

01 说说 MOSS 中的母版页

MOSS 中有两种页面：Site Pages 和 Application Pages，他们分别使用不同的母版页，Site Pages 使用的是

Default.master， Application Pages 使用的是 Application.master。我们下面讨论的主要是针对 default.master 的，因为 Application.master 是不支持被定制的。

Default.master 在安装目录的 C:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\GLOBAL\目录下，站点中引用的相对路径为

/_catalogs/masterpage/default.master。MOSS 中

母版页是可以嵌套的：<%@ Master master=MyParent.master %>。不能添加 webpart 区域，但可以添加静

态的 wetpart 。

该母版页中主要有以下几部分：

1. 链接，菜单，图标和导航控件。

MOSS 已经给我们提供了很多封装好的控件，比如 SPWebPartManager，导航的控件等等。

2. 占位符(Named Placeholds)

我们可以利用占位符来给继承自 Default.master 的页面添加内容，下面会有例子。

3. Delegate controls

这个词就不翻译了，使用它可以替换掉母版页中的内容。

下面是 Default.master 中的一段代码：

```
%@Master language="C#"%  
  
<%@ Register Tagprefix="SharePoint" Namespace="Microsoft.SharePoint.WebControls"  
Assembly="Microsoft.SharePoint, ..." %>  
  
<HTML runat="server">  
  
  <HEAD runat="server">  
  
    <!-- SharePoint 控件-->  
  
    <SharePoint:CssLink ID="CssLink1" runat="server"/>  
  
    <SharePoint:Theme ID="Theme1" runat="server"/>
```

```

    <SharePoint:ScriptLink language="javascript" name="core.js" Defer="true"
runat="server"/>

    <!-- Named Placeholders -->

    <Title ID=onetidTitle>

    <asp:ContentPlaceHolder id=PlaceHolderPageTitle runat="server"/>

    </Title>

    <asp:ContentPlaceHolder id="PlaceHolderAdditionalPageHead" runat="server"/>

    <!-- Named Delegate Control -->

    <SharePoint:DelegateControl ID="DelegateControl1" runat="server"
ControlId="AdditionalPageHead"
    AllowMultipleControls="true"/>

</HEAD>

```

Head 中的 CssLink 和 Theme, ScriptLink 都是 moss 中标准的服务器控件, 这两个控件在 application.master 中也是有的.

Head 中有两个 Named Placeholders, ID 分别为 PlaceHolderPageTitle 和 PlaceHolderAdditionalPageHead,

PlaceHolderPageTitle 表示页面的标题, PlaceHolderAdditionalPageHead 表示页面的 Head 区域, 你可以添加类似

<meta>的标签, 示例如下:

```

<%@ Page MasterPageFile="~/masterurl/default.master" %>

<asp:Content ID="PageTitle" runat="server" ContentPlaceHolderID="PlaceHolderPageTitle">

    My Custom Page Title

</asp:Content>

<asp:Content ID="AdditionalPageHead" runat="server"
ContentPlaceHolderID="PlaceHolderAdditionalPageHead">

    <META name="keywords" content="Software, Consulting, Money, Fame" />

</asp:Content>

```

下面是 MOSS 中已定义的 Named placeholders

占位符的 Name	描述
PlaceholderAdditionalPageHead	需要写在页面<head>标签里的附加内容，如引用的脚本或样式文件
PlaceholderBodyAreaClass	附加在页面顶部的 body 中的样式
PlaceholderBodyLeftBorder	页面 body 的边框元素
PlaceholderBodyRightMargin	页面 body 的右边距
PlaceholderCalendarNavigator	在页面中有日历时为其显示一个日期选择框
PlaceholderFormDigest	这是页面中必备的 "form digest"安全组件
PlaceholderGlobalNavigation	站点导航
PlaceholderHorizontalNav	导航标签
PlaceholderLeftActions	左侧导航区下面的动作区
PlaceholderLeftNavBar	左侧导航区
PlaceholderLeftNavBarBorder	左侧导航区的边框元素
PlaceholderLeftNavBarDataSource	左侧导航区菜单的数据源
PlaceholderLeftNavBarTop	左侧导航区上面的导航区
PlaceholderMain	页面主体
PlaceholderMiniConsole	一个放置页面级命令的地方，比如在 WIKI 站点里的 Edit Page, History, Incoming Links
PlaceholderNavSpacer	左侧导航区的宽度
PlaceholderPageDescription	页面描述区
PlaceholderPageImage	页面左上的图标
PlaceholderPageTitle	页面的<Title>，通常显示在浏览器的标题栏
PlaceholderSearchArea	搜索框
PlaceholderSiteName	站点名称
PlaceholderTitleAreaClass	TitleArea 附加的样式
PlaceholderTitleAreaSeparator	TitleAreaSeparator 区
PlaceholderTitleBreadcrumb	TitleBreadcrumb 区
PlaceholderTitleInTitleArea	Breadcrumb 区下面的标题
PlaceholderTitleLeftBorder	Title 区左侧边框
PlaceholderTitleRightMargin	Title 区右侧空白
PlaceholderTopNavBar	标签导航区
PlaceholderUtilityContent	页面底部需要的一块特殊内容
SPNavigation	在 Windows SharePoint Services 中默认为空，用于附加的页面编辑控件
WSSDesignConsole	页面编辑控件，当页面进入编辑页面模式时使用(当我们点 Site Actions, Edit Page 后)

