第 12 章　ASP.NET 的皮肤、主题和母版页

在 Web 应用程序开发中，一个良好的 Web 应用程序界面能够让网站的访问者耳目一新，当用户访问 Web 应用时，网站的界面和布局能够提升访问者对网站的兴趣和继续浏览的耐心。ASP.NET 提供了皮肤、主题和模板页的功能增强了网页布局和界面优化的功能，这样即可轻松的实现对网站开发中界面的控制。

12.1　皮肤和主题

皮肤和主题是自 ASP.NET 2.0 就包括的内容，使用皮肤和主题，能够将样式和布局信息分解到单独的文件中，让布局代码和页面代码相分离。主题可以应用到各个站点，当需要更改页面主题时，无需对每个页面进行更改，只需要针对主题代码页进行更改即可。

12.1.1　CSS 简介

在任何 Web 应用程序的开发过程中，CSS（Cascading Style Sheets，级联样式表）都是非常重要的页面布局方法，而且 CSS 也是最高效的页面布局方法。CSS 发展于 1994 年 10 月，是为了补救 HTML 3.2 语法中的不足，但是由于当时网络的发展的不足和浏览器的支持率较低，直到 1996 年底，才正式发表了 CSS 1.0 规格，也正是 1996 年之后，浏览器才开始正式的支持 CSS。

在网页布局中，CSS 经常被使用于页面样式布局和样式控制。熟练的使用 CSS 能够让网页布局更加的方便，在页面维护时，也能够减少工作量。通常 CSS 能够支持三种定义方式，一是直接将样式控制放置于单个 HTML 元素内，称为内联式；二是在网页的 head 部分定义样式，称为嵌入式；三是以扩展名为.css 文件保存样式，称为外联式。

这三种样式适用于不同的场合，内联式适用于对单个标签进行样式控制，这样的好处就在于开发方便，而在维护时，就需要针对每个页面进行修改，非常的不方便；而嵌入式可以控制一个网页的多个样式，当需要对网页样式进行修改时，只需要修改 head 标签中的 style 标签即可，不过这样仍然没有让布局代码和页面代码完全分离；而外联式能够将布局代码和页面代码相分离，在维护过程中，能够减少工作量。

12.1.2　CSS 基础

CSS 能够通过编写样式控制代码来进行页面布局，在编写相应的 HTML 标签时，可以通过 Style 属性进行 CSS 样式控制，示例代码如下所示。

```
<body>
    <div style="font-size:14px;">这是一段文字</div>
</body>
```

上述代码使用内联式进行样式控制，并将属性设置为 font-size:14px，其意义就在于定义文字的大小为 14px；同样，如果需要定义多个属性时，可以同写在一个 style 属性中，示例代码如下所示。

```
<body>
    <div style="font-size:14px;">这是一段文字 1</div>
        <div style="font-size:14px; font-weight:bolder">这是一段文字 2</div>
            <div style="font-size:14px; font-style:italic">这是一段文字 3</div>
        <div style="font-size:14px; font-variant:small-caps">This is My First CSS code</div>
    <div style="font-size:14px; color:red">这是一段文字 5</div>
</body>
```

上述代码分别定义了相关属性来控制样式，并且都使用内联式定义样式，这些CSS 的属性的意义如下所示：

- 字体名称属性（font-family）：该属性设定字体名称，如 Arial,、Tahoma,、Courier 等，可以定义字体的名称。
- 字体大小属性（font-size）：该属性可以设置字体的大小。字体大小的设置可以有多种方式，最常用的就是 pt 和 px。
- 该属性有三个值可选：normal,、italic、oblique、normal 是默认值，italic、oblique 都是斜体显示。
- 字体粗细属性（font-weight）：该属性常用值是 normal 和 bold，normal 是默认值，bold 是粗体。
- 字体变量属性（font-variant）：该属性有两个值 normal 和 small-caps，normal 是默认值。small-caps 表示字体将被显示成大写。
- 字体属性（font）：该属性是各种字体属性的一种快捷的综合写法。
- 字体颜色(color)：该属性用来控制字体颜色。

这些属性分别定义了字体属性，如图 12-1 所示。

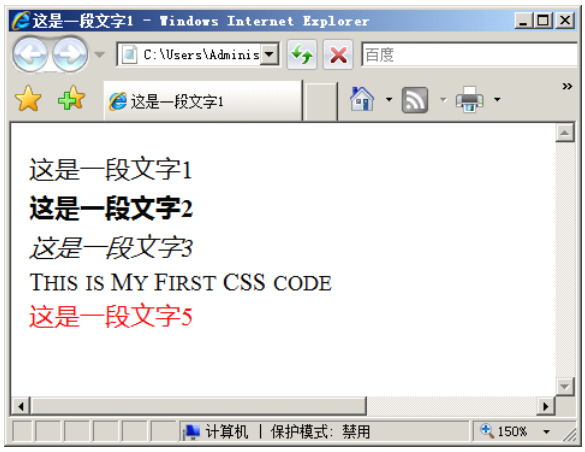

图 12-1　CSS 样式控制

用内联式的方法进行样式控制固然简单，但是在维护过程中却是非常的复杂和难以控制。当需要对页面中的布局进行更改时，则需要对每个页面的每个标签的样式进行更改，这样无疑增大的工作量，当需要对页面进行布局时，可以使用嵌入式的方法进行页面布局，示例代码如下所示。

```
<head>
    <meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
        <title>这是一段文字 1</title>
        <style type="text/css">
        .font1
        {
            font-size:14px;
        }
```

```
        .font2
        {
            font-size:14px;
            font-weight:bolder;
        }
        .font3
        {
            font-size:14px;
            font-style:italic;
        }
        .font4
        {
            font-size:14px;
            font-variant:small-caps;
        }
        .font5
        {
            font-size:14px;
            color:red;
        }
        </style>
</head>
```

上述代码分别定义了 5 种字体样式，这些样式都是通过“.”号加样式名称定义的，在定义了字体样式后，就可以在相应的标签中使用 class 属性来定义样式，示例代码如下所示。

```
<body>
    <div class="font1">这是一段文字 1</div>
        <div class="font2">这是一段文字 2</div>
            <div class="font3">这是一段文字 3</div>
        <div class="font4">This is My First CSS code</div>
    <div class="font5">这是一段文字 5</div>
</body>
```

其运行后的结果依然同 11-12 所示，但是这样编写代码在维护起来更加的方便，只需要找到 head 中的 style 标签，就可以对样式进行全局控制。虽然嵌入式能够解决单个页面的样式问题，但是这样只能针对单个页面进行样式控制，而在很多网站的开发应用中，大量的页面样式基本相同，只有少数的页面不尽相同，所以使用嵌入式还是有不足，这里就可以使用外联式。使用外联式，必须创建一个.css 文件后缀的文件，并在当前页面中添加引用，.css 页面代码如下所示。

```
.font1
{
    font-size:14px;
}
.font2
{
    font-size:14px;
    font-weight:bolder;
}
.font3
{
    font-size:14px;
    font-style:italic;
```

```
}
.font4
{
    font-size:14px;
    font-variant:small-caps;
}
.font5
{
    font-size:14px;
    color:red;
}
```

在.css 文件中，只需要定义如 head 标签中的 style 标签的内容即可，其编写方法也与内联式和内嵌式相同。在编写完成 CSS 文件后，需要在使用的页面的 head 标签中添加引用，示例代码如下所示。

```
<link href="css.css" type="text/css" rel="stylesheet"></link>
```

上述代码添加了一个 css.css 文件的引用，意在告诉浏览器当前页面的一些样式可以在 css.css 中找到并解析。在使用了外联式后，当前页面的 HTML 代码就能够变得简单和整洁，示例代码如下所示。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
    <title>这是一段文字 1</title>
    <link href="css.css" type="text/css" rel="stylesheet"></link>
</head>
<body>
    <div class="font1">这是一段文字 1</div>
        <div class="font2">这是一段文字 2</div>
            <div class="font3">这是一段文字 3</div>
        <div class="font4">This is My First CSS code</div>
    <div class="font5">这是一段文字 5</div>
</body>
</html>
```

使用外联式能够很好的将页面布局的代码和 HTML 代码相分离，这样不仅能够让多个页面同时使用一个 CSS 样式表进行样式控制，同样在维护的过程中，只需要修改相应的 CSS 文件中的样式的属性即可实现该样式在所有的页面中都进行更新的操作。这样无疑是减少了工作量，提高的代码的可维护性 。可见外联式能够让样式控制既方便又灵活。

12.1.3 CSS 常用属性

CSS 不仅能够控制字体的样式，CSS 还具有强大的样式控制功能，包括背景，边框，边距等属性，这些属性能够为网页布局提供良好的保障，熟练的使用这些属性能够极大的提高 Web 应用的友好度。

1. CSS 背景属性

CSS 能够描述背景，包括背景颜色、背景图片、背景图片重复方向等属性，这些属性为页面背景的样式控制提供了强大的支持，这些属性包括如下所示：

- 背景颜色属性（background-color）：该属性为 HTML 元素设定背景颜色。
- 背景图片属性（background-image）：该属性为 HTML 元素设定背景图片。
- 背景重复属性（background-repeat）：该属性和 background-image 属性连在一起使用，决定背景

图片是否重复。如果只设置 background-image 属性，没设置 background-repeat 属性，在缺省状态下，图片既 x 轴重复，又 y 轴重复。

- 背景附着属性（background-attachment）：该属性和 background-image 属性连在一起使用，决定图片是跟随内容滚动，还是固定不动。
- 背景位置属性（background-position）：该属性和 background-image 属性连在一起使用，决定了背景图片的最初位置。
- 背景属性（background）：该属性是设置背景相关属性的一种快捷的综合写法。

通过这些属性能够为网页背景进行样式控制，示例代码如下所示。

