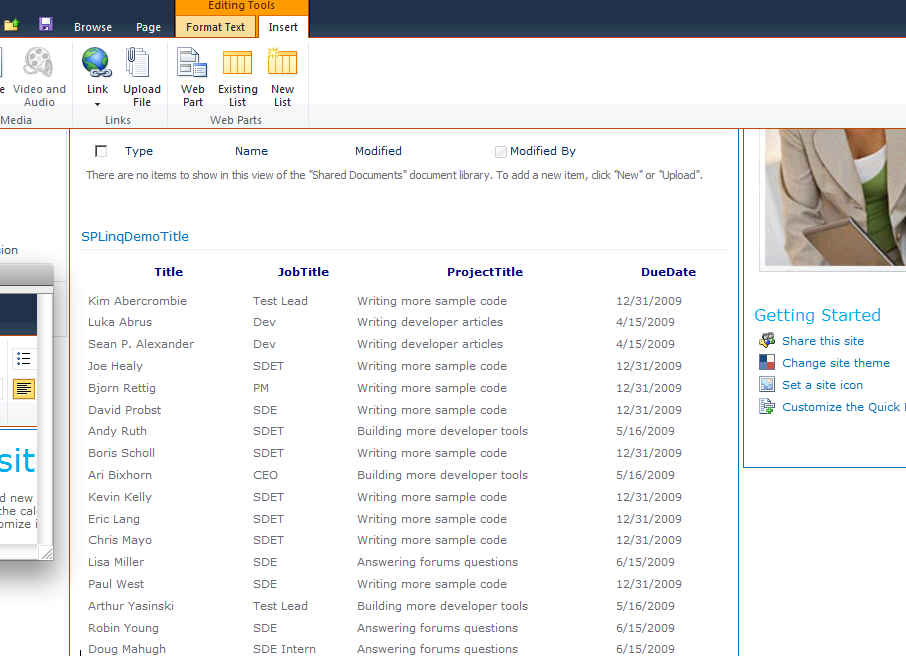


Figure 15 - Selecting SPlinqDemoTitle Web Part

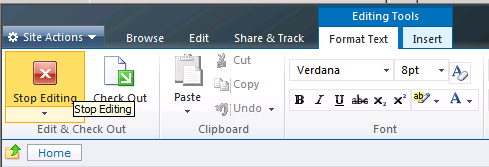
Put your cursor in the area of the page where you want the Web Part to appear. This must be a zone that accepts Web Parts. In this case, put your cursor in the zone below the Shared Documents control.

*Figure 16 – The insert cursor*

1. Click **Add** to add the web part to the page. This will add the **SPLinqDemoTitle** web part to the selected layout zone.

*Figure 17 - SPLinqDemoTitle Web Part added to the layout zone*

1. Click on **Page**, click the **down arrow** on the “**Save and Close”** button, and select **Stop Editing** to save the page and stop editing. Click **Yes** when prompted to save the changes you made.



*Figure 18 - SharePoint Ribbon - Stop Editing*

1. Close Internet Explorer.
2. Close Visual Studio 2010.

In the past exercise, you built and deployed a Visual Web Part that uses Linq to SharePoint to retrieve data from a SharePoint list.

## Exercise 2 – Connecting Web Parts

Estimated time to complete this exercise: **20 minutes**

In this exercise, you will:

1. Build two basic SharePoint Web Parts.
2. Configure a web part to participate in a web part connection as a provider.
3. Configure a web part to participate in a web part connection as a consumer.

### Task 1 – Create a new Empty SharePoint Project

In this task, you will create a new Empty SharePoint Project.

1. Open **Visual Studio 2010** by going to **the Start Menu | All Programs | Microsoft Visual Studio 2010 | Microsoft Visual Studio 2010**.
2. From the menu, select **File | New | Project**.
3. In the **New Project** dialog window, choose **Visual C# | SharePoint | 2010** from the Installed Templates.
4. Select **Empty SharePoint Project** from the Project Items.