导航组件：

MOSS 提供了很多标准的导航组件，如 `NavigationProvider`, `SPSiteMapProvider`, `SPContentMapProvider`, 和 `SPXmlContentMapProvider`，所有被激活的你都可以在站点的 `web.config` 的 `siteMap` 节点中配置，示例如下：

```
<siteMap defaultProvider="CurrentNavSiteMapProvider" enabled="true">
  <providers>
    <add name="SPNavigationProvider"
      type="Microsoft.SharePoint.Navigation.SPNavigationProvider,
        Microsoft.SharePoint, Version=12.0.0.0, Culture=neutral, PublicKeyToken=71e9bce111e9429c"
    />
  </providers>
</siteMap>
```

顶部链接栏和快速启动菜单是 `Default.master` 中定义的两个主导航组件。顶部链接栏是即 `AspMenu`。

它由数据

源为 `SiteMapDataSource` 控件的 `SPNavigationProvider` 的定义的。快速启动和顶部链接栏是以同样的方式定义

的，他们之间的区别是顶部链接的 `SiteMapDataSource` 中的 `StartingNodeId` 属性使用值为 `sid:1002`，

而快速启

动菜单对应的 `StartingNodeId` 属性值为 `sid:1025`。`1002` 和 `1025` 是用于跟踪导航节点的，顶部链接

的顶部节点的

`id` 是 `1002`，快速启动的顶部节点的 `id` 为 `1025`。

顶部链接和快速启动的节点的添加也是非常方便的，在后台的"网站设置"即可以灵活的设置.也可以使用

MOSS 对

象模型来添加，示例代码如下：

```
SPWeb site = (SPWeb)properties.Feature.Parent;
SPNavigationNodeCollection topNav = site.Navigation.TopNavigationBar;
// 创建下拉菜单 SPNavigationNode DropDownMenu1;
```

```
DropDownMenu1 = new SPNavigationNode("SitePages", "", false);  
topNav[0].Children.AddAsLast(DropDownMenu1);  
  
// 添加节点  
DropDownMenu1.Children.AddAsLast(new SPNavigationNode("Page1", "SitePages/Page1.aspx"));  
DropDownMenu1.Children.AddAsLast(new SPNavigationNode("Page2", "SitePages/Page2.aspx"));
```

Delegate Controls

我们可以设置它影响的范围，有四个级分别是：site scope, site collection scope, Web application scope, 和

farm scope。下面我们就以 MOSS 中的搜索栏为例说明，下面是 MOSS 中默认的：

```
<SharePoint:DelegateControl  
    ID="DelegateControl5"  
    runat="server"  
    ControlId="SmallSearchInputBox"  
/>
```

MOSS 中使用名为 ContentLightup 的 feature 来实现这功能的，SearchArea.xml 中对应的代码为：

```
<Control  
    Id="SmallSearchInputBox"  
    Sequence="100"  
    ControlSrc="~/_controltemplates/searcharea.ascx"  
/>
```

下面我们就用我们自己定制搜索栏来替默认的，我们就需要在我们自定义的 feature 中做如下设置：

```
<Control  
    Id="SmallSearchInputBox"  
    Sequence="10"  
    ControlSrc="~/_controltemplates/OurSearchArea.ascx"  
/>
```

在这里要注意一点，Sequence 属性的值一定要小于默认提供的，这个 Sequence 的值在 wss 中是 100，

在 moss

标准版中是 50，企业版是 25。这个你可以去 MOSS 的安装目录下搜索 feature 中对应的 xml 文件来核

对（名字：

SearchArea.xml）。由于我们定义的这个是站点的级别的，那么你整个站点页面中的搜索控件都会被

替换为

你自己定制的风格。

下面就是我们自定义的搜索样式的页面.ascx 代码：

```
<%@ Control Language="C#" %>

<script runat="server">

protected void OurSearchArea_Click(object sender, EventArgs e)

{ }

</script>

<table>

<tr><td><asp:Button ID="cmdRunSearch" runat="server" Text="Search"

OnClick="OurSearchArea_Click" /></td>

<td><asp:TextBox ID="txtSearchText" runat="server" Width="120" /></td></tr>

</table>
```

下图是替换的效果：



上面我们使用了用户控件方式(.ascx)，我们还可以使用控件类的方式来自定义，下面是 MOSS 提供的标准的

QuickLaunchDataSource.

```
<SharePoint:DelegateControl
```

```

        ID="DelegateControl8" runat="server"
        ControlId="QuickLaunchDataSource">
<Template_Controls>
    <asp:SiteMapDataSource
        SiteMapProvider="SPNavigationProvider"
        ShowStartingNode="False"
        id="QuickLaunchSiteMap"
        StartingNodeUrl="sid:1025"
        runat="server" />
</Template_Controls>
</SharePoint:DelegateControl>

