第 3 章 HTML 相关技术基础知识

纵观各种动态页面开发技术，无论是 JSP、ASP 还是 PHP 都无法摆脱 HTML 的影子。这些动态的页面开发技术无非是在静态 HTML 页面的基础上添加了动态的可以交互的内容。HTML 是所有动态页面开发技术的基础。在接下来的章节将要详细介绍的就是 HTML 相关的一系列技术，包括 HTML、JavaScript 和 CSS。其中 HTML 是一组标签，负责网页的基本表现形式；JavaScript 是在客户端浏览器运行的语言，负责在客户端与用户的互动；CSS 是一个样式表，起到美化整个页面的功能。

本书不是详细介绍 HTML 的专著，在本章也只是讲解 Web 开发中最常见的 HTML 知识，目的在于使读者能尽快进入 Web 开发的状态。如果读者有更深层次的需求可以参考专门讲解 HTML 的书籍。

3.1 HTML 基础知识

在各种 Web 开发技术中，HTML 无疑是最为基础的。任何动态语言都离不开 HTML 的支持。所以在开始 Web 开发的学习之前，读者还是需要静下心来打好这个基础。在这个章节中将会概述 HTML 的框架知识。

3.1.1 什么是 HTML

HTML（Hyper Text Markup Language）即超文本标记语言，用来描述 Web 文档数据。用户可以通过 URL 链接来访问这种 Web 文档，从而达到信息展示、信息共享的目的。下面就是一个简单的 HTML 文档的例子。

```
//--------文件名：First.html--------------------
<html>
  <head>
    <title>这是第一个 HTML 例子</title>
  </head>
  <body>
    欢迎光临！这是我的第一个 HTML 文档。 <br/>
  </body>
</html>
```

在这个 HTML 文档中，可以看出 HTML 的简单结构，每个 HTML 文档都包括一对<html></html>标签，这是所有 HTML 文档所必需的。在这个标签中间还包括着其他两对<head></head>、<body></body>，其中在<head></head>中是 HTML 文档的头信息，包括标题、关键字、页面编码格式、引入的 CSS 或者是 JavaScript 文件的路径等基本信息。在<body></body>中间放置的是文档要表述展示的内容，在上面这个例子中我们要展示的仅仅是“ 欢迎光临！这是我的第一个 HTML 文档。”这句话。所以在<body></body>这对标签中间没有其他内容。

注意：一般情况下，可以包括其他内容的 HTML 标签都是成对出现的，例如上面例子中的<title></title>这对标签，它包含了一个文字的标题信息，所以成对出现。而
这样的标签仅仅是一个回车换行的作用，它不包含其他内容，所以不成对出现。

3.1.2 HTML 运行原理

在上面的例子中我们看到的仅仅是几行代码，而在大家的印象中 HTML 文档就是网页，究竟如何把 HTML 文档转化成我们常见的网页，这就是接下来要介绍的内容。

我们平时所看的丰富多彩的网页都是通过浏览器看到的，当你在浏览器中敲入网址，然后就可以看到对应的一个网页，上面有各种各样的内容。其实在本质上你敲入的网址就是互联网上一个 HTML 文档的访问路径，浏览器可以根据这个路径取回这个 HTML 文档，取回的文档格式和上面例子中的类似。只不过把这些代码翻译成了一个形象的网页。

前面介绍 HTML 定义的时候就说过，HTML 是一种标记语言，每一种 HTML 标签都是有一定表现含义的。浏览器就是按照 HTML 标签的语义规则把 HTML 代码翻译成漂亮的网页。就像上面的 HTML 文档示例会被翻译成如图 3.1 所示的效果。

图 3.1 First.html 在浏览器中的运行效果

在图 3.1 中可以明显看到，上面文档中“<title>这是第一个 HTML 例子</title>”这行代码被翻译成了网页的标题。而<body></body>标签中包含的内容被翻译成了网页只能够显示的内容，在这里就只有一句话。等接下来常用的 HTML 标签介绍结束之后，就可以构造出丰富表现形式的网页了。

3.1.3 HTML 常用标签

在本节要介绍的是常用标签的基本用法。

1．<table>

在 HTML 的布局标签中，<table>标签是使用频率最高的一个。它可以把一组信息用表格的形式表示出来，具体使用示例参考下面这个示例。

```
//--------文件名：Table.html--------------------
<html>
  <head>
    <title>表格标签示例</title>
  </head>
  <body>
   <table border="1">
  <tr>
    <td>第一行第一列</td>
    <td>第一行第二列</td>
  </tr>
  <tr>
```

```
    <td>第二行第一列</td>
    <td>第二行第二列</td>
  </tr>
</table>
 </body>
</html>
```

在这个示例程序中展示了表格的基本用法，其中<table></table>标签的含义是表格，这个表格还有一个属性 border='1'，含义是这个表格的边框是 1 像素。在表格中间有<tr></tr>标签，这个标签的含义是表格的行，在行中间有<td></td>，这个标签的含义是表格的列。在<td></td>中间有文字，这些是这个单元格要显示的内容。其中<td></td>标签必需用在<tr></tr>之间，而<tr></tr>标签只可以用在<table></table>之间。如果这三对标签的位置或者是顺序用错，整个表格就不能正常显示。