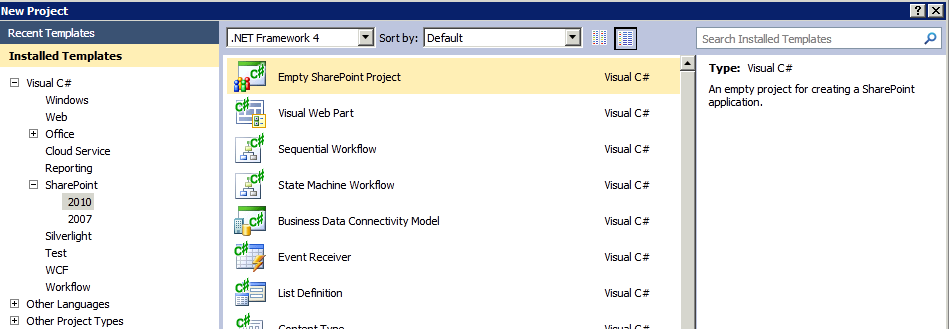
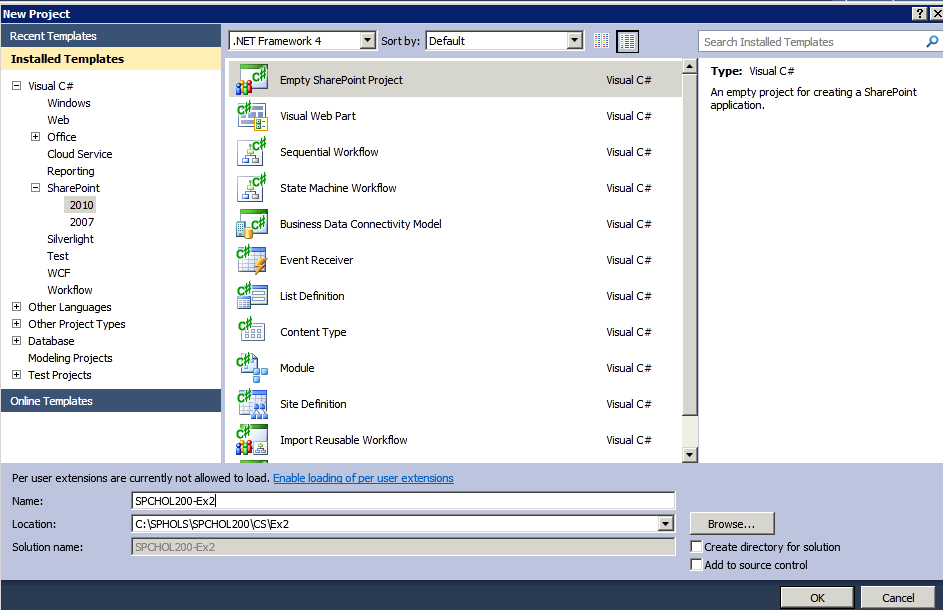


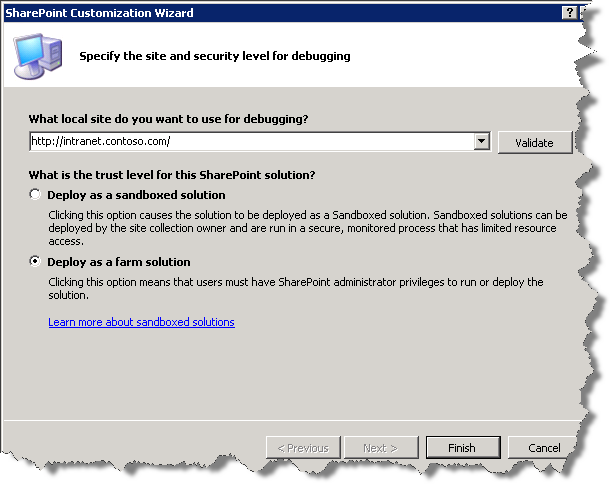
Figure 19 - SharePoint Empty Project

1. Enter ***SPCHOL200-Ex2*** in the **Name** textbox.
2. Enter ***C:\SPHOLS\SPCHOL200\CS\Ex2***in the **Location** textbox.



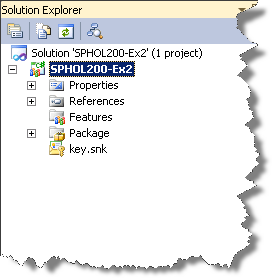
*Figure 20 - Visual Studio New Project Dialog Window*

1. Make sure **Create directory for solution** and **Add to source control** are both deselected.
2. Click **OK**.
3. In the **SharePoint Customization Wizard**:
   * Enter [**http://intranet.contoso.com**](http://intranet.contoso.com) for the local site.
   * Set the trust level to **Deploy as a farm solution.**



*Figure 21 - SharePoint Customization Wizard*

1. Click **Finish** to complete the wizard.
2. Visual Studio will create the new **SPCHOL200-Ex2** project and add the necessary files.

