TURING 图灵程序设计丛书 Web开发系列

jQuery Recipes A Problem-Solution Approach

jQuery攻略

[印] B. M. Harwani 著
侯伯薇 陈宁 译

人民邮电出版社
北京

图书在版编目（CIP）数据

jQuery攻略 / （印）哈瓦尼（Harwani,B.M.）著；侯伯薇，陈宁译. -- 北京 : 人民邮电出版社，2010.10
（图灵程序设计丛书）
书名原文：jQuery Recipes: A Problem-Solution Approach
ISBN 978-7-115-23830-6

Ⅰ. ①j… Ⅱ. ①哈… ②侯… ③陈… Ⅲ. ①JAVA语言－程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2010)第182499号

内 容 提 要

本书对使用jQuery过程中遇到的各类问题给出了解决方案，比如，如何使用jQuery框架、CSS选择器、DOM、事件处理、动画效果，以及如何开发Ajax应用程序、如何使用jQuery工具函数、如何使用插件扩展jQuery。

本书非常适合想利用最少的代码创建交互性网站的开发人员学习和参考，也适合懂少量HTML知识又想创建动态网站的初学者学习。

图灵程序设计丛书
jQuery攻略

◆ 著 [印] B. M. Harwani
译 侯伯薇 陈 宁
责任编辑 朱 巍
执行编辑 陈彦辛

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
中国铁道出版社印刷厂印刷

◆ 开本：800×1000 1/16
印张：21.5
字数：522千字 2010年10月第1版
印数：1－3 000册 2010年10月北京第1次印刷

著作权合同登记号 图字：01-2010-2363号
ISBN 978-7-115-23830-6

定价：59.00元

读者服务热线：(010)51095186 印装质量热线：(010)67129223
反盗版热线：(010)67171154

版 权 声 明

译 者 序

回顾一下自己的工作生涯，忽然发现我大部分时间都在从事 B/S 应用程序的开发，因此也就与 JavaScript 结下了不解之缘。最初使用最基本的 JavaScript 开发页面的脚本，之后出现了 Ajax，使用了 Prototype，直到发现并开始使用 jQuery，越来越觉得开发 JavaScript 脚本是一件快乐的事！

使用 jQuery，不仅仅可以让我们更加高质高效地编写 JavaScript 代码，而且其中内建的功能，让我们可以很轻松地实现诸如表单验证、表格处理、Ajax、视觉特效等很棒的功能，让我们的用户有更好的体验。

本书由浅入深，介绍了使用 jQuery 的方方面面，不仅包括最基本的安装配置、处理字符串和数组、事件处理等比较基本的内容，也包括了表单验证、页面导航、视觉特效，处理表格等进阶的技巧，另外还讲述了如何使用它来实现 Ajax 功能，如何使用插件得到更高的开发效率，最终作者还将 CSS 方面的内容作为必要的补充提供给我们。

所有的内容都非常实用，我们可以在遇到具体问题的时候，将此书作为参考，使用其中提供的代码快速实现所需要的功能。而针对每个具体问题的解决方案的剖析，让我们不仅能够知其然，而且知其所以然。

不论是初学者，还是已经有 JavaScript 开发经验，想要使用 jQuery 来提高开发效率和质量的程序员，或者已经有了一定 jQuery 开发经验的开发者，在本书中都会找到所需要的知识。

愿大家享受本书，享受 jQuery！

侯伯薇

2010 年 6 月于大连

前　言

内容

jQuery是功能丰富的JavaScript库，可以帮助用户毫不费力地把动态功能应用到网页。jQuery具有许多强大的功能，包括访问部分网页、快速修改网页内容、添加动画、应用AJAX技术，等等。

本书以问题描述+解决方案的方式，帮助读者理解jQuery这个开源项目的各种功能。本书开头介绍利用选择器把效果应用到段落和列表，从而教会你如何设置网页布局。之后学习事件处理和对不同表单元素执行验证的技术。应用视觉效果、导航、Ajax和jQuery的其他方面，也是以这种问题描述+解决方案的方式进行讲解。所有示例中用到的代码及每个步骤，都会通过屏幕截图进行透彻的讲解。如果你对HTML、CSS和jQuery稍有了解，那么这本书正是为你而准备的。因为本书涵盖了利用jQuery展开工作时可能遇到的大多数问题。

读者对象

本书是为Web开发人员、专业人士、培训讲师和学生准备的。在把某些功能应用到网页时经常遇到的问题，大部分都能在本书中快速地找到解决方案。

从本书能学到什么

- 把效果应用到段落和列表
- 设置布局
- 事件处理
- 表单验证
- 页面导航
- 视觉效果
- 表格处理
- Ajax
- 使用插件

致　谢

我要向编辑部主任助理 Steve Anglin 道一声感谢，他对本书进行初步验收，并给予我创作本书的机会。非常感谢在创作本书过程中 Apress 出版团队的精诚合作和贡献。

感谢责任编辑 Matthew Moodie，他反馈了大量重要的信息以帮助改善本书。他在本书的结构和质量改善上起到了重要作用。

我要感谢技术编辑 Massimo Nardone，感谢他出色而细致地审读了本书，提出了许多很有帮助的意见和建议。

特别感谢文字编辑 Candace English，感谢她在结构和文字方面一流的编辑水平，感谢她对提升本书内容和润色文字作出的努力。

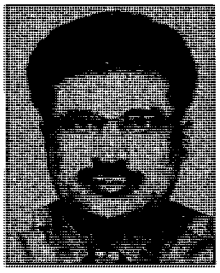
十分感谢协调编辑 Kelly Moritz，她为整个团队的协作付出了真诚的努力，使本书得以按时出版。