按照上面介绍的标签的语义，上面这段程序表示的就是一个两行两列的表格。在浏览器中的运行效果如图 3.2 所示。

图 3.2 表格示例程序运行效果

上面这个示例程序仅仅展示了一个简单的布局，而常见的复杂布局都可以是利用表格的嵌套来实现，这也是在很长一段时间内开发人员采取最多的一种布局手段。

2．DIV

在以往的 Web 页面开发中，表格是首选的布局元素，但是近来 DIV 元素越来越受开发者的欢迎，究其愿意就是定位比较方便，可以采取相对定位，也可借助于 CSS 绝对定位，同时采用 DIV 作为布局元素可以避免像表格那样的层层嵌套。

其实层也就是一个容器，在里面可以放任何需要展示的内容。和表格的使用方法类似。在章节 3.3.3 中我们将结合 CSS 详细解释 DIV 的使用方法。

3．<a>

在浏览一个网站的时候，我们经常会遇到一些链接，单击这些链接就会导航的其他的页面。这个链接使用的就是<a>超链接标签，这个标签可以像下面这个程序中这样使用。

```
//--------文件名：Link.html--------------------
<html>
  <head>
    <title>超链接示例</title>
  </head>
  <body>
    <a href="http://www.sohu.com" target="_blank">搜狐</a>
  </body>
</html>
```

在这个程序中定义了一个超链接。

```
<a href="http://www.sohu.com" target="_blank">搜狐</a>
```

<a>是超链接的标签，其中 href 是链接的地址，在这里我们指向 http://www.sohu.com，tartget="_blank"指的是在一个新的浏览器窗口打开这个链接，这个属性的值还可以选择_self、_parent、_top。“搜狐”是我们显示在这个链接标签中的内容，单击这两个字的时候就跳转到链接地址。上面这个例子中 href 属性是必不可少的。targert 是可选的属性。这个程序的运行效果如图 3.3 所示。

单击“搜狐”这个链接就会在新窗口打开搜狐网站主页面。

注意：在 HTML 中对字母的大小写不敏感，同样一个标签大些小写都不影响显示的效果。

4．<img>

在目前的网站开发中，对图片的依赖是其他元素所不能替代的，一个漂亮的网页往往是由一系列图片组合而成。在这里介绍 HTML 中图像标签<img>的使用方法，具体方法参考下面例子。

```
//--------文件名：Image.html--------------------
<html>
  <head>
    <title> 图片使用示例</title>
  </head>
  <body>
   <img src="hand.gif"/>
  </body>
</html>
```

<img>是图像标签，src 属性指明了图像文件的位置，上面这个程序运行效果如图 3.4 所示。

图 3.3 超链接示例程序运行效果

图 3.4 图片使用示例程序运行效果

注意：HTML 文档的源文件都是以.html 或者.htm 作为后缀名。在上面这个程序中，图片和 HTML 文档在放在同一个文件夹中，所以只用给 src 属性提供文件名即可，如果图片和 HTML 文档不在同一个文件夹中就需要提供相对的访问路径。

3.1.4 HTML 表单标签

前面讲述的都是 HTML 向用户展示信息的标签，在本节要介绍的内容就是 HTML 用来收集用户输入的标签。<form></form>是表单标签，只有在这个标签中的用户输入才会被提交给服务器。表单标签的使用方法类似下面这种格式。

```
<form action="" method="get"></form>
```

其中 action 属性指明了处理这个表单数据的对象。method 属性指明传送这个表单中的数据所使用的方法，method 属性有两个值可选，get 提交表单的时候，表单的所有的输入都会显示在浏览器的地址栏中，而且 get 对所能提交数据的大小也比较小，因为所有提交的内容都要显示在地址栏，而 URL 的大小是有限制的。method 还可以选择 post 方法，这个方法对提交数据的时候不在浏览器的地址栏中显示，而且所能支持数据量比较大。

在表单中可以包含用输入标签收集用户输入的数据，其中<input></input>是输入标签，使用方法类

似下面这种格式。

```
<input type="text"></input>
```

其中 type 属性指明了用户的输入方式，type 可以选择的值如图 3.5 所示。

图 3.5 输入标签 type 属性取值范围

其中 button 就是一个简单的按钮；checkbox 是复选框,file 是文件选择对话框；hidden 是隐藏域，当要提交一个值又不想在页面显示的时候用到这个选项；image 是构造图片按钮；password 是密码输入框，当输入的时候输入的值自动变为小黑点;radio 是单选按钮；reset 是重置按钮，这个按钮实现的功能是把当前表单中所有已经输入的值清空；submit 是提交按钮，单击这个按钮就会把表单中所有的输入数据提交给 action 对应的处理对象，对于一个表单来说这个按钮是必不可缺的，因为表单的目的就是提交用户输入到服务器，当然少不了提交按钮；text 是文本输入框，在表单中也是用的比较多的。