```
body
{
    background-color:green;
}
```

上述代码设置了网页的背景颜色为绿色，如图 12-2 所示。同样，设计人员能够使用 background-image 属性设置背景图片，并说明图片是否重复，如图 12-3 所示。

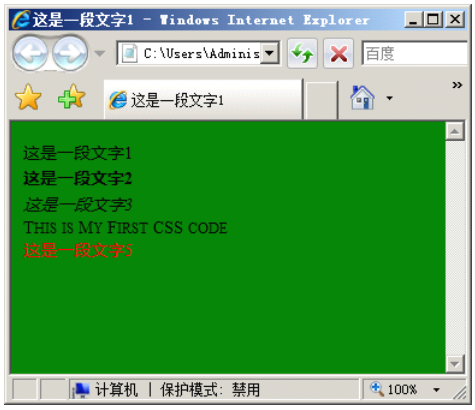

图 12-2　修改背景颜色

图 12-3　背景图片

当使用 background-image 属性设置背景图片时，还需要使用 background-repeat 属性进行循环判断，示例代码如下所示。

```
body
{
    background-image:url('bg.jpg');
    background-repeat:repeat-x;
}
```

上述代码将 bg.jpg 作为背景图片，并且以 x 轴重复，如果不编写 background-repeat 属性，则默认是即 x 轴重复也 y 轴重复。上述代码还可以简写，示例代码如下所示。

```
body
{
    background:green url('bg.jpg') repeat-x;
}
```

2．CSS 边框属性

CSS 还能够进行边框的样式控制，使用 CSS 能够灵活的控制边框，边框属性包括有：

- 边框风格属性（border-style）：该属性用来设定上下左右边框的风格。
- 边框宽度属性（border-width）：该属性用来设定上下左右边框的宽度。

- 边框颜色属性（border-color）：该属性设置边框的颜色。
- 边框属性（border）：该属性是边框属性的一个快捷的综合写法。

通过这些属性能够控制边框样式，示例代码如下所示。

```
.mycss
{
    border-bottom:1px black dashed;
    border-top:1px black dashed;
    border-left:1px black dashed;
    border-right:1px black dashed;
}
```

上述代码分别设置了边框的上部分、下部分、左部分、右部分的边框属性，来形式一个完整的边框，同样可以使用边框属性来整合这些代码，示例代码如下所示。

```
.mycss
{
    border:1px black dashed;
}
```

3．CSS 边距和间隙属性

CSS 的边距和间隙属性能够控制标签的位置，CSS 的边距属性使用的是 margin 关键字，而间隙属性使用的是 padding 关键字。CSS 的边距和间隙属性虽然都是一种定位方法，但是边距和间隙属性定位的对象不同，也就是参照物不同，如图 12-4 所示。

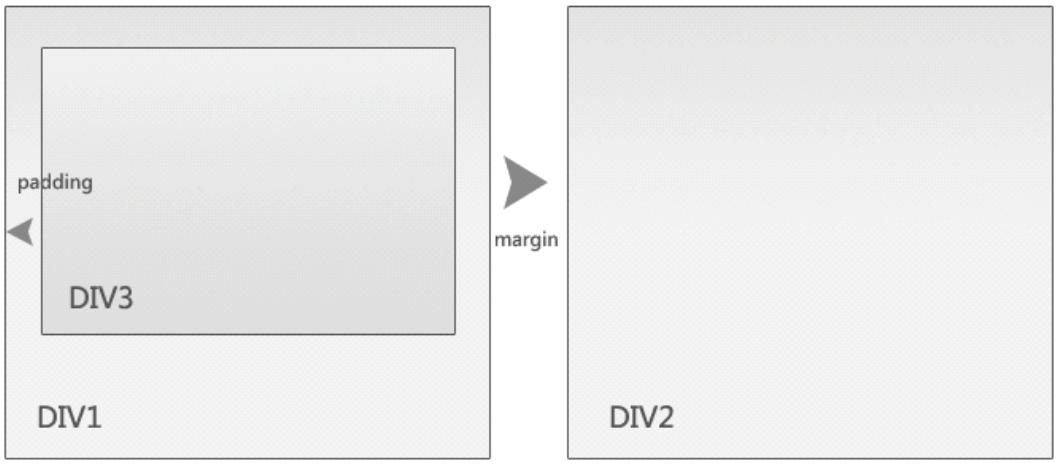

图 12-4　边距属性和间隙属性的区别

边距属性（margin）通常是设置一个页面中一个元素所占的空间的边缘到相邻的元素之间的距离，而间隙属性（padding）通常是设置一个元素中间的内容（或元素）到父元素之间的间隙（或距离）。对于边距属性（margin）有以下属性：

- 左边距属性（margin-left）：该属性用来设定左边距的宽度。
- 右边距属性（margin-right）：该属性用来设定右边距的宽度。
- 上边距属性（margin-top）：该属性用来设定上边距的宽度。
- 下边距属性（margin-bottom）：该属性用来设定下边距的宽度。
- 边距属性（margin）：该属性是设定边距宽度的一个快捷的综合写法，用该属性可以同时设定上下左右边距属性。

对于间隙属性，基本同边距属性，只是 margin 改为了 padding，其属性如下所示。

- 左间隙属性（padding-left）：该属性用来设定左间隙的宽度。
- 右间隙属性（padding-right）：该属性用来设定右间隙的宽度。
- 上间隙属性（padding-top）：该属性用来设定上间隙的宽度。示例如下：
- 下间隙属性（margin-bottom）：该属性用来设定下间隙的宽度。示例如下：
- 间隙属性（padding）：该属性是设定间隙宽度的一个快捷的综合写法，用该属性可以同时设定上下左右间隙属性。

使用边距属性和间隙属性能够进行页面布局，其中 HTML 页面代码如下所示。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
    <title>这是一段文字 1</title>
    <link href="css.css" type="text/css" rel="stylesheet"></link>
</head>
<body>
    <div class="div1">
     DIV1
          <div class="div3">DIV3</div>
    </div>
    <div class="div2">DIV2</div>
</body>
</html>
```

HTML 代码制作完毕后，就可通过 css.css 文件为该页面编写样式，示例代码如下所示。

```
.div1
{
    float:left;
    margin-left:10px;                                    //和左边元素距离为 10px
    background:white url('bg.jpg') repeat-x;
    border:1px solid #ccc;
    width:300px;
    height:200px;
    padding:30px;                                        //内部对齐 30px
}
.div2
{
    float:left;
    margin-left:20px;                                    //和左边元素距离为 20px
    background:white url('bg.jpg') repeat-x;
    border:1px solid #ccc;
    width:300px;
    height:260px;
}
.div3
{
    background:white;                                    //背景为白色
}
```

页面布局完成后，运行图如图 12-5 所示。

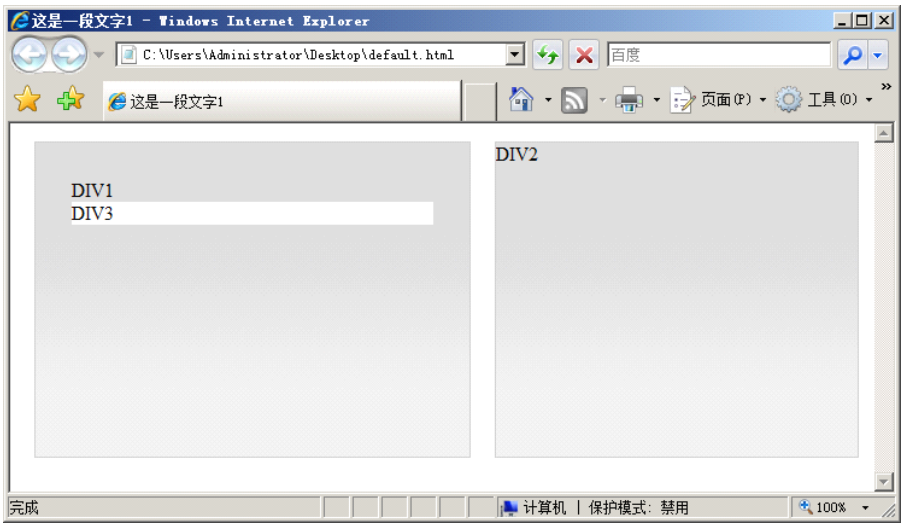

图 12-5　边距属性和间隙属性

CSS 不仅提供了诸如此类的强大布局功能，CSS 还提供了很多其他的布局功能，这些功能非常的多，能够为页面布局起到美化作用。CSS 还包括盒子模式、列表属性和伪类等高级技巧，这些技巧就不在本书中一一介绍了。