深深地感谢 Apress 出版社的编辑和制作人员，以及为出版本书付出努力的整个团队。与你们每个人一起工作都很愉快。

我也感谢我的家庭——我的小世界：Anu（我的妻子）、Chirag 和 Naman（我的两个小宝贝），给我时间编著本书，虽然我本应该陪他们的。

不应该忘记感谢我亲爱的同学们，他们也是我的“好老师”。他们让我了解了在面对新课题时学生们遇到的基本问题，使我能够直截了当地处理那些问题。学生们无穷而有趣的提问让我能够切实可行和有所侧重地编写本书。

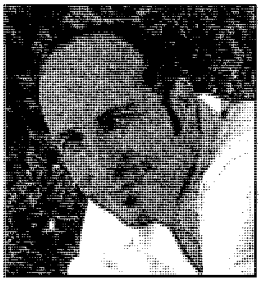
关于作者



B.M.Harwani 是印度阿杰梅尔的微芯计算机教育中心（MCE）总经理。他毕业于 Pune 大学的计算机工程系获得学士学位，拥有印度政府机构 DOEACC 颁发的 C 级证书（计算机硕士文凭）。

在教学领域工作的 15 年多来，他开发出了富有艺术性的教学方法，以人人都能懂的方式去解释哪怕最复杂的课题。他成功地编写了多本图书，包括 *Programming & Problem Solving through 'C'*（BPB, 2004）、*Learn Tally in Just Three Weeks*（Pragya, 2005）、*Data Structures and Algorithms through C*（CBC, 2006）、*Master Unix Shell Programming*（CBC, 2006）、*Business Systems*（CBC, 2006）、*Practical Java Project*（Shroff, 2007）、*Practical Web Services*（Shroff, 2007）、*Java for Professionals*（Shroff, 2008）、*C++ for Beginners*（Shroff, 2009）、*Practical ASP.NET 3.5 Projects*（Shroff, 2009）、*Java Server Faces - A Practical Approach for Beginners*（PHI Learning, 2009）、*Practical JSF Project using NetBeans*（PHI Learning, 2009）、*Foundation Joomla*（friendsof ED, 2009）和 *Practical EJB Project*（Shroff, 2009）。他还撰写了有关计算机学科的广泛课题的文章。想要了解更多，就请访问 http://bmharwani.com。

关于技术审稿人



Massimo Nardone 出生在维苏威火山下，拥有意大利萨勒诺大学颁发的计算机科学硕士学位。他目前是 IT 安全和基础设施高级架构师、芬兰 IBM 的芬兰发明开发小组组长。他是领导级的 IT 架构师，主要负责 IT 安全事务，广泛涉及 IT 基础设施、安全审计和评估、PKI / WPKI、安全隧道、LDAP 安全和智能卡安全。

Nardone 在移动、安全和 Web 技术领域有着超过 15 年的工作经验，在国内或国际项目中担当过项目经理、软件工程师、研发工程师、首席安全架构师和软件专家等职位。他还担任赫尔辛基理工大学的客座讲师（讲授“通信协议安全”课程）和网络实验室的实验总监。Nardone 熟谙安全通信协议的测试工具和方法，并曾利用多种编程语言开拓性地开发因特网和移动应用。

Massimo Nardone 在多个不同的 IT 领域担任技术审稿人，包括安全、互联网和数据库技术。他研究、设计和实现了安全工具和方法，涉及 Standard BS7799、PKI /WPKI、Security Java (JAAS、JSSE、JCE 等)、BEA Web Logic Security、J2EE Security、LDAP Security、SSO、Apache Security、MS SQL Server Security、XML Security 和 SmartCard Security。Nardone 目前拥有 PKI、SIP、SAML 和 Proxy 领域的 4 个国际专利。