除了<input></input>输入标签外，在表单中还有<select></select>标签，这个标签是一个下拉列表，其中每一个下拉选项都是由<option></option>组成的。下面给出一个简单的表单示例，仅仅是展示如何使用上面介绍的各种常用的输入标签，具体代码如下。

```
//--------文件名：Form.html--------------------
<html>
  <head>
    <title>表单使用示例</title>
  </head>
  <body>
      <form action="" method="post">
            用户名：<input type="text" name="userName"></input><br>
            密码：<input type="password" name="password"></input><br>
            重新输入密码:<input type="password" name="rePassword"></input><br>
            性别：<input type="radio" name="sex" value="男">男
                 <input type="radio" name="sex" value="女">女<br>
            出生日期：<select name="birth">
                        <option value="0">－请选择－</option>
                        <option value="1981">1981</option>
                        <option value="1982">1982</option>
                        <option value="1983">1983</option>
                        <option value="1984">1984</option>
                        <option value="1985">1985</option>
                        <option value="1986">1986</option>
                    </select>年<br>
           兴趣：<input name="habit" type="checkbox" value="1">音乐</input>
                <input name="habit" type="checkbox" value="2">动漫</input>
                <input name="habit" type="checkbox" value="3">电影</input><br>
          <input type="submit" value="提交"/>
```

```
            <input type="reset" value="取消" />
        </form>
    </body>
</html>
```

这个程序展示了一个简单的表单，展示了最长用的几种输入标签的用法。其中。

```
性别：<input type="radio" name="sex" value="男">男
<input type="radio" name="sex" value="女">女<br>
```

单选按钮的使用需要注意，属于一组的单选按钮名称必需一样，例如在这个示例中男、女选项只能选择一个，那么只有它们的名字相同才可以实现。否则两个按钮都可以选择。

程序中的提交按钮如果不提供 value 值的话会在按钮上显示默认的“提交查询内容”；同样重置按钮如果不提供 value 值，也将显示默认值“重置”。上面这个表单的运行效果如图 3.6 所示。

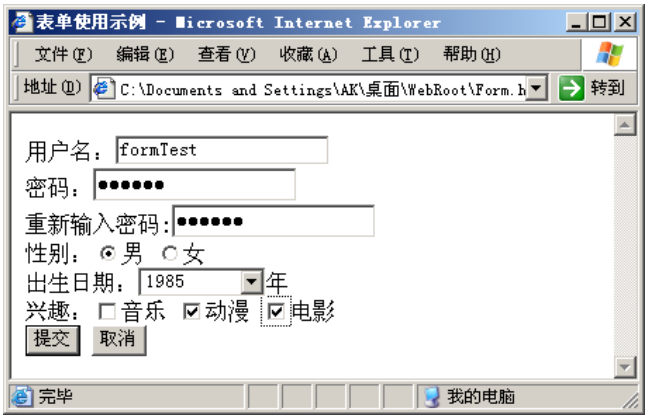

图 3.6 表单使用示例程序运行效果

3.1.5 HTML 其他标签

在本章只是介绍了 HTML 最基本最常用的几个标签，HTML 还有很多其他标签，例如应用程序标签中可以加入不同动态程序代码，多媒体标签中可以加入多媒体文件，Flash 标签中可以加入 Flash 动画，文本标签可以用各种方式组织文本内容的显示。

读者如果要深入全面的研究 HTML 标签，可以参考这方面的参考手册。在本书中不再对这些内容进行详细的介绍。

3.2 JavaScript 基础知识

JavaScript 的出现给静态的 HTML 网页带来很大的变化。JavaScript 增加了 HTML 网页的互动性，使以前单调的静态页面变得富有交互性，它可以在浏览器端实现一系列动态的功能，仅仅依靠浏览器就可以完成一些与用户的互动。在下面的章节中将要简单介绍这种技术的基础知识。

3.2.1 什么是 JavaScript

JavaScript 是一种简单的脚本语言，可以在浏览器中直接运行，无需服务器端的支持。这种脚本语言可以直接嵌套在 HTML 代码中，它响应一系列的事件，当一个 JavaScript 函数响应的动作发生时，浏

览器就会执行对应的 JavaScript 代码，从而在浏览器端实现与客户的交互。

3.2.2 JavaScript 中的事件

在 HTML 的标签中，绝大部分都可以触发一些事件，例如鼠标单击、双击、鼠标经过、鼠标离开等一系。JavaScript 最主要的功能就是与用户的交互，而用户只能在表单中提交输入内容，所以表单的所有输入标签都可以出发鼠标事件、键盘事件等 JavaScript 所能响应的常见事件。

在这里我们不在一一列举每一个 HTML 标签所能触发的事件，在下面仅仅以一个普通按钮所能触发的事件为例介绍 JavaScript 的事件，其他方法的处理过程都是相同的。