```

这个 DelegateControl 使用 QuickLaunchDataSource 控件来填充他的内容, 并不是 asc.aspx 用户控件的

方式。他默认使用了 id 为 QuickLaunchSiteMap 的 SiteMapDataSource 控件, 如果你想替换这个控件, 你

可以在 feature 中添加一下代码

```

<Control Id="QuickLaunchDataSource" Sequence="1" ControlAssembly="System.Web, ..."
ControlClass="System.Web.UI.WebControls.SiteMapDataSource">
    <Property Name="ID">QuickLaunchSiteMap</Property>
    <Property Name="SiteMapProvider">SPSiteMapProvider</Property>
    <Property Name="ShowStartingNode">False</Property>
</Control>

```

这个 Control 的 id 要和你要替换的 id 是一样的才可以。

自定义 Default.master

我们可以使用 SPD 来定制母版页，母版页和其他的 Site Page 的原理是一样的，你定制过后就会被存到内容数

据库中。所以我们不要去文件目录下直接修改 default.master 文件，修改了也是没有用的。

下面我们就自定义一个母版页实现简单的换肤功能，在 elements.xml 中定义如下：

```
<Module Name="MasterPages" List="116" Url="_catalogs/masterpage">  
  <File Url="OurCustom.master" Type="GhostableInLibrary" />  
</Module>
```

Url: "_catalogs/masterpage" 是母版页的相对路径。

Type: "GhostableInLibrary" 会在文档库中存储该母版页。

List: 属性代表不同的列表类型，下表是 MOSS 中已经定义的，微软建议我们自定义的最好大于 10000，以避免

和已定义的冲突。

ID	Description
100	Generic list
101	Document library
102	Survey
103	Links list
104	Announcements list
105	Contacts list
106	Events list
107	Tasks list
108	Discussion board
109	Picture library
110	Data sources
111	Site template gallery
112	UserInformation
113	Web Part gallery
114	List template gallery
115	XML Form library
116	MasterPageCatalog
117	NoCodeWorkflows
118	WorkflowProcess
119	WebPageLibrary
120	Custom grid for a list

130	DataConnectionLibrary
140	WorkflowHistory
150	GanttTasks
200	Meeting Series list
201	Meeting Agenda list
202	Meeting Attendees list
204	Meeting Decisions list
207	Meeting Objectives list
210	Meeting text box
211	Meeting Things To Bring list
212	Meeting Workspace Pages list
300	Portal Sites list.
301	Posts
302	Comments
303	Categories
1100	Issue tracking
1200	AdminTasks
2002	Personal document library
2003	Private document library

Page 的 MasterPageFile 属性指定使用的母版页，MOSS 中的母版页默认提供了一些 token，主要有动态 tokens (“~masterurl/default.master” 和 “~masterurl/custom.master”) 和静态 tokens (“~site/default.master” and “~sitecollection/default.master”), 说明如下:

Dynamic token “~masterurl/default.master”

可以通过 SPWeb 的 MasterUrl 属性来指定网站的母版页的 URL，即指定 page 中的 MasterPageFile, 页面的 <%@

Page MasterPageFile=“~masterurl\default.master”%>中的

“~masterurl/default.master”在运

行时将会被 SPWeb 的 MasterUrl 属性取代

Dynamic token “~masterurl/custom.master”

与上面的类似，他是通过 SPWeb 的 CustomMasterUrl 属性来指定的。

Static tokens “~site/default.master” 和 “~sitecollection/default.master”

如果你的内容页在 <http://siteColl/subsite1/subsite2/default.aspx>，你使用了静态 tokens“~sitecollection/mypage.master”，你内容页使用的母版页在这个位置 <http://siteColl/>

mypage.master。如果你用的 tokens 是“~site/mypage.master”，你的母版页就在这个位置 <http://siteColl/subsite1/subsite2/mypage.master>。

注意：MasterUrl 和 CustomMasterUrl 默认都是这个值

“/_catalogs/masterpage/default.master”，如果你要

改变的话，要两个都改变，必须一致才行。

下面我们就来实现动态设定母版页，由于我们定义的是站点级的，而站点中的每个子站点都是应用的自己独立

的一个 master，所以我们使用递归来替换所有的站点的母版页代码如下：

```
protected void btnApplyOurCustomMaster_Click(object sender, EventArgs e)
{
    SPWeb site = SPContext.Current.Site.RootWeb;

    string MasterUrlPath = site.ServerRelativeUrl;

    if (!MasterUrlPath.EndsWith("/"))
        MasterUrlPath += "/";

    MasterUrlPath += @"_catalogs/masterpage/OurCustom.master";

    ApplyCustomBrand(MasterUrlPath, site);
}

protected void ApplyOurCustomMaster(string MasterUrlPath, SPWeb site) {
    site.MasterUrl = MasterUrlPath;

    site.Update();

    foreach (SPWeb child in site.Webs) {
        ApplyCustomBrand(MasterUrlPath, child);
    }
}
```

下图是前后的效果：