计算机精品学习资料大放送

软考官方指定教材及同步辅导书下载 | 软考历年真是解析与答案

软考视频 | 考试机构 | 考试时间安排

Java 一览无余：Java 视频教程 | Java SE | Java EE

.Net 技术精品资料下载汇总：ASP.NET 篇

.Net 技术精品资料下载汇总：C#语言篇

.Net 技术精品资料下载汇总：VB.NET 篇

撼世出击：C/C++编程语言学习资料尽收眼底 电子书+视频教程

Visual C++(VC/MFC)学习电子书及开发工具下载

Perl/CGI 脚本语言编程学习资源下载地址大全

Python 语言编程学习资料(电子书+视频教程)下载汇总

最新最全 Ruby、Ruby on Rails 精品电子书等学习资料下载

数据库精品学习资源汇总：MySQL 篇 | SQL Server 篇 | Oracle 篇

最强 HTML/xHTML、CSS 精品学习资料下载汇总

最新 JavaScript、Ajax 典藏级学习资料下载分类汇总

网络最强 PHP 开发工具+电子书+视频教程等资料下载汇总

UML 学习电子资下载汇总 软件设计与开发人员必备

经典 LinuxCBT 视频教程系列 Linux 快速学习视频教程一帖通

天罗地网：精品 Linux 学习资料大收集(电子书+视频教程) Linux 参考资源大系

Linux 系统管理员必备参考资料下载汇总

Linux shell、内核及系统编程精品资料下载汇总

UNIX 操作系统精品学习资料<电子书+视频>分类总汇

FreeBSD/OpenBSD/NetBSD 精品学习资源索引 含书籍+视频

Solaris/OpenSolaris 电子书、视频等精华资料下载索引

目　录

计算机精品学习资料大放送

软考官方指定教材及同步辅导书下载 | 软考历年真是解析与答案

软考视频 | 考试机构 | 考试时间安排

Java 一览无余：Java 视频教程 | Java SE | Java EE

.Net 技术精品资料下载汇总：ASP.NET 篇

.Net 技术精品资料下载汇总：C#语言篇

.Net 技术精品资料下载汇总：VB.NET 篇

撼世出击：C/C++编程语言学习资料尽收眼底 电子书+视频教程

Visual C++(VC/MFC)学习电子书及开发工具下载

Perl/CGI 脚本语言编程学习资源下载地址大全

Python 语言编程学习资料(电子书+视频教程)下载汇总

最新最全 Ruby、Ruby on Rails 精品电子书等学习资料下载

数据库精品学习资源汇总：MySQL 篇 | SQL Server 篇 | Oracle 篇

最强 HTML/xHTML、CSS 精品学习资料下载汇总

最新 JavaScript、Ajax 典藏级学习资料下载分类汇总

网络最强 PHP 开发工具+电子书+视频教程等资料下载汇总

UML 学习电子资下载汇总 软件设计与开发人员必备

经典 LinuxCBT 视频教程系列 Linux 快速学习视频教程一帖通

天罗地网：精品 Linux 学习资料大收集(电子书+视频教程) Linux 参考资源大系

Linux 系统管理员必备参考资料下载汇总

Linux shell、内核及系统编程精品资料下载汇总

UNIX 操作系统精品学习资料<电子书+视频>分类总汇

FreeBSD/OpenBSD/NetBSD 精品学习资源索引 含书籍+视频

Solaris/OpenSolaris 电子书、视频等精华资料下载索引

第1章

jQuery 基础知识

本章将讲述 jQuery 的基础知识，从 jQuery 的安装到操作 DOM 节点。这些基本的“攻略”将有助于刷新你的记忆和填补你的知识空白。本章将涵盖以下攻略：

- 安装 jQuery；
- 选择 DOM 节点；
- 延迟 JavaScript 的执行；
- 把 CSS 应用到元素上；
- 选择一系列非标准的 HTML 元素；
- 计数 DOM 节点的数目和显示其文本；
- 获得一个元素的 HTML；
- 更改一个 DOM 节点的内容；
- 快速创建 DOM 节点；
- 为不同的 HTML 元素分配相同的类名，并对它们应用相应的样式。

注意 第 10 章将更深入地讲解 CSS。为了完善你的 jQuery 代码，我们将讲述 CSS 技术。由于 CSS 是一种基本的 jQuery 技能，我们还将给出一些基本的 CSS 攻略。

1.1 jQuery 的安装

问题描述

请你安装 jQuery，这样就可以在应用程序中使用 jQuery，并照着本书中的攻略“依葫芦画瓢”。

解决方案

jQuery 其实是一个.js（JavaScript）文件。可以轻而易举地下载 jQuery 并安装到 Web 应用程序中。可从官方网站 http://jquery.com/下载 jQuery 的最新版本。本书使用的 jQuery 版本是 1.3.2。你想在哪里写 jQuery 程序，就把下载的文件复制到哪个文件夹。

把加载jQuery库的声明写在HTML文档的head标记里，这个声明如下所示：

```
<script src="jquery-1[1].3.2.js" type="text/javascript"></script>
```

在几乎所有的攻略中，我们将在HTML文档中添加以下3行代码（在head标记里）：

```
<link rel="stylesheet" href="style.css" type="text/css" />
<script src="jquery-1[1].3.2.js" type="text/javascript"></script>
<script src="d1.js" type="text/javascript"></script>
```

知其所以然

第一行是用来把外部的样式表文件style.css（文件名随意）链接到HTML文档。第二行是加载jQuery库，第三行是指定JavaScript文件名（在这里是d1.js，但它可以是任何文件名），其中包含了来自本书攻略的jQuery代码。

1.2　选择DOM节点

问题描述

请利用jQuery来访问DOM（文档对象模型）元素，以便操作那些元素。

解决方案

jQuery利用CSS选择器来访问DOM元素。当利用选择器来访问DOM中元素的时候，元素以jQuery对象的形式被返回。使用CSS选择器来访问DOM中元素时，要使用$()函数。

使用jQuery代码来选择一个段落，典型的JavaScript脚本如下所示：

```
//d1.js
$(document).ready(function() {
  $('p').addClass('highlight');
});
```

知其所以然

$()函数用于选择文档的一部分。它接受任何CSS选择器表达式、0个或多个DOM节点，返回所有匹配的元素，以便我们操作这些元素，改变其外观。

下面是一些示例。

- $('p')——访问HTML文档中所有的段落元素
- $('div')——访问HTML文档中所有的div元素
- $('#A')——访问所有id=A的HTML元素
- $('.b')——访问所有class=b的HTML元素

在下面的攻略中，我们将选择所有的段落元素，并对其调用addClass()方法。这将把CSS类highlight应用到文档中所有的段落。

1.3 延迟 JavaScript 的执行

问题描述

在 HTML 文档 head 部分中引用了 JavaScript 文件。一旦浏览器发现脚本行，就会执行该 JavaScript 文件。但样式无法应用于 HTML 元素（因为 JavaScript 文件在 head 部分中被引用，而 HTML 元素在 body 部分中才出现，此时尚未加载）。因此，我们需要延迟 JavaScript 代码的执行，直到 DOM 加载完毕。

解决方案

用来通知我们 DOM 准备就绪的方法是$(document).ready()。在 DOM 加载完成后，此方法执行函数调用（函数作为它的参数）：

```
$(document).ready(function() {
  $('p').addClass('highlight');
});
```

知其所以然

我们使用关键字 function（不带函数名）来定义函数体。函数体在 DOM 加载之后被调用。为什么函数体被用作方法的参数？很简单，因为我们希望函数立即执行，且只执行一次。我们不希望该函数体被再次使用。换句话说，$(document).ready()给文档注册了一个 ready（就绪）事件。