12.1.4　将 CSS 应用在控件上

CSS 不仅能够用来进行页面布局，同样也可以应用在控件中，使用 CSS 能够让控件更具美感。在空间上使用 CSS 基本和在页面上使用 CSS 的方法相同。在控件界面的编写中，可以使用控件的默认属性，例如 BackColor、ForeColor、BorderStyle 等，同样也可以通过 style 属性编写控件的属性，示例代码如下所示。

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>无标题页</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <asp:Button ID="Button1" runat="server" Text="Button" />
        <br />
        <br />
        <asp:TextBox ID="TextBox2" runat="server" style="border:1px solid #ccc;"></asp:TextBox>
        <asp:Button ID="Button2" runat="server" Text="Button"
        style="border:1px solid #ccc; background:white; color:Black"/>
    </div>
    </form>
</body>
</html>
```

上述代码分别编写了 4 个控件，其中 2 个控件是输入框和按钮控件，另外 2 个控件也是输入框和按钮控件。不同的是，另外 2 个控件通过 style 属性进行了样式控制，运行后如图 12-6 所示。

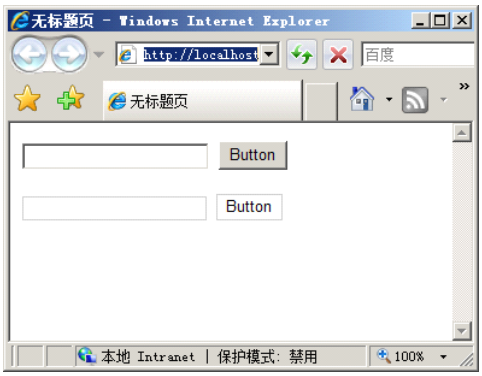

图 12-6　控件的样式控制

除了通过 style 标签以外，控件自己还带有“样式”属性，通过配置相应的属性，即可为控件进行样式控制。其中最典型的包括日历控件，日历控件能够套用默认格式以呈现更加丰富的样式，示例代码如下所示。

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>无标题页</title>
    <style type="text/css">
        .style1
        {
            width: 100%;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <table class="style1">
            <tr>
                <td>
                    默认样式</td>
                <td>
                    选择样式</td>
            </tr>
            <tr>
                <td>
                    <asp:Calendar ID="Calendar1" runat="server"></asp:Calendar>
                </td>
                <td>
                    <asp:Calendar ID="Calendar2" runat="server"
                    BackColor="#FFFFCC" BorderColor="#FFCC66"
                    BorderWidth="1px" DayNameFormat="Shortest" Font-Names="Verdana" Font-Size="8pt"
                    ForeColor="#663399" Height="200px" ShowGridLines="True" Width="220px">
                    <SelectedDayStyle BackColor="#CCCCFF" Font-Bold="True" />
                    <SelectorStyle BackColor="#FFCC66" />
                    <TodayDayStyle BackColor="#FFCC66" ForeColor="White" />
                        <OtherMonthDayStyle ForeColor="#CC9966" />
                            <NextPrevStyle Font-Size="9pt" ForeColor="#FFFFCC" />
```

```
                        <DayHeaderStyle BackColor="#FFCC66" Font-Bold="True" Height="1px" />
                        <TitleStyle BackColor="#990000" Font-Bold="True" Font-Size="9pt"
                    ForeColor="#FFFFCC" />
                </asp:Calendar>
            </td>
        </tr>
    </table>
</div>
</form>
</body>
</html>
```

上述代码通过属性为日历控件进行了样式控制，运行后如图 12-7 所示，默认的样式和配置了样式之后的差别巨大。

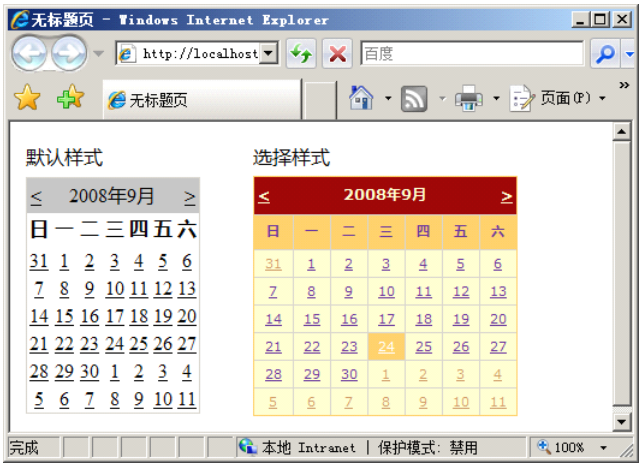

图 12-7　属性样式控制

通过编写样式能够让控件的呈现更加的丰富，让用户体验更加友好，当用户访问页面时，能够提高用户对网站的满意程度。控件的样式控制，不仅能够使用默认的属性进行样式控制，同样可以使用 style 属性进行样式控制，但是 style 属性的样式控制在很多地方不能操作，例如日历控件中的当前日期样式，而通过控件的属性的配置，却能够快速配置当前日期的样式。

12.1.5　主题和皮肤

主题是属性设置的集合，通过使用主题的设置能够定义页面和控件的样式，然后在某个 Web 应用程序中应用到所有的页面，以及页面上的控件，以简化样式控制。主题包括一系列元素，这些元素分别是皮肤、级联样式表（CSS）、图像和其他资源。主题文件的后缀名称为.skin，创建主题后，主题文件通常保存在 Web 应用程序的特殊目录下，创建主题文件如图 12-8 所示。创建外观文件，Visual Studio 会提示是否将文件存放到特殊目录，如图 12-9 所示。

图 12-8　创建外观文件

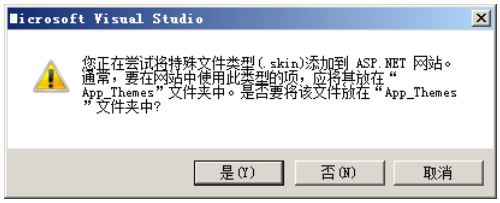

图 12-9　将主题文件存放到特殊目录

单击【是】按钮后主题文件会存放到 App_Themes 文件夹中。主题文件通常都保存在 Web 应用程序的特殊目录下，以便这些文件能够在页面中进行全局访问。在主题文件中编写代码可以对控件进行主题配置，示例代码如下所示。

```
<asp:Calendar runat="server" BackColor="White"
    BorderColor="Black" BorderStyle="Solid" CellSpacing="1" Font-Names="Verdana"
    Font-Size="9pt" ForeColor="Black" Height="250px" NextPrevFormat="ShortMonth"
    SkinID="blue" Width="330px">
    <SelectedDayStyle BackColor="#333399" ForeColor="White" />
    <TodayDayStyle BackColor="#999999" ForeColor="White" />
    <OtherMonthDayStyle ForeColor="#999999" />
    <DayStyle BackColor="#CCCCCC" />
    <NextPrevStyle Font-Bold="True" Font-Size="8pt" ForeColor="White" />
    <DayHeaderStyle Font-Bold="True" Font-Size="8pt" ForeColor="#333333"
    Height="8pt" />
    <TitleStyle BackColor="#333399" BorderStyle="Solid" Font-Bold="True"
    Font-Size="12pt" ForeColor="White" Height="12pt" />
</asp:Calendar>
<asp:Calendar runat="server" BackColor="White"
    BorderColor="#999999" CellPadding="4" DayNameFormat="Shortest"
    Font-Names="Verdana" Font-Size="8pt" ForeColor="Black" Height="180px"
    SkinID="now" Width="200px">
    <SelectedDayStyle BackColor="#666666" Font-Bold="True" ForeColor="White" />
    <SelectorStyle BackColor="#CCCCCC" />
    <WeekendDayStyle BackColor="#FFFFCC" />
    <TodayDayStyle BackColor="#CCCCCC" ForeColor="Black" />
    <OtherMonthDayStyle ForeColor="#808080" />
    <NextPrevStyle VerticalAlign="Bottom" />
    <DayHeaderStyle BackColor="#CCCCCC" Font-Bold="True" Font-Size="7pt" />
    <TitleStyle BackColor="#999999" BorderColor="Black" Font-Bold="True" />
</asp:Calendar>
```

上述代码创建了两种日历控件的主题，这两个日历控件的主题分别为 SkinID=“blue”和 SkinID=“now”。值得注意的是，SkinID 属性在主题文件中是必须且惟一的，因为这样才可以在相应页面中为控件配置所需要使用的主题，示例代码如下所示。

```
<asp:Calendar ID="Calendar1" runat="server" SkinID="blue"></asp:Calendar>
<asp:Calendar ID="Calendar2" runat="server" SkinID="now"></asp:Calendar>
```

上述控件并没有对控件进行样式控制，只是声明了 SkinID 属性，当声明了 SkinID 属性后，系统会自动在主题文件中找到相匹配的 SkinID，并将主题样式应用到当前控件。在使用主题的页面，必须声明主题，如果不声明主题，则页面无法找到页面中控件需要使用的主题，示例代码如下所示。

