第 3 章　HTML、JavaScript 简介

网络给人们带来了一个缤纷绚丽的世界，那么上网浏览的网页是用什么语言编写的呢？这些页面主要是用 HTML 语言编写。HTML 是英文 Hypertext Marked Language 的缩写，即超文本标记语言，就是该类文档有别于纯文本的单个文件的浏览形式。超文本文档中提供的超级链接能够让浏览者在不同的页面之间跳转。

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。在接下来的章节中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。

本章的主要任务就是通过任务需求掌握最新的 HTML 技术如 JSON、jQuery 和 AJAX 等，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。

本章要点：

- 了解 HTML 的基本格式；
- 掌握 HTML 在文本及版面风格的控制；
- 熟悉图像、超链接和表格的使用；
- 掌握 HTML 表单的制作；
- 熟悉 HTML 5.0 技术；
- 掌握 JavaScript 使用和 JQuery 使用；
- 了解 JSON 概念，熟悉 AJAX 技术和 DWR 框架。

3.1　任务 1　引入性案例

本书不是详细介绍 HTML 的专著，因此本章主要通过实例任务完成对 HTML 相关技术的理解与掌握。

任务描述及任务目标：在 Windows 中，从头到尾做一个最简单的 HTML 的例子，完整地展示 HTML 文档的制作过程。

1．编写HTML文本

可以直接用文本编辑器如 Windows 中的记事本来编写 HTML 文件，当然通过一些开发工具如 Dreamweaver、FrontPage 编写也是可以的。在 Windows 中，寻找合适的文件夹，在空白处右击，在弹出的快捷菜单中选择“新建”命令，在之后弹出的子菜单中选择“文

本文档”命令，此时会在指定的文件夹下创建一个文本文件，打开该文件，在其中编写 HTML 文本，代码如下：

```
<!-- --------文件名：First.html-------------------- -->
<html>
      <head>
         <title>这是第一个 HTML 例子的标题</title>
      </head>
      <body>
         欢迎光临！这是我的第一个 HTML 文档。//这是文档主体，正文部分
       </body>
</html>
```

注意：

- 代码中“<、>、/”等字符都是英文半角字符。HTML 标签一般是成对出现的。
- 第一行中“<!-- -->”为 HTML 的注释符号。

2．保存

编辑之后，将文件重命名为 First.html 或 First.htm。

注意：扩展名一定是.html 或.htm，不能是.txt。

3．运行

保存完毕，在当前文件夹中就会有一个 First.html 文件，双击该文件，系统会自动用操作系统默认的浏览器打开，运行结果如图 3-1 所示。

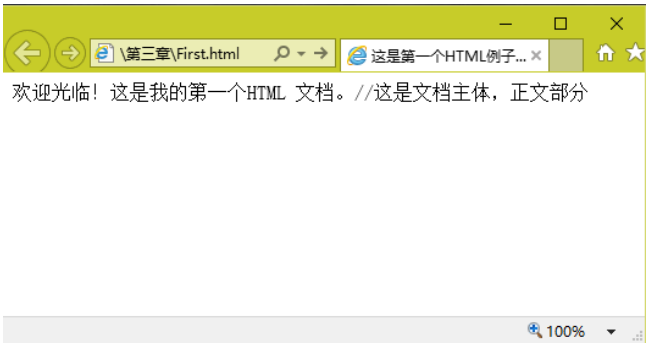

图 3-1　第一个 HTML 例子

该案例给出了一个简单的 HTML 页面，想让该页面丰富起来，需要更多的 HTML 知识，下面将分别介绍。

3.2　任务 2　HTML 的基本结构

任务描述：一个完整的 HTML 文件包括标题、段落、列表、表格以及各种嵌入对象，这些对象统称为 HTML 元素。在 HTML 中，使用标记或标签（tag）来分割并描述这些元

素。因此，HTML 文件就是由各种 HTML 标记和元素组成的。

任务目标：了解并掌握 HTML 的基本结构。

从任务 1 可以知道，一个 HTML 文件的基本结构如下：

```
<html>
       <head>
          头部信息
</head>
<body>
          文档主体部分
</body>
</html>
```

一个 HTML 文件分为两部分：头和体。标签<head>和</head>之间的是头部分，是关于整个页面的一些设置信息；标签<body>和</body>之间的是体部分，是要在浏览器中显示的页面内容。

在这个结构中引入了 3 对最重要的 HTML 标记，分别是：

（1）<html>和</html>是最外层的标记。任何 HTML 文件都应该以<html>开始，以</html>结束，表示这对标记间的内容是 HTML 文档。这对标记可以省略，因为.html 或.htm 文件被 Web 浏览器默认为是 HTML 文档，但最好写上。

（2）<head>和</head>之间是文档的头部信息，如文档标题等，若不需头部信息，可省略此标记。

（3）<body>和</body>之间是文档的正文内容。

3.3 任务 3 文本及版面风格的控制

任务描述及任务目标：处理文本、字号、外版面风格是制作精美网页所需要的基本功，为了使网页版面风格更加美观吸引人，则需进一步进行标签控制。该任务主要讲解如何在 HTML 中编排段落和修饰文字。

3.3.1 文本控制

1．字体控制

文本控制主要用于控制文字的字体大小、颜色，这些可以通过<font>标签实现。另外，还有一些设置字体某个特点的标签，如、、<u>、<sup>、<sub>、<strike>、<code>等，它们都要成对出现。

<font>标签的主要格式如下：

```
<font face = 字体类型 size = 字体大小 color = 颜色值 style = 样式>……</font>
```

其中，

（1）face：指定字体类型，如宋体、Times New Roman 等。但只有对方的计算机中可以设置相同的字体，才可以在其浏览器中出现预先设计的风格，所以最好指定常用字体。

（2）size：设置字号大小，有效值的范围为 1～7 的整数，默认值为 3。可以在 size 属性值之前加上+、—字符，来指定相对于当前字号值的增量或减量。

（3）color：指定字体颜色。颜色值既可以是十六进制数（最好用#作前缀），也可以是颜色名称。

（4）style：指定字体样式。

为了让文字富有变化，或者为了着重强调某一部分，HTML 还提供了一些标签，这些标签的格式和含义如表 3-1 所示。

表 3-1　常用标签的格式和含义

标 签 格 式	含　义
…	文字使用粗体形式显示
…	文字使用斜体形式显示
<u>…</u>	为当前文字添加下画线
<strike>…</strike>	为当前文字添加中画线
<big>…<big>	以比当前字号大一号的字体显示
…	以比当前字号小一号的字体显示
…	表示强调，一般为斜体
…	表示特别强调，一般为粗体
[…]	字体上标
_…	字体下标
<tt>…</tt>	打字机或等宽字体
<blink>…</blink>	闪烁效果，仅用于 Netscape Navigator
<cite>…</cite>	用于引证，举例，一般为斜体
<var>…</var>	变量字体，一般为斜体
<dfn>…</dfn>	定义字体，一般为斜体
<code>…</code>	代码字体，字体等宽

实例 3-1　阅读下面代码，了解<font>标签的使用方法。不同标签设置效果的运行结果如图 3-2 所示。

```
<html>
   <head>
      <title>HTML、JavaScript 简介—实例 3-1：font 标签的使用</title>
   </head>
   <body bgcolor = "#eeeeee">
      <center>
        <font>默认字体</font><br>
        <font size = "1">1 号字体</font><br>
        <font size = "2" face = "Arial" color = blue>2 号 Arial 蓝色字体
        </font><br>
        <font size = "3" face = "Times New Roman">3 号 Times New Roman
        字体</font><br>
       <font size = "4" face = "楷体_gb2312">4 号楷体（未必能显示）</font><br>
          <font size = "5" face = "Comic Sans MS, 仿宋_gb2312">5 号仿宋字体
          （未必能显示），英文字体是 Comic Sans MS</font><br>
          <font size = "6" style = "color:green">6 号绿色字体</font><br>
          <font size = "7" style = "color:red">7 号红色字体</font>
          <font size = 3>
```

```
            <p><b>黑体字</b><i>斜体字</i>
            <u>加下画线</u><strike>加中画线</strike></p>
             <p><big>大一号字体</big>原字体<small>小一号字体</small></p>
            <p><em>你好，</em><strong>欢迎学习 HTML!</strong></p>
            <p><cite>Welcome!</cite></p>
            <p>a<sub>1</sub> = x<sup>2</sup>+ y<sup>2</sup></p>
            </font>
        </center>
    </body>
</html>
```

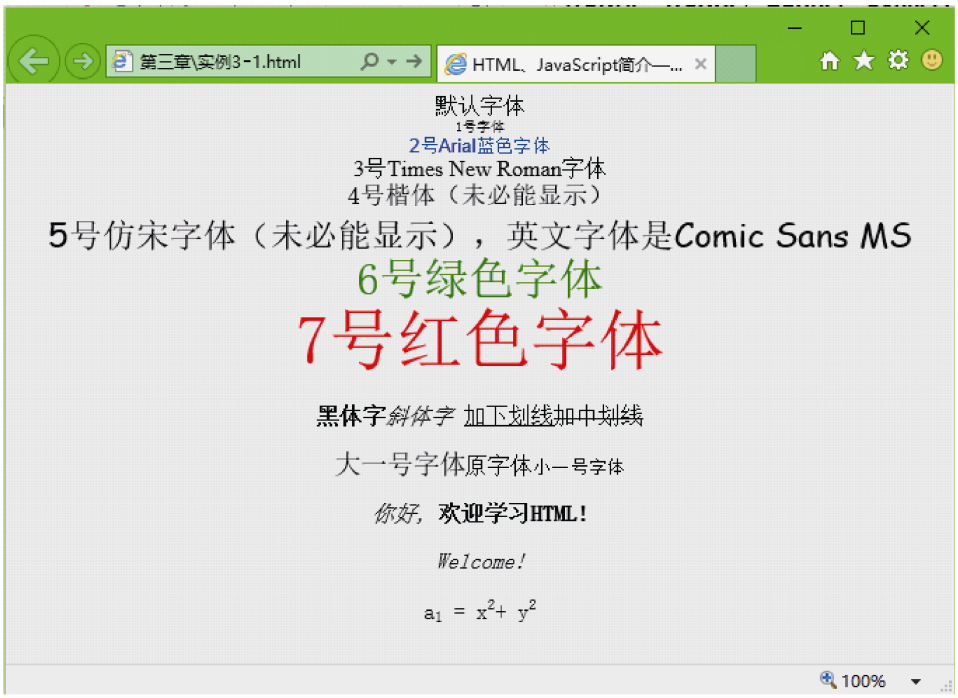

图 3-2　各种大小类型字体

2．标题

一般文章都有标题、副标题、章和节等结构，HTML 中也提供了相应的标题标记<hn>和</hn>，其中 n 为标题的等级。HTML 一共提供 6 个等级的标题，分别从<h1>到<h6>。n 越小，标题字号就越大，主要格式为：

```
<hn align = 对齐方式>……</hn>
```

其中，对齐方式有 left、right、center 3 种，即左对齐、右对齐和居中，默认为 left。

实例 3-2　各级标题<hn>的使用，代码如下，运行结果如图 3-3 所示。

```
<html>
<head>
<title>HTML、JavaScript 简介——实例 3-2：hn 标签的使用</title>
</head>
<body>
    <h1>一级标题</h1>
    <h2>二级标题</h2>
```

```
    <h3 align = center>三级标题</h3>
    <h4 align = left>四级标题</h4>
    <h5 align = right>五级标题</h5>
    <h6>六级标题</h6>
</body>
</html>
```

图 3-3　各级标题

从结果可以看出，每一个标题的字体均为黑体，内容文字前后都插入空行。

3.3.2　版面风格控制

1．网页整体风格控制

对页面整体风格的控制主要通过<body>标签的相关属性实现，常用的属性主要有以下几种。

- bgcolor 属性：用于设置网页的背景颜色。
- text 属性：设置网页文本的背景颜色。
- background 属性：用于设置网页的背景图形，图形以平铺方式作为网页背景。
- bgproperties 属性：该属性只有一个取值，即 FIXED，用于锁定网页的背景图形。锁定后再滚动页面时，背景图片不会跟着滚动。其语法如下：

```
<body background="背景图"  bgproperties="FIXED">
```

- Link、VLink 和 ALink 属性：该组属性用于设置超链接在不同状态下的颜色。Link 设置链接的颜色，VLink 设置已经访问过的链接的颜色，ALink 设置活动链接的颜色。

2．分段和换行控制

（1）分段标记（<p>）

段落分段使用<P>标记实现。 表示一段落的开始； 表示段落的结束，通常可以省略不写。单独的一个 标记可以产生一个空行。

（2）换行（
）和禁止换行标记（<nobr>文本</nobr>）

换行用标记
来实现，没有对应的结束标记，它指示浏览器在标记处执行一个换行动作。
标记后的文本将显示换行，换行后的文本，与前面的文本仍属于一个段落，因此换行后字符和段落格式不会改变，这与 标记不同。