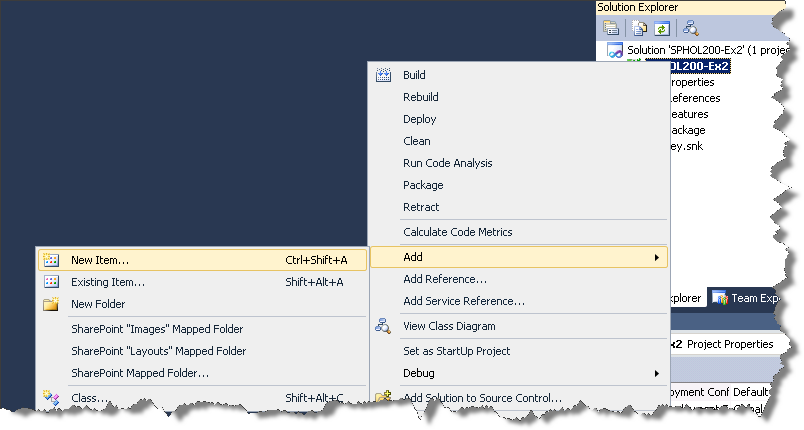


*Figure 22 - SPCHOL200-Ex2 Project*

### Task 2 – Create the Web Part Connection Interface

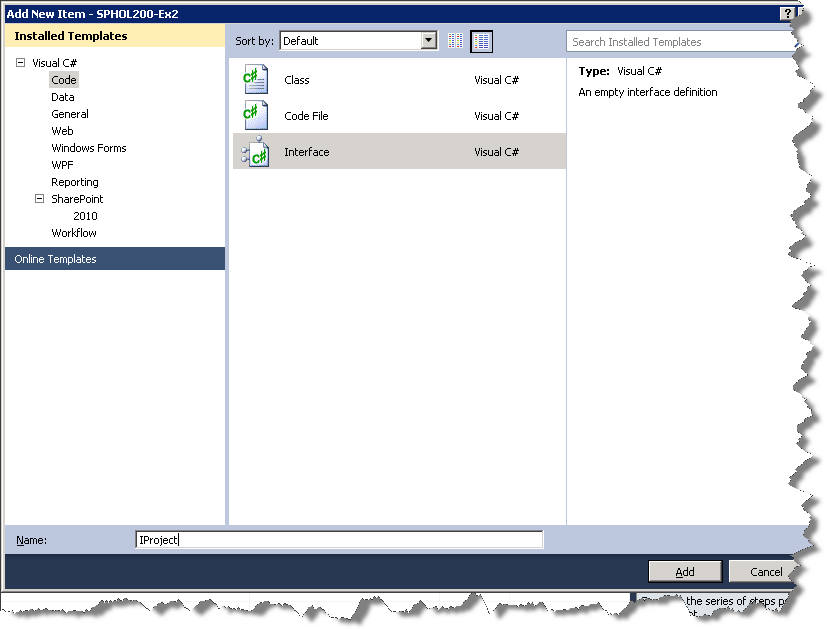
In this task, you will create the web part connection interface **IProject** responsible for exchanging connection information between a provider and consumer web part.

1. In the Solution Explorer, right click on **SPCHOL200-Ex2** and select **Add | New Item…**



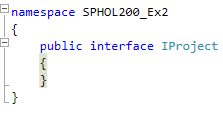
*Figure 23 - Add New Item*

1. In the **Add New Item** dialog window, select **Visual C# | Code** from the **Installed Templates**.
2. Select **Interface** from the available templates.
3. Enter ***IProject*** in the **Name** textbox and click the **Add** button.



*Figure 24 - IProject Interface Creation*

1. Visual Studio will add the new **IProject.cs** file to the project.
2. In the Solution Explorer, double-click on **IProject.cs** file.
3. Change the visibility of the interface to Public**.** Prefix the keyword ***public*** to the interface declaration:



*Figure 25 - Prefix public Keyword*

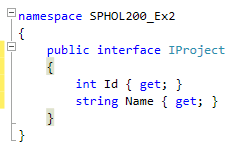
1. Insert the following code block inside the **IProject** interface:

int Id { get; }

string Name { get; }

**Code Snippet**: *My Code Snippets | spchol200\_ex2\_iproject*

1. The **IProject.cs** file should look like this after adding the above code block:

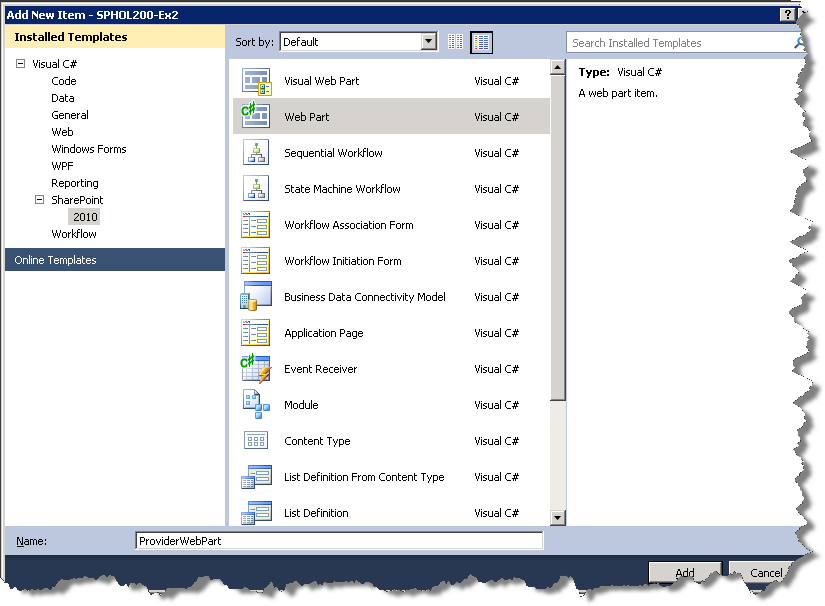


*Figure 26 - IProject Interface*

### Task 3 – Create the Provider Web Part

In this task, you will create a web part to participate in a web part connection as a provider.

1. In the Solution Explorer, right click on **SPCHOL200-Ex2** and select **Add | New Item…**
2. Select **Visual C# | SharePoint | 2010** from the **Installed Templates.**
3. Select **Web Part** from the available Item Templates.
4. Enter ***ProviderWebPart*** in the **Name** textbox and click **Add**.



*Figure 27 - Adding the Provider Web Part*

1. Visual Studio will add the new **ProviderWebPart** to the solution.

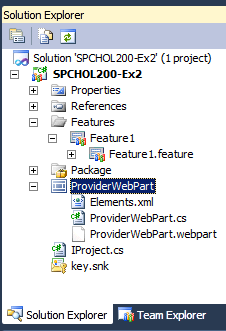


Figure 28 - Provider Web Part

1. In the Solution Explorer, double-click on **ProviderWebPart.cs** to open the code behind file.
2. In the **ProviderWebPart** class declaration, implement IProject by replacing the WebPart base class in the ProviderWebPart’s class inheritance block with the following:

Microsoft.SharePoint.WebPartPages.WebPart, IProject

**Code Snippet**: *My Code Snippets | spchol200\_ex2\_provider\_class*

1. The **ProviderWebPart** class declaration should like this after making the above code change:



Figure 29 – IProject implemented

1. Insert the following code after the **ProviderWebPart** class declaration. This code block implements the **IProject** web part connection interface and adds a local variable to the web part:

DropDownList \_projectPicker = null;

int IProject.Id

{

get { return int.Parse(\_projectPicker.SelectedValue); }

}

string IProject.Name

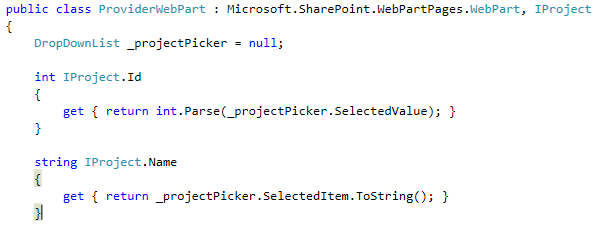
{

get { return \_projectPicker.SelectedItem.ToString(); }

}

**Code Snippet**: *My Code Snippets | spchol200\_ex2\_provider\_properties*

1. The **ProviderWebPart** class should look like this after making the above code change:



*Figure 30 - IProject Interface implemented and DropDownList Variable*

1. Insert the following code in the **CreateChildControls** method just after the **base.CreateChildControls** method call:

try

{

\_projectPicker = new DropDownList();

using (SPSite spSite = new SPSite(SPContext.Current.Web.Url))

using (SPWeb spWeb = spSite.OpenWeb())

{

SPList projectsList = spWeb.Lists["Projects"];

foreach (SPListItem project in projectsList.Items)

{

\_projectPicker.Items.Add(new ListItem(project.Title, project.ID.ToString()));

}

}

\_projectPicker.AutoPostBack = true;

this.Controls.Add(\_projectPicker);

}

catch (Exception ex)

{

this.Controls.Clear();

this.Controls.Add(new LiteralControl(ex.Message));

}

**Code Snippet**: *My Code Snippets | spchol200\_ex2\_provider\_createchildcontrol*

1. Insert the following **ConnectionProvider** property below the **CreateChildControls** method. This provides the Connection Provider interface point for the **ProviderWebPart:**

[ConnectionProvider("Project Name and ID")]

public IProject NameDoesNotMatter()

{

return this;