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="_12_1._Default" Theme="Theme1"%>
```

在页面声明主题后，控件就能够使用.skin 文件中的主题，通过 SkinID 属性，控件可以选择主题文件中的主题。运行如图 12-10 所示。

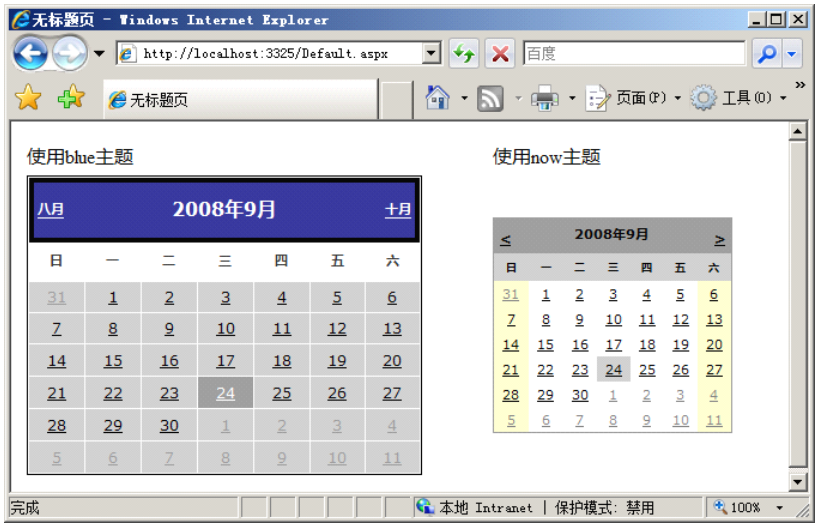

图 12-10　选择不同的主题

主题还可以包括级联样式表（CSS 文件），将.css 放置在主题目录中，样式表则会自动的应用为主题的一部分，不仅如此，主题还可以包括图片和其他资源。

12.1.6　页面主题和全局主题

用户可以为每个页面设置主题，这种情况被称为“页面主题”。也可以为应用程序的每个页面都使用主题，在每个页面使用默认主题，这种情况被称为“全局主题”。

页面主题是一个主题文件夹，其中包括控件的主题、层叠样式表、图形文件和其他资源文件，这个文件夹是作为网站中的“\App_Themes”文件夹和子文件夹创建的。每个主题都是“\App_Themes”文件夹的一个子文件夹，如图 12-11 所示。

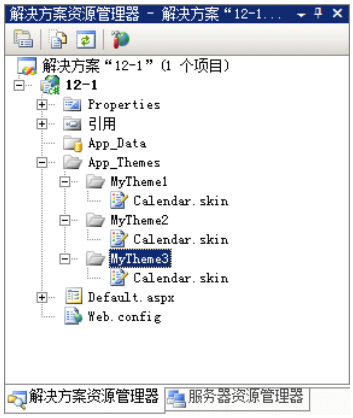

图 12-11　多个主题

使用全局主题，可以让应用程序中的所有页面都能够使用该主题，当维护同一个服务器上的多个网站时，可以使用全局主题定义应用程序的整体外观。当需要使用全局主题时，则可以通过修改 Web.config 配置文件中的<pages>配置节进行主题的全局设定。

使用全局主题和使用页面主题的方法基本相同，它们都包括属性设置、样式设置和图形。但是全局主题和页面主题不同的是，全局主题存放在服务器上的公共文件夹中，这个文件夹通常命名为 Theme。服务器上的任何 Web 应用程序都能够使用 Theme 文件夹中的主题。主题能够和 CSS 文件一样，进行页面布局和控件样式控制，但是主题和 CSS 文件的描述不同，所能够完成的功能也不同，其主要区别如下所示：

- 主题可以定义控件的样式，不仅能够定义样式属性，还能够定义其他样式，包括模板。
- 主题可以包括图形等其他主题元素文件。
- 主题的层叠方式与 CSS 文件的层叠方式不同。
- 一个页面只能应用与一个主题，而 CSS 可以被多个文件应用。

主题不仅能够进行控件的样式定义，还能够定义模板，这样减少了相同类型的控件的模板编写操作。但是主题也有缺点，一个页面只能应用一个主题，而无法应用多个主题。与之相反的是，一个页面能够应用多个 CSS 文件。

主题与 CSS 相比，主题在样式控制上还有很多不够强大的地方，而 CSS 页面布局的能力比主题更加强大，样式控制更加友好。

12.1.7　应用和禁用主题

通常情况下，可以在网站目录下的“App_Themes”文件夹下定义主题，然后在页面中进行主题的使用声明，这样在页面中就能够使用主题了。制作主题的过程也非常简单，在“App_Themes”文件夹下新建一个文件夹，则这个文件夹的名称就会作为主题名称在应用程序中保存。同样，开发人员能够在文件夹中可以新增“.skin”文件，以及“.css”文件和图形图像文件来修饰主题，这样一个主题就制作完毕并能够在页面中使用了。

在很多情况下，在 Web 开发中需要定义全局主题，这样 Web 应用程序就能够使用这个主题，全局主题通常放在一个特殊的目录下，放在这个目录下的主题能够被服务器上的任何网站，以及网站中的任何应用所引用。全局主题存放的目录如下所示。

```
lisdefaultroot\aspnet_client\system_web\version\Themes
```

在全局主题目录下，可以创建任何主题文件，这样在网站上的其他 Web 应用也能够使用全局主题作为主题。在主题的编写过程中，通常需要以下几个步骤：

- 添加项目，包括.skin、css 以及其他文件。
- 创建皮肤，包括对控件属性的定义。
- 在页面中声明并使用皮肤。

通过以上三个步骤能够创建并使用皮肤，但是值得注意的是，在创建皮肤文件时，必须保存为.skin 文件并且主题中控件的定义必须包括 SkinID 属性且不能包括 ID。在皮肤中，对控件的属性的描述同样必须要包括 runat=“server”标记，这样才能够保证皮肤文件中控件的皮肤的描述是正确和可读的，示例代码如下所示。

```
<asp:Calendar    runat="server" BackColor="White"
    BorderColor="Black" BorderStyle="Solid" CellSpacing="1" Font-Names="Verdana"
    Font-Size="9pt" ForeColor="Black" Height="250px" NextPrevFormat="ShortMonth"
```

```
    SkinID="blue" Width="330px">
    <SelectedDayStyle BackColor="#333399" ForeColor="White" />
    <TodayDayStyle BackColor="#999999" ForeColor="White" />
    <OtherMonthDayStyle ForeColor="#999999" />
    <DayStyle BackColor="#CCCCCC" />
    <NextPrevStyle Font-Bold="True" Font-Size="8pt" ForeColor="White" />
    <DayHeaderStyle Font-Bold="True" Font-Size="8pt" ForeColor="#333333"
    Height="8pt" />
    <TitleStyle BackColor="#333399" BorderStyle="Solid" Font-Bold="True"
    Font-Size="12pt" ForeColor="White" Height="12pt" />
</asp:Calendar>
```

在定义了控件的皮肤后，就可以在单个页面进行皮肤的声明和使用，示例代码如下所示。

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="_12_1._Default"
Theme="MyTheme1"%>
```

同样也可以使用 StyleSheetTheme 属性进行页面主题的设置，示例代码如下所示。

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="_12_1._Default"
StylesheetTheme="MyTheme1"%>
```

如果需要使用全局主题，则需要在 Web.config 配置文件中定义全局主题，示例代码如下所示。

```
<system.web>
     <pages theme="MyTheme1">
     </pages>
</system.web>
```

在使用主题后，对于控件的属性的编写是没有任何效果的，示例代码如下所示。

```
    <asp:Calendar ID="Calendar1" runat="server" SkinID="blue" BackColor="#FFFFCC"
        BorderColor="#FFCC66" BorderWidth="1px" DayNameFormat="Shortest"
        Font-Names="Verdana" Font-Size="8pt" ForeColor="#663399" Height="200px"
        ShowGridLines="True" Width="220px">
        <SelectedDayStyle BackColor="#CCCCFF" Font-Bold="True" />
        <SelectorStyle BackColor="#FFCC66" />
        <TodayDayStyle BackColor="#FFCC66" ForeColor="White" />
        <OtherMonthDayStyle ForeColor="#CC9966" />
        <NextPrevStyle Font-Size="9pt" ForeColor="#FFFFCC" />
        <DayHeaderStyle BackColor="#FFCC66" Font-Bold="True" Height="1px" />
        <TitleStyle BackColor="#990000" Font-Bold="True" Font-Size="9pt"
        ForeColor="#FFFFCC" />
    </asp:Calendar>
```

上述代码编写了一个控件的属性，其中某些属性被主题覆盖。简单的说，局部的设置将会服从全局的设置，即页面上的控件已经具备自己的属性设置，但是当指定了 SkinID 属性后，部分属性将会服从全局属性设置，如图 12-12 和图 12-13 所示。

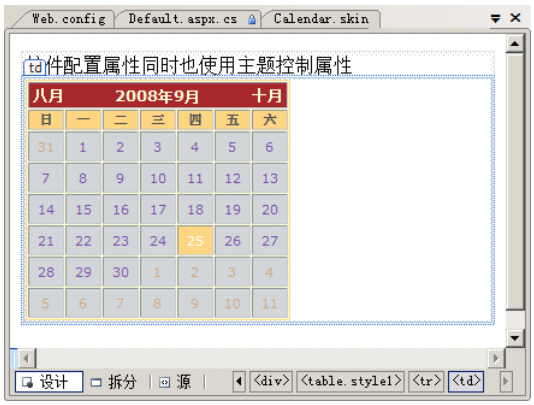

图 12-12　本地属性和全局属性

图 12-13　运行后的控件样式

虽然本地属性设置为另一种样式，但是运行后的控件样式却不是本地属性配置的样式，因为其中的某些属性已经被主题更改。在设置页面或者全局主题的 StyleSheetTheme 属性时，将主题作为样式表主题应用的话，本地页的设置将优先于主题中定义的设置，即局部设置将会覆盖全局设置。

对于主题而言，如果本地主题，以及全局主题都存在时，这种情况如控件本身的属性和使用的主题属性都存在一样，本身的属性将会被全局属性更改，全局属性中没有的属性将继续保留。而相对与 CSS 文件而言，如果本地 CSS 描述和全局 CSS 描述都存在，包括控件本身的 CSS 描述和内嵌式 CSS 文件的描述都一样时，相反的，本地 CSS 描述会替代全局的 CSS 描述。

对于有些情况，主题会重写也和控件外观的本地设置。当控件或页面已经定义了外观，而又不希望全局主题将本地主题进行重写和覆盖，可以禁用主题的覆盖行为。对于页面，可以用声明的方法进行禁用，示例代码如下所示。

