

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

This walkthrough shows how to create a more full-featured site definition that incorporates several elements that you might add to a site definition. It is an employee locator site definition that enables you to locate an employee on a Bing map based on the employee's name and the region of the country where they work.

This walkthrough demonstrates the following tasks:

- Creating a site definition by using the Visual Studio project template.
- Adding and incorporating a custom master page in the solution.
- Adding a visual Web part to the solution.
- Adding custom fields to the solution.
- Adding a content type that includes the custom fields.
- Adding a list definition based on the content type.
- Adding a list instance and default data for the list that deploys with the site definition.
- Customizing the site's default.aspx page by adding a visual Web part to it.
- Customizing a user control on the visual Web part.
- Adding to the solution a custom logo image that appears on the visual Web part.

Note: Your computer might show different names or locations for some of the Visual Studio user interface elements in the following instructions. The Visual Studio edition that you have and the settings that you use determine these elements.

Prerequisites

You need the following components to complete this walkthrough:

- Supported editions of Microsoft Windows and SharePoint. For more information, see Requirements for Developing SharePoint Solutions.
- SharePoint Designer 2010.
- Visual Studio 2010.
- Bing Maps Platform Developer Account credentials. You can sign up for a free developer account at the Bing Maps Account Center.

Creating a Site Definition Solution

First, create the site definition project in Visual Studio.

To create a site definition project

1. Display the **New Project** dialog box by pointing to **New** on the **File** menu and then clicking **Project**.
2. Expand the **SharePoint** node under either **Visual C#** or **Visual Basic**, and then click **2010**.
3. In the templates pane, select **Site Definition**.
4. In the **Name** box, type Testsitedef2 and then click **OK**. The **SharePoint Customization Wizard** appears.

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

5. On the **Specify the site and security level for debugging** page, enter the URL for the SharePoint server site where you want to debug the site definition, or use the default location (`http://system name/`).
6. In the **What is the trust level for this SharePoint solution?** section, use the default value of **Deploy as a farm solution**. All site definition projects must be deployed as farm solutions. For more information about sandboxed solutions versus farm solutions, see *Sandboxed Solution Considerations*.
7. Click **Finish**. The project appears in **Solution Explorer**.

Adding a Logo Image

Next, add an image to the solution for use as a corporate logo in the site definition.

To add an image

1. Right-click the project node in **Solution Explorer**, point to **Add**, and then click **SharePoint "Images" Mapped Folder**.

Because the **Images** folder maps directly to the SharePoint file system, files added to it deploy to the SharePoint /images folder and become available for use in SharePoint.

2. Right-click the **TestSiteDef2** folder under **Images**, point to **Add**, and then click **Existing Item**. Select an image file to use as a logo and then click **Add**.

Alternatively, you can click **New Item** on the shortcut menu and create a new image, although your choice of image types is limited. It is typically better to create images by using another tool.

3. Repeat step 2 if you want to add other images to the solution. For example, you could add another image to act as a placeholder for the image control added later in this topic.

Adding a Custom Master Page

If you want your site definition to use a custom master page, you can add one to the solution.

To add a custom master page

1. Create a master page. For more information, see *ASP.NET Master Pages*.
2. Export and then import the custom master page into Visual Studio, as outlined in *Walkthrough: Import a Custom Master Page and Site Page with an Image*.
3. Add the master page to a module. To do this, right-click the project node in **Solution Explorer**, point to **Add**, and then click **New Item**.
4. In the **Add New Item** dialog box, in the list of SharePoint templates, select **Module**. Give the module a name.
5. In the module, delete the default `Sample.txt` file.
6. Add a folder to the module named `_catalogs`, and then another one under it called `masterpage`. This matches the file location of other master pages in SharePoint.

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

7. Add the master page under the **masterpage** folder. To do this, select the module node and then, on the **Project** menu, click **Add Existing Item**. Locate the master page and select it. Master page files have a .master file name extension.
8. Change the master page's **Deployment Conflict Resolution** setting to **Automatic**.
9. Double-click Elements.xml in the module to open it in the XML Designer. You must update the Elements.xml file to reference the master page.
10. Replace the existing module markup with the following markup.

```
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <Module Name="[Module Name]"
    Url="_catalogs/masterpage">
    <File Path="[Module Name]\_catalogs\masterpage\[Master Page
      Name].master"
      Url="[Master Page Name].master"
      Type="GhostableInLibrary" />
  </Module>
</Elements>
```

Be sure to replace the placeholder values with the actual names of the module and the master page.