为了保证某一单词的完整性，有时候需要禁止某处换行，此时使用<nobr>标记实现。其用法为<nobr>文本</nobr>，该标记作用的文本将在同一行显示。

3．预格式化文本（<pre>文本块</pre>）

预格式化文本是文本块以等宽字体显示，在网页中显示时可保留预排版的格式。预格式化文本使用<pre>标记来实现，使用方法是<pre>文本块</pre>。

4．画分隔线（ <hr> ）

在网页中产生的分隔线使用<hr>标记来实现，其用法为：

```
<hr size = 宽度 width = 长度 align = 对齐方式 color = 颜色 noshade>
```

各个属性说明如下。

- size：用于设置水平线的宽度，以像素为单位，默认值为 1。
- width：用于表示水平线的长度，可以以像素为单位，如 100；也可以是浏览器窗口宽度的百分比，如 80%，默认值为 100%。
- align：水平线的对齐方式，有 left、right、center 3 种，默认为 center。
- color：指定线条颜色。颜色值既可以是一个十六进制数（最好用#作前缀），也可以是颜色名称，默认为灰色。
- noshade：线段无阴影属性，为实心线段，默认为空心线段。

<hr>是一个单标记，也就是说没有对应的</hr>。

3.4 任务 4 图像、超链接和表格

任务目标及任务描述：一个优秀的网页往往是绚丽缤纷的，图像自然必不可少。想让多个网页相互连接在一起，实现网页和网页的互动，超链接就起到了举足轻重的作用。信息在表格中显示，将带来清晰、明确、简单、大方的效果，表格的作用就得到了体现。该任务的职责就是要解决图像、超链接以及表格方面的问题。

3.4.1 图像和超链接

1．图像标记

如果把图像嵌入到网页中，网页将变得五光十色。能实现这一需求的是<img>标记。<img>标记的主要语法为：

```
<img src = 图像文件的地址 alt = 文字 border = 边框宽度 width = 图像宽度 height =
图像高度 align=对齐方式>
```

各个属性说明如下。

- src：用来设置图像文件所在的路径。可以是相对路径，也可以是绝对路径。
- alt：当鼠标放在图片上时，显示小段提示文字，一般是此图片的标题或主要内容。当图像文件无法在网页中显示时，在图像的位置也会显示 alt 所设置的文字。
- border：图像边框的宽度，单位是像素。在默认情况下图像无边框，即 border=0。
- width 和 height：图像的宽度和高度，单位是像素。在默认情况下，如果改变其中一个值，另一个值也会等比例进行调整，除非同时禁止两个属性。
- align 属性：指定图像和周围文字的对齐方式。图像的绝对对齐方式与相对文字的对齐方式不同，绝对对齐方式包括 left、center 和 right 3 种，而相对文字对齐方式则是指图像与一行文字的相对位置。align 的取值情况如表 3-2 所示。

表 3-2　align取值

值	描　述	值	描　述
left	将图像对齐到左边	top	将图像与顶部对齐
right	将图像对齐到右边	bottom	将图像与底部对齐
middle	将图像与中央对齐		

注意：

- 不赞成使用 align，但是几乎所有的浏览器都支持该属性。
- 如果不想使用 align 和 时该怎么办呢？可以使用 HTML/XHTML 表格来对齐内容。不过更好的方案是使用样式。例如，margin-left 属性可以对图像进行缩进，而 float 属性 可以实现文本包围图像的效果。

2．超链接

所谓超链接（hyperlink），就是当单击某个字或图片时，就可以打开另一个网页或画面。它的作用对网页来说极其重要，可以说是互联网的灵魂，是 HTML 最强大和最有价值的功能。超链接简称链接（link）。

（1）定义超链接

超链接的语法根据其链接对象的不同而有所变化，但都是基于<a>标记的，主要语法为：

```
<a href = 本机上绝对或相对路径的文件名 target = 目标>……</a>
或者：
<a href = Internet 上的带 URL 的文件名 target = 目标>……</a>
```

其中，href 是 hypertext reference（超文本引用）的缩写。target 用于指定如何打开链接的网页，有以下几个值。

- _blank：打开一个新的浏览器窗口显示。
- _self：用网页所在的浏览器窗口显示，是默认设置。

❑ _parent：在上一级窗口打开，常用在框架页面中。

❑ _top：在浏览器的整个窗口打开，将会忽略所有的框架结构。

在<a>和之间，是超链接要显示的文字或图片。当用户把鼠标光标放在这些文字或图片上时，光标会变成手的形状，此时单击文字或图片，超链接就会发生作用了。

实例 3-3 下面为一个文字、图形的超链接应用，运行结果如图 3-4 所示。

```
<html>
  <head>
  <title>HTML、JavaScript 简介——实例 3-3：超链接演示</title>
  </head>
  <body>
    <p><a href="a.html">一个简单超链接页面</a></p>
    <p><a href="a.html"><img src="3.3.jpg"> </a></p>
    <p><a href = "http://www.sina.com.cn" target = "_blank">用新窗口打开新浪
    </a></p>
    <p> <a href = "http://www.sohu.com">用本窗口打开搜狐</a> </p>
  </body>
</html>
```

图 3-4 超链接示例

（2）定义锚点

当创建的网页内容多、页面很长时，还可以在网页内跳转，此时就要定义“锚（anchor）”。有的书上称为“书签”，但作者还是觉得“锚”更形象，因为当船只靠岸后，一般需要扔进水里一个巨大的铁钩，以便钩住水底的石头或沉于淤泥中，这样船只就不容易飘走了，这就是锚。

同样，在一个篇幅巨大的网页中，也可以定义一些锚，以便快速定位。定义锚的主要语法为：

```
<a name = 锚名>文字</a>
```

其中，属性 name 是不可缺少的，它的值也就是锚名，文字并非是必须的。

定义了锚之后，就可以用下面的语法格式进行跳转了：

```
<a href = "#锚名">链接的文字</a>
```

3.4.2　表格

绘制表格是 HTML 一项非常重要的功能。表格是网页排版的灵魂。同时由于表格包含的功能比较多，读者需要仔细体会才能掌握。

1．表格的基本语法格式

表格由列和行组成。在 HTML 中，表格用<table>、</table>表示。一个表格可以分为很多行（row），各行用<tr>、</tr>表示；每行又可以分为很多单元格（cell）或列（column），用<td>、</td>表示。它们是创建表格最常用的标记，需要统一的使用方法，语法为：

```
<table>
   <tr>
   <td>单元格内的文字</td>
   <td>单元格内的文字</td>
   …
   <td>单元格内的文字</td>
   </tr>
   …
</table>
```

2．<table>标记

<table>标记中的属性很多，用于控制表格的整体显示。常用语法为：

```
<table align= bgcolor= border= bordercolor= height= width= cellpadding=n
cellspacing=n nowrap>
…
</table>
```

各个属性说明如下。

- align：表格在上一层容器控件中的对齐方式，有 center、left、right 共 3 个值，其中 left 是默认对齐方式。
- bgcolor：设置表格的背景色，默认是上级容器的背景色。
- border：设置表格线的宽度，单位是像素，默认值是 1。
- bordercolor：设置表格线的颜色。如果没包含 border 属性，或者 border 属性值是 0，则忽略此属性值。
- height：设置表格的高度，以像素或页面高度的百分比为单位。但如果表格内容大于设置的高度，则表格会自动扩张，以便容纳所要显示的内容。
- width：表格的宽度，以像素或页面宽度的百分比为单位。但如果表格内容大于设置的宽度，则表格会自动扩张，以便容纳所要显示的内容。

- cellpadding：设置单元格内部所显示的内容与表格线之间的距离，单位是像素。
- cellspacing：设置表格线的“厚度”，单位是像素或百分比。这些属性是<table>标记最常用、最基本的属性。

<table>标记中一些不常用的标记如下。

- background：设置定义表格的背景图案。一般选浅颜色的图案。该属性不要与 bgcolor 同用。
- bordercolorlight：设置表格边框向光部分的颜色，只适用于 IE。如果忽略 border 属性或 border 属性值为 0，则此属性不起作用。
- bordercolordark：设置表格边框向光部分的颜色，只适用于 IE。如果忽略 border 属性或 border 属性值为 0，则该属性不起作用。使用 bordercolorlight 或 bordercolordark 时，bordercolor 将会失效。

3．<tr>标记

用<table>标记可以设置整个表格的属性，但如果要设置表格各行的属性，就需要详细了解<tr>标记的各个属性。<tr>标记的常用语法为：

```
<td align = 水平对齐方式 valign = 垂直对齐方式 height = 行高 background = 背景图案 bgcolor = 背景色 bordercolor = 边界颜色 bordercolordark = 边界背光部分的颜色 bordercolorlight =边界向光部分的颜色 colspan = 跨列数 rowspan = 跨行数 nowrap>
…
</td>
```

可以看出，<tr>的很多属性和<table>的相应属性是一样的。所不同的是<table>的各个属性用于设置整个表格的显示情况，而<tr>属性只用于设置相应行的显示情况。当<tr>属性值的设置和<table>的同名属性值不同时，以<tr>属性值为准。也就是说，低层的属性设置会“屏蔽”高层属性设置。

<tr>标记一些和<table>不同的属性的意义如下。

- align：文本在单元格中的水平对齐方式，有 center、left、justify、right 共 4 个值，其中 left 是默认对齐方式，justify 是指合理调整单元格中的内容，以恰当显示。
- valign：文本在单元格中的垂直对齐方式，有 baseline、top、middle、bottom 共 4 个值，默认值是 middle，即垂直居中对齐。baseline 是指单元格中的内容以基线（baseline）为准垂直对齐，类似于 bottom（底端对齐）。

和<table>中的相应属性类似，bordercolorlight 和 bordercolordark 属性是 IE 浏览器的独有属性，并且它们会屏蔽 bordercolor 属性，但在其他浏览器中，bordercolorlight 和 bordercolordark 属性不起作用。

4．<td>标记

<td>标记设置的是一行内某一单元格的各项属性，也遵从低层的属性设置会“屏蔽”高层属性设置的原则，具体属性与<tr>相似，不再赘述。

实例 3-4　使用 HTML 语言绘制的表格如图 3-5 所示。

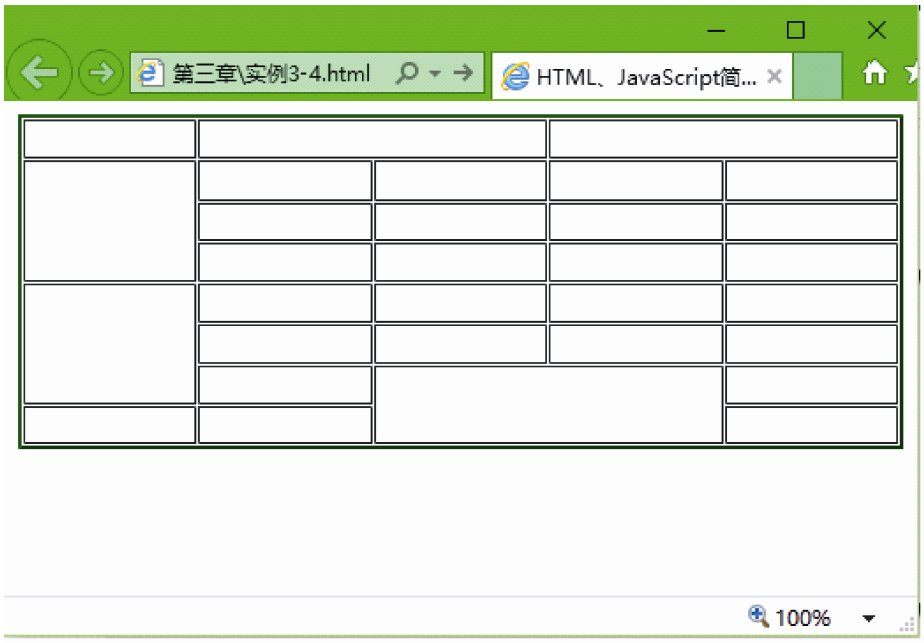

图 3-5　表格示例

```
<html>
<head>
       <title>HTML、JavaScript 简介—实例 3-4：绘制表格</title>
    </head>
    <body bgcolor = "#ffffff">
       <table border="2" width="100%" bordercolor="#008000" cellspacing="1">
          <tr> <!--第 1 行-->
             <td>  </td>
             <td colspan="2">  </td>   <!--第 2、3 个单元格合并-->
             <td colspan="2">  </td>   <!--第 4、5 个单元格合并-->
          </tr>
          <tr> <!--第 2 行-->
             <td rowspan="3">  </td>  <!--和下面两行的相应单元格合并-->
             <td>  </td>
             <td>  </td>
             <td>  </td>
             <td>  </td>             <!--第 2、3、4、5 个单元格-->
          </tr>
             <tr> <!--第 3 行-->
             <td>  </td>  <!--只有 4 个单元格,因为第 1 个已经和第 2 行第 1 个合并-->
             <td>  </td>
             <td>  </td>
             <td>  </td>
          </tr>
          <tr> <!--第 4 行-->
             <td>  </td>  <!--只有 4 个单元格,因为第 1 个已经和第 2 行第 1 个合并-->
             <td>  </td>
             <td>  </td>
             <td>  </td>
          </tr>
          <tr> <!--第 5 行-->
             <td rowspan="3">  </td>  <!--和下面两行的相应单元格合并-->
```

```
            <td>  </td>
            <td>  </td>
            <td>  </td>
            <td>  </td>
        </tr>
        <tr> <!--第 6 行-->
            <td>  </td>  <!--只有 4 个单元格,因为第 1 个已经和第 5 行第 1 个合并-->
            <td>  </td>
            <td>  </td>
            <td>  </td>
        </tr>
        <tr> <!--第 7 行-->
            <td>  </td>  <!--本应有 4 个单元格,因为第 1 个已经和第 5 行第 1 个合并-->
            <td colspan="2" rowspan="2">  </td>
            <!--但现在只有 3 个，因为两行两列合并成一个单元格-->
            <td>  </td>
        </tr>
        <tr> <!--第 8 行-->
            <td>  </td>  <!--第 1 个-->
            <td>  </td>  <!--第 2 个-->
            <td>  </td>  <!--第 5 个，因为第 3、4 个已经和上一行 3、4 个合并-->
        </tr>
    </table>
  </body>
</html>
```

