Lab 04: Creating Master Pages & Customizing Navigation

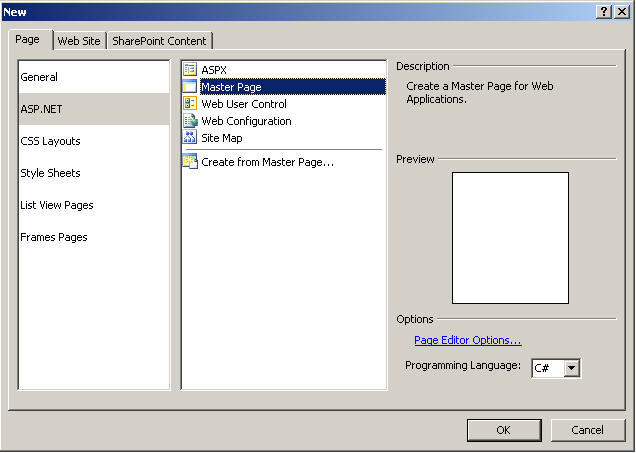
**Lab Time:** 45 minutes

**Lab Overview:** In this lab you will walk through creating and adding a master page to the Master Page Gallery two ways: first using SharePoint Designer and second, using Visual Studio and Features.

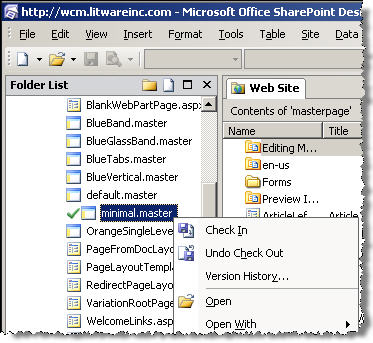
Exercise 1: Creating a minimal master page with SharePoint Designer

In this exercise you will create a minimal master page using Office SharePoint Designer 2007. Creating master pages this way will start the page off as a customized page rather than one based off a template.

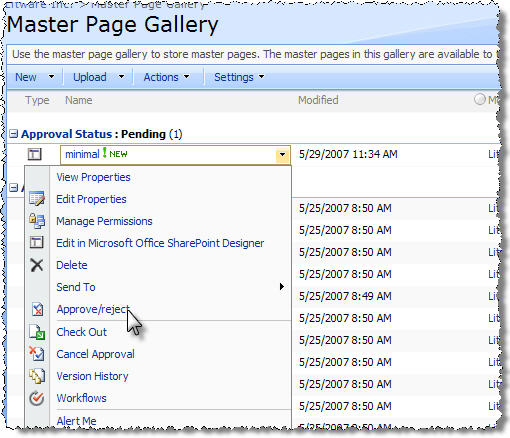
1. Open **SharePoint Designer** by selecting **Start » All Programs » Microsoft Office » SharePoint Designer 2007**.
2. If SharePoint Designer automatically loads a site when started, close the site by selecting **File » Close Site**.
3. Open the Publishing site created in a prior lab by selecting **File » Open Site...**. In the **Open Site** dialog, enter **http://wcm.litwareinc.com** in the **Site name** field and click **Open**.
4. To create a new master page, select **File » New...**. In the **New** dialog, select the **Page** tab, then **ASP.NET** in the first column, **Master Page** in the middle column and finally click **OK** (refer to the image below):



1. Next, open the sample minimal master page **minimal.master** file located in the **Resources** folder within this lab (**c:\Student\Labs\04\_MasterPagesNavigation\Resources\minimal.master**) in something other than SharePoint Designer (notepad or Visual Studio will work just fine) because it is not made to open SharePoint master pages from the file system.
2. Copy the entire contents of the **minimal.master** file and paste them into the new master page created in step 4, replacing the default contents added by SharePoint Designer.
3. Save the changes to the master page by selecting **File » Save As**.... You want to save the master page into the site's **Master Page Gallery**. Make sure the **Current Site** button is selected in the **Save As** dialog on the left panel and browse to the following path within the site: **http://wcm.litwareinc.com/\_catalogs/masterpage**. Name the new master page **minimal.master** using the **Field name:** field and click **Save**.
4. At this point the master page is still checked out and unpublished. While you are the one that created it so you can immediately use it within the site, anyone else hitting the site will receive an error because the file will not be checked in and/or published. In your local development environment this may not be an issue, but for good practice, go through the process of checking in and publishing the master page. Right-click **minimal.master** in the **Folder List** tool window and select **Check In** as shown in the following image:



Then select **Publish a major version** and click **OK**. A dialog will appear asking if you want to view/modify the approval status of the master page. Click **Yes** which will open a new browser window loading the Master Page Gallery with your master page at the top of the list. From the ECB menu of the minimal master page, select **Approve/reject**, as shown in the following image:



1. On the **Master Page Gallery: minimal page**, select **Approved**. This item will become visible to all users.
2. At this point the master page has been approved and can now be seen by anyone browsing the site. Configure the site to use the new minimal master page by selecting **Site Actions » Site Settings » Modify All Site Settings**.
3. On the **Site Settings** page, select **Master page** from the **Look and Feel** section, select **minimal.master** for the **Site Master Page** and click **OK**. Click the **Litware Inc.** logo at the top-left corner of the page to browse back to the homepage of the site to see the new master page in use.