}

**Code Snippet**: *My Code Snippets | spchol200\_ex2\_provider\_connection*

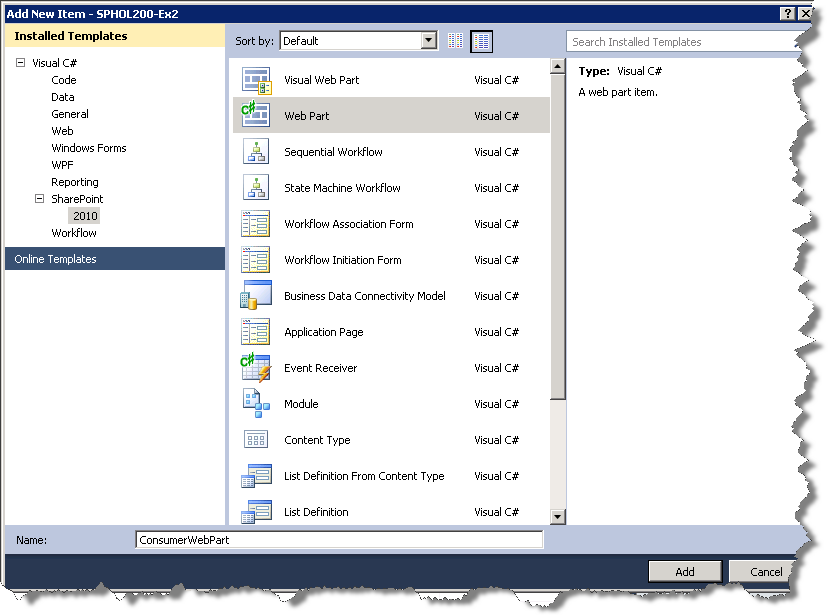
1. Do a build in Visual Studio 2010 by going to the **Build** menu and selecting **Build Solution**. If you’ve done everything correctly you will get a successful build message in the Output window.

In the past few minutes, you saw how to create a provider web part and add a connection provider interface point so that the consumer web part can receive messages from a provider web part.

### Task 4 – Create the Consumer Web Part

In this task, you will create a web part to participate in a web part connection as a consumer.

1. In the Solution Explorer, right click on **SPCHOL200-Ex2** and select **Add | New Item…**
2. Select **Visual C# | SharePoint | 2010** from the **Installed Templates.**
3. Select **Web Part** from the available Item Templates.
4. Enter ***ConsumerWebPart*** in the **Name** textbox and click **Add**.



*Figure 31 - Adding the ConsumerWebPart*

1. Visual Studio will add the new **ConsumerWebPart** to the solution.

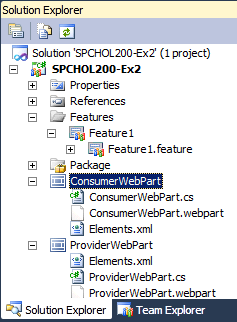


Figure 32 – ConsumerWebPart

1. Insert the following code inside the **ConsumerWebPart** class declaration:

IProject \_provider = null;

Label \_lbl = null;

**Code Snippet**: *My Code Snippets | spchol200\_ex2\_consumer\_variables*

1. Insert the following code in the **CreateChildControls** method:

try

{

\_lbl = new Label();

if (\_provider != null)

{

if (\_provider.Id > 0)

{

\_lbl.Text = \_provider.Name + " was selected.";

}

else

{

\_lbl.Text = "Nothing was selected.";

}

}

else

{

\_lbl.Text = "No Provider Web Part Connected.";

}

this.Controls.Add(\_lbl);

}

catch (Exception ex)

{

this.Controls.Clear();

this.Controls.Add(new LiteralControl(ex.Message));

}

**Code Snippet**: *My Code Snippets | spchol200\_ex2\_consumer\_createchildcontrol*

1. Insert the following **ConnectionConsumer** property below the **CreateChildControls** method. This provides the Connection Consumer interface point for the **ConsumerWebPart:**

[ConnectionConsumer("Project Name and ID")]

public void ThisNameDoesNotMatter(IProject providerInterface)

{

\_provider = providerInterface;

}

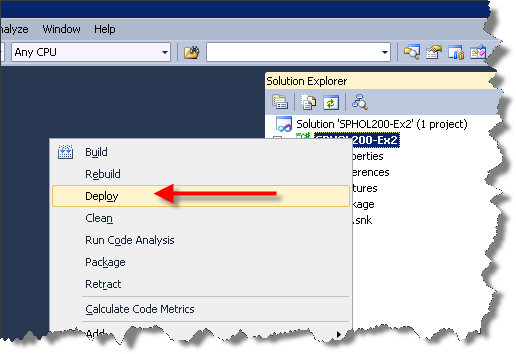
**Code Snippet**: *My Code Snippets | spchol200\_ex2\_consumer\_connection*

1. In the past few minutes you saw how to create a consumer web part and create a new connection consumer interface point, so that the consumer web part can receive messages from a provider web part.