如图 3.7 就是一个简单的按钮所能触发的事件的列表。

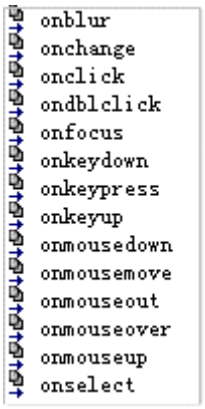

图 3.7 按钮标签所能触发的事件列表

下面介绍如何通过这样一个事件调用一个 JavaScript 函数，具体代码如下所示。

```
//--------文件名：Event.html--------------------
<html>
  <head>
    <title>事件触发示例</title>
    <script language="javascript">
        function test()
        {
            alert("事件触发测试！");
        }
    </script>
  </head>
  <body>
      <form action="" method="post">
          <input type="button" value="单击事件测试" onclick="test()">
      </form>
  </body>
</html>
```

在上面这个程序中可以看出，单选按钮触发一个鼠标单击事件，当鼠标单击这个按钮的时候，浏览器就会调用 JavaScript 中的 test()这个方法，test()方法的功能就是弹出一个如图 3.8 所示的对话框，其中对话框中显示的内容就是 test()方法中 alert 中间的参数内容。

图 3.8　JavaScript 鼠标单击测试弹出的对话框

```
<script language="javascript">
    function test()
    {
        alert("事件触发测试！");
    }
</script>
```

上面这段代码就是 JavaScript 代码的典型格式，所有的 JavaScript 代码都是以函数（function）的方式存在的，HTML 触发的动作对应的就是这里面的方法。

注意：当 JavaScript 的代码量比较大的时候，或者在多个页面都会用到同样功能的 JavaScript 代码的时候，可以把这些 JavaScript 代码放在一个以.js 为后缀名的文件中。然后在 HTML 页面中按照<script src=" "></script>这种格式进行引用，其中 src 是.js 文件放置的相对路径。

3.2.3　JavaScript 中的对象简介

JavaScript 所实现的动态功能，基本上都是对 HTML 文档或者是 HTML 文档运行的环境进行的操作。那么要实现这些动态功能就必需找到相应的对象。JavaScript 中有已经定义过的对象供开发者调用，在了解这些对象之前先看图 3.9 所示的内容。

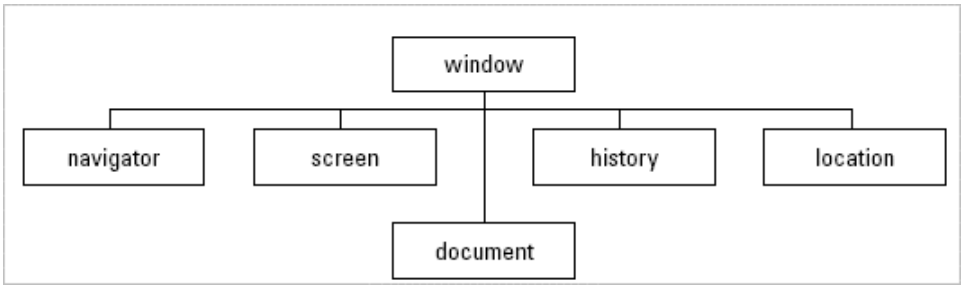

图 3.9 在浏览器窗口中的文档对象模型

图 3.9 中的内容是一个简单的 HTML 文档在浏览器窗口中的文档对象模型，其中 window、navigator、screen、history、location 都是 HTML 文档运行所需的环境对象，document 对象才是前面讲述的 HTML 文档，当然这个 document 对象还可以划分出 html、head、body 等分支。

- window 对象是所有对象中最顶层的对象，HTML 文档在 window 对象中显示。
- navigator 对象可以读取浏览器相关的信息。
- screen 对象可以读取浏览器运行的物理环境，例如屏幕的宽和高，此处的单位为像素。
- document 对象是整个网页 HTML 内容，每个 HTML 文档被浏览器加载以后都会在内存中初始化一个 document 对象。
- history 对象可以控制浏览器的前进和后退。
- location 对象可以控制页面的跳转。

在接下来的章节中将对最常用的 window 对象、location 对象、document 对象进行详细的介绍。

3.2.4 window 对象简介

如图 3.9 所示，window 对象是所有 JavaScript 对象中最顶层的对象，整个 HTML 文档就是在一个浏览器的一个窗口，即 window 对象中显示。当打开一个浏览器窗口以后，甚至是一个空白窗口，这时候在内存中间已经定义了一个 window 对象。window 对象所提供的方法很多，在下面的内容中将进行对最常用的几种方法进行介绍。

1．窗体的创建和关闭

利用 window 对象可以新建浏览器窗口，也可以关闭浏览器窗口，下面来看具体的操作代码。

```
//--------文件名：Window.html--------------------
<html>
<head>
<title>窗体的创建和关闭示例</title>
<script type="text/javascript">
      var win;
      function createWin() {
          win = window.open("","","width=300,height=200");
      }
      function closeWin() {
           if (win) {
                win.close();
           }
      }
</script>
</head>
<body>
<form>
<input type="button" value="创建新窗口" onclick="createWin()">
<input type="button" value="关闭新窗口" onclick="closeWin()">
</form>
</body>
</html>
```