```
<%@ Page Language="C#" AutoEventWireup="true" EnableTheming="false" %>
```

当页面需要某个主题的属性描述，而又希望单个控件不被主题描述时，同样可以通过 EnableTheming 属性进行主题禁止，示例代码如下所示。

```
<asp:Calendar ID="Calendar3" runat="server" EnableTheming="False">
</asp:Calendar>
```

这样就可以保证该控件不会被主题描述和控制，而页面和页面的其他元素可以使用主题描述中的相应的属性。

12.1.8　用编程的方法控制主题

当主题被制作完成后，很多场合用户希望能够自行更改主题，这种方式非常的实用，通过编程手段，只需要更改 StyleSheetTheme 属性就能够对页面的主题进行更改。通过编程的方法不仅能够更改页面的主题，同样可以更改控件的主题，达到动态更改控件主题的效果。当需要更改页面的主题时，可以更改页面的 StyleSheetTheme 属性即可实现页面主题更改的效果，StyleSheetTheme 属性的更改代码只能编写在 PreInit 事件中，示例代码如下所示。

```
protected void Page_PreInit(object sender, EventArgs e)
{
    switch (Request.QueryString["theme"])                    //获取传递的参数
```

```
        {
            case "MyTheme1":                                    //判断主题
                Page.Theme = "MyTheme1"; break;                 //更改主题
            case "MyTheme2":                                    //判断主题
                Page.Theme = "MyTheme2"; break;                 //更改主题
        }
    }
```

上述代码则通过更改 Page 的 StyleSheetTheme 属性对页面的主题进行更改，在编程的过程中，同样可以使用更加复杂的编程方法实现主题的更改。在更改页面的代码中，必须首先重写 StyleSheetTheme 属性，然后通过其中的 get 访问器返回样式表的主题名称，示例代码如下所示。

```
    public override String StyleSheetTheme
    {
        get                                                     //获取主题
        {
            return "MyTheme1";                                  //返回主题名称
        }
    }
```

对于控件，可以通过更改控件的 SkinID 属性来对控件的主题进行更改，示例代码如下所示。

```
    protected void Page_PreInit(object sender, EventArgs e)
    {
        Calendar3.SkinID = "blue";                              //更改 SkinID 属性
    }
```

上述代码通过修改控件的 SkinID 属性修改控件的主题，在控件中，SkinID 属性是能够将控件与主题进行联系的关键属性。

12.2 母版页

在 Web 应用开发过程中，经常会遇到 Web 应用程序中的很多页面的布局都相同这种情况。在 ASP.NET 中，可以使用 CSS 和主题减少多页面的布局问起，但是 CSS 和主题在很多情况下还无法胜任多页面的开发，这时就需要使用母版页。

12.2.1 母版页基础

开发人员能够使用母版页定义某一组页面的呈现样式，甚至能够定义整个网站的页面的呈现样式，Visual Studio 2008 能够轻松的创建母版页文件，对网站的全部或部分页面进行样式控制。单击【添加项】选项，选择【母版页】项目，即可向项目中添加一个母版页，如图 12-14 所示。

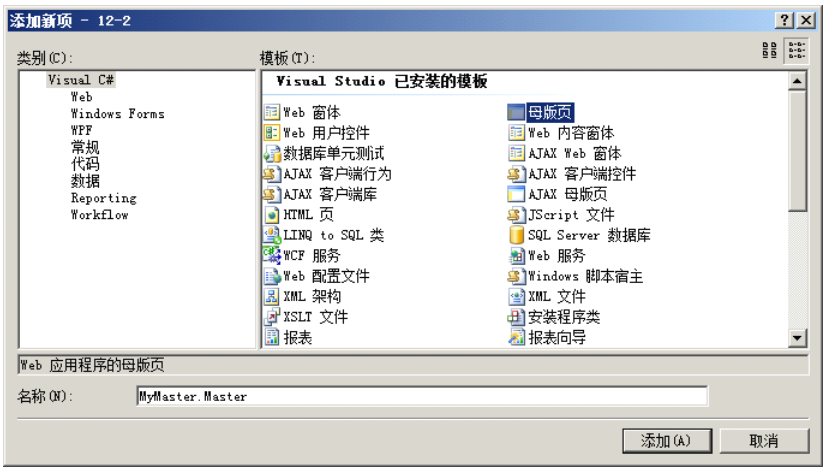

图 12-14　添加母版页

母版页的后缀名为.master。母版页同 Web 窗体在结构上基本相同，与 Web 窗体不同的是，母版页的声明方法不是使用 Page 的方法声明，而是使用 Master 关键字进行声明，示例代码如下所示。

```
<%@ Master Language="C#"
AutoEventWireup="true" CodeBehind="MyMaster.master.cs" Inherits="_12_2.MyMaster" %>
```

母版页的结构基本同 Web 窗体，但是母版页通常情况下是用来进行页面布局。当 Web 应用程序中的很多页面的布局都相同，甚至中间需要使用的用户控件、自定义控件、样式表都相同时，则可以在一个母版页中定义和编码，对一组页面进行样式控制。编写母版页的方法非常简单，只需要像编写 HTML 页面一样就可以编写母版页。在编写网站页面时，首先需要确定通用的结构，并且确定需要使用控件或 CSS 页面，如图 12-15 所示。

图 12-15　母版页页面布局

在确定了母版页布局的通用结构后，就可以编写母版页的结构了。这里使用 Table 进行布局，在布局前，首选需要定义若干样式，示例代码如下所示。

```
<style type="text/css">
    body
    {
      font-size:12px;
      text-align:center;
```

```
        }
        .style1
        {
            width: 100%;
            height: 129px;
        }
        .style2
        {
            background:url('images/bg.jpg') repeat-x;
            height: 111px;
            text-align: center;
            font-size:18px;
            font-weight:bolder;
        }
        .style3
        {
        background:url('images/bg.jpg') repeat-x;
            height: 94px;
        }
        .style4
        {
        background:url('images/bg2.jpg') repeat-x;
            width: 129px;
        }
        .style5
        {
        background:url('images/bg2.jpg') repeat-x;
            width: 476px;
        }
        .style6
        {
        background:url('images/bg2.jpg') repeat-x;
        }
    </style>
</head>
```

这些样式规定了一些基本样式，用来 Table 以及页面的布局，整页布局代码如下所示。

```
<body>
    <form id="form1" runat="server">
    <div>
        <table class="style1">
            <tr>
                <td class="style2">
                    标题</td>
            </tr>
            <tr>
                <td>
                    <table class="style1">
                        <tr>
                            <td class="style4">
                                左侧</td>
```