### Task 5 – Build and Deploy the Web Parts

In this task, you will build and deploy the provider and consumer web part and also create a new Web Part Page to add the web parts.

1. In the Solution Explorer, right click on **SPCHOL2-Ex2** and select **Deploy.**



*Figure 33 - Deploy SPCHOL200-Ex2*

1. Open Internet Explorer and browse to the following site:

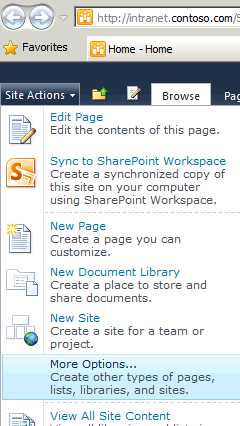
<http://intranet.contoso.com>

1. If prompted for authentication, enter the following details.

**Username**: Administrator

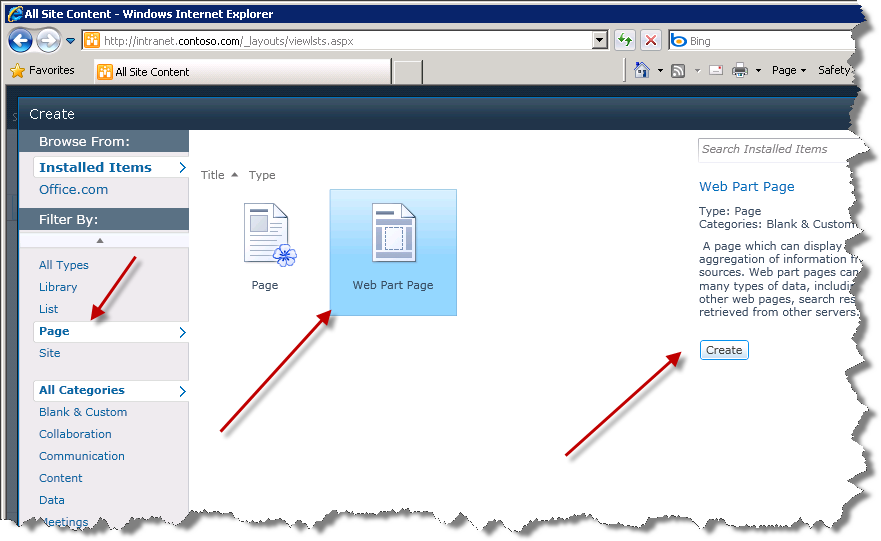
**Password**: pass@word1

1. Click on **Site Actions** menu and select **More Options…**



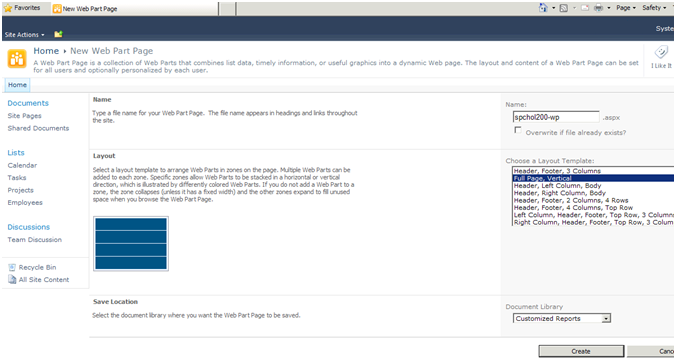
*Figure 34 - Site Actions - More Options*

1. Select **Web Part Page** from the **Pages** list. Press **Create**.



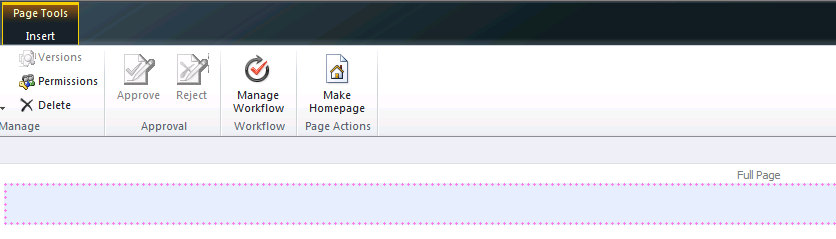
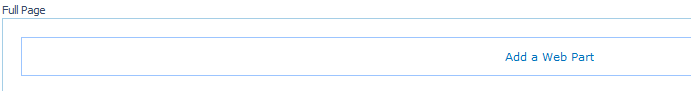
*Figure 35 - Create a new Web Part page*

1. In the **New Web Part Page**:
   * Enter ***spchol200-wp*** in the **Name:** textbox.
   * Choose **Full Page, Vertical** as the **Layout Template.**
   * Select **Customized Reports** from the **Document Library** drop-down menu.
   * Click **Create** button.