11. Change the name of the master page in the default.aspx file. For example, if the master page is called "newmaster.master", you would use the following.

```
<%@ Page language="C#" MasterPageFile="~/_catalogs/masterpage/newmaster.master" In
herits="Microsoft.SharePoint.WebPartPages.WebPartPage,Microsoft.SharePoint,Version
=14.0.0.0,Culture=neutral,PublicKeyToken=71e9bce111e9429c" %>
```

12. Save the project.

Defining Custom Fields

Define custom fields that are used later to create a custom list. These fields provide additional data required by the code in the site definition.

To define custom fields

1. Right-click the site definition node in **Solution Explorer**, point to **Add**, and then click **New Item**.
2. Select **Empty Element** in the list of templates and name the element SiteColumns.
3. In the Elements.xml file, replace the markup with the following.

```
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <Field ID="{587DE9D5-618C-42C4-A754-06EA36CF7496}"
    Type="Text"
    Name="Latitude"
    DisplayName="Latitude"
    Group="Coordinates"/>
  <Field ID="{5D1CFF2C-8032-4792-AB34-8E0A0520B478}"
    Type="Text"
```

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

```
Name="Longitude"
  DisplayName="Longitude"
  Group="Coordinates"/>
<Field ID="{8B7E3CC0-40B5-465E-81FE-D63B7471B7CE}"
  Type="Text"
  Name="Region"
  DisplayName="Region"
  Group="Coordinates"/>
<Field ID="{8A152314-DB62-4EC7-96E7-E28F8A253B1D}"
  Type="Text"
  Name="StaffNumber"
  DisplayName="StaffNumber"
  Group="Coordinates"/>
</Elements>
```

This markup adds four custom fields to SharePoint: Latitude, Longitude, Region, and StaffNumber. These fields are used to identify employees and their locations. You can add more fields if you want. Use the **Create GUID** tool on the **Tools** menu to generate a unique GUID for each additional field.

Note: If you re-run this solution, comment out the field definitions to avoid errors.

4. Save the project.

Adding a Content Type

Create a content type that references the new fields.

To add a content type

1. Right-click the site definition node in **Solution Explorer**, point to **Add**, and then click **New Item**.
2. Select **Content Type** in the list of templates and name the new content type StaffListContentType.
3. On the **Choose Content Type Settings** page, select **Contact** in the list for the base contact type.
4. In the Elements.xml file for the content type, replace the markup with the following:

```
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <!-- Parent ContentType: Contact (0x0106) -->
  <ContentType ID="0x010600a024d6b9cf214430a254b1272eaedc4f"
    Name="TestSiteDef2 - StaffListContentType"
    Group="Custom Content Types"
    Description="My Content Type"
    Inherits="TRUE"
    Version="0">
    <FieldRefs>
      <FieldRef ID="{587DE9D5-618C-42C4-A754-06EA36CF7496}"
        Name="Latitude"
        DisplayName="Latitude"/>
      <FieldRef ID="{5D1CFF2C-8032-4792-AB34-8E0A0520B478}"
        Name="Longitude"
        DisplayName="Longitude"/>
      <FieldRef ID="{8B7E3CC0-40B5-465E-81FE-D63B7471B7CE}"
        Name="Region"
        DisplayName="Region"/>
      <FieldRef ID="{8A152314-DB62-4EC7-96E7-E28F8A253B1D}"
        Name="StaffNumber"
        DisplayName="StaffNumber"/>
    </FieldRefs>
  </ContentType>
</Elements>
```

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

```
</ContentType>  
</Elements>
```

5. Save the project.

Adding a List Definition

Create a list definition that references the new content type.