```
                        <td class="style5">
                            中间</td>
                        <td class="style6">
                            右侧</td>
                    </tr>
                </table>
            </td>
        </tr>
        <tr>
            <td class="style3">
            底部说明
            </td>
        </tr>
    </table>
</div>
</form>
</body>
```

上述代码对页面进行了布局，并定位了头部、中部和底部三个部分，而中部又分为左侧、中间和右侧三个部分，布局完成后效果如图 12-16 所示。

图 12-16　母版页最终布局效果

通过编写 HTML，就能够进行母版页的布局，不仅如此，母版页还能够嵌入控件、用户控件和自定义控件，方便母版页中通用模块的编写。母版页提供一个对象模型，其他页面能够通过母版页快速的进行样式控制和布局，使用母版页具有以下好处。

- 母版页可以集中的处理页面的通用功能，包括布局和控件定义。
- 使用母版页可以定义通用性的功能，包括页面中某些模块的定义，这些模块通常由用户控件和自定义控件实现。
- 母版页允许控制占位符控件的呈现方式。
- 母版页能够为其他页面提供对项目型，其他页面能够使用母版页进行二次开发。

母版页能够将页面布局集中到一个或若干个页面中，这样无需在其他页面中过多的关心页面布局。

12.2.2 内容窗体

使用母版页的页面被称作内容窗体（也称内容页）。内容窗体不是专门负责设计的页面，它们只需要关注一般页面的布局、事件以及窗体结构即可，所以内容窗体无需过多的考虑页面布局。当用户请求内容窗体时，内容窗体将与母版页合并并且将母版页的布局和内容窗体的布局组合在一起呈现到浏览器。

创建内容窗体的方法基本同 Web 窗体一样，在 Visual Studio 2005 中创建 Web 窗体时，必须勾选【选择母版页】选项，而在 Visual Studio 2008 中，有单独的内容页可以选择，如图 12-17 所示。单击【添加】按钮，系统会提示选择相应的母版页，选择相应的母版页后，单击【确定】按钮即可创建内容窗体，如图 12-18 所示。

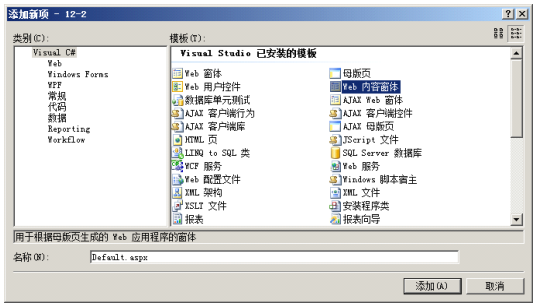

图 12-17　创建 Web 内容窗体

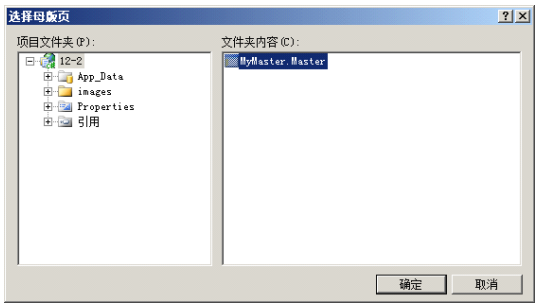

图 12-18　选择母版页

选择母版页后，系统会自动将母版页和内容整合在一起，如图 12-19 所示。

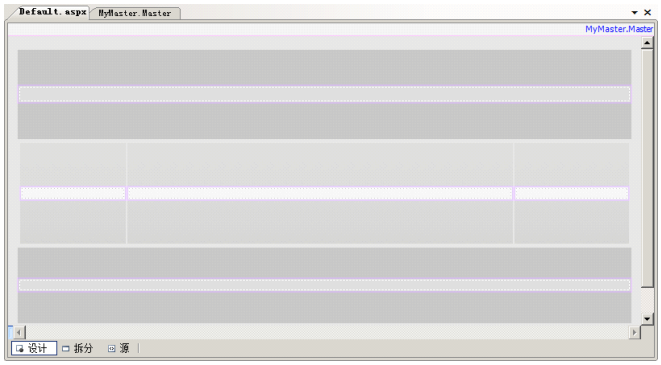

图 12-19　使用母版页

在使用母版页之后，内容窗体不能够修改母版页中的内容，也无法向母版页中新增 HTML 标签，在编写母版页时，必须使用容器让相应的位置能够在内容页中被填充。例如图 12-16，按照其方法编写母版页，内容窗体不能够对其中的文字进行修改，也无法在母版页中插入文字。在编写母版页，如果需要在某一区域能够允许内容窗体能够新增内容，就必须使用 ContentPlaceHolder 控件作占位，在母版页中，其代码如下所示。

```
<asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
</asp:ContentPlaceHolder>
```

在母版页中无需编辑此控件，当内容窗体使用了相应的母版页后，则能够通过编辑此控件并向此占位控件中添加内容或控件。单击 ContentPlaceHolder 控件，并单击 Content 任务，可在占位控件中增加控件或自定义内容，如图 12-20 所示。

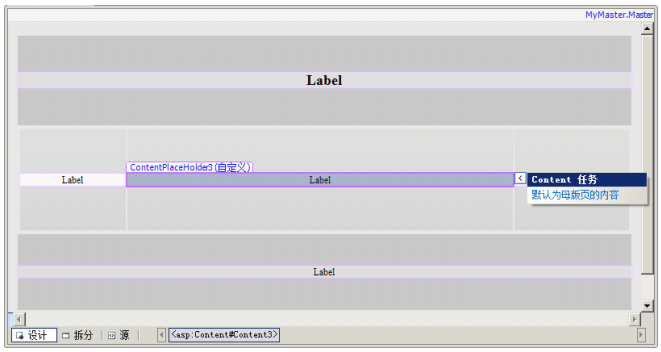

图 12-20　编辑内容窗体

编辑完成后，整个内容窗体就编写完毕了。内容窗体无需进行页面布局，也无法进行页面布局，否则会抛出异常。在内容窗体中，只需要按照母版页中的布局进行控件的拖放即可。

12.2.3　母版页的运行方法

在使用母版页时，母版页和内容页通常是一起协调运作的，母版页和内容也协调运作图如 12-21 所示。

图 12-21　母版页和内容窗体

在母版页运行后，内容窗体中 ContentPlaceHolder 控件会被映射到母版页的 ContentPlaceHolder 控件，并向母版页中的 ContentPlaceHolder 控件填充自定义控件。运行后，母版页和内容窗体将会整合形成结果页面，然后呈现给用户的浏览器。母版页运行的具体步骤为：

- 通过 URL 指令加载内容页面。
- 页面指令被处理。
- 将更新过内容的母版页合并到内容页面的控件树里。
- 单独的 ContentPlaceHolder 控件的内容被合并到相对的母版页中。
- 合并的页面被加载并显示给浏览器。

从浏览者的角度来说，母版页和内容窗体的运行并没有什么本质的区别，因为在运行的过程中，其URL 是惟一的。而从开发人员的角度来说，实现的方法不同，母版页和内容窗体分别是单独而离散的页面，分别进行各自的工作，在运行后合并生成相应的结果页面呈现给用户。在内容页中使用，母版页无需存放在特殊的目录中，只需放在普通的目录文件中即可，内容页需要使用母版页时，只需要使用MasterPageFile 属性即可，示例代码如下所示。

```
<%@ Page Language="C#"
MasterPageFile="~/MyMaster.Master"
AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="_12_2.Default" Title="无标题页" %>
```

使用 MasterPageFile 属性能够声明母版，Page 指令中的 MasterPageFile 属性会解析为一个.master 页面，在运行时，就能够将母版页和内容窗体合并为一个 Web 窗体并呈现给浏览器。

12.2.4 嵌套母版页

母版页与母版页之间能够嵌套运行，让一个母版页作为另一个母版页的子母版，能够方便的将页面进行模块化。当编写 Web 应用时，可以使用母版页进行较大型的框架布局，对一个页面进行整体的样式控制。同样可是使用母版页进行嵌套，对细节的地方进行细分。

母版页的结构和 Web 窗体的结构十分相似，与任何母版页一样，母版页也可以包含母版页，被包含的母版页被称为子母版。子母版通常会包含一些控件，这些控件将映射到父母版上的内容占位符。在MyMaster 页面中，可以编写相应的代码进行嵌套，示例代码如下所示。

```
<body>
    <form id="form1" runat="server">
    <div>
        <table class="style1">
            <tr>
                <td class="style2">
                    <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
                    </asp:ContentPlaceHolder>
                </td>
            </tr>
            <tr>
                <td>
                    <table class="style1">
                        <tr>
                            <td class="style4">
                                <asp:ContentPlaceHolder ID="ContentPlaceHolder2" runat="server">
                                </asp:ContentPlaceHolder>
                            </td>
                            <td class="style5">
                                <asp:ContentPlaceHolder ID="ContentPlaceHolder3" runat="server">
                                    Master 母版页:
                                    <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
                                </asp:ContentPlaceHolder>
                            </td>
                            <td class="style6">
                                <asp:ContentPlaceHolder ID="ContentPlaceHolder4" runat="server">
                                </asp:ContentPlaceHolder>
                            </td>
```

```
                        </tr>
                    </table>
                </td>
            </tr>
            <tr>
                <td class="style3">
                    <asp:ContentPlaceHolder ID="ContentPlaceHolder5" runat="server">
                    </asp:ContentPlaceHolder>
                </td>
            </tr>
        </table>
    </div>
    </form>
</body>
```

上述代码创建了 MyMaster 母版页，并使用了 Content 控件进行占位控件的编写。右击当前项目并单击【新建项】选项，创建一个 Child.master 母版页并为母版页编写相应的 HTML 代码，示例代码如下所示。

```
<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="Child.master.cs" Inherits="_12_2.Child"
MasterPageFile="~/MyMaster.Master"%>
<asp:Content ID="ContentPlaceHolder4" ContentPlaceHolderID="ContentPlaceHolder4" runat="server">
子母版页:<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
</asp:Content>
```

上述代码在子母版页中创建了一个文本框。在父母版页中则可以使用此母版页，使用方法同样是使用 MasterPageFile 属性进行声明。在 Child 子母版中已经声明了 MyMaster 母版页，在使用和加载 Child 页面时，可以使用 MasterPageFile="~/MyMaster.Master"语法对子母版页的母版进行声明，在上述代码中 Child 母版页使用了 MyMaster 母版页，并使用 ContentPlaceHolderID="ContentPlaceHolder4"属性对该控件进行占位控件的填充，如图 12-22 所示。

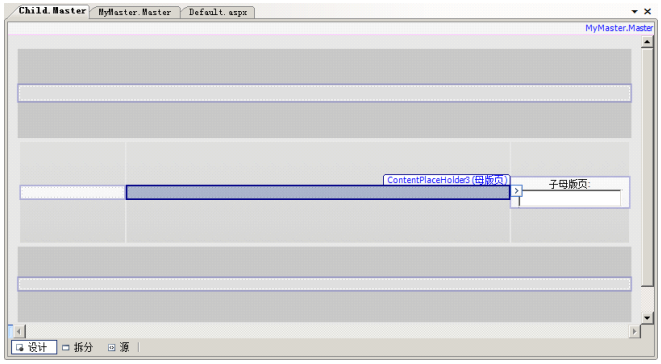

图 12-22　嵌套母版页

母版页嵌套完毕后，使用母版页的页面也应该进行相应的修改，在使用嵌套后，子母版页应该被声明到需要使用的页面，而不是母页面。简单的说，需要使用的页面应该声明的是子页面，而不是母母版页，在这里应该为 Child.master，示例代码如下所示。