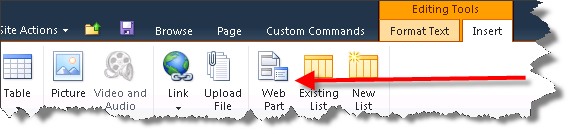
*Figure 36 - New Web Part Page Options*

1. SharePoint will create the new Web Part page and open the page in Edit mode.
2. Click on the Web Part zone (the blue box that says “Add a Web Part”).

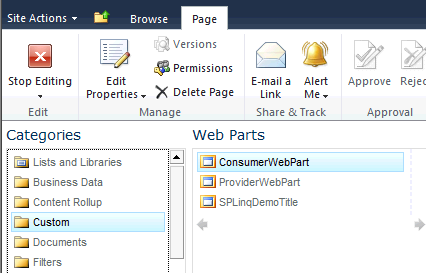
*Figure 37 - Web Part Zone*

1. Click on the **Insert** tab in the Ribbon and click **Web Part**



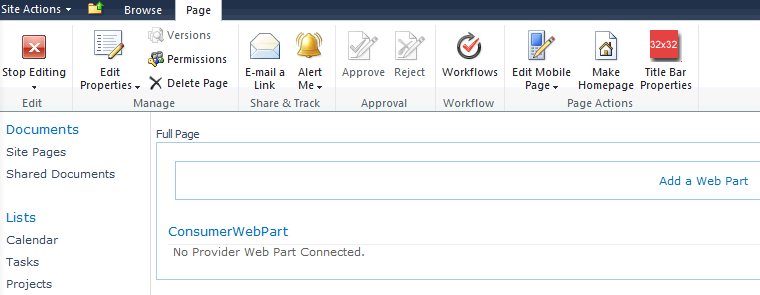
*Figure 38 - Insert Web Part*

1. Select **Custom** from **Categories**.
2. Select **ConsumerWebPart** and click **Add**.



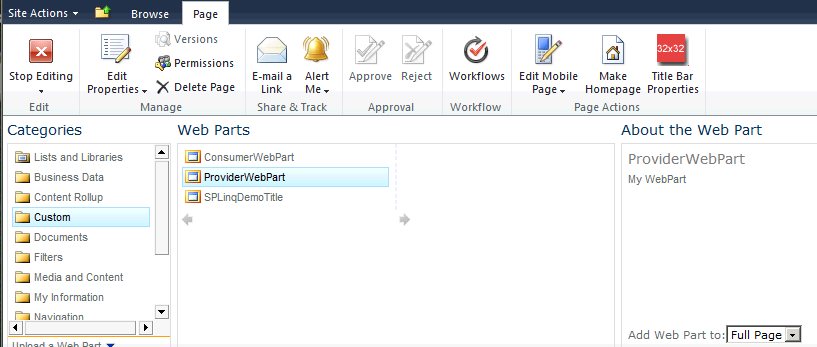
*Figure 39 - Add Consumer Web Part*

1. You should see the **ConsumerWebPart** added to the page.



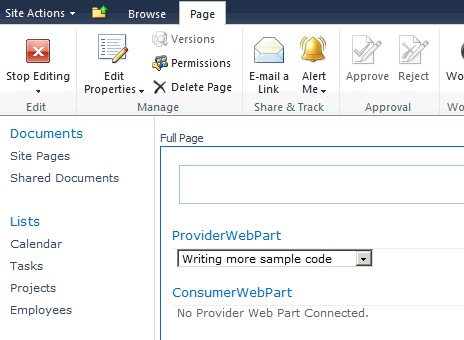
*Figure 40 - Consumer Web Part added to the page*

1. Click on **Add a Web Part** in the main body of the screen (see Figure 41).
2. Select **Custom** from **Categories**.
3. Select **ProviderWebPart** and click **Add**.