这个程序在浏览器中运行以后，界面上会有两个按钮，鼠标单击“创建新窗口”按钮会弹出一个新的浏览器窗口，这个窗口的宽为 300 像素，高为 200 像素。鼠标单击“关闭新窗口”，这个弹出窗口就会被关闭。

上面这个程序中用到的就是 window 对象的 open 和 close 两个方法，open 方法新建一个窗口，close 方法关闭指定窗口。

2. 三种常用的对话框

在 window 对象中，有三种常用的对话框，第一种是警告对话框，第二种是确认对话框，第三种是输入对话框。在面这个示例中展示了这三个对话框的用法。

```
//--------文件名：Dialog.html--------------------
<html>
<head>
<title>三种常用的对话框</title>
<script type="text/javascript">
      function alertDialog() {
```

```
        alert("您成功执行了这个操作。");
    }
    function confirmDialog() {
        if(window.confirm("您确认要进行这个操作吗？"))
            alert("您选择了确定！");
        else
            alert("您选择了取消");
    }
    function promptDialog() {
       var input = window.prompt("请输入内容：");
       if(input !=null)
       {
            window.alert("您输入的内容为"+input);
       }
    }
</script>
</head>
<body>
<form>
<input type="button" value="警告对话框" onclick="alertDialog()">
<input type="button" value="确认对话框" onclick="confirmDialog()">
<input type="button" value="输入对话框" onclick="promptDialog()">
</form>
</body>
</html>
```

上面这个程序在浏览器中运行以后，鼠标单击“警告对话框”按钮，会弹出如图 3.10 所示的对话框。鼠标单击“确认对话框”按钮，会弹出如图 3.11 所示的对话框。

鼠标单击“输入对话框”按钮，会弹出如图 3.12 所示的对话框。

图 3.10 警告对话框

3.11 确认对话框

图 3.12 输入对话框

在上面这个程序中，对后两种对话框的返回值也进行了示例处理，读者可以参照上面的格式稍加修改就可以用到自己的程序中去。

3.2.5 document 对象简介

document 对象是在具体的开发过程中用的最频繁的对象，利用 document 对象可以访问页面上任何的元素。通过控制这些元素可以完成与用户的互动。

1．利用 document 定位 HTML 页面元素

所有的 HTML 页面元素都可以用 document.getElementById()这个方法访问，还有一部分 HTML 页面元素可以使用数组来访问，例如表单元素就可以使用 document.forms[“formName”]或者是 document.forms[“formIndex”]来访问，其中 formName 是表单的名称，formIndex 是表单的序号。

当 HTML 页面中使用了 iframe 时，使用 document 对象定位 iframe 中的元素时，首先要取得 iframe

的 document 对象，然后在这个对象上继续操作。这个 document 对象可以这样获得：document.frames[“framesName”].document，这里的 frameName 是 iframe 的名称，取得的这个 iframe 的 document 对象使用方法和其他 document 的使用方法是一样的，在这个 document 基础上可以继续定位 iframe 中的元素。

同样道理，如果在页面中使用了框架集 frameset 的时候，也可以采用像上面 iframe 一样的处理方法。

2．利用 document 对象动态生成 HTML 页面

用 document 对象不仅仅可以取出或者设置 HTML 页面元素的值，而且可以动态的生成整个新的 HTML 文档。下面的例子就是利用 document 对象生成一个新的 HTML 文档。

```
//--------文件名：CreateHtml.html--------------------
<html>
<head>
<title>动态生成 HTML 页面</title>
<script type="text/javascript">
function create() {
      var content = "<html><head><title>动态生成的 HTML 文档</title></head>";
      content += "<body><font size='2'><b>这个文档的内容是利用 document 对象动态生成的</b></font></h1>";
      content += "</body></html>";

      var newWindow = window.open();
      newWindow.document.write(content);
      newWindow.document.close();
}
</script>
</head>
<body>
<form>
<input type="button" value="创建 HTML 文档" onclick="create()">
</form>
</body>
</html>
```

在上面这个示例程序中，利用 JavaScript 动态生成一个 HTML 代码串，并且利用 document 对象把这段代码串写入新建窗口的 document 对象中，这样就完成动态生成 HTML 页面的功能，此处如果是在原窗体显示，只需要把新建的窗体对象替换成当前窗体的 window 对象即可。

上面这个程序在浏览器中打开以后，鼠标单击“创建 HTML 文档”按钮，就会弹出一个如图 3.13 所示的新窗体，窗体内容是利用 document 对象动态创建的。

图 3.13 利用 document 对象动态生成的 HTML 页面

注意：在 JavaScript 的字符串操作中，不允许在单引号中嵌套单引号或者在双引号中嵌套双引号，但是这两种引号可以交叉使用，可以在双引号中嵌套单引号，也可以在单引号中嵌套双引号。