```
<%@ Page Language="C#"
MasterPageFile="~/Child.Master"
AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="_12_2.Default" Title="无标题页" %>
```

上述代码声明了该页的母版页为 Child.master，运行结果如图 12-23 所示。

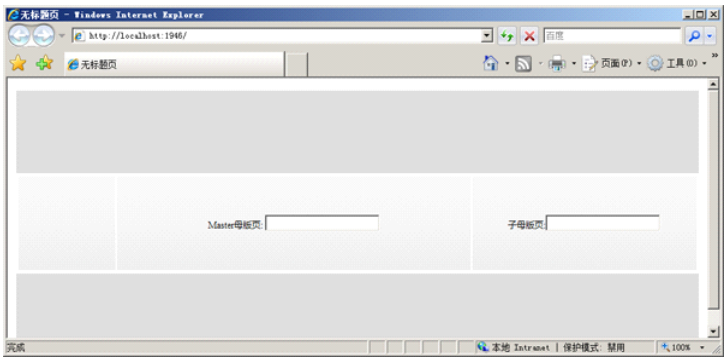

图 12-23　嵌套母版页

嵌套母版页之后，使用子母版页的页面将不能直接进行页面编辑，在 Visual Studio 2008 中，使用子母版页的页面将显示为空白，但并不表示页面显示将为空白。

12.3　Microsoft Expression 2

Microsoft Expression 2 是微软推出的一套专业的设计软件。Microsoft Expression 2 Studio 包括 Expression Web 2，Expression Blend 2，Expression Design 2，Expression Media 2 和 Expression Encoder 2，它们可以协调的同 Visual Studio 2008 一起协同合作，并支持 Vista 和 Window Server 2008 操作系统。

12.3.1　Microsoft Expression 2 简介

Microsoft Expression 2 是微软推出的强大的设计软件，不仅能够设计和进行网页布局，同时还支持 XAML 语言，能够进行 Silverlight 设计，WPF 设计，Microsoft Expression 2 Studio 包括的软件如下所示：

- Microsoft Expression Web：网页设计工具，用于网页设计、页面布局。
- Microsoft Expression Blend：交互设计工具，可以用于 Silverlight、WPF 的设计和开发。
- Microsoft Expression Design：平面图形设计工具，可以对图像进行编辑和设计。
- Microsoft Expression Media：多媒体编辑工具，可以对多媒体进行编辑、剪切和设计。
- Expression Encoder：音频编辑工具，可以对音频进行编辑、剪切和设计。

Microsoft Expression 2 包含的软件为微软的产品开发和设计做出了强有力的保障，其中对于 ASP.NET 开发人员最常用的就包括 Microsoft Expression Web、Microsoft Expression Blend 和 Microsoft Expression Design。

Microsoft Expression Web 提供了对 ASP.NET 中控件的支持，这弥补了传统的 Dreamware 系列对 ASP.NET 的控件不支持，造成 ASP.NET 页面设计困难，Microsoft Expression Web 还提供了页面的调试环境，通过 Microsoft Expression Web 也能够进行基本的网页调试。在开源的影响力之下，Microsoft Expression Web 还支持 php 的脚本编写。

Microsoft Expression Blend 提供了对.NET 中 Silverlight，以及 WPF 的设计和开发的支持。在 Visual Studio 2005 甚至是 Visual Studio 2008 中，Silverlight 以及 WPF 都不能很好的进行设计和可视化开发，因为 Silverlight 和 WPF 都是较新的技术，而 Microsoft Expression Blend 提供了对 Silverlight 和 WPF 的开发支持。

Microsoft Expression Design 用于平面设计，Microsoft Expression Design 不仅能够像传统的 Photoshop

一样设计和开发 JPG、GIF 格式的图片，也能够为 Silverlight 和 WPF 应用程序开发资源文件。这种资源文件可以是不规则的窗体，也可以是一段动画，Microsoft Expression Design 能够保存为资源文件所需要的文件类型。

Microsoft Expression 2 是微软推出的设计软件，在传统的开发过程中，虽然 Visual Studio 提供了可视化编程的解决方案，但是 Visual Studio 中可视化开发的效率依旧不高，对于计算机配置不是很高的用户更是如此。而另一方面，在开发过程中，开发小组很难将开发人员和设计人员完全的分离开，而使用 Microsoft Expression 2，可以使设计人员专注于设计，使开发人员专注于代码的编写。

12.3.2 安装 Microsoft Expression 2

Microsoft Expression 2 并不是免费的软件，但是开发人员可以在微软的官方主页上下载试用版本下载地址为 http://www.microsoft.com/expression/try-it/default.aspx，其中 Expression® Studio 2 包括了所有的软件包，如果无需其他软件包，可以选择单独的软件包进行下载。下载完成后，单击 msi 安装程序，即可安装 Microsoft Expression 2。系统会提取文件，当文件提取完毕后，即会加载进入安装界面，如图 12-24 所示。

等待安装程序初始化，安装程序会进入下一步，提示要求输入密钥并进行激活，安装程序会确定 Expression Studio 2 的安装状态，并继续安装。图 12-25 所示。

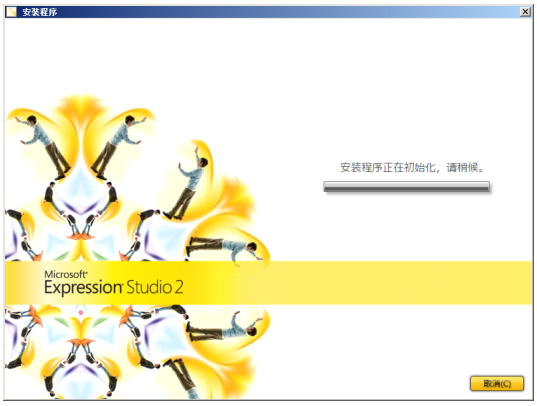

图 12-24　安装 Microsoft Expression 2

图 12-25　确定 Microsoft Expression 2 安装状态

按着安装程序的提示，基本上只需要单击下一步就能够将 Microsoft Expression 2 自行安装到本地计算机，Microsoft Expression 2 安装完毕后，就能够选择相应的应用程序做相应的开发。Microsoft Expression 2 的界面为黑色界面，看上去比较清新，但可能传统的用户很难适应这样的布局，如图 12-26 所示。

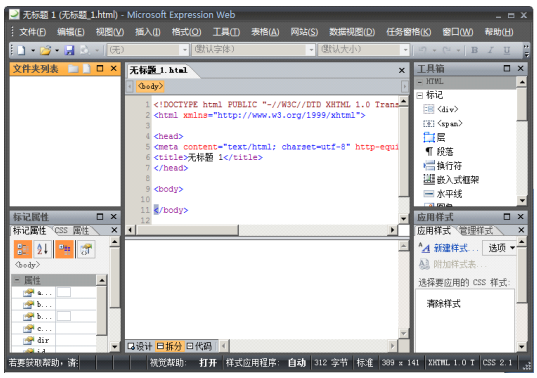

图 12-26　Microsoft Expression Web

12.4　使用 Microsoft Expression Web 2 制作页面

Microsoft Expression Web 2 是属于 Microsoft Expression 2 Studio 软件包中对 ASP.NET 开发人员来说最为强大的开发工具，Microsoft Expression Web 2 不仅提供了基本的网页布局功能，还支持 ASP.NET 中控件的拖动。

12.4.1　创建 ASPX 页面

通过 Microsoft Expression Web 2 能够快速的创建 ASPX 页面。在菜单栏中单击【文件】选项，单击【新建】按钮，可以选择创建相应的项目。Microsoft Expression Web 2 支持新建项目和新建网站，新建项目是为现有项目添加文件，也可以通过新建网站来新建另一个项目，在这里建立一个文件即可。单击【新建】按钮，系统会弹出对话框，用于创建新项目，如图 12-27 所示。

Microsoft Expression Web 2 不仅支持创建 ASPX 页面，也能够创建母版页、XML、动态 Web 模板，甚至能够支持创建 PHP 页面。这里可以选择一个 ASPX 页面进行创建，单击确定，创建一个 ASPX 页面。创建 ASPX 页面后，在 Microsoft Expression Web 2 的工具箱中，就可以看到 Microsoft Expression Web 2 为开发人员提供了 HTML 控件和 ASPX 控件，如图 12-28 所示。

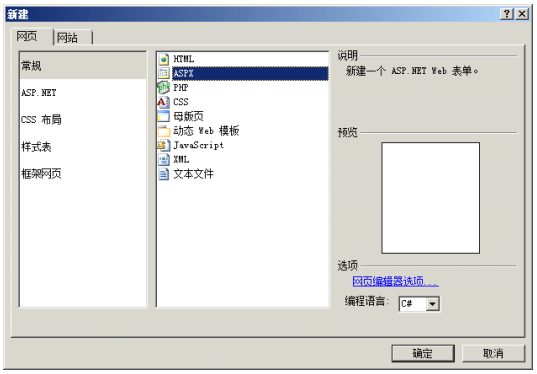

图 12-27　新建文件

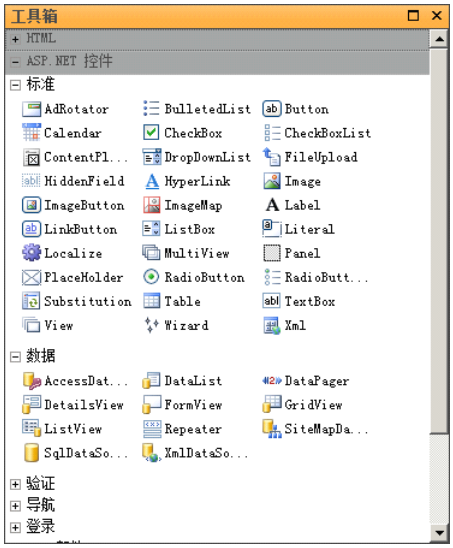

图 12-28　HTML 控件和 ASP.NET 控件

设计人员能够将页面布局进行设计，同时设计人员也能够拖动 ASP.NET 控件到页面布局中，这样就极大的方便了设计人员在前台界面的设计开发。而编程人员只需获取相应的页面，然后对页面进行逻辑代码的编写，即可组成一个完整的 ASPX 页面。Microsoft Expression Web 2 不仅能够支持设计人员对现有的页面进行控件的拖放，还能够支持进行数据源配置和数据绑定，如图 12-29 所示。

图 12-29　配置数据源

在可视化开发中，Microsoft Expression Web 2 的效率比 Visual Studio 2008 较高，因为 Microsoft Expression Web 2 只负责页面布局，并负责配置相应的数据源和数据绑定，虽然 Microsoft Expression Web 2 不能负责页面逻辑的开发，但是对于 ASP.NET 页面设计的支持已经非常强大了。

12.4.2　创建 CSS 层叠样式表

CSS 层叠样式表是在网站设计和开发中必不可少的，通过 Microsoft Expression Web 2 同样能够创建和使用 CSS 层叠样式表，在菜单栏中找到并单击【文件】选项，在下拉菜单中单击【新建】选项，在弹出对话框中选择【CSS】选项，单击【确定】按钮就能够创建 CSS 层叠样式表。CSS 文件能够对现有的页面进行样式控制，开发人员可以在 CSS 层叠样式表中编写样式控制代码，示例代码如下所示。