3.5 任务 5 HTML 的表单

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。在 Java EE 的开发中表单非常重要，其是客户端向服务器发送请求数据的一个主要途径。表单的标签是<form>…</form>，中间可以加入文本框、单选按钮、复选框等表单元素。

3.5.1 表单定义

表单在网页中负责数据采集功能，表单由 3 个部分组成：表单标签、表单域和表单按钮。表单是为了处理动态数据交互，将不同数据在不同页面间传递。

表单标签包含了处理表单数据所使用 CGI 程序的 URL 以及数据提交服务器的方法。

form 标记是众多表单标记中的重要一种，其是表单窗口的开始，功能是设置表单的基础数据。语法格式如下：

```
<form action="" method="post/get" enctype="application/
x-www-form-urlencoded" name="form1" target="_parent"></form>
```

其中，

- name：表单名字。
- method：数据的传送方式，有 get 和 post 两种方法（post 表示从发送表单内直接传输数据；get 表示将发送数据附加到 URL 的尾部，将新的 URL 送至服务器）。
- enctype：传输数据的 MMIE 类型。传输数据的 MMIE 有两种，一种是默认方式，

即 application/x-www-form-urlencoded；另一种是 multipart/form-data，是上传文件或图片的专用方式。

- target：是处理文件的打开方式。

实例 3-5　一个登录身份验证的表单实例，代码如下：

```
<form method="post" action="login.jsp">
   请输入用户名：<input type="text" name="username" size="20" />
     <br />
   请输入密码：<input type="password" name="password" size="20" />
     <br />
           <input type="submit" value="登录" />
 </form>
```

如图 3-6 所示为实例 3-5 表单应用的实际效果。

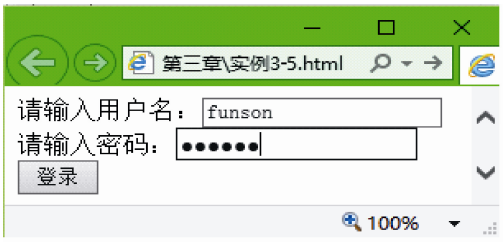

图 3-6　实例 3-5 表单的运行效果

注意：

- 这是程序中通常用来身份登录验证的表单代码。代码从<form>开始，到</form>结束。
- 中间包括了 3 个<input>控件，一个是文本框，用于输入用户名，另一个是密码框，用于输入密码，还有一个是提交按钮，用于将表单中用户输入的数据提交到网站服务器端。
- 该实例中 action 后的 login.jsp 文件只是表示跳转目的，需第 4 章的 JSP 技术实现。

3.5.2　文本框控件

文本框控件是在网页上显示可输入文本的方框，并在提交处理时，将其内容读出，并交由网页进行处理。文本框控件有 3 个，分别为单行文本框，密码框以及多行文本框。语法如下：

（1）单行文本框

```
<input type = "text" name = 控件名称 size = 控件长度 maxlength = 最长输入字符数 value = 初始值>
```

（2）密码框

```
<input type = "password" name = 控件名称 size = 控件长度 maxlength = 最长输入字符数 value = 初始值>
```

（3）多行文本框

```
<textarea name = 控件名称 value = 初始值 rows = 行数 cols = 列数></textarea>
```

其中，

- type：标记文本框类型，有 10 个属性，该类型必须标明。
- name：为文本框命名，命名最好将该文本框的功能表示出来，以后亦可以用该属性调用文本框中内容。
- size：用于指定输入字段宽度，可以是文字，也可以是像素。
- maxlength：用于指定输入字段可输入文字个数（单行文本框和密码框默认没有字数限制）。
- value：用于指定输入字段的数据值（当 type 为 checkbox 和 radio 时，不可省略该属性，其他可省略）。
- rows：指定多行文本框显示行数。
- cols：指定多行文本框显示列数。
- warp：用于控制多行文本框是否自动换行（若未设置该属性则默认自动换行）。

上述 type 的 10 个属性如下。

- text：文字输入域（输入型）。
- password：也是文字输入域，但是输入的文字以密码符号'*'显示（输入型）。
- file：可以输入一个文件路径（输入型）。
- checkbox：复选框，可以选择零个或多个（选择型）。
- radio：单选按钮，只可以选择一个而且必须选择一个（选择型）。
- hidden：代表隐藏域，可以传送一些隐藏的信息至服务器。
- button：按钮（单击型）。
- image：使用图片来显示按钮，使用 src 属性指定图像的位置（就像 img 标签的 src 属性，单击型）。
- submit：提交按钮，表单填写完毕可以提交，把信息传送至服务器。可以使用 value 属性来显示按钮上的文字（单击型）。
- reset：重置按钮，可以把表单中的信息清空（单击型）。

3.5.3 单选按钮和复选框

单选按钮和复选框是给用户提供已有选择条件，在这些选择条件中进行选择，无须像文本框一样输入文字。单选按钮只能在一组选项中选择一个，而复选框可以选择多个。

语法如下：

（1）单选按钮

```
<input type = "radio" name = 控件名称 value = 单选按钮的取值 checked>
```

（2）复选框

```
<input type = "checkbox" name = 控件名称 value = 复选框的值 checked>
```

其中，type、name、value 前面已介绍，此处意义相同。checked 表示在单选按钮和复选框中设置的默认选择项，可以省略。

3.5.4　下拉菜单和列表

<select>标记可以在页面中创建下拉列表框，此时列表框为空框，要使用<option>标记向列表中添加内容。

下拉菜单语法如下：

```
<select name = 控件名称 size = 显示的项数 multiple>
   <option value = 选项值 1 selected>显示内容 1</option>
   <option value = 选项值 2>显示内容 2</option>
   …
   <option value = 选项值 n>显示内容 n</option>
</select>
```

其中，multiple 表示用于多行列表框支持多选。

3.5.5　按钮

按钮是为了进行页面之间的处理，如提交按钮，其会将该页面的数据提交到指定页面进行处理。语法如下：

（1）普通按钮

```
<input type = "button" name = 控件名称 value = 按钮值 onclick = 处理程序>
```

（2）提交按钮

```
<input type = "submit" name = 控件名称 value = 按钮值>
```

（3）重置按钮

```
<input type = "reset" name = 控件名称 value = 按钮值>
```

其中，onclick 表示单击这个按钮之后运行的处理程序。

实例 3-6　一个综合应用上述表单元素的实例来了解它们的使用，实例代码如下：

```
<form>
      <p>姓名：<input type="text" name="name" size="10">
      <p>密码：<input type="password" name="pass" size="10">
      <p>性别：<input type="radio" name="gender" value="m" checked>男
         <input type="radio" name="gender" value="f">女
      <p>爱好：<input type="checkbox" name="hobby" value="literature">文学
         <input type="checkbox" name="hobby" value="music">音乐
         <input type="checkbox" name="hobby" value="sport">运动
      <p>班级：<select name="class">
            <option value="1">计科 1 班
            <option value="2">计科 2 班
            <option value="3">软工 1 班
            <option value="4">软工 2 班
            <option value="5">网工 1 班
            <option value="6">网工 2 班
         </select>
```

```
    <p>自我介绍:
    <p><textarea name="introduce" rows="5" cols="20"></textarea>
    <p><input type="submit" value="确定">
       <input type="reset" value="重填">
</form>
```

实例 3-6 在浏览器中运行结果如图 3-7 所示。

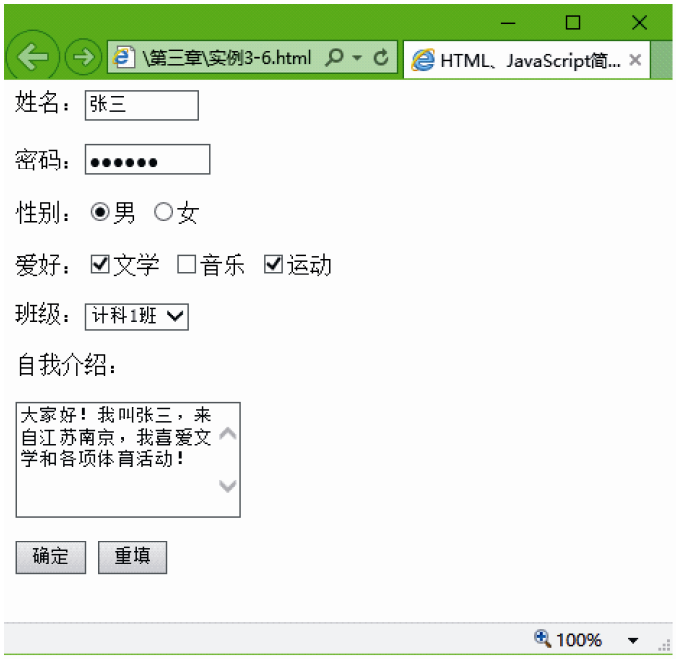

图 3-7　表单综合应用实例

提示：

- 表单元素必须掌握。
- 在后续 JSP 开发过程中，表单是最重要的人机交互的实现方式，因此必须熟练掌握以上各表单元素的使用。

3.5.6　图像域、隐藏域和文件域

图像域是创建一个图像控件，该控件单击后将导致表单立即提交（最好用相对路径，直接在 WebRoot 文件夹下创建文件名 image，然后用相对路径 image\..来引用，用绝对路径需要 Tomcat 的绝对路径，而非本机绝对路径）。

隐藏域是为了显示数据，但不给浏览用户看到，该数据值提供给设计页面的程序员控制页面使用。

文件域由文本框和浏览器组成，用户既可以在文本框中直接输入文件路径和文件名，也可以通过单击浏览按钮从磁盘上查找和选择所需文件。语法如下：

（1）图像域

```
<input type = "image" src = 图像文件地址 name = 控件名称>
```

（2）隐藏域

```
<input type = "hidden" name = 控件名称 value = 提交的值>
```

（3）文件域

```
<input type = "file" name = 控件名称>
```

其中，src 表示图像文件位置，可以是相对位置，也可以是绝对位置。

3.5.7　分组标记

如果一个页面表单项太多，可以用 fieldset 和 legend 标签对表单内容分组。当一组表单元素放到<fieldset>标签内时，浏览器会以特殊方式显示它们，有特殊边界、3D 效果，或者可以创建一个子表单来处理这些元素。语法如下：

```
<fieldset>
  <legend>
    …
  </lengend>
</fieldset>
```

实例 3-7　分组表单，代码如下：

```
<form>
  <fieldset>
    <legend>个人健康信息</legend>
    年龄：<input type="text" />
    身高：<input type="text" />
    体重：<input type="text" />
    血压：<input type="text" />
  </fieldset>
</form>
```