上面解决方案的$('p')是一个选择器，用于访问 HTML 文档的所有段落元素，而 addClass() 方法将把指定的 CSS 类应用于这些元素。

1.4 把 CSS 应用到元素上

问题描述

应用一个 CSS 类到网页的某些元素上。

解决方案

addClass()方法用于应用一个 CSS 类到网页的选中部分。它包含类的名称，作为方法的参数。通过在单独的样式表中定义 CSS 类的样式规则，这个方法用起来就非常容易。在样式表中给不同的 CSS 类编写样式规则，如下所示：

```
.highlight {
  font-style: italic;
  background-color: #0f0;
}
```

以下是应用CSS类的代码：

```
$('div').addClass('highlight');
$('body').addClass('highlight');
```

知其所以然

在第一行jQuery代码中，$('div')选择HTML文档的所有div元素，并对这些元素应用highlight类的样式规则。在第二行中，选择器$('body')访问HTML文档的body元素，然后为该元素添加CSS类highlight。

1.5　选择一系列非标准的HTML元素

问题描述

选择非常规的HTML元素，比如包含一段给定文本的HTML元素、在序列中特定位置的元素、在HTML文档中的奇数或偶数元素。

解决方案

要选择非常规的HTML元素，我们需要使用自定义选择器。自定义选择器帮助我们选择HTML元素组，比如包含一段给定文本的HTML元素、在特定位置的元素（例如在第三段落）。自定义选择器还可以选择HTML文档中的奇数或偶数元素。

以下是如何选择包含文本Life的所有元素：

```
$('span:contains(Life)').addClass('highlight');
```

在下面的示例中，我们给HTML文档中的偶数和奇数div元素、给定序号的一个段落分别应用不同的样式：

```
$('div:odd').addClass('highlight');
$('div:even').addClass('boundary');
$('p:eq(1)').addClass('linkstyle');
```

知其所以然

第一个示例选择包含单词Life的span元素，对其应用CSS类highlight。在第二个示例中:odd和:even是jQuery的自定义选择器，帮助选择所需元素。在JavaScript中从0开始编号，0是偶数，1是奇数，依此类推。上面第一个示例选择了奇数div元素（编号为1、3……）并对奇数元素应用CSS类highlight：

```
$('div:odd').addClass('highlight');
```

同样，第二个示例选择了偶数div元素（编号为0、2……）并应用CSS类boundary：

```
$('div:even').addClass('boundary');
```

最后，第三个示例使用了自定义选择器:eq来选择第二段落（因为第一个段落的编号为0），并对其应用CSS类linkstyle：

```
$('p:eq(1)').addClass('linkstyle');
```

1.6 计数 DOM 节点和显示其文本

问题描述

请通过 jQuery 访问 DOM 及其节点。

解决方案

在 DOM 中，网页被表示为一个根节点（亲）和一些分支节点（子）的树结构形式，其中每个 HTML 元素被表示为一个节点。通过利用 jQuery 你可以随心所欲地访问和操纵这些节点。

请看下面的 HTML 页面：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title>JQuery Examples</title>
    <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
    <script src="d1.js" type="text/javascript"></script>
  </head>
  <body>
    <div id="root">
      <div>Darjeeling</div>
      <div>Assam</div>
      <div>Kerala</div>
    </div>
  </body>
</html>
```

该 HTML 文档的前面部分包括加载 jQuery 库的<script>标记以及包括 JavaScript 文件（即包含 jQuery 代码的 d1.js）的<script>标记，还包括一个 id="root"的 div 元素。此 div 元素里的所有元素都是其子节点。也就是说，id="root"的 div 是所有在它当中定义的 div 元素的亲节点。为了计数 DOM 节点并显示其文本，我们编写 jQuery 代码如下：

```
$(document).ready(function() {
  var $nodes = $('#root').children();
  alert('Number of nodes is '+$nodes.length);
  var txt="";
  $('#root').children().each( function() {
    txt+=$(this).text();
  });
  alert(txt);
});
```

知其所以然

以上代码访问 id="root"的 div 的所有子节点，并分配给变量$nodes。利用 alert 语句显

示子节点数组的长度。然后利用 each()方法逐个访问 id="root"的 div 的所有子节点。使用 text()方法来访问并连接每个子节点的文本，并存储在字符串变量 txt 中。最后通过 alert()方法显示所有子节点的文本。下面逐一讲解以上 jQuery 代码示例用到的方法。

1. children()

children()是遍历树的方法，它搜索指定元素的直接子节点，并返回一个新的 jQuery 对象。此方法仅在 DOM 树中向下遍历一层。在该示例中，此方法访问根 div 元素中定义的所有 DOM 节点，并将它们分配给变量$nodes。$nodes 是一个 jQuery 对象，它包含根 div 元素中定义的 3 个 div 元素。利用$nodes 对象的 length 属性，显示该 HTML 文档中 DOM 节点（作为 id="root"的 div 的子节点）的数目。

2. each()

each()方法用于循环访问包装集合中的每个元素。它包含一个迭代函数。在迭代函数中，我们编写代码，应用于集合的每个单独元素。

3. text()

text()是 jQuery 对象的一个方法，用于访问指定元素的文本内容。它合并指定元素的文本内容，并以字符串的形式返回。为了查看一个段落元素的文本内容，我们编写 jQuery 代码如下：