```
body
{
background:silver;
font-size:12px;
}
.div1
{
background-color:white;
padding:10px 10px 10px;
font-size:16px;
font-weight:bolder;
}
```

上述代码编写了 body 的样式，以及 class=“div1”的样式。在相应的页面则能够通过此样式表对页面中的标签进行样式控制。在 Microsoft Expression Web 2 中，可以智能的添加 CSS 层叠样式表。单击菜单栏中的【格式】选项，选择【CSS 样式】，在下拉菜单中选择【附加样式表】选项则会弹出附加对话框，单击【浏览】按钮选择相应的 css 文件即可，如图 12-30 所示。

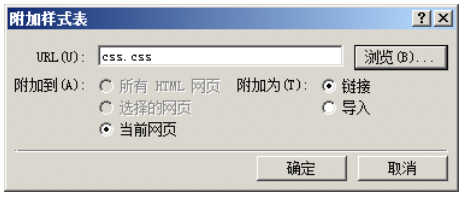

图 12-30　附加样式表

附加完成后，在页面的 HTML 代码中会自动增加 CSS 文件引用代码，示例代码如下所示。

```
<link href="css.css" rel="stylesheet" type="text/css" />
```

上述代码就为页面中声明的外联式 CSS 样式表，当声明了相应的外联式样式文件后，该页面就能够使用外联式 CSS 样式表提供的样式进行样式控制。

12.4.3 创建框架集

包含框架的页面被称为框架集。框架集是一个单独的文件，用于定义页面上所有框架的布局和属性，包括框架数量、框架的大小和位置，以及最初显示在每个框架中的页面的。框架集通常用于后台页面的开发，也用于帮助文档的开发，例如 MSDN 中对函数的查询，就是使用了框架集。

框架集是一个单独的页面，框架集是对页面中所有的框架的布局，每个框架都是另一个页面，框架集只是负责将页面组织并呈现在同一个页面中。单击【新建】按钮，在弹出窗口中选择框架集，如图 12-31 所示。

框架集能够预览，相应的框架集为不同的页面进行布局，这里创建一个目录类型的框架集，创建完成后，框架集页面会智能提示用户所需填充的页面，通过填充页面，能够填充相应的框架，如图 12-32 所示。

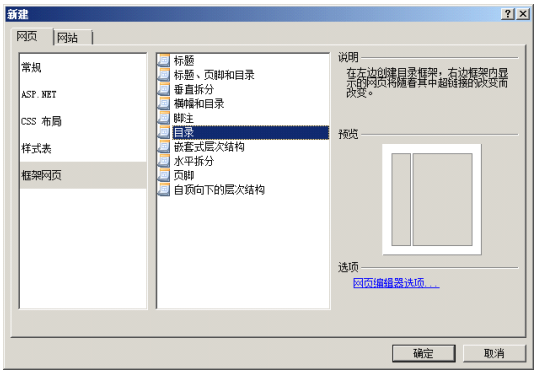

图 12-31 创建框架集

图 12-32 填充框架集

单击设置初始网页可以选择框架集所需要的网页，但是网页必须存在，若网页不存在，则可以单击新建网页填充框架集。左侧的框架可以用于导航，右侧的框架可以用于目录的显示，编写完成后如图 12-33 所示。

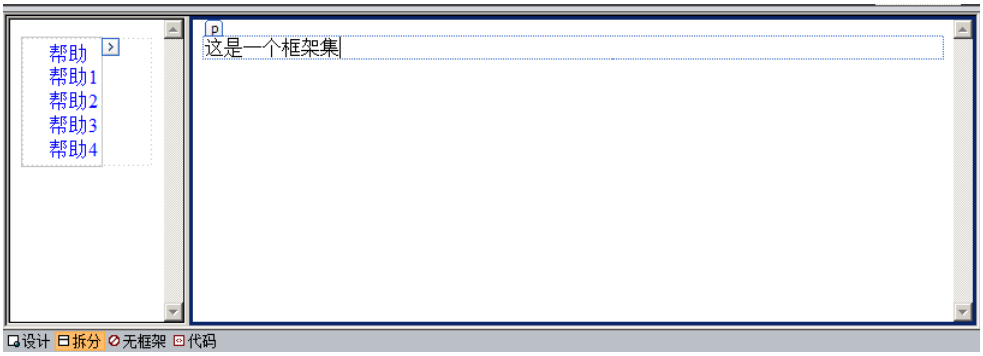

图 12-33 编写框架集

框架集创建后，框架集的页面 HTML 代码将自动生成，示例代码如下所示。

```
<html>
<head>
```

```
<meta content="text/html; charset=utf-8" http-equiv="Content-Type">
<title>无标题 1</title>
</head>
<frameset cols="150,*">
<frame name="contents" src="无标题_3.aspx" target="main">
<frame name="main" src="无标题_2.html">
<noframes>
<body>
<p>此网页使用了框架，但您的浏览器不支持框架。</p>
</body>
</noframes>
</frameset>
</html>
```

从上述代码可以看出，框架集就是将不同的页面呈现在同一页面的一种布局方法，上述代码中包括两个框架，这两个框架的页面分别为“无标题_3.aspx”和“无标题_2.html”。

注意：在使用框架集时，框架都包括 name 属性，这个属性在进行超链接时非常有用，通过编写超链接的 target 属性，能够指定相应的框架中的连接将能够对目标框架的页面进行更改。

12.5 小结

本章对 CSS，皮肤，主题做了详细的介绍，通过使用 CSS，能够优化网页代码布局，提高网页的友好度，增加用户粘度。同样，使用皮肤和主题能够控制控件的样式，并能够通过编程的方法动态的更改皮肤和主题，增强了代码的复用性。同时，本章还介绍了母版页，通过母版页能够将页面布局和控件进行分离，母版页只需对页面进行布局和样式控制，而内容窗体只需要镶嵌相应的控件即可。本章还包括：

- CSS 常用属性。
- 将 CSS 应用在控件上：在控件上使用 CSS。
- 应用和禁用主题：使用主题和禁用主题的方法。
- 母版页的运行方法：讲解了母版页是如何运行的。
- 嵌套母版页：讲解了如何进行母版页的嵌套。
- Microsoft Expression 2 简介。
- 使用 Microsoft Expression Web 2 制作页面。

本章还讲解了微软强大的设计工具——Microsoft Expression 2，通过 Microsoft Expression 2 能够为 ASP.NET 开发人员进行页面设计提供支持。