表单分组演示运行效果如图 3-8 所示。

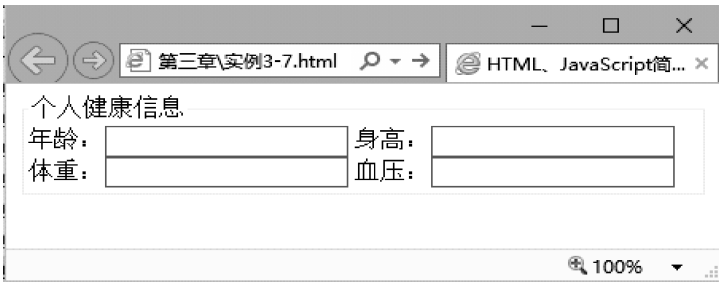

图 3-8　表单分组演示

3.6　任务 6　CSS 使用

层叠样式表 CSS（Cascading Style Sheets）是一种用来表现 HTML 或 XML 等文件样式的计算机网页格式化语言。CSS 目前的最新版本为 CSS 3，是能够真正做到网页表现与内

容分离的一种样式设计语言。

样式通常保存在外部的.css 文件中。通过仅编辑一个简单的 CSS 文档，就可以同时改变站点中所有页面的布局和外观。由于 CSS 允许同时控制多重页面的样式和布局，因此可以称得上 Web 设计领域的一个突破。作为网站开发者，只需要为每个 HTML 元素定义样式，并将之应用于希望的任意多的页面中即可。如需进行全局的更新，只需简单地改变样式，然后网站中的所有元素均会自动更新。

CSS 的使用方式有以下 3 种。

1．行间样式表

行间样式表是指将 CSS 样式编码写在 HTML 标签中，格式代码如下：

```
<h1 style="font-size:12px;color:#000FFF">
   我的 CSS 样式。
</h1>
```

行间样式表由 HTML 元素的 HTML 元素的 style 支持，只需将 CSS 代码用分号隔开写在 style=""之中。这是最基本的形式，但是其没有实现表现与内容分离且不能灵活地控制多个页面，所以我们只是在调试 CSS 代码的时候使用。

2．内部样式表

内部样式表与行间样式表相似，都是把 CSS 代码写在 HTML 页面中，稍微不同的是前者可以将样式表放在一个固定的位置，格式代码如下：

```
<html>
<head>
<title>内部样式表</title>
  <style type="text/css">
     h1{font-size:12px;
       color:#000FFF
       }
  </style>
</head>
<body>
  <h1>我的 CSS 样式。</h1>
</body>
</html>
```

内部样式表编码是初级的应用形式，不能达到跨页面使用所以不适合使用。

3．外部样式表

外部样式表是 CSS 应用中最好的一种形式，其将 CSS 样式代码单独放在一个外部文件中，再由网页进行调用。多个网页可以调用一个样式文件表，这样能够实现代码的最大限度的重用及网站文件的最优化配置，

实例 3-8　采用方法 3 的格式代码如下：

```
<html>
<head>
<title>实例 3-8：外部样式表演示</title>
  <link rel="stylesheet" rev="stylesheet" href="style.css">
```

```
</head>
<body>
  <h1>我的 CSS 样式。</h1>
</body>
</html>
```

另外，在实例 3-8.html 的相同目录下创建一个名为 style.css 的文件，其代码如下：

```
h1{font-size:12px;
    color:#000FFF
    }
```

如上所示，在<head>中使用了<link>标签来调用外部样式表文件。将 link 指定为 stylesheet 方式，并使用了 href="style.css"指明样式表文件的路径，便可将该页面应用到在 style.css 中定义的样式。

实例 3-8 的运行效果如图 3-9 所示。

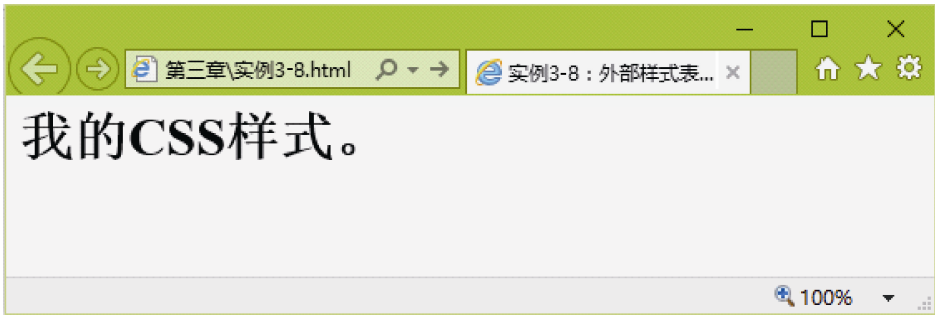

图 3-9　外部样式表使用演示

3.7　任务 7　HTML 5 应用

HTML 5 是 HTML 最新的修订版本，2014 年 10 月由万维网联盟（W3C）完成标准制定。HTML 5 的设计目的是为了在移动设备上支持多媒体。HTML 5 简单易学，是下一代 HTML 标准。HTML 4.01 的上一个版本诞生于 1999 年，从那以后，Web 世界已经经历了巨变。目前 HTML 5 仍处于完善之中，然而大部分现代浏览器已经具备了某些 HTML 5 功能的支持。HTML 5 中的一些有趣的新特性如下。

- 用于绘画的 canvas 元素。
- 用于媒介回放的 video 和 audio 元素。
- 对本地离线存储的更好的支持。
- 新的特殊内容元素，如 article、footer、header、nav、section。
- 新的表单控件，如 calendar、date、time、email、url、search。

编写代码时其<!DOCTYPE> 声明必须位于 HTML 5 文档中的第一行，使用非常简单。一个简单的 HTML 5 文档代码格式如下：

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
```

```
  <title>文档标题</title>
</head>

<body>
  主体内容…
</body>

</html>
```

实例 3-9　下面是一个通过 min、max 和 step 属性用于为包含数字或日期的 input 类型规定限定（约束）的例子，实例 3-9 源代码如下：

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>实例 3-9：日期输入验证</title>
</head>
<body>
  <form action="demo-form.jsp">
    输入 1980-01-01 之前的日期:
    <input type="date" name="bday" max="1979-12-31"><br>
     输入 2000-01-01 之后的日期:
    <input type="date" name="bday" min="2000-01-02"><br>
      数量 (在 1 和 5 之间):
    <input type="number" name="quantity" min="1" max="5"><br>
    <input type="submit">
</form>

<p><strong>注意:</strong> Internet Explorer 9 及更早 IE 版本，Firefox 不支持 input 标签的 max 和 min 属性。</p>
<p><strong>注意:</strong>
  在 Internet Explorer 10 中 max 和 min 属性不支持输入日期和时间，IE 10 不支持这些输入类型。</p>
</body>
</html>
```

实例 3-9 的运行效果如图 3-10 所示。

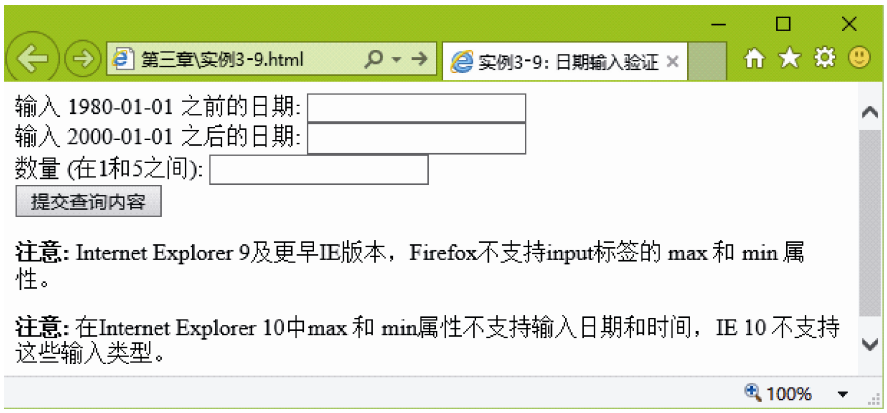

图 3-10　HTML 5 日期输入验证

当在表单中输入不符合条件的日期和数值时就会提示输入出错信息，要求重新输入。

注意：对于中文网页需要使用 <meta charset="utf-8"> 声明编码，否则会出现乱码。

3.8　任务 8　JavaScript 使用

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，其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与用户的一些互动。JavaScript 是通过嵌入或调入在标准 HTML 语言中实现的，弥补了 HTML 语言的不足，是 Java 与 HTML 的折中选择方案。下面简单介绍 Java Script 技术的基础知识。

3.8.1　JavaScript 概述

JavaScript 是一种基于对象（Object）和事件驱动（Event Driven）并具有安全性能的脚本语言。嵌入在 HTML 语言中实现。能够在客户端执行，具有解释性、基于对象、事件驱动、安全性和跨平台等特点。JavaScript 无须编译，可直接嵌入 HTTP 页面中，把静态页面转变成支持用户交互并响应应用事件的动态页面，经常用于数据验证、控制浏览器以及生成时钟，日历和时间戳文件。

JavaScript 的标签是 <script language="javascript">…</script>，一般放在页面的 <head>…</head>之间。

注意：

- Java 和 JavaScript 虽然名字相似，但两者没有任何关系。
- 两门语言是由两家不同公司开发的，Java 是 Sun 公司（现由 Oracle 公司收购），而 JavaScript 是 Netscape 公司，最初名字为 LiveScript; Java 是面向对象语言 OOP，所有对象和类都要求用户自己定义，JavaScript 就基于对象的语言，所有对象都是由浏览器提供给用户的，直接调用即可。
- 之所以将 LiveScript 更名为 JavaScript，主要也是借助 Java 的名声，便于推广使用。

3.8.2　函数

在 JavaScript 开发中最重要的部分就是函数，也是在代码中最常使用的一种形式。

1．变量

在 JavaScript 中，可以用关键字 var 声明变量，其语法格式如下：

```
var variable;
```

在 JavaScript 里，变量都是用 var 来声明，不区分数据类型，为这个变量赋值什么类型的数据，就是什么类型，如 int、double 等。可以用一个 var 声明若干个变量，并赋值，例如：

```
var i= 5, j="你好！", k=true;
```

2．函数定义

在 JavaScript 中运算符和流程控制语句和 Java 一致，在这里就不多介绍了。

函数由关键字 function、函数名加一组参数以及置于大括号中需要执行的一段代码定义的。定义函数的基本语法如下：

```
function functionname([parameter1,parameter2,……]){
  Statements;
  [return expression;]
}
```

参数说明：

- functionname：必选项，用于指定函数名。
- parameter：可选项，用于指定参数。
- Statements：必选项，是函数体，用于实现函数功能的语句。
- expression：可选项，用于返回函数值。

从函数定义格式中可以发现，在 JavaScript 中定义的函数不需要声明返回值，而如果一个函数需要有返回值的话，则直接通过 return 语句返回即可。

3．函数调用

如果函数调用不带参数的函数，使用函数名加上括号即可；如果要调用的函数带参数，则在括号内加上需要传递的参数；如果包含多个参数，各个参数间用逗号隔开。

实例 3-10　验证表单输入字符串是否为汉字，JavaScript 代码如下：

```
<script language="javascript">
function check(){
  var str=form1.Name.value;
  if(str==""){
    alert("请输入姓名！");
    form.Name.focus();
    return;
        }
  else{
   var obj=/[\u4E00-\u9FA5]{2,}/;
   if(obj.test(str)==true){
      alert("你输入正确的姓名！");
                    }
  else{
      alert("你输入姓名不正确！");
      }
   }
 }
</script>
<html>
<head>
   <title>实例 3-10：JavaScript 输入验证</title>
</head>
<body>
<div>
  <form  name="form1" method="post" action="" >
名字：<input type="text" name="Name" id="name" size="40" /><br>
```

```
    <input type="button" name="Button" onclick="check()" value="检测"/>
  </form>
</div>
</body>
</html>
```

当输入错误的名字（非汉字）与汉字时，运行结果如图 3-11 所示。

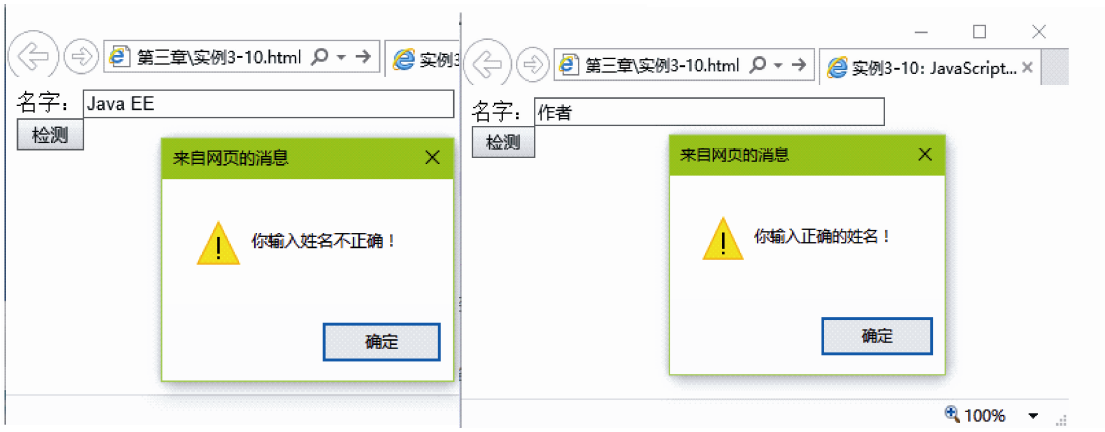

图 3-11　运行结果

3.8.3　事件处理

在 JavaScript 中，除了调用 JavaScript 函数外，还可以通过触发事件执行 JavaScript 语句。事件可以使 JavaScript 的程序变得更灵活，使页面具备更好的交互效果。在 JavaScript 的事件处理过程中主要也是围绕函数展开的，一旦发生事件后，则会根据事件的类型来调用相应的函数，以完成事件的处理。