3.2.6 location 对象简介

在 HTML 标签中可以用<a>超链接标签来控制网页中的跳转，在 JavaScript 中如果要实现类似的网页跳转功能只能选择 location 对象，这个对象的使用方法非常简单，只需要在 JavaScript 代码中添加下面这行代码即可。

```
window.location.href = “http://www.sohu.com”;
```

window 对象就是要控制的目标窗体，赋值的内容就是窗体将要跳转到的页面，这行代码可以实现类似超链接标签的效果。

3.2.7 JavaScript 输入验证

在前面章节 3.1.4 中我们简单介绍了 HTML 表单的基本知识，在本章将介绍在浏览器端对用户输入的简单验证，这种验证仅仅局限于输入格式等方面。

在这里我们仍然使用 3.1.4 中的表单内容，只是添加了输入验证的内容，假设验证规则为：用户名、密码、重新输入密码这三项不能为空，用户名长度不能小于 6 位，两次密码输入必需相同。

下面是添加按照上面定义的这也规则验证的表单代码。

```
//--------文件名：ValidateForm.html--------------------
<html>
  <head>
    <title>表单输入验证示例</title>
    <script type="text/javascript">
        function validate()
        {
            var userName=document.forms[0].userName.value;
            var password=document.forms[0].password.value;
            var rePassword=document.forms[0].rePassword.value;

            if(userName.length<=0)
                alert("用户名不能为空！");
            else if(password<=0)
                alert("密码不能为空！");
            else if(rePassword.length<=0)
                alert("重新输入密码不能为空！");
            else if(userName.length<6)
                alert("用户名不能小于 6 位！");
            else if(password!=rePassword)
                alert("两次输入密码不一致！");
            else
                {
                    alert("验证通过，表单可以提交！");
                    document.forms[0].submit();
                }
        }
    </script>
  </head>
  <body>
      <form action="" method="post">
```

```
            用户名：<input type="text" name="userName"></input><br>
            密码：<input type="password" name="password"></input><br>
            重新输入密码:<input type="password" name="rePassword"></input><br>
            性别：<input type="radio" name="sex" value="男">男
                 <input type="radio" name="sex" value="女">女<br>
            出生日期：<select name="birth">
                        <option value="0">－请选择－</option>
                        <option value="1981">1981</option>
                        <option value="1982">1982</option>
                        <option value="1983">1983</option>
                        <option value="1984">1984</option>
                        <option value="1985">1985</option>
                        <option value="1986">1986</option>
                      </select>年<br>
          兴趣：<input name="habit" type="checkbox" value="1">音乐</input>
               <input name="habit" type="checkbox" value="2">动漫</input>
               <input name="habit" type="checkbox" value="3">电影</input><br>
        <input type="button" value="提交" onClick="validate()"/>
        <input type="reset" value="取消" />

      </form>
  </body>
</html>
```

这个程序针对上面的验证规则，对输入的各项进行检查，如果有一条不满足就不提交表单，例如用户没有输入密码就提交表单，就会弹出如图 3.14 所示的对话框，其他各种错误提示跟这个密码提示类似。

图 3.14 没有输入密码的提示信息

注意：在进行表单输入验证的时候，必需把<input type="submit" value="提交"/>中间的 type 换为 button，同时给这个 button 添加一个 JavaScript 事件，这时候在输入验证中使用 JavaScript 提交窗体。如果不把输入的类型改为 button，则无论输入是否合法窗体都会被提交。

3.2.8 JavaScript 高级应用探讨

上面介绍的示例中，JavaScript 都没有和服务器进行互动，都是在浏览器中独立执行，这样所能实现的与客户互动的功能就比较有限了，例如现在用户注册的时候需要验证这个用户名是否已经被占用，这个功能便需要到服务器中进行查询。然而在我们上面的验证中，只有当表单提交的时候服务器才会相应请求，其他情况下，如果没有刷新整个页面是不能实现与服务器之间的通信的。

现在有这样一种解决方案，那就是使用 Ajax，利用 Ajax 就可以实现页面的局部刷新，当输入用户名的时候可以同时进行与服务器进行通信，在数据库中进行查询，这时候只需要刷新局部的页面即可，这种技术的核心语言就是 JavaScript，通过 JavaScript 操作 XMLHttpRequest 对象来实现与服务器之间的局部通信。这种技术在第二十章中将会详细介绍。

3.3 CSS 基础知识

在前面的内容中讲解了 HTML 和 JavaScript，现在我们已经基本可以编出具有简单互动的网页，但是仅仅这样还是不够的，一个专业的网页需要在字体、颜色、布局等方面进行各种设置，需要给用户带来视觉的冲击。接下来的内容将要介绍这种美化页面的技术。

3.3.1 什么是 CSS

CSS（Cascading Style Sheets）即层叠样式表，也就是通常所说样式表。CSS 是一种美化网页的技术。通过使用 CSS，可以方便、灵活地设置网页中不同元素的外观属性，通过这些设置可以使网页在外观上达到一个更高的级别。