To add a list definition

1. Right-click the site definition node in **Solution Explorer**, point to **Add**, and then click **New Item**.
2. Select **List Definition from Content Type** in the list of templates and name the list definition StaffListDefinition.
3. On the **Choose List Definition Settings** page, use the default values for the display name and base content type. Clear the **Add a list instance for this list definition** box and then click **Finish**. This bases the list definition on the new content type. You will later create an instance of the list in the onet.xml file.
4. Double-click Schema.xml to open it. Notice that the custom fields are already referenced at the top of the file.

Adding Default List Data to onet.xml

To provide an instance of the new list definition and some default employee information to the employee list after the site definition is deployed, add data to the site definition's onet.xml file.

To add list data to onet.xml

1. Double-click the onet.xml file to open it.
2. Replace the <Lists/> tag with the following markup.

```
<Lists>  
  <List Title="StaffList"  
    FeatureId="fea29b33-8752-45d4-a5bd-6f7a54db7d49"  
    Url="Lists/TestSiteDef2-StaffList"  
    Description="" Type="10000">  
    <Data>  
      <Rows>  
        <Row>  
          <Field Name="FirstName">David</Field>  
          <Field Name="FullName">David Pelton</Field>  
          <Field Name="Region">Northwest</Field>  
          <Field Name="Longitude">-122.33</Field>  
          <Field Name="Latitude">47.60</Field>  
          <Field Name="StaffNumber">100200</Field>  
        </Row>  
        <Row>  
          <Field Name="FirstName">Joe</Field>  
          <Field Name="FullName">Joe Healy</Field>  
          <Field Name="Region">Southwest</Field>  
          <Field Name="Longitude">-115.13</Field>  
          <Field Name="Latitude">36.15</Field>  
          <Field Name="StaffNumber">100300</Field>  
        </Row>  
      </Data>  
    </List>  
</Lists>
```

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

```
<Row>
  <Field Name="FirstName">Lisa</Field>
  <Field Name="FullName">Lisa Andrews</Field>
  <Field Name="Region">Northeast</Field>
  <Field Name="Longitude">-73.98</Field>
  <Field Name="Latitude">40.75</Field>
  <Field Name="StaffNumber">100400</Field>
</Row>
<Row>
  <Field Name="FirstName">Chris</Field>
  <Field Name="FullName">Chris Ashton</Field>
  <Field Name="Region">Southeast</Field>
  <Field Name="Longitude">-84.38</Field>
  <Field Name="Latitude">33.76</Field>
  <Field Name="StaffNumber">100500</Field>
</Row>
</Rows>
</Data>
</List>
</Lists>
```

3. Set the scope for the feature to site level. To do this, double-click the feature file in **Solution Explorer** to open it in the Feature Designer. In the Feature Designer, set **Scope** to **Site**.
4. Save the project.

Adding a Reference to the Bing Maps Service

Call the Bing Maps Imagery service to provide a map to display the location of the selected employee.

To add the Bing Maps Imagery service to the solution

1. Add a service reference to the project. To do this, right-click the project node in **Solution Explorer** and click **Add Service Reference**.
2. In the **Add Service Reference** dialog box, enter the following URL in the **Address** box.

`http://dev.virtualearth.net/webservices/v1/imageryservice/imageryservice.svc?wsdl`
3. Click **Go**, type ImageryService in the **Namespace** box, and then click **OK**.
4. Delete the app.config file from the site definition. This file, added by the service, provides necessary endpoints and binding. However, it does not work properly with SharePoint solutions, so its functionality is added later through code.
5. The Bing Maps Web Services require that you have a Bing Maps key to make requests. Obtain a key by signing into your account at Bing Maps Account Center.

Creating a Visual Web Part with Controls

Next, create a visual Web part with controls to appear on the site definition's main page.