```
alert($('p').text());
```

假设该段落元素的内容如下所示：

```
<p>Styles make the formatting job much easier and more efficient. To give an attractive
look to web sites, styles are heavily used.
<span>jQuery is a powerful JavaScript library that allows us to add dynamic elements
to our
web sites. </span>Not only is it easy to learn, but it's easy to implement too.<br>
<a href="a1.htm"> jQuery Selectors</a> are used for selecting the area of the document
where we want to apply styles </p>
```

以上 jQuery 代码的输出如图 1-1 所示。

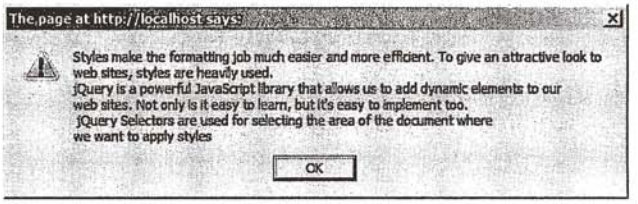


图 1-1　HTML 文档的段落元素的文本内容

通过使用如下语句，可以看到段落元素的子元素的文本内容：

```
$(document).ready(function() {
  alert($('p').children().text());
});
```

4. parent()

parent()是遍历树的方法，用于搜索每个指定元素的直接亲元素，并返回一个新的 jQuery

对象。此方法仅在 DOM 树中向上遍历一层。为了获得 span 元素的亲元素的文本内容，我们编写 jQuery 代码如下：

```
alert($('span').parent().text());
```

1.7 获得一个元素的 HTML 代码

问题描述

请显示指定元素的 HTML 代码。

解决方案

假设 HTML 文档包含下面示例的段落元素：

```
<p>Styles make the formatting job much easier and more efficient. To give an attractive
look to web sites, styles are heavily used.
<span>jQuery is a powerful JavaScript library that allows us to add dynamic elements
to our web sites. </span>Not only is it easy to learn, but it's easy to implement too.<br>
<a href="a1.htm"> jQuery Selectors</a> are used for selecting the area of the document
where we want to apply styles </p>
```

显示该段落元素的 HTML 代码的 jQuery 代码如下：

```
$(document).ready(function() {
  alert($('p').html());
});
```

知其所以然

以上代码访问段落元素的内容，利用 html()方法显示其 HTML 代码。html()方法从指定元素中的第一个元素获取 HTML 内容。它以字符串形式返回 HTML 内容。html()和 text()的区别是 text()方法可用于 XML 文档和 HTML 文档，而 html()只能用于 HTML 文档。另一个区别是 html()方法不仅显示文本，还连同文本一起显示标记。

输出如图 1-2 所示。可以看到，输出包括标记和文本。

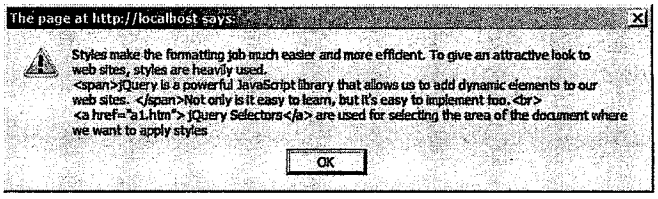


图 1-2 一个 HTML 文档的段落元素的 HTML 内容

为了获取 span 元素的 HTML 内容，我们使用以下语句：

```
alert($('span').html());
```

为了获取 span 元素的亲元素的 HTML 内容，我们使用以下 jQuery 代码：

```
alert($('span').parent().html());
```

1.8　改变 DOM 节点的内容

问题描述

请动态地改变一个 DOM 节点的内容。假设某 HTML 文档中的 h2 元素显示特定文本，请你使用 jQuery 代码改变该 h2 元素所显示的文本。

解决方案

下面是一个 HTML 文档，包含段落元素和 h2 元素。该段落元素包含两个子元素，即 span 元素和 anchor（锚）元素。该 HTML 文档如下所示：

```
<body>
<p>Styles make the formatting job much easier and more efficient. To give an attractive
look to web sites, styles are heavily used.
<span>jQuery is a powerful JavaScript library that allows us to add dynamic elements
to our web sites. </span>Not only is it easy to learn, but it's easy to implement too.<br/>
<a href="a1.htm"> jQuery Selectors</a> are used for selecting the area of the document
where we want to apply styles </p>
<h2>Using jQuery</h2>
</body>
```

以上 HTML 文档没有应用任何 jQuery 代码，执行之后原始输出如图 1-3 所示。

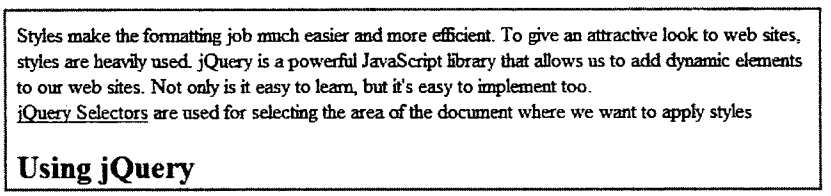


图 1-3　h2 元素的原始文本内容

可以看到，目前 h2 元素显示了文本 Using jQuery。这不是我们想要的，请看解决此问题的两个方法，即利用 jQuery 代码改变其内容。

1. text(text)

下面利用 text(text)方法快速改变 h2 元素的内容。

```
$(document).ready(function() {
  $('h2').text('JavaScript Libraries');
});
```

2. html(HTML)

.html(HTML)方法类似于 text(text)方法。此方法为指定的元素设置 HTML 内容。

为了设置 h2 元素的 HTML 内容，我们使用以下语句：