同时，CSS 与 JavaScript 结合可以在给用户带来更具变化的外观体验，例如下拉菜单等功能的实现都是通过 CSS 与 JavaScript 来实现的。

3.3.2 CSS 属性设置

CSS 美化网页就是通过设置页面元素的属性来实现的，在下面的内容中将介绍 CSS 属性设置的基本方法。

1．字体、颜色、背景等属性设置

字体属性是 CSS 最基本的属性，其最常用的属性包括 font-family、font-style、font-weight、font-size 等，其中 font-family 定义使用哪种字体，font-style 定义是否使用斜体，font-weight 定义字体的粗细，font-size 定义字体的大小。

color 定义前景颜色，background-color 定义背景颜色，background-image 定义背景图片，background-repeat 定义背景图片重复方式，background-attacement 定义滚动，background-position 定义背景图片的初始位置。下面通过一个示例程序展示这些属性的设置方法。

```
FontColor.html
<html>
    <head>
        <title>字体、颜色、背景属性设置示例</title>
    </head>
    <body>
      <div align="left">字体属性设置：</div>
      <span style="font-family:幼圆;font-style:italic;font-weight:bold;font-size:10pt;">幼圆、斜体、
      黑体、10pt</span><br>
      <span style="font-family:隶书;font-size:16pt;">隶书、黑体、16pt</span><br>
      <div align="left">字体属性设置：</div>
      <span style="color:red;background-color:yellow;">前景红色、背景黄色</span><br>
      <span style="background-image:url('hand.gif');background-repeat:no-repeat;">    
      图片背景、不重复</span><br>
    </body>
</html>
```

这个页面在浏览器中的运行效果入股 3.15 所示。

2．鼠标样式属性设置

在一些网页中，我们经常会遇到这样一种情况，当把鼠标移到不同区域，或者是在执行不同功能的时候，鼠标的形状都会发生变化。这种功能的实现其实非常简单，就是控制 CSS 中的 cursor 属性来实现的，其中 cursor 的属性列表如图 3.16 所示。

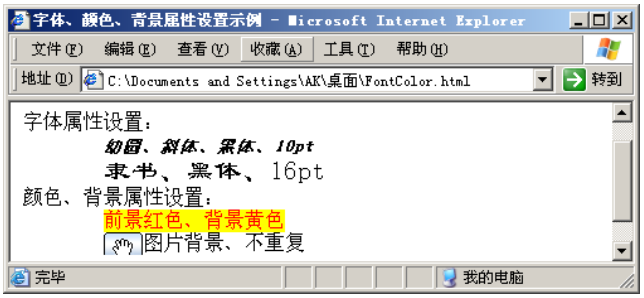

图 3.15 字体、颜色、背景属性设置运行效果

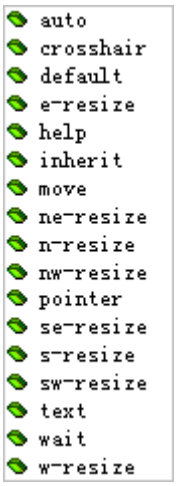

图 3.16 cursor 属性列表

在下面的例子中将要展示几种鼠标样式的设置方法，其他属性的使用方法都是相同。具体的设置方法参考下面的代码。

```
//--------文件名：Cursor.html--------------------
<html>
    <head>
        <title>设置鼠标样式</title>
    </head>
    <body>
        <div style="font-family:宋体;font-size:10pt;">
          <span style="cursor:hand">手形状</span><br>
          <span style="cursor:move">移动</span><br>
          <span style="cursor:ne-resize">反方向</span><br>
          <span style="cursor:wait">等待</span><br>
          <span style="cursor:help">求助</span><br>
          <span style="cursor:text">文本</span><br>
          <span style="cursor:crosshair">十字</span><br>
          <span style="cursor:s-resize">箭头朝下</span>
          </div>
    </body>
</html>
```

上面这个网页在浏览器中打开以后，鼠标移动到不同区域就会变成不同样式。

3.3.3 CSS 绝对定位示例

在 HTML 中布局一般情况下需要使用表格，如果要定位只有通过表格的嵌套来实现，这种方法的确可以在一定程度上解决问题，但是却不能随意定位页面元素，而且对某个元素位置的改变有可能影响到整个页面的布局。

在 CSS 中提供了灵活的定位的方法，所以在页面布局中我们又多了一种可以选择的方案。在 CSS 中常用的定位属性有 position、left、top、width、height、overflow、z-index、visibility 等，其中 position 定义采用绝对定位（absolute）、相对定位（relative）还是静态定位（static），left 和 top 定义横纵坐标的位置，width 和 height 定义宽和高，overflow 定义内容超出的处理方法，z-index 定义立体效果，visibility 定义可见性。 在下面的示例程序中就展示了怎么使用 CSS 实现绝对定位。

```
//--------文件名：Locate.html--------------------
<html>
  <head>
    <title>CSS 定位示例</title>
  </head>
  <body>
    <div style="position:absolute;left=100;top=20;visibility=visible">
     下面这个图片是可见的：
        <img src="hand.gif"/>
    </div>
    <div style="position:absolute;left=100;top=60;visibility=hidden">
     下面这个图片是不可见的：
        <img src="hand.gif"/>
    </div>
  </body>
</html>
```