1．JavaScript常用事件

JavaScript 常用事件如表 3-3 所示。

表 3-3　JavaScript常用事件

事　件	何 时 触 发
onabort	对象载入被中断时触发
onblur	元素或窗口本身失去焦点时触发
onchange	改变<select>元素中选项或其他表单元素失去焦点，并且在其获得焦点后内容发生改变时触发
onclick	单击鼠标左键时触发
ondbclick	双击鼠标左键时触发
onerror	出现错误时触发
onfocus	任何元素或窗口获得焦点时触发
onkeydown	键盘上按键被按下时触发，当返回 false 时，取消默认动作
onkeypress	键盘被按下时触发，并产生一个字符
onkeyup	释放键盘上按键时触发

续表

事　件	何 时 触 发
onload	页面完全载入后，在 window 对象上触发；所有框架都载入后，在框架集上触发；<img>标记指定的图像完全载入后，在其上触发；或<object>标记指定的对象完全载入后，在其上触发
onmousedown	单击任何一个鼠标按键时触发
Onmousemove	鼠标在某个元素上移动时触发
onmouseout	鼠标在某个元素上移开时触发
onreset	单击重置按钮时在<form>上触发
onresize	窗口或框架大小被改变时触发
onscroll	在任何滚动条的元素或窗口上滚动时触发
onselect	选中文本时触发
onsubmit	单击提交按钮在<form>上触发
onunload	页面完全载卸载后，在 window 对象上触发

2．事件处理程序调用

在使用事件处理程序对页面进行操作时，最主要的是如何通过对象的事件来指定事件处理程序。指定方式主要有以下两种。

（1）在 JavaScript 中

在 JavaScript 中调用事件处理程序，首先需要获得要处理对象的引用，然后将要执行处理的函数值给对应的事件。

代码如下：

```
<input name="but"  type="button"  value="保存">
<script language="javascript">
  Var button=document.getElementById("but");
  button.onclick=function(){
  alert("单击了保存按钮");
}
</script>
```

（2）在 HTML 中

在 HTML 中分配事件处理程序，只要在 HTML 标记中添加相应的事件，并在其中指定要执行的代码或函数即可。

部分应用代码如下：

```
<input name="but" type="button"  value="保存"  onclick=" alert('单击了保存按钮');" >
```

3．JavaScript 输入验证实例

前面介绍了在浏览器端对用户输入进行简单验证的 HTML 代码，这种验证仅局限于输入格式等方面。下面通过一个完整的表单内容验证实例来了解 JavaScript 用法，输入验证的内容的设验证规则为：用户名、密码、重新输入密码这 3 项不能为空，用户名长度不能小于 6 位，两次密码输入必须相同。

实例 3-11　下面是添加按照上面定义的规则验证的表单代码，代码如下：

```
<html>
<head>
  <title>实例 3-11: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">文学</input>
    <input name="habit" type="checkbox" value="3">体育</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，用户没有输入用户名或密码就提交表单，就会分别弹出如图 3-12 所示的对话框，其他各种错误提示与这个提示类似。实例 3-11 运行结果如图 3-12 所示。

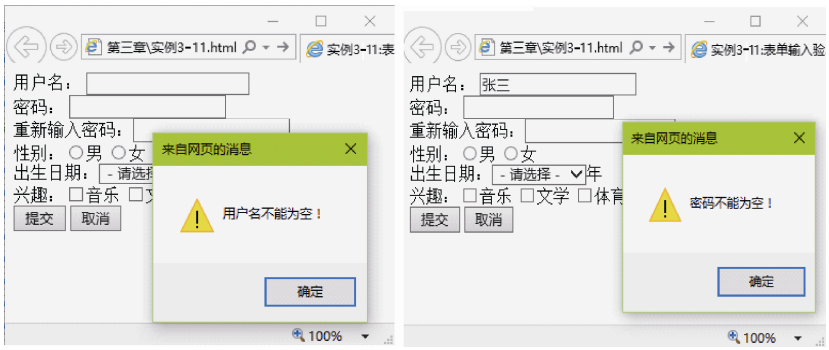

图 3-12　不同验证运行结果

提示：如果经常要在不同文件中调用同一函数，这样的函数可以放在一个 JavaScript 文件里定义，然后在用到的页面里引入这个 JavaScript 文件。JavaScript 文件的后缀是.js。

3.8.4　JSON 简介

如果开发过程中直接用 JavaScript 解析 XML 文件的话，常常会导致复杂的代码和极低的开发效率。为此，JSON 为 Web 应用开发者提供了另一种数据交换格式。

JSON 指的是 JavaScript 对象表示法（JavaScript Object Notation）。JSON 是轻量级的文本数据交换格式，和 XML 一样也是纯文本的，具有独立于语言，自我描述性，更易理解等特点，具有层级结构（值中存在值）可通过 JavaScript 进行解析数据，可使用 AJAX 进行传输。JSON 使用 JavaScript 语法来描述数据对象，但是 JSON 仍然独立于语言和平台。JSON 解析器和 JSON 库支持许多不同的编程语言。

Object 对象在 JSON 中是用 {} 包含一系列无序的 Key-Value 键值对表示的，实际上此处的 Object 相当于 Java 中的 Map<String, Object>，而不是 Java 的 Class 。注意 Key 只能用 String 表示。

例如，一个 employee 对象是包含 3 个员工记录（对象）的数组表示如下：

```
{
"employees": [
  { "firstName":"Bill" , "lastName":"Gates" },
  { "firstName":"George" , "lastName":"Bush" },
  { "firstName":"Thomas" , "lastName":"Carter" }
  ]
}
```

要把 JSON 文本转换为 JavaScript 对象。JSON 最常见的用法之一，是从 Web 服务器上读取 JSON 数据（作为文件或作为 HttpRequest），将 JSON 数据转换为 JavaScript 对象，然后在网页中使用该数据。

实例 3-12　在 JavaScript 中创建 JSON 对象的代码如下：

```
<html>
<head>
  <title>实例 3-12:创建 JSON 对象</title>
</head>
<body>
```

```
  <h2>在 JavaScript 中创建 JSON 对象</h2>
    <p>
    姓名: <span id="jname"></span><br />
    年龄: <span id="jage"></span><br />
    地址: <span id="jstreet"></span><br />
    电话: <span id="jphone"></span><br />
    </p>
    <script type="text/javascript">
    var JSONObject= {
    "name":"张山",
    "street":"江苏省南京市江北新区",
    "age":36,
    "phone":"025 1234567"};
    document.getElementById("jname").innerHTML=JSONObject.name
    document.getElementById("jage").innerHTML=JSONObject.age
    document.getElementById("jstreet").innerHTML=JSONObject.street
    document.getElementById("jphone").innerHTML=JSONObject.phone
    </script>
</body>
</html>
```

实例 3-12 的运行效果如图 3-13 所示。

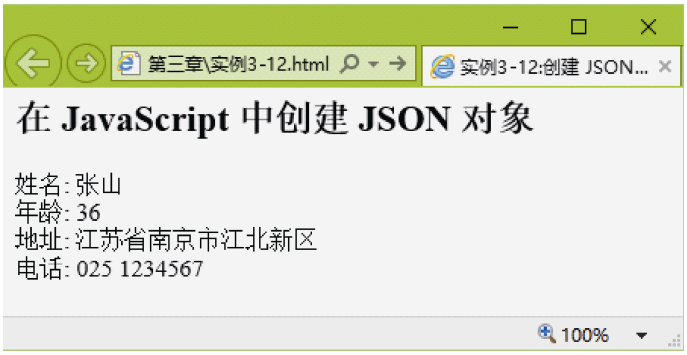

图 3-13　创建 JSON 对象的运行结果

JSON 已经是 JavaScript 标准的一部分。目前，主流的浏览器对 JSON 支持都非常完善。应用 JSON，可以从 XML 的解析中摆脱出来，对那些应用 AJAX 的 Web 2.0 网站来说，JSON 确实是目前最灵活的轻量级方案。

3.9　任务 9　jQuery 基础

jQuery 是一个 JavaScript 函数库。jQuery 极大地简化了 JavaScript 编程。jQuery 库可以通过一行简单的标记被添加到网页中。虽然 jQuery 上手简单，比其他库容易学习，但是要全面掌握却不轻松。因为其涉及网页开发的方方面面，提供的各种方法和内部变化有上千种之多。初学者常常感到入门很方便，提高很困难。

jQuery 是一个轻量级的“写得少，做得多”的 JavaScript 库。jQuery 库包含以下功能：

- HTML 元素选取；
- HTML 元素操作；

- CSS 操作；
- HTML 事件函数；
- JavaScript 特效和动画；
- HTML DOM 遍历和修改；
- AJAX；
- Utilities。

提示：除此之外，jQuery 还提供了大量的插件。

目前网络上有大量开源的 JS 框架，但是 jQuery 是目前最流行的 JS 框架，而且提供了大量的扩展。很多大公司都在使用 jQuery，如 Google、微软、IBM 和 Netflix 等。

3.9.1 jQuery 安装

可以通过多种方法在网页中添加 jQuery。一般可以使用以下两种方法：

- 从 jquery.com 下载 jQuery 库。
- 从 CDN 中载入 jQuery, 如从 Google 中加载 jQuery。

若下载 jQuery 库使用，则有两个版本的 jQuery 可供下载。

- Production version：用于实际的网站中，已被精简和压缩。
- Development version：用于测试和开发（未压缩，是可读的代码）

以上两个版本都可以从 jquery.com 中下载，最新版本是 jquery-3.1.0.js。

jQuery 库是一个 JavaScript 文件，开发人员可以直接使用 HTML 的<script>标签进行引用。其使用格式如下：

```
<head>
  <script src="jquery-1.10.2.min.js"></script>
</head>
```

提示：将下载的文件放在网页的同一目录下，就可以使用 jQuery。

另外，如果开发人员不希望下载并存放 jQuery 库文件，那么也可以通过 CDN（内容分发网络）进行引用，如百度、又拍云、新浪、谷歌和微软的服务器都存有 jQuery。

如果是国内用户，建议使用百度、又拍云、新浪等国内 CDN 地址，如果站点用户是国外用户，则可以使用 Google 和微软。例如，想从百度、又拍云、新浪、Google 或微软引用 jQuery，可使用以下代码之一。

Baidu CDN 如下：

```
<head>
  <script src="http://libs.baidu.com/jquery/1.10.2/jquery.min.js">
  </script>
</head>
```

又拍云 CDN 如下。

```
<head>
 <script
src="http://upcdn.b0.upaiyun.com/libs/jquery/jquery-2.0.2.min.js">
```

```
 </script>
</head>
```

新浪 CDN 如下。

```
<head>
 <script
src="http://lib.sinaapp.com/js/jquery/2.0.2/jquery-2.0.2.min.js">
 </script>
</head>
```

Google CDN 如下。

```
<head>
 <script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
 </script>
</head>
```

注意，不推荐使用 Google CDN 来获取版本，因为 Google 产品在中国很不稳定。

Microsoft CDN 如下。

```
<head>
 <script
src="http://ajax.htmlnetcdn.com/ajax/jQuery/jquery-1.10.2.min.js">
 </script>
</head>
```

使用百度、又拍云、新浪、Google 或微软的 jQuery，有一个很大的优势：

许多用户在访问其他站点时，已经从百度、又拍云、新浪、Google 或微软加载过 jQuery，所以结果是，当用户再次访问站点时，会从缓存中加载 jQuery，这样可以减少加载时间。同时，大多数 CDN 都可以确保当用户向其发送请求文件时，会从离用户最近的服务器上返回响应，这样也可以提高加载速度。

3.9.2　jQuery 语法

jQuery 语法是通过选取 HTML 元素，并对选取的元素执行某些操作。也就是说 jQuery 的基本设计和主要用法，就是“选择某个网页元素，然后对其进行某种操作”。这是 jQuery 它区别于其他函数库的根本特点。

基础语法为：

```
$(selector).action()
```

- 美元符号$表示定义 jQuery。
- 选择符（selector）表示“查询”和“查找”HTML 元素。
- jQuery 的 action() 表示执行对元素的操作。

实例用法如下：