```
$('h2').html('JavaScript Libraries');
```

知其所以然

text(text)方法把指定元素的内容设置为指定的文本，并返回该元素作为新的jQuery对象。以上 jQuery 代码使用 text(text)把 HTML 文档中所有 h2 元素的文本内容设置为 JavaScript Libraries。由于该 HTML 文档中只有一个 h2 元素，该元素的文本内容将被替换，输出如图 1-4 所示。

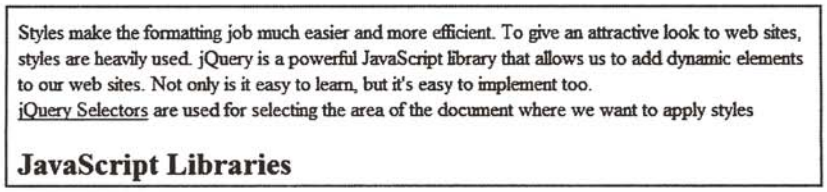


图 1-4 使用.text(text)方法，设置 h2 元素的文本内容

与 text(text)方法不同的是，html(HTML)方法还可以指定包含文本的 HTML 标记。例如，为了改变段落元素的内容，我们编写 jQuery 代码如下：

```
$('p').html('<b>We can create Rich Internet Applications </b><br/>by making AJAX
requests');
```

示例中的 html(HTML)访问 HTML 文档的段落元素，将其内容设置为两行，一行显示加粗文本 We can create Rich Internet Applications，另一行显示普通文本 by making AJAX requests。换句话说，段落元素的原始内容被这两行所取代。输出如图 1-5 所示。

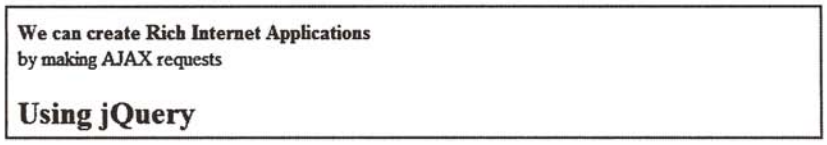


图 1-5 为段落元素设置 HTML 内容

1.9 快速创建 DOM 节点

问题描述

请动态创建一个 DOM 节点。本攻略将快速创建一个 h2 元素，并在 HTML 文档中现有段落元素之前添加该元素。

解决方案

HTML 文档中包括段落元素和 h2 元素，如下所示：

```
<body>
<p>Styles make the formatting job much easier and more efficient. To give an attractive
```

```
look to web sites, styles are heavily used.
<span>jQuery is a powerful JavaScript library that allows us to add dynamic elements
to our web sites. </span>Not only is it easy to learn, but it's easy to implement too.<br/>
<a href="a1.htm"> jQuery Selectors</a> are used for selecting the area of the document
where we want to apply styles </p>
<h2>Using  jQuery </h2>
</body>
```

解决方案中用于创建 DOM 节点的方法是 prepend()、prependTo()和 clone()。其他可用于快速创建 DOM 节点的方法有 append()、appendTo()、before()、insertBefore()、after()和 insertAfter()。让我们先看 prepend()方法。

1. prepend()

此方法在指定元素前面插入指定的内容，并返回一个 jQuery 对象。内容可以是文本、HTML 元素或 jQuery 对象。

在段落元素之前插入一个 h2 元素，其文本为 Power of selectors。编写 jQuery 代码如下所示：

```
$('p').prepend('<h2> Power of selectors </h2>');
```

2. prependTo()

与 prepend()类似，prependTo()也用于动态添加 DOM 节点。它在指定目标之前插入指定元素。目标可以是如下形式：HTML 元素、字符串或 jQuery 对象。该方法返回一个 jQuery 对象。以下代码执行结果与调用 prepend()方法的上一示例相同。

```
$('<h2> Power of selectors </h2>').prependTo('p');
```

本示例中要插入的内容（即 h2 元素）处于方法之前。

3. clone()

为了快速添加 DOM 节点（该节点是已有元素的副本），我们使用 clone()方法。此方法复制指定元素并返回一个新的 jQuery 对象。复制一个 h2 元素，在段落元素之前插入该段元素，jQuery 代码如下所示：

```
$('h2').clone().prependTo('p');
```

知其所以然

前一个 jQuery 代码示例中 prepend()访问 HTML 文档的段落元素，并在其前面插入一个包含文本 Power of selectors 的 h2 元素。输出如图 1-6 所示。

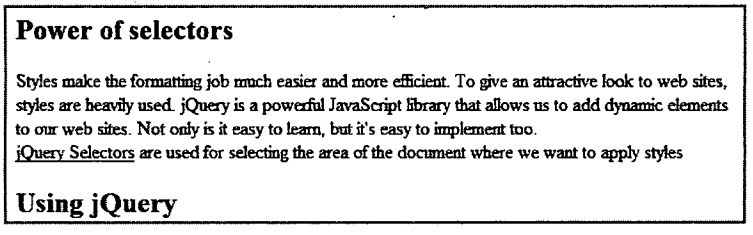


图 1-6　在段落元素前面快速添加 DOM 节点（h2 元素）

注意 如果使用 prepend()方法，则插入内容也出现在元素的开头。换句话说，如果我们把样式应用于段落元素，该样式也将被应用于 h2 元素。

为了验证是否样式被应用于合并的段落元素和 h2 元素，我们新建外部样式表文件，命名为 style.css，并编写 CSS 类如下：

```
.highlight {
font-style: italic;
background-color: #0f0;
}
```

以上样式规则使文本显示为斜体，并将文字的颜色变为绿色。在段落元素前插入一个 h2 元素，把样式应用于段落元素。jQuery 代码如下所示：