*Figure 41 – Add Provider Web Part*

1. You should see the **ProviderWebPart Title** added to the page.

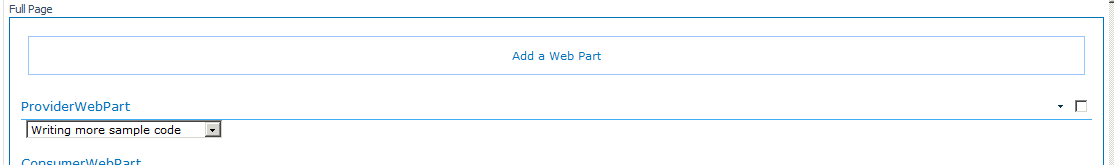


*Figure 42 - Provider Web Part added to the page*

### Task 6 – Connect the Web Parts

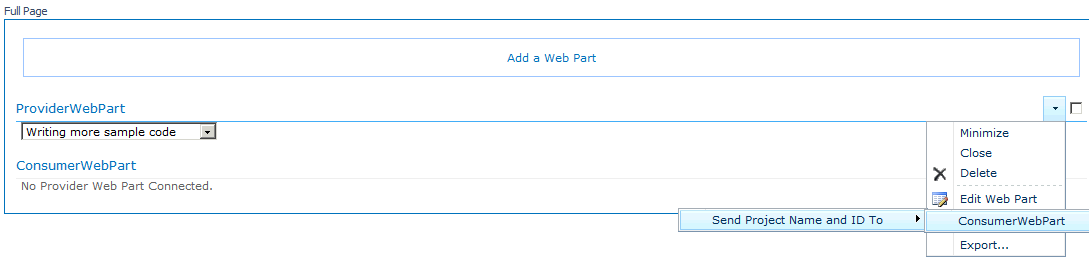
In this task, you will connect the provider and consumer web parts.

1. Hover over the **ProviderWebPart**, and a drop-down menu arrow will appear on the right side of the window,.Click on it to open the drop-down list, and move down to **Connections**..



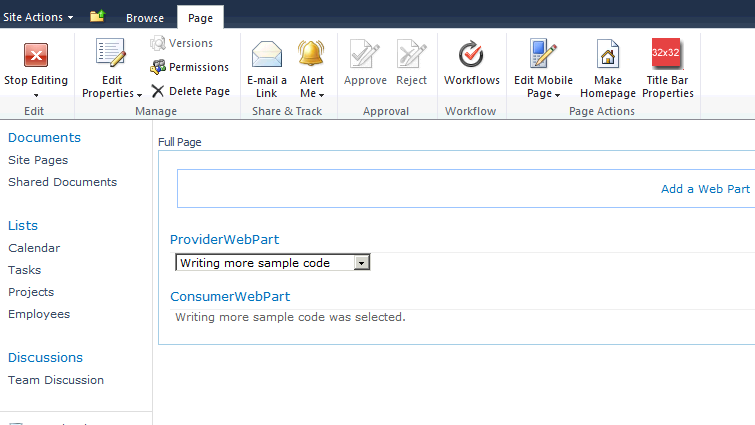
*Figure 43 - Provider Web Part Edit Button*

1. Hover your mouse to **Connections | Send Project Name and ID to | ConsumerWebPart** and click on **ConsumerWebPart.**



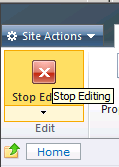
*Figure 44 - Web Part Connection Menu*

1. This will create a web part connection with the **ConsumerWebPart** web part, and the text of ConsumerWebPart will now say “Writing more sample code was selected”.



*Figure 45 - WebPart connection completed!*

1. Click on **Stop Editing** in the SharePoint Ribbon.

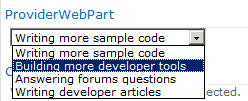


*Figure 46 - Stop Editing*

### Task 7 – Verify the Web Part Connection

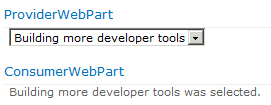
In this task, you will verify the web part connection between the provider and consumer web parts added to the web part page in **Task 6.**

1. In the **ProviderWebPart** web part, select **Building more developer tools** from the drop-down list.



*Figure 51 - Selecting a Project*

1. You should see the page being refreshed and **ConsumerWebPart** web part updated with the project selected (Building more developer tools) in the **ProviderWebPart** drop-down list.



*Figure 47 - Updated Consumer Web Part*

In the past few minutes, you saw how to connect two web parts and send a message from the provider web part to the consumer web part.

## Lab Summary

In this lab, you performed the following exercises:

Created a Visual Web Part Project.

Generated Linq-to-SharePoint proxy code.

Created Linq query to retrieve SharePoint List data.

Created two basic SharePoint Web Parts.

Configured a web part to participate in a web part connection as a provider.

Configured a web part to participate in a web part connection as a consumer.

In this lab, you learned how to create a SharePoint Visual Web Part from scratch. You also learned how to write a Linq-to-SharePoint query using the new Linq-to-SharePoint mode. You also learned how to create and configure web parts for web part connections.