```
$(this).hide() - 隐藏当前元素
$("p").hide() - 隐藏所有 <p> 元素
$("p.test").hide() - 隐藏所有 class="test" 的 <p> 元素
$("#test").hide() - 隐藏所有 id="test" 的元素
```

实例中的所有 jQuery 函数位于一个 document ready 函数中：

```
$(document).ready(function(){
--- jQuery functions 代码 ----
});
```

这是为了防止文档在完全加载（就绪）之前运行 jQuery 代码。如果在文档没有完全加载之前就运行函数，操作可能失败。

实例 3-13 下面通过一个简单的实例了解其语法结构，其功能是单击按钮后隐藏<p>元素中的文字。代码如下：

```
<html>
<head>
  <script type="text/javascript" src="jquery/jquery.js"></script>
  <script type="text/javascript">
    $(document).ready(function(){
      $("button").click(function(){
        $("p").hide();
      });
    });
  </script>
</head>

<body>
<h2>这是实例 3-13：jQuery 演示</h2>
  <p>下面是段落内容！.</p>
  <p>另一段落内容！.</p>
  <button>单击我</button>
</body>
```

在上面的例子中，当按钮的单击事件被触发时会调用一个函数：

```
$("button").click(function() {..some code... } )
```

该方法隐藏所有 <p> 元素内容：

```
$("p").hide();
```

</html>实例 3-13 的运行结果如图 3-14 所示。

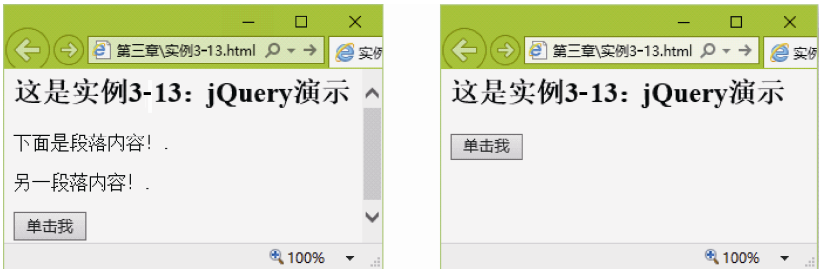

图 3-14 运行结果

3.9.3 jQuery 选择器

本节的关键点是学习 jQuery 选择器是如何准确地选取希望应用效果的元素。jQuery 元

素选择器和属性选择器允许通过标签名、属性名或内容对 HTML 元素进行选择。选择器允许对 HTML 元素组或单个元素进行操作。

常用选择器实例如表 3-4 所示。

表 3-4　常用选择器实例

选　择　器	实　　例	选　　取
*	$("*")	所有元素
#id	$("#lastname")	id="lastname" 的元素
.class	$(".intro")	所有 class="intro" 的元素
element	$("p")	所有 <p> 元素
.class.class	$(".intro.demo")	所有 class="intro" 且 class="demo" 的元素
:first	$("p:first")	第一个 <p> 元素
:last	$("p:last")	最后一个 <p> 元素
:even	$("tr:even")	所有偶数 <tr> 元素
:odd	$("tr:odd")	所有奇数 <tr> 元素
:eq(index)	$("ul li:eq(3)")	列表中的第四个元素（index 从 0 开始）
:gt(no)	$("ul li:gt(3)")	列出 index 大于 3 的元素
:lt(no)	$("ul li:lt(3)")	列出 index 小于 3 的元素
:not(selector)	$("input:not(:empty)")	所有不为空的 input 元素
:header	$(":header")	所有标题元素 <h1> - <h6>
:animated		所有动画元素
:contains(text)	$(":contains('W3School')")	包含指定字符串的所有元素
:empty	$(":empty")	无子（元素）节点的所有元素
:hidden	$("p:hidden")	所有隐藏的 <p> 元素
:visible	$("table:visible")	所有可见的表格
s1,s2,s3	$("th,td,.intro")	所有带有匹配选择的元素
[attribute]	$("[href]")	所有带有 href 属性的元素
[attribute=value]	$("[href='#']")	所有 href 属性的值等于“#”的元素
[attribute!=value]	$("[href!='#']")	所有 href 属性的值不等于“#”的元素
[attribute$=value]	$("[href$='.jpg']")	所有 href 属性的值包含“.jpg”的元素
:input	$(":input")	所有 <input> 元素
:text	$(":text")	所有 type="text" 的 <input> 元素
:password	$(":password")	所有 type="password" 的 <input> 元素
:radio	$(":radio")	所有 type="radio" 的 <input> 元素
:checkbox	$(":checkbox")	所有 type="checkbox" 的 <input> 元素
:submit	$(":submit")	所有 type="submit" 的 <input> 元素
:reset	$(":reset")	所有 type="reset" 的 <input> 元素
:button	$(":button")	所有 type="button" 的 <input> 元素
:image	$(":image")	所有 type="image" 的 <input> 元素
:file	$(":file")	所有 type="file" 的 <input> 元素
:enabled	$(":enabled")	所有激活的 input 元素
:disabled	$(":disabled")	所有禁用的 input 元素
:selected	$(":selected")	所有被选取的 input 元素
:checked	$(":checked")	所有被选中的 input 元素

3.9.4 jQuery 事件操作

jQuery 是为事件处理特别设计的。jQuery 事件处理方法是 jQuery 中的核心函数。事件处理程序指的是当 HTML 中发生某些事件时所调用的方法。只需把所有 jQuery 代码置于事件处理函数中，把所有事件处理函数置于文档就绪事件处理器中，把 jQuery 代码置于单独的 .js 文件中即可。如果存在名称冲突，则重命名 jQuery 库。

jQuery 可以对网页元素绑定事件。根据不同的事件，运行相应的函数。通常会把 jQuery 代码放到 <head>部分的事件处理方法中：

```
$('p').click(function(){
    alert('Hello');
});
```

目前，jQuery 主要支持以下事件：

- .blur() 表单元素失去焦点。
- .change() 表单元素的值发生变化。
- .click() 鼠标单击。
- .dblclick() 鼠标双击。
- .focus() 表单元素获得焦点。
- .focusin() 子元素获得焦点。
- .focusout() 子元素失去焦点。
- .hover() 同时为 mouseenter 和 mouseleave 事件指定处理函数。
- .keydown() 按下键盘（长时间按键，只返回一个事件）。
- .keypress() 按下键盘（长时间按键，将返回多个事件）。
- .keyup() 松开键盘。
- .load() 元素加载完毕。
- .mousedown() 按下鼠标。
- .mouseenter() 鼠标进入（进入子元素不触发）。
- .mouseleave() 鼠标离开（离开子元素不触发）。
- .mousemove() 鼠标在元素内部移动。
- .mouseout() 鼠标离开（离开子元素也触发）。
- .mouseover() 鼠标进入（进入子元素也触发）。
- .mouseup() 松开鼠标。
- .ready() DOM 加载完成。
- .resize() 浏览器窗口的大小发生改变。
- .scroll() 滚动条的位置发生变化。
- .select() 用户选中文本框中的内容。
- .submit() 用户递交表单。
- .toggle() 根据鼠标单击的次数，依次运行多个函数。
- .unload() 用户离开页面。

以上这些事件在 jQuery 内部，都是.bind()的便捷方式。使用.bind()可以更灵活地控制

事件，例如为多个事件绑定同一个函数：

```
$('input').bind(
    'click change', //同时绑定 click 和 change 事件
    function() {
      alert('Hello');
    }
);
```

有时只想让事件运行一次，这时可以使用.one()方法。

```
$("p").one("click", function() {
    alert("Hello"); //只运行一次，以后的单击不会运行
});
```

.unbind()用来解除事件绑定。

```
$('p').unbind('click');
```

所有的事件处理函数，都可以接受一个事件对象（event object）作为参数，例如下面例子中的 e。

```
$("p").click(function(e) {
    alert(e.type); // "click"
});
```

这个事件对象有一些很有用的属性和方法如下。

- event.pageX：事件发生时，鼠标距离网页左上角的水平距离。
- event.pageY：事件发生时，鼠标距离网页左上角的垂直距离。
- event.type：事件的类型（如 click）。
- event.which：按下了哪一个键。
- event.data：在事件对象上绑定数据，然后传入事件处理函数。
- event.target：事件针对的网页元素。
- event.preventDefault()：阻止事件的默认行为（如单击链接，会自动打开新页面）。
- event.stopPropagation()：停止事件向上层元素冒泡。

在事件处理函数中，可以用 this 关键字，返回事件针对的 DOM 元素：

```
$('a').click(function() {
    if ($(this).attr('href').match('evil')) { //如果确认为有害链接
        e.preventDefault(); //阻止打开
        $(this).addClass('evil'); //加上表示有害的 class
    }
});
```

有两种方法可以自动触发一个事件，一种是直接使用事件函数，另一种是使用.trigger()或.triggerHandler()函数。

```
$('a').click();
$('a').trigger('click');
```

另外，jQuery 允许对象呈现某些特殊效果。

```
$('h1').show(); //展现一个 h1 标题
```

常用的特殊效果如下。

- .fadeIn()：淡入；
- .fadeOut()：淡出；
- .fadeTo()：调整透明度；
- .hide()：隐藏元素；
- .show()：显示元素；
- .slideDown()：向下展开；
- .slideUp()：向上卷起；
- .slideToggle()：依次展开或卷起某个元素；
- .toggle()：依次展示或隐藏某个元素。

实例 3-14　下面通过一个例子进一步了解事件处理函数的使用。代码如下：