To create a visual Web part

1. Right-click the site definition node in **Solution Explorer**, point to **Add**, and then click **New Item**.

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

2. Select **Visual Web Part** in the list of templates and use the default value, VisualWebPart1, for the name of the Web part.
3. Under the existing markup in VisualWebPart1UserControl.ascx, add the following.

```
<style type="text/css">
    .style4
    {
        height: 205px;
        width: 187px;
    }
    .style5
    {
        height: 396px;
        width: 187px;
    }
    .style7
    {
        height: 205px;
        width: 291px;
    }
    .style8
    {
        height: 396px;
        width: 291px;
    }
</style>

<table style="width: 100%;">
    <tr>
        <td align="center" class="style4" valign="middle">
<asp:Image ID="Image1" runat="server" Height="200px"
    ImageUrl="~/_layouts/images/TestSiteDef2/logo.jpg" Width="350px" />
        </td>
        <td class="style7">
<asp:Label ID="Label1" runat="server" Font-Size="XX-Large"
    Font-Underline="True" Text="Tailspin Toys"></asp:Label>
            <br />
            <br />
<asp:Label ID="Label7" runat="server" Font-Size="X-Large"
    Font-Underline="False" Text="Employee Locator"></asp:Label>
        </td>
    </tr>
    <tr>
        <td class="style5" valign="top">
            <br />
<asp:Label ID="Label2" runat="server" Font-Size="Large" Text="Region:"></asp:Label
>
            <br />
            <asp:DropDownList ID="DropDownList1" runat="server" Height="24px"
    Width="352px" onselectedindexchanged="DropDownList1_SelectedIndexChanged"
                AutoPostBack="True">
                <asp:ListItem>Northeast</asp:ListItem>
                <asp:ListItem>Northwest</asp:ListItem>
                <asp:ListItem>Southeast</asp:ListItem>
```


Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

```
// Add the SharePoint server name below.
string webName =
    Context.Request.Url.AbsoluteUri.Replace("http://localhost/", "");
webName = webName.Replace("http://[SharePoint Server Name]/", "");
webName = webName.Replace("/default.aspx", "");

SPWeb web = site.AllWebs[webName];
SPList list = web.Lists["StaffList"];

ListBox1.Items.Clear();
foreach (SPListItem item in list.Items)
{
    if (item["Region"].ToString() == selectedRegion)
    {
        // LinkTitle == LastName -> see schema.xml
        ListBox1.Items.Add(item["StaffNumber"] + ": " +
            item["FullName"].ToString());
    }
}

protected void ListBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    // Reference the site based on its URL.
    SPSite site = new SPSite(Context.Request.Url.AbsoluteUri);

    // Add the SharePoint server name below.
    string webName =
        Context.Request.Url.AbsoluteUri.Replace("http://localhost/", "");
    webName = webName.Replace("http://[SharePoint Server Name]/", "");
    webName = webName.Replace("/default.aspx", "");

    SPWeb web = site.AllWebs[webName];

    // Reference the new list "StaffList" (as defined in onet.xml).
    SPList list = web.Lists["StaffList"];

    // Get the string entered in the listbox, split it up.
    // String format: ("XXXXXX: First Last")
    string fullName = ListBox1.SelectedItem.Text;
    string[] splitStr = fullName.Split(':');

    string longitude = "";
    string latitude = "";
    // Iterate through list, get the latitude/longitude
    // values for the selected employee's StaffNumber.
    foreach (SPListItem item in list.Items)
    {
        if (splitStr[0] == item["StaffNumber"].ToString())
        {
            longitude = item["Longitude"].ToString();
            latitude = item["Latitude"].ToString();

            break;
        }
    }
}
```

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

```
// Fetch the Bing map based on the selected employee's
// latitude and longitude.
ShowMap(longitude, latitude);
}

public void ShowMap(string Longitude, string Latitude)
{
    Image2.ImageUrl = GetMap(Latitude, Longitude, Latitude + ","
        + Longitude);
}

private string GetMap(string Latitude, string Longitude, string locationSt
ring)
{
    // Set the address to the Bing Maps imagery service.
    EndpointAddress address = new
        EndpointAddress("http://dev.virtualearth.net/webservices/v1/imager
yservice/imageryservice.svc");

    // Set up the binding, channel, and Bing Maps key.
    BasicHttpBinding binding = new BasicHttpBinding();
    binding.UseDefaultWebProxy = true;
    ChannelFactory<IImageryService> factory =
        new ChannelFactory<IImageryService>(binding, address);
    IImageryService channel = factory.CreateChannel(address);
    // Add your key below.
    string key = "[Bing Maps key]";
    MapUriRequest mapUriRequest = new MapUriRequest();

    // Set credentials using a valid Bing Maps key
    mapUriRequest.Credentials = new ImageryService.Credentials();
    mapUriRequest.Credentials.ApplicationId = key;

    // Set the location of the requested image.
    mapUriRequest.Center = new ImageryService.Location();
    string[] digits = locationString.Split(',');
    mapUriRequest.Center.Latitude = double.Parse(digits[0].Trim());
    mapUriRequest.Center.Longitude = double.Parse(digits[1].Trim());

    // Set the map style and zoom level.
    MapUriOptions mapUriOptions = new MapUriOptions();
    mapUriOptions.Style = MapStyle.AerialWithLabels;
    mapUriOptions.ZoomLevel = 17;

    // Set the size of the requested image in pixels.
    mapUriOptions.ImageSize = new ImageryService.SizeOfint();
    mapUriOptions.ImageSize.Height = 388;
    mapUriOptions.ImageSize.Width = 465;
    mapUriOptions.Style = MapStyle.AerialWithLabels;

    mapUriRequest.Options = mapUriOptions;

    // Add a pushpin to the current location.
    ImageryService.Pushpin[] MapPins = new ImageryService.Pushpin[1];
    MapPins[0] = new Pushpin();
    MapPins[0].IconStyle = "34";
    MapPins[0].Location = new Location();
```

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

```
MapPins[0].Location.Latitude = Convert.ToDouble(Latitude);
MapPins[0].Location.Longitude = Convert.ToDouble(Longitude);
mapUriRequest.Pushpins = MapPins;

// Open the channel and retrieve the map.
((IChannel)channel).Open();
MapUriResponse mapUriResponse = channel.GetMapUri(mapUriRequest);
((IChannel)channel).Close();
return mapUriResponse.Uri;
}
}
```

Adding the Visual Web Part to the default.aspx Page

Next, add the visual Web part to the site definition's default.aspx page.

To add a visual Web part to the default.aspx page

1. Open the default.aspx page and add the following markup under the WebPartPages tag.

```
<%@ Register Tagprefix="MyWebPartControls" Namespace="TestSiteDef2.SiteDefinition.
VisualWebPart1" Assembly="$SharePoint.Project.AssemblyFullName$" %>
This line associates the name MyWebPartControls with the Web part and its code.
The Namespace parameter is the same as the namespace used in the
VisualWebPart1Usercontrol.ascx code file.
```

2. After the </asp:Content> element, replace the entire ContentPlaceHolderId="PlaceHolderMain" section and its contents with the following markup.

```
<asp:Content ID="Content1" ContentPlaceHolderId="PlaceHolderMain" runat="s
erver">
  <h1>
    Welcome to Tailspin Toys
  </h1>
  <MyWebPartControls:VisualWebPart1 runat="server" />
</asp:Content>
```

This markup creates a reference to the visual Web part that you created earlier.

Running and Deploying the Site Definition Solution

Next, run the project and deploy it to SharePoint.

To run and deploy the site definition

Press F5. Visual Studio compiles the code, adds the project's features, packages all of the files into a .wsp file, and deploys the .wsp file to SharePoint Server. SharePoint then installs the files, activates the features, and displays the **New SharePoint Site** page.

Creating a Site Based on the Site Definition

Create a new site by using the new site definition.

To create a site by using the site definition

1. On the SharePoint site, the **New SharePoint Site** page appears.

Creating a Site Definition with Additional Content in Visual Studio 2010

(Microsoft Corporation)

2. In the **Title and Description** section, enter My New Site for the title and a description of the site.
3. In the **Web Site Address** section, enter mynewsite in the **URL name** box.
4. In the **Template** section, click the **SharePoint Customizations** tab, and then select **TestSiteDef2** in the **Select a template** list.
5. Leave the other settings at their default values and then click **Create**.
The new site appears.

Testing the New Site

Next, test the new site to make sure that it works correctly.

To test the new site

1. In the box under **Region**, click one of the locations. An employee name for that region appears in the **Staff** list.
2. Click the employee's name in the **Staff** list. A map location for the employee appears in the image control.

Note: If the Bing Maps service is unavailable, you can get an "EndPointNotFoundException" error. If this happens, try again later.