```
$('p').prepend('<h2> Power of selectors </h2>');
$('p').addClass('highlight');
```

我们看到，jQuery 代码访问 HTML 文档的段落元素，并在其前面插入一个包含文本 Power of selectors 的 h2 元素。接下来，把 CSS 类 highlight 应用于段落元素（该段落元素包括新插入的 h2 元素）。输出如图 1-7 所示。

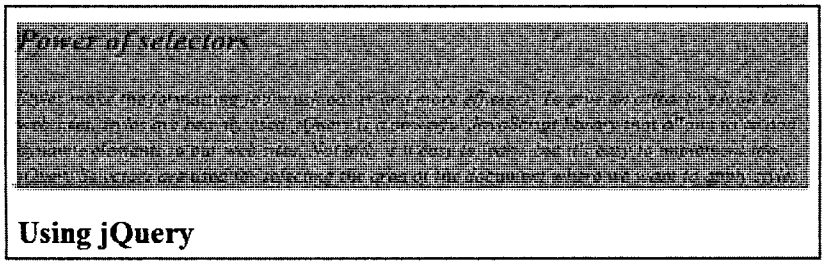


图 1-7 使用 prepend()方法插入内容，把样式应用于该结合体

prependTo()示例与前面的 prepend()示例非常相似。prepend()和 prependTo()的区别是，内容和目标刚好互换。prepend()方法，目标元素（即在其前面添加某些内容）在该方法之前，而 prependTo()方法则是添加的内容在该方法之前。

最后，jQuery 代码 clone()访问 HTML 文档的 h2 元素并复制副本。然后在段落元素之前插入该副本。输出如图 1-8 所示。

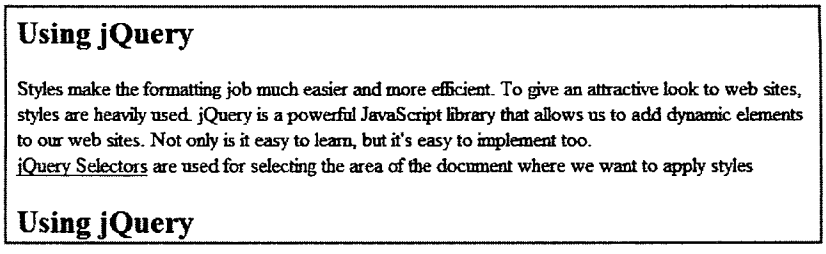


图 1-8 在段落元素前面插入 h2 元素的副本

1.10　为不同 HTML 元素分配相同类名并对它们应用样式

问题描述

请为两个 HTML 元素分配相同的类名，并为它们应用样式。例如一个段落元素和一个 h1 元素。

解决方案

请看下面的 HTML 文档，其中类名 features 被分配给段落元素和 h1 元素：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
        <title>JQuery Examples</title>
        <link rel="stylesheet" href="style.css" type="text/css" media="screen" />
        <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
        <script src="d1.js" type="text/javascript"></script>
  </head>
<body>
<p class="features">Styles make the formatting job much easier and more efficient.</p>
To give an attractive look to web sites, styles are heavily used.
<h1 class="features">Using jQuery</h1>
</body>
</html>
```

为了把样式应用于以上 HTML 文档中包含类 features 的元素，创建外部样式表（style.css）文件如下：

```
.features{color:green;font-style:italic}
```

如果想要通过 jQuery 代码把样式规则应用于 HTML 元素（非自动），我们需要在样式表中为样式规则分配别的名称。

```
.highlight{color:green;font-style:italic}
```

然后我们需要编写 jQuery 代码如下：

```
$('.features').addClass('highlight');
```

知其所以然

在 HTML 文档中，段落元素和 h1 元素都包含类 features。在以上样式表中，样式规则拥有选择器.highlight，意味着在此规则中定义的属性将应用于所有包含类 highlight 的 HTML 元素。该样式规则定义了两个属性，color 和 font-style，分别应用绿色和斜体样式。

以上 jQuery 代码将为包含类名 features 的所有 HTML 元素设置 CSS 类 highlight。输出如图 1-9 所示。

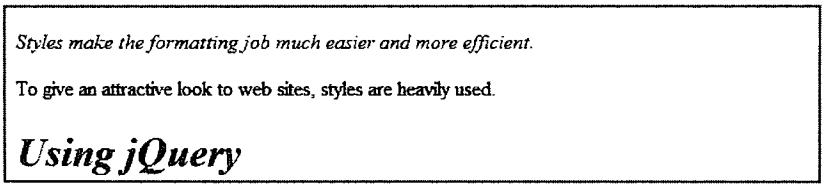


图 1-9 相同的类被应用于<p>和<h1>标记

1.11 小结

在本章中，我们学习了如何安装 jQuery、选择 DOM 节点以及把 CSS 应用于元素。我们还学习了如何选择非标准系列的 HTML 元素、计算 DOM 节点的数量并显示其文本。我们学会了获取元素的 HTML 代码的方法、改变 DOM 节点的内容、快速创建一个 DOM 节点和为不同的 HTML 元素分配同一个类名，并为它们应用样式。下一章我们将学习处理数组的攻略，以及学习如何筛选、排序、拆分和操作数组。

第2章

数组和字符串

本章将讲述在 jQuery 中如何处理数组、映射和字符串。我们将学习以下攻略：

- 利用数组在列表中显示名字；
- 操作数组元素；
- 筛选数组，只显示所需数据；
- 为字符串和数值数组排序；
- 拆分数组；
- 合并数组；
- 把数值数组转换为字符串，查找子字符串；
- 创建对象数组；
- 为对象数组排序。

2.1 利用数组在列表中显示名字

问题描述

请利用数组来显示名字列表。也就是说，假设有一个名字数组，请在网页上显示其元素。

解决方案

创建 HTML 文件，在名字列表的上方显示标题。该文件还包含一个空的段落元素，如下所示：

```
<body>
<h3> Members of my Group are </h3>
<p></p>
</body>
```

下面为段落元素添加数组中包含的名字。jQuery 代码如下：

```
$(document).ready(function() {
  var members = [ "John", "Steve", "Ben", "Damon", "Ian" ];
  $('p').text(members.join(", "));
});
```

1. 纵向显示名字

也可以纵向显示名字，jQuery 代码如下：

```
$(document).ready(function() {
  var members = [ "John", "Steve", "Ben", "Damon", "Ian" ];
  $('p').html(members.join("<br/>"));
});
```

2. 显示名字及其在数组中的位置

为了显示名字及其在数组中的位置，需要修改 HTML 文件，包含空的有序列表元素，如下所示：

```
<body>
<h3> Members of my Group are </h3>
<ol id="list">
</ol>
</body>
```

从数组中获取名字并追加到有序列表的 jQuery 代码如下：

```
$(document).ready(function() {
  var memlist = $( "#list" );
  var members = [ "John", "Steve", "Ben", "Damon", "Ian" ];
  $.each(members,function( index, value ){
    memlist.append($( "<li>" + value + "</li>" ));
  });
});
```

3. 利用HTML元素创建数组和计算数组长度

在这个解决方案中，假设名字列表以 `li` 元素形式存在于 HTML 文件中。获取所有存放在 `li` 元素的名字并保存到数组中，然后查找并显示数组中的名字的数目。包含 `li` 元素名字的 HTML 文件如下所示：

```
<body>
<p></p>
<ul>
  <li>John</li>
  <li>Steve</li>
  <li>Ben</li>
  <li>Damon</li>
  <li>Ian</li>
</ul>
</body>
```

在 `ul` 元素前面的段落元素用于显示名字的数目。目前，它没有包含文本。利用 jQuery 代码为它分配文本，与在 `li` 元素中名字的数目一起显示：

```
$(document).ready(function() {
  var names = $("li").get();
  $('p').text("Following are the " + names.length + " members of my Group");
});
```

知其所以然

第一个解决方案中，创建的 names 数组包含 5 个名字：John、Steve、Ben、Damon 和 Ian。利用 text()方法把所有的数组元素分配到段落元素。元素之间使用 join()方法以逗号（,）连接。换句话说，数组中的所有名字以逗号分隔，并分配到段落元素。运行结果是在网页上显示所有的名字（以逗号分隔），如图 2-1 所示。

为了纵向显示名字，在使用 join()方法时，在数组元素之间使用换行符
而不是逗号。此外，为了把
元素解析为 HTML 标签和而不是文本，需要使用 html()方法而不是 text()方法来为段落元素分配数组元素。否则，
元素将原原本本地显示在屏幕上，而不是创建换行符。

```
$('p').html(members.join("<br/>"));
```

图 2-2 显示了数组元素如何纵向显示：

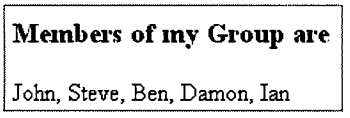


图 2-1　在一行中（横向）显示数组中的名字

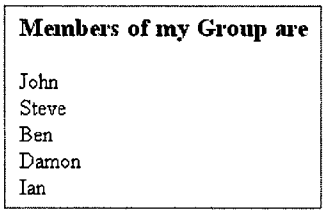


图 2-2　数组中的名字之间以换行符分隔显示

在有序列表的解决方案中，为有序列表分配了 id 为 list。数组中的名字正是追加到这个列表中。回忆一下，有序列表为列表项自动显示序号。换句话说，为了给名字（存储在数组中）自动分配序号，将从数组中一次提取一个名字，并将其附加到有序元素 ol：

```
var memlist = $( "#list" );
var members = [ "John", "Steve", "Ben", "Damon", "Ian" ];
$.each(members,function( index, value ){
  memlist.append($( "<li>" + value + "</li>" ));
});
```

在以上 jQuery 代码中，把 HTML 文件中 id 为 list 的元素（有序列表元素）分配到变量 memlist。也就是说，memlist 引用 HTML 文件中的 ol 元素。定义名为 members 的数组，并将 5 个名字存储在数组中。然后利用 each()方法提取数组中的每个元素（名字）并追加到 ol 元素。

运行结果是分配了序号的名字列表，如图 2-3 所示。

最后那个示例从 div 元素提取名字并存入数组中。以下语句用 get()方法提取所有 div 元素的文本，并存储在名为 names 的数组中，如下所示：

```
var names = $("div").get();
```

以下语句将文本分配到段落元素，并插入 names 数组的元素个数：

```
$('p').text("Following are the " + names.length + " members of my Group");
```

运行结果是显示名字的个数和名字列表，如图 2-4 所示。

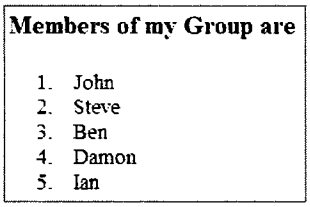


图 2-3 以列表项形式显示数组元素

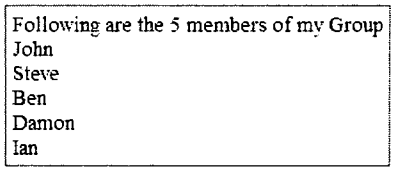


图 2-4 数组的长度即是 div 元素的个数（数组成员的个数）

2.2 操作数组元素

问题描述

操作数组元素，比如为数组元素应用序号、将数组元素转换为大写以及其他类似的任务。

解决方案

假设 HTML 文件中包含标题元素（显示消息 Members of my Group are）和空的段落元素，如下所示：

```
<body>
<h3> Members of my Group are </h3>
<p></p>
</body>
```

这个空白的段落元素将连同序号一起显示数组中的名字。连同序号一起显示数组元素的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var members = [ "John", "Steve", "Ben", "Damon