```
<html>
<head>
<script type="text/javascript" src="jquery/jquery.js"></script>
  <script type="text/javascript">
  $(document).ready(function(){
    $("button").click(function(){
      $("p").css({"background-color":"red","font-size":"200%"});
    });
  });
  </script>
</head>
<body>
  <h2>这是实例 3-14：jQuery 事件演示</h2>
  <p>第一段内容.</p>
  <p>第二段内容.</p>
  <button type="button">单击我</button>
</body>
</html>
```

实例 3-14 运行效果如图 3-15 所示。

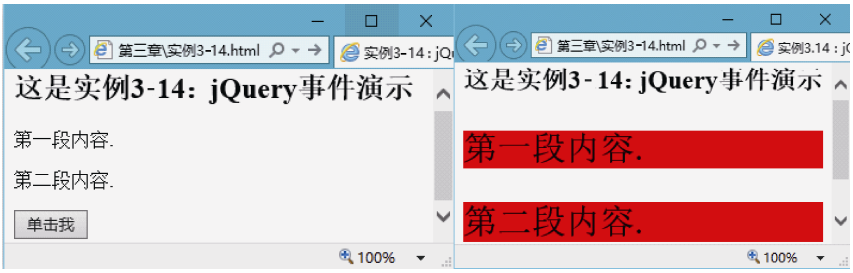

图 3-15　运行结果

3.10　任务 10　AJAX 基础应用

在 Web 应用程序开发中，页面重载循环是最大的一个使用障碍，对于 Java 开发人员来说也是一个严峻的挑战。而 AJAX（异步 JavaScript 和 XML）是一种编程技术，允许

为基于 Java 的 Web 应用程序把 Java 技术、XML 和 JavaScript 组合起来，从而打破页面重载的范式。

3.10.1　AJAX 概述

AJAX 全称为 Asynchronous JavaScript and XML（异步 JavaScript 和 XML），是一种创建交互式网页应用的网页开发技术，类似于 DHTML 或 LAMP。AJAX 不是指一种单一的技术，而是有机地利用了一系列相关的技术。

AJAX 不是一种新的编程语言，而是一种用于创建更好更快以及交互性更强的 Web 应用程序的技术。通过 AJAX 技术 JavaScript 可使用 JavaScript 的 XMLHttpRequest 对象直接与服务器进行通信。通过这个对象，JavaScript 可在不重载页面的情况与 Web 服务器交换数据。AJAX 在浏览器与 Web 服务器之间使用异步数据传输（HTTP 请求），这样就可使网页从服务器请求少量的信息，而不是整个页面。Ajax 可使互联网应用程序更小、更快、更友好。

AJAX 是一种独立于 Web 服务器软件的浏览器技术。AJAX 基于下列 Web 标准：JavaScript、XML、HTML、CSS。在 AJAX 中使用的 Web 标准已被良好定义，并被所有的主流浏览器支持。AJAX 应用程序独立于浏览器和平台。采用 AJAX 的 MVC 设计模式的工作原理往返过程，如图 3-16 所示。

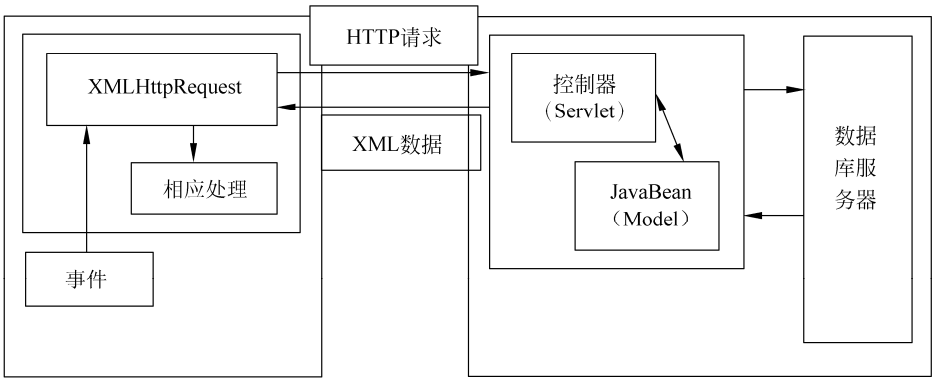

图 3-16　AJAX 工作原理的 MVC 模式图

图 3-16 中，AJAX 交互开始于叫作 XMLHttpRequest 的 JavaScript 对象。它允许客户端脚本执行 HTTP 请求，并解析 XML 服务器响应。AJAX 往返过程的第一步是创建 XMLHttpRequest 的实例。在 XMLHttpRequest 对象上设置请求使用的 HTTP 方法（GET 或 POST）以及目标 URL。

在发送 HTTP 请求时，不想让浏览器“挂着”等候服务器响应，相反，想让浏览器继续对用户与页面的交互进行响应，并在服务器响应到达时再进行处理。为了实现这个要求，可以在 XMLHttpRequest 上注册一个回调函数进行相应处理，然后异步地分派。之后控制就会返回浏览器，当服务器响应到达时，会调用回调函数。在 Java Web 服务器上，请求同其他 HttpServletRequest 一样到达。在解析了请求参数之后，控制器 Servlet 调用必要的应用程序逻辑，把响应序列化成 XML，并把 XML 写入 HttpServletResponse。

回到客户端时，先调用注册在 XMLHttpRequest 上的回调函数，处理服务器返回的 XML 文档。最后，根据数据库服务器返回的数据，用 JavaScript 操纵页面的 HTML DOM，将用户界面更新。

3.10.2 XMLHttpRequest 对象

AJAX 基本上就是把 JavaScript 技术和 XMLHttpRequest 对象放在 Web 表单和服务器之间。当用户填写表单时，数据发送给一些 JavaScript 代码而不是直接发送给服务器。相反，JavaScript 代码捕获表单数据并向服务器发送请求。同时用户屏幕上的表单也不会闪烁、消失或延迟。换句话说，JavaScript 代码在幕后发送请求，用户甚至不知道请求的发出。更好的情况是，请求是异步发送的，就是说 JavaScript 代码（和用户）不用等待服务器的响应。因此用户可以继续输入数据、滚动屏幕和使用应用程序。

然后，服务器将数据返回 JavaScript 代码（仍然在 Web 表单中），JavaScript 决定如何处理这些数据，其可以迅速更新表单数据，让人感觉应用程序是立即完成的，表单没有提交或刷新，而用户得到了新数据。JavaScript 代码甚至可以对收到的数据执行某种计算，再发送另一个请求，完全不需要用户干预！这就是 XMLHttpRequest 的强大之处。它可以根据需要自行与服务器进行交互，用户甚至可以完全不知道幕后发生的一切。结果就是类似于桌面应用程序的动态、快速响应、高交互性的体验，但是背后又拥有互联网的全部的强大力量。

创建新的 XMLHttpRequest 对象代码如下：

```
<script language="javascript" type="text/javascript">
var xmlHttp = new XMLHttpRequest();
</script>
```

通过 XMLHttpRequest 对象与服务器进行对话的是 JavaScript 技术。这不是一般的应用程序流，恰恰是 AJAX 的强大功能的来源。得到 XMLHttpRequest 的句柄后，其他的 JavaScript 代码就非常简单了。事实上，我们将使用 JavaScript 代码完成非常基本的任务：

- 获取表单数据：JavaScript 代码很容易从 HTML 表单中抽取数据并发送到服务器。
- 修改表单上的数据：更新表单也很简单，从设置字段值到迅速替换图像。

下面给出将要用于 XMLHttpRequest 的常用方法和属性。

- open()：建立到服务器的新请求。
- send()：向服务器发送请求。
- abort()：退出当前请求。
- readyState：提供当前 HTML 的就绪状态。
- responseText：服务器返回的请求响应文本。

创建具有错误处理能力的 XMLHttpRequest 代码如下：

```
<script language="javascript" type="text/javascript">
var request = false;
  try {
    request = new XMLHttpRequest();
  } catch (failed) {
    request = false;
  }
```

```
if (!request)
  alert("Error initializing XMLHttpRequest!");
</script>
```

上述代码含义解释如下：

- 创建一个新变量 request 并赋值 false。后面将使用 false 作为判定条件，表示还没有创建 XMLHttpRequest 对象。
- 增加 try/catch 块：尝试创建 XMLHttpRequest 对象，如果失败（catch (failed)）则保证 request 的值仍然为 false。
- 检查 request 是否仍为 false（如果一切正常就不会是 false）。
- 如果出现问题（request 是 false）则使用 JavaScript 警告通知用户出现了问题。

3.10.3　AJAX 应用实例

下面通过一个 Web 开发过程中经常遇到的用户登录注册验证的实例，进一步理解 AJAX 工作原理，在输入过程中能动态检查错误并实时显示。在 MyEclipse 中创建 Java Web 项目，并创建两个文件，第一个文件名为“实例 3-15.html”的代码如下：

```
<html>
<head>
  <title>实例 3-15:AJAX 登录验证实例</title>
</head>
<body>
<script language="JavaScript">
       var req = null;
       function test() {
              //初始化
              var code = document.all.code.value;
              var name = document.all.name.value;
              req = new ActiveXObject("Microsoft.XMLHTTP");
              //设置属性，当后台处理完成后，回来调用 myDeal()方法
              req.onreadystatechange = myDeal;
              //发出请求
              req.open("GET", "b.jsp?code=" + code + "&name=" + name,
              "false");

              req.send(null);
       }
       function myDeal() {
              if (req.readyState == 4) {
                     //接收服务端返回的数据
                     var ret = req.responseText;
                     //处理数据
                     document.all("myDiv").innerHTML = ret;
              }
       }
</script>

用户注册：<br>
用户编号：<input type="text" name="code" onblur="test();">* <div id="myDiv"
name="myDiv"></div><br>
用户名称：<input type="text" name="name"><br>
<input type="button" value="注册" onclick="test();">
```

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

新建另一个文件名为 b.jsp 的代码如下：

```
    <%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%
        //接收参数
        String code = request.getParameter("code");
        String name = request.getParameter("name");
        //控制台输出表单数据
        System.out.println("code=" + code + ",name=" + name);
        //检查 code 的合法性
        if (code == null || code.trim().length() == 0) {
                out.println("用户代码不能为空！");
        } else if (code != null && code.equals("admin")) {
                out.println("不能用管理员代码！");
        } else {
                out.println("正确！");
        }
%>
```

将上述 AJAX 项目按第 2 章介绍的方法部署到 Tomcat 应用服务器后，打开浏览器输入项目地址后，运行结果如图 3-17 所示。

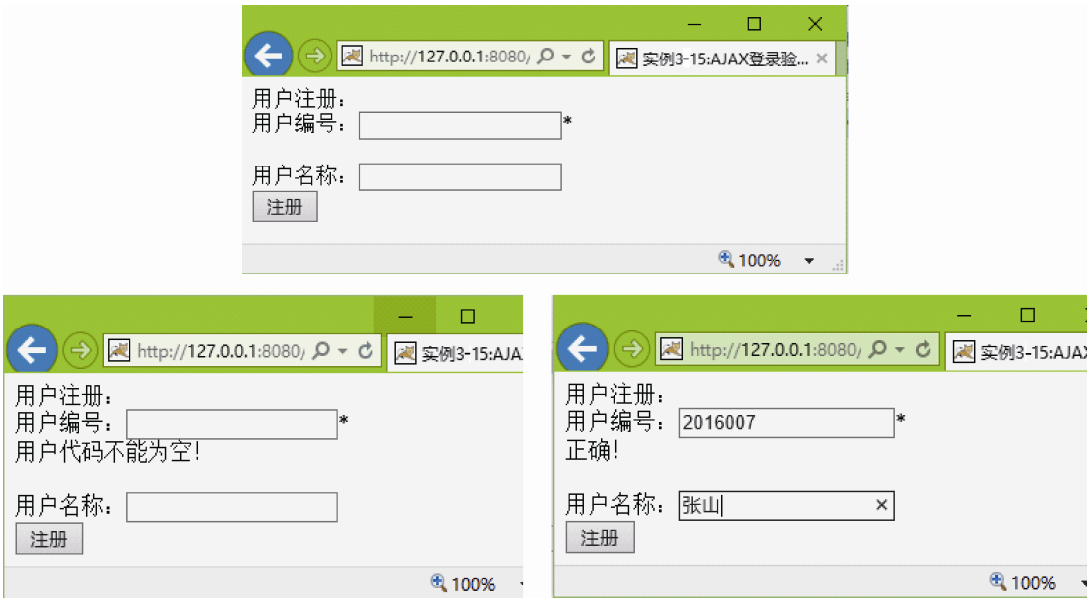

图 3-17　运行效果

3.10.4　开源 AJAX 框架 DWR 介绍

开发 Web IM 系统普遍采用 DWR（Direct Web Remoting）框架，它的实现是基于 AJAX，可以实现无刷新效果。DWR 是一个 Web 远程调用框架，利用这个框架可以让 AJAX 开发变得很简单。利用 DWR 可以在客户端利用 JavaScript 直接调用服务端的 Java 方法并返回值给 JavaScript，就像直接从本地客户端调用一样（DWR 根据 Java 类动态生成 JavaScript

代码）。它的最新版本 DWR 3.0.1 添加了许多特性如：支持 Dom Trees 的自动配置，支持 Spring（JavaScript 远程调用 spring bean），更好的浏览器支持，还支持一个可选的 commons-logging 日记操作。

其开发过程大概如下：

（1）编写业务代码，该代码是和 DWR 无关的。

（2）确认业务代码中哪些类、哪些方法是由 JavaScript 直接访问的。

（3）编写 DWR 组件，对步骤（2）的方法进行封装。

（4)配置 DWR 组件至 dwr.xml 文件中，如果有必要，配置 convert，进行 Java 和 JavaScript 类型互转。

（5）通过反射机制，DWR 将步骤（4）的类转换成 JavaScript 代码，提供给前台页面调用。

（6）编写网页，调用步骤（5）的 JavaScript 中的相关方法（间接调用服务器端的相关类的方法），执行业务逻辑，将执行结果利用回调函数返回。

（7）在回调函数中，得到执行结果后，可以继续编写业务逻辑的相关 JavaScript 代码。

注意：因 DWR 项目代码较多，完整的 DWR 官方演示代码，请参见本书配套电子资源第 3 章代码。将课本配套资源中的 dwrdemo.war 文件直接复制到 Tomcat 应用服务器的 webapps 目录下，然后启动 Tomcat 服务，在浏览器地址栏中输入以下不同地址就可以进行测试，其完整源代码及使用说明在本书配套资源中。

实例 1 地址：http://127.0.0.1:8080/dwrdemo/simple/text.html;

实例 2 地址：http://127.0.0.1:8080/dwrdemo/people/search.html;

实例 3 地址：http://127.0.0.1:8080/dwrdemo/reverseajax/java-chat.html。

3.11　本 章 小 结

本章力求对 HTML 和 JavaScript 最新开发知识进行介绍，使读者了解到制作静态网页 HTML 可以胜任，但动态漂亮的网页必须加入 CSS，可以说 HTML 是编写框架，CSS 是对网页美化，JavaScript 是对网页的操作。在本章的讲解中，还对 HTML 及 JavaScript 最新 Web 开发技术如 HTML5.0、jQuery、JSON、AJAX、DWR 等进行了介绍，使读者可以迅速对最新 Java Web 开发的基础知识有个宏观、清楚的认识，从而可以快速进入后面章节的学习，如果读者对这方面的基础知识想更深地了解，需要参考相关的专题书籍。

3.12　习　　题

一、选择题

1．在 HTML 中，样式表按照应用方式可以分为 3 种类型，其中不包括_____。

A．内嵌样式表　　B．行内样式表

C．外部样式表文件　　D．类样式表

2．在 HTML 中，可以使用_____标记向网页中插入 GIF 动画文件。

A．<FORM>　　B．<BODY>

C．<TABLE>　　D．<IMG>

3．以下说法正确的是_____。

A．<P>标签必须以</P>标签结束

B．
标签必须以</BR>标签结束

C．<TITLE>标签应该以</TITLE>标签结束

D．<FONT>标签不能在<PRE>标签中使用

4．关于下列代码片段的说法中，正确的是_____。