这个程序在页面上不同的位置放置了两个 DIV，每个 DIV 中都有一个图片，但是第二个 DIV 设置为隐藏的，所在浏览器中的运行的时候只能显示一个图片。

3.3.4 JavaScript+DIV+CSS 实现下拉菜单

在 Web 应用中，下拉菜单的可以说是随处可见，在学习了 JavaScript 和 CSS 以后实现起来毫无难度。其原理就是在用 JavaScript 控制不同 DIV 的显示和隐藏，其中所有的 DIV 都是用 CSS 定位方法提前定义好位置和表现形式，下拉的效果只是当鼠标经过的时候触发一个事件，把对应的 DIV 内容显示出来而已。下面的例子中将会实现一个简单的下拉菜单。

```
//--------文件名：Menu.html--------------------
<html>
  <head>
    <title>下拉菜单示例</title>
    <script language="javascript">
        //当鼠标移到菜单选项的时候显示对应的 DIV
        function show(menu)
        {
            document.getElementById(menu).style.visibility="visible";
        }
        //当鼠标移出的时候隐藏所有的 DIV
        function hide()
        {
            document.getElementById("menu1").style.visibility="hidden";
            document.getElementById("menu2").style.visibility="hidden";
            document.getElementById("menu3").style.visibility="hidden";
       }
```

```
    </script>
  </head>
  <body>
      <table>
          <tr bgcolor="#9999FF">
              <td width="80" onMouseMove="show('menu1')" onMouseOut="hide()">菜单一</td>
              <td width="80" onMouseMove="show('menu2')" onMouseOut="hide()">菜单二</td>
              <td width="80" onMouseMove="show('menu3')" onMouseOut="hide()">菜单三</td>
          </tr>
      </table>
       <div id="menu1"    onMouseMove="show('menu1')"
      onMouseOut="hide()" style="background:#9999FF;position:absolute;left=12;top=38;width=80;
      visibility=hidden">
            <span>子菜单一</span><br>
             <span>子菜单二</span><br>
             <span>子菜单三</span><br>
       </div>
      <div id="menu2" onMouseMove="show('menu2')"
      onMouseOut="hide()" style="background:#9999FF;position:absolute;left=95;top=38;width=80;
      visibility=hidden">
             <span >子菜单一</span><br>
              <span>子菜单二</span><br>
              <span>子菜单三</span><br>
       </div>
      <div id="menu3"    onMouseMove="show('menu3')"
     onMouseOut="hide()" style="background:#9999FF;position:absolute;left=180;top=38;width=80;
     visibility=hidden">
             <span >子菜单一</span><br>
              <span>子菜单二</span><br>
              <span>子菜单三</span><br>
       </div>
  </body>
</html>
```

在这个程序中可以看出，所有的子菜单都是已经写好的 DIV，当鼠标移到指定区域的时候显示对应的 DIV，当鼠标移出的时候隐藏所有的 DIV，从而就有了下拉菜单的效果。这个程序在浏览器中的效果如图 3.17 所示。

图 3.17 下拉菜单示例程序运行效果

3.3.5 JavaScript+CSS 实现表格变色

在一些 Web 应用中间经常会用表格来展示数据，当表格行数比较多的时候，就容易后看错行的情况发生，所以需要一种方法来解决这个问题。

在这里我们采取这样一种措施，当鼠标移到某一行的时候，这行的背景颜色发生变化，这样当前行就会比较突出，不容易出错。具体的实现代码如下。

```
//--------文件名：ColorTable.html--------------------
<html>
    <head>
        <title>变色表格示例</title>
          <script language="javascript">
              function changeColor(row)
               {
                    document.getElementById(row).style.backgroundColor='#CCCCFF';
               }
                 function resetColor(row)
               {
                    document.getElementById(row).style.backgroundColor='';
               }
          </script>
    </head>
    <body>
    <table width="200" border="1" cellspacing="1" cellpadding="1">
     <tr>
        <th>学校</th>
        <th >专业</th>
        <th>人数</th>
     </tr>
     <tr align="center" id="row1" onMouseOver="changeColor('row1')" onMouseOut="resetColor('row1')">
        <td>北大</td>
        <td>法律</td>
        <td>2000</td>
     </tr>
    <tr align="center" id="row2" onMouseOver="changeColor('row2')" onMouseOut="resetColor('row2')">
       <td>清华</td>
       <td>计算机</td>
       <td>5000</td>
    </tr>
      <tr align="center" id="row3" onMouseOver="changeColor('row3')" onMouseOut="resetColor('row3')">
       <td>人大</td>
       <td>经济</td>
       <td>6000</td>
    </tr>
</table>
</body>
</html>
```

这个页面的在浏览器中的运行结果如图 3.18 所示。

图 3.18 变色表格运行效果

3.4 小结

HTML 是组织展示内容的标记语言，JavaScript 是客户端的脚本语言，CSS 是美化页面的样式表，这三种技术结合在一起构成了 Web 开发最基础的知识，所有的 Web 应用开发都是在这个基础之上进行的。

在本章的讲解中，仅仅对这三种技术的大体情况进行了介绍，使读者可以迅速对 Web 开发的基础知识有一个宏观的清楚的认识，从而可以快速进入后面章节的学习，如果读者对这方面基础知识有更深一步了解的需要，就有必要参考相关的专题书籍。