At this point you have now created a master page using SharePoint Designer. The downside to this approach is that your master page has started off as a customized page that lives exclusively in the SharePoint content database. In the next exercise you will create a master page that starts as a page template using Visual Studio and WSS Features.

Exercise 2: Creating a minimal master page with Visual Studio and Features

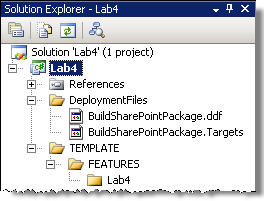
In this exercise you will create a master page template and deploy it with a SharePoint Feature. Similar to the Visual Studio project created in lab 1 exercise 2, in this exercise you will create a Visual Studio project that does not compile any code but simply packages all the files into a WSS solution package.

1. Open Visual Studio and create a new **C# Empty Project** project named **Lab4** (the **Empty Project** template is available when you highlight the **Visual C#\Windows** project type in the **New Project** dialog). You should create the project within the following path so the project files will reside within the same directory structure as all the other lab exercises:

c:\Student\Labs\04\_MasterPagesNavigation\Lab

*Notice in this case you are using the Empty Project template rather than the Class Library template for this project. The reason why is that the Empty Project template is a little more slimmed down.*

1. Now you need to create the folder structure that will contain the Feature. The Feature you will create is going to live within the Features folder nested within SharePoint's "12" system folder. To make your life easier, create a mirror of the "12" system folder structure from the TEMPLATE folder on down (**TEMPLATE\FEATURES\Lab4**). In addition, create a folder at the root of the project named **DeploymentFiles** and copy **BuildSharePointPackage.ddf** and **BuildSharePointPackage.targets** from the lab **Resources** folder into this directory. The following image shows what your project should look like after creating the necessary folder structure:



1. The next step is to create the necessary files for your feature. The first file to add is the master page. Copy the files **Sample.master** & **SampleMasterPreview.gif** from the lab **Resources** folder into the **Lab4** Feature folder within the "12" system folder in the Visual Studio project.
2. Now create the feature definition file by adding a new **XML** file to the project and naming it **feature.xml**. This file should reside in the same **Lab4** folder as the master page and image. Add the following XML markup to the feature.xml file:

<?xml version="1.0" encoding="utf-8" ?>

<Feature xmlns="http://schemas.microsoft.com/sharepoint/"

Id="30D0CA6D-899A-4d31-B8D8-1FE4844E2380"

Title="Lab 4 - Working with Provisioned Master Pages"

Hidden="FALSE"

Scope="Site"

Version="1.0.0.0">

<ElementManifests>

<ElementManifest Location="elements.xml" />

<ElementFile Location="Sample.master" />

<ElementFile Location="SampleMasterPreview.gif" />

</ElementManifests>

</Feature>

1. Next, you need to add the elements file that will provision the master page and preview image into the Master Page Gallery and associate the preview image with the master page. Add a new XML file named **elements.xml** and add it to the same **Lab4** folder as the other files. Add the following XML markup to the elements.xml file (the HTML/XML comments are not necessary to enter... they are there to explain each piece of the XML):

<?xml version="1.0" encoding="utf-8" ?>

<Elements xmlns="http://schemas.microsoft.com/sharepoint/">

<!-- add the master page to the Master Page Gallery -->

<Module Url="\_catalogs/masterpage" RootWebOnly="TRUE">

<!-- provision the master page into the Master Page Gallery -->

<File Url="Sample.master"

Name="SampleFromFeature.master"

Type="GhostableInLibrary">

<!-- specify the master page's list item content type -->

<Property Name="ContentType"

Value="$Resources:cmscore,contenttype\_masterpage\_name;"/>

<!-- specify the URL to the preview image, provisioned below -->

<Property Name="PublishingPreviewImage"

Value="~SiteCollection/\_catalogs/masterpage/Preview Images/Litware/SampleMasterPreviewFeature.gif, ~SiteCollection/\_catalogs/masterpage/Preview Images/Litware/SampleMasterPreviewFeature.gif" />

<!-- specify the title of the master page -->

<Property Name="Title"

Value="SampleFromFeature.master" />

</File>

</Module>

<!-- Add the preview image to the Master Page Gallery -->

<Module Url="\_catalogs/masterpage/Preview Images/Litware"

RootWebOnly="TRUE">

<!-- provision the preview image into the Master page gallery -->

<File Url="SampleMasterPreview.gif"

Name="SampleMasterPreviewFeature.gif"

Type="GhostableInLibrary">

<!-- specify the title of the preview image -->

<Property Name="Title"

Value="SampleMasterPreviewFeature.gif" />

</File>

</Module>

</Elements>

1. Now you need to create the WSS solution package. The following few steps are similar to the steps outlined in the first lab, but not exact. Because you used the Empty Project template, there will be a slight difference as your build process in this lab won't actually compile anything. To start, you need to edit the project file to call the custom targets file.
2. You first need to unload the project by right-clicking it in the **Solution Explorer** tool window and selecting **Unload Project**. With the project unloaded, right-click the project again in the **Solution Explorer** tool window and select **Edit Lab4.csproj**.
3. Here is where things are a bit different from the first lab. In the opening XML **<Project>** element, change the attribute **DefaultTargets** value from **Build** to **BuildSharePointPackage**.
4. Now, scroll down to the only XML **<Import>** element. Change the **Project** attribute value to be the following:

<Import Project="DeploymentFiles\BuildSharePointPackage.targets" />

Save all your changes and close the **Lab4.csproj** file.

1. Right-click the project in the **Solution Explorer** tool window and select **Reload project**. You may receive prompts to close the project file (select **OK**) and a security warning (select **Load project normally**, uncheck **Ask me for every project in this solution** and click **OK**).
2. SharePoint expects to find a manifest.xml file within the root of every WSS solution package. Add a new XML file named **manifest.xml** to the **DeploymentFiles** folder in the project and add the following XML code to the file:

<?xml version="1.0" encoding="utf-8" ?>

<Solution xmlns="http://schemas.microsoft.com/sharepoint/"

SolutionId="29254E7F-954B-4312-9108-66C3919754F0"

DeploymentServerType="WebFrontEnd"

ResetWebServer="FALSE">

<FeatureManifests>

<FeatureManifest Location="Lab4\feature.xml" />

</FeatureManifests>

</Solution>

1. At this point all the files necessary for deployment have been created, filled with the necessary information (including our Feature) and your Visual Studio project has been configured to automatically build WSS solution packages. Now you need to edit the \*.ddf file to tell makecab.exe to include the necessary files. Add the following code between the comments (the two lines with semicolons) in the **DeploymentFiles\BuildSharePointPackage.ddf** file:

DeploymentFiles\manifest.xml

.Set DestinationDir=Lab4

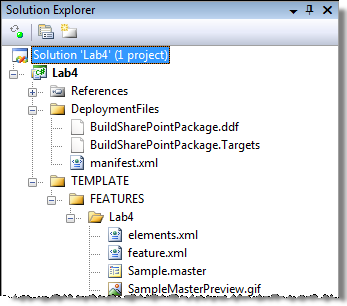
TEMPLATE\FEATURES\Lab4\feature.xml

TEMPLATE\FEATURES\Lab4\elements.xml

TEMPLATE\FEATURES\Lab4\Sample.master

TEMPLATE\FEATURES\Lab4\SampleMasterPreview.gif

The Visual Studio project should now look like the following image:



1. Now it's time to build deploy the WSS solution package. Build the project in **Visual Studio** to create the WSP file.
2. First the WSS solution package must be deployed. Open a command prompt and navigate to the following directory:

c:\Program Files\Common Files\Microsoft Shared\web server extensions\12\BIN

1. Enter the following command into the command line window and hit Enter:

stsadm -o addsolution -filename c:\Student\Labs\04\_MasterPagesNavigation\Lab\wsp\Debug\Lab4.wsp

1. Launch **Central Administration** by selecting **Start » All Programs » Microsoft Office Server » SharePoint 3.0 Central Administration**.
2. From the **Central Administration** site, select the **Operations** tab and then select **Solution management** under the **Global Configuration** section.
3. On the **Solution Management** page, click the link on **lab4.wsp**.
4. On the **Solution Properties** page, select **Deploy Solution**.
5. On the **Deploy Solution** page, specify **Now** in the **Deploy When?** section and click **OK**.
6. Test the Feature by browsing to the **http://wcm.litwareinc.com/** site and select **Site Actions » Site Settings » Modify All Site Settings**.
7. On the **Site Settings** page, select **Site collection features** under the **Site Collection Administration** section.
8. On the **Site Collection Features** page, click **Activate** on the **Lab 4 - Working with Provisioned Master Pages Feature**. After activating the Feature, click the **Home** link in the top navigation bar.
9. Now change the master page for the Publishing site. Select **Site Actions » Site Settings » Modify All Site Settings**, then select **Master page** from the **Look and Feel** section on the **Site Settings** page. Select the **SampleFromFeature.master** master page in the **Site Master Page** section (yes, the preview image isn't the right size, but you see how to create a preview image now). Finally, browse to the homepage of **http://wcm.litwareinc.com** to see the new master page. You should see a slightly different master page with an orange bar in the upper left (*this is a copy of the OrangeSingleLevel.master master page*).
10. Finally, take a look at the files provisioned. Select **Site Actions » Manage Content and Structure**. On the **Site Content** **and Structure** page, select the **Master Page Gallery**. Once the page refreshes, notice the master page you added at the bottom of the list. Additionally, select the subfolder **Preview Images** in the **Master Page Gallery**. Within the **Litware** folder you will find the preview image.

You have now created a master page using the template method. The file within the Master Page Gallery is still just referencing the template on the file system (within the Feature folder **Lab4**) until it is customized using SharePoint Designer.