```
<HR size= "5" color="#0000FF" width="50%">
```

A．size 是指水平线的长度

B．size 是指水平线的宽度

C．width 是指水平线的宽度

D．width 是指水平线的高度

5．以下说法正确的是_____。

A．<A>标签是页面链接标签，只能用来链接到其他页面

B．<A>标签是页面链接标签，只能用来链接到本页面的其他位置

C．<A>标签的 src 属性用于指定要链接的地址

D．<A>标签的 href 属性用于指定要链接的地址

二、填空题

1．创建一个 HTML 文档的开始标记符_____；结束标记符是_____。

2．设置文档标题以及其他不在 Web 网页上显示的信息的开始标记符_____；结束标记符是_____。

3．设置文档的可见部分开始标记符_____；结束标记符是_____。

4．网页标题会显示在浏览器的标题栏中，则网页标题应写在开始标记符_____和结束标记符_____之间。

5．预格式化文本标记<pre></pre>的功能是_____。

6．jQuery 基础语法为_____。

三、简答题

1．简要说明表格与框架在网页布局时的区别。

2．表单是实现动态交互式的可视化界面，在表单开始标记中一般包含哪些属性?其含义分别是什么？

3．JavaScript 的常用数据类型有哪些？并举例说明。

4．简述 JSON 概念。

5．简述 AJAX 开源框架 DWR 大概开发过程。

四、上机操作题

1．练习本章任务 1 到任务 8 的 HTML 基础实验；

2．练习本章任务 9 的 jQuery 实验；
3．练习本章任务 10 的 AJAX 实验。

实训 3　HTML 和 JavaScript 综合应用

一、实验目的

1．掌握 HTML 表单的编写。
2．掌握 HTML 框架的使用。
3．了解 jQuery 的使用。
4．掌握 AJAX 的使用。
5．实验学习 DWR 框架使用（选做）。

二、实验内容

综合应用 HTML 框架技术和 jQuery 的程序主界面项目，只须将课程源代码第 3 章中的 testajax 目录复制到 Tomcat 的 webapps 目录下，然后启动 Tomcat 服务后运行即可。也可以在 MyEclipse 中新建一个 Java Web 项目，导入到新项目中编辑、发布和运行。

项目的实现效果如图 3-18 所示，其中整体主界面用到了 HTML 框架技术，左侧下拉列表框则用到了 jQuery 技术，单击相应选项，则自动伸缩拉开菜单供用户选择。注册页面中则用到了 HTML 表单技术，并通过 JavaScript 对表单内容进行验证，非法输入则不能注册。

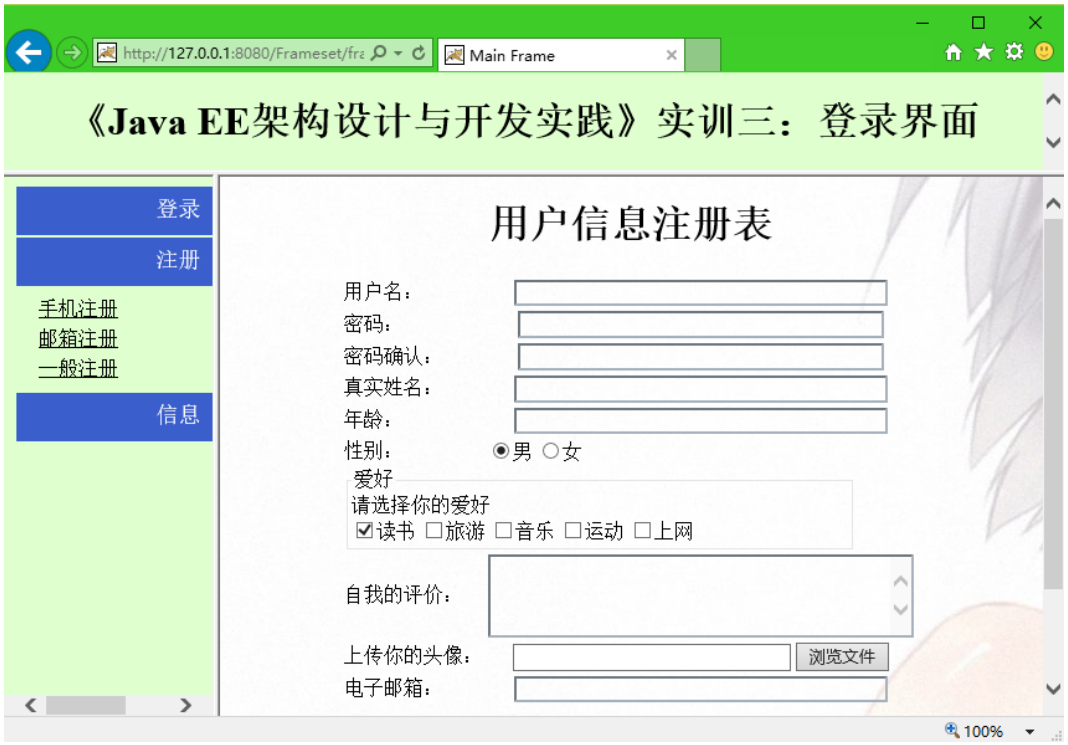

图 3-18　登录界面主界面

（1）主体框架文件 Frameset.jsp 代码如下。

```
<frameset rows="80,*">
  <frame src="top.jsp" name="top">
  <frameset cols="20%,*">
  <frame src="left.jsp" name="left">
  <frame src="right.jsp" name="right">
  </frameset>
</frameset>
```

（2）头部文件 top.jsp 代码如下。

```
<style>
body{
 background-image:url(image/bj.jpg);
 }
</style>
<body style="text-align:center" >
<h1>《Java EE 架构设计与开发实践》实训三：登录界面</h1>
</body>
```

（3）左部文件 left.jsp 代码如下。

```
<title>jQuery 技术实现单击伸缩、展开的菜单</title>
<style type="text/css">
body { font-family: Arial; font-size: 16px;  background-image:url(image/
6.jpg);}
dl { width: 250px; }
dl,dd { margin: 0; }
dt { background-color:#3A5FCD;  background-position:5px 13px; font-size:
18px; padding: 5px 5px 5px 20px; margin: 2px; height:29px; line-height:28px;
    text-align:center;
}
dt a { color: #FFF; text-decoration:none; }
dd a { color: #000;  }
ul{ list-style: none; padding:5px 5px 5px 20px; margin:0; }
li{ line-height:24px;}
.bg{ background-position:5px -16px;}
</style>
<script type="text/javascript" src="jquery/2.js">
</script>
<script type="text/javascript">
$(document).ready(function(){
  $("dd").hide();
  $("dt a").click(function(){
  $(this).parent().toggleClass("bg");
  $(this).parent().prevAll("dt").removeClass("bg")
  $(this).parent().nextAll("dt").removeClass("bg")
  $(this).parent().next().slideToggle();
  $(this).parent().prevAll("dd").slideUp("slow");
  $(this).parent().next().nextAll("dd").slideUp("slow");
  return false;
  });
});
</script>
</head>
<body>
<dl>
  <dt><a href="/">登录</a></dt>
  <dd>
```

```
  <ul>
      <li><a href="login2.jsp" target="right">手机登录</a></li>
      <li><a href="login3.jsp" target="right">邮箱登录</a></li>
      <li><a href="login.jsp" target="right">一般登录</a></li>
  </ul>
  </dd>
  <dt><a href="/">注册</a></dt>
  <dd>
  <ul>
      <li><a href="#">手机注册</a></li>
      <li><a href="#">邮箱注册</a></li>
      <li><a href="register.jsp" target="right">一般注册</a></li>
  </ul>
  </dd>
  <dt><a href="/">信息</a></dt>
  <dd>
  <ul>
      <li><a href="#">个人信息</a></li>
      <li><a href="#">注册须知</a></li>
  </ul>
  </dd>
</dl>
</body>
```

（4）框架右部文件 right.jsp 代码如下。

```
<style>
div{text-align:center;}
</style>
<body>
<div>
<form method="post" action="main.jsp">
<caption><h1>用户登录</h1></caption>
用户: <input style="background-color:transparent;border-width:1;
border-color:#DDEEFF;"  type="text" name="user"  size="50" /><br><br>
密码: <input style="background-color:transparent;border-width:1;border-
color:#DDEEFF;"  type="password" name="password"  size="50" /><br><br>
<input type="checkbox" name="remember" value="remember_name" />记住账号
<input type="checkbox" name="remember" value="remember_password" />记住密
码<br><br>
<input type="submit" name="Submit" value="提交" />   
<input type="reset" name="delete" value="取消"/>
</form>
</div>
</body>
```

（5）注册文件 register.jsp 代码如下。

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE   html   PUBLIC   "-//W3C//DTD   HTML   4.01   Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>用户信息注册界面</title>
</head>
<script language="javascript">
```

```
function checkuser(value){
  if(value==""){
   document.getElementById("usernameError").innerHTML= "内容不能为空！" ;
  }else{
     document.getElementById("usernameError").innerHTML= "" ;
  }
}
function checkpassword(value){
  if(value==""){
      document.getElementById("passwordError").innerHTML= "内容不能为空！";
  }else{
     document.getElementById("passwordError").innerHTML= "" ;
  }
}function checkage(value){
  if(value==""){
      document.getElementById("ageError").innerHTML= "内容不能为空！" ;
  }else{
     document.getElementById("ageError").innerHTML= "" ;
  }
}
  function checkrepassword(){
   var str1=form1.password.value;
   var str2=form1.repassword.value;
   if(str1!=str2){
   document.getElementById("repasswordError").innerHTML= "密码输入不一致!";
      }else{
         document.getElementById("repasswordError").innerHTML= "" ;
      }
   }
  function checkname(){
     var name=form1.name.value;
     var obj=/[\u4E00-\u9FA5]{2,}/;
     if(obj.test(name)){
     document.getElementById("nameError").innerHTML= "你输入姓名正确！" ;
     }else{
     document.getElementById("nameError").innerHTML= "你输入姓名不正确" ;
     }
  }
  function checkall(){
  var  str1=form1.user.value;
  var  str2=form1.password.value;
  var  str3=form1.name.value;
  var  str4=form1.age.value;
  if(str1||str2||str3||str4==""){
     alert("请填写完整信息！");
     return false;
    }
  }
</script>
</tr>
<tr>
<td align="left">性别：</td>
<td align="left">
<input type="radio" name="sex" value="man" checked />男
<input type="radio" name="sex" value="woman"  />女
</td>
</tr>
<tr>
<td colspan="2" align="left">
```

```
<fieldset style="width:400px">
<legend>爱好</legend>
请选择你的爱好<br>
<input type="checkbox" name="hobby" value="reading" checked/>读书
 <input type="checkbox" name="hobby" value="journey" />旅游
 <input type="checkbox" name="hobby" value="music" />音乐
 <input type="checkbox" name="hobby" value="transport" />运动
 <input type="checkbox" name="hobby" value="surfing" />上网
</fieldset>
</td>
</tr>
<tr>
<td align="left">自我的评价：</td>
<td><textarea
style="background-color:transparent;border-width:1;border-color:#DDEEFF
;"  name="evaluate" rows="4" cols="40"></textarea> </td>
</tr>
<tr>
<td align="left">上传你的头像：</td>
<td><input type="text" id="photo" name="myphoto" size="32"/>
<input type="button"  name="button" onclick="myfile.click()" value="浏览文
件" />
<input    type="file"    id="myfile"    onchange="photo.value=this.value"
style="display:none" />
</td>
</tr>
<tr>
<td align="left">电子邮箱：</td>
<td><input
style="background-color:transparent;border-width:1;border-color:#DDEEFF
;"  type="text" name="email" size="45" /></td>
</tr>
</table><br><br>
         
<input type="submit" name="Submit" value="注册"  />
<input type="reset"  name="Reset" value="重置" />
<input type="button"  name="drop"  value="取消" onclick="window.close();"
/>
<br><br><br>
</div>
</form>
</body>
```

提示：

- 因本书项目代码较多，本章其他项目代码就不一一列出了，请参见本书配套电子资源第 3 章实训代码目录中。
- 如其中 jQuery 多级展开手风琴竖向菜单 DEMO 演示，这是一款基于 jQuery 的多级展开手风琴竖向菜单，菜单是垂直的，单击菜单项即可垂直展开，显示对应菜单项的相关描述。另外，这款 jQuery 垂直手风琴菜单的菜单项还有一个漂亮的小图标，是一款比较实用的 jQuery 菜单插件，可在实际项目中直接修改后选用。
- DWR 官方演示程序的使用，内有大量 DWR 源代码，可根据需要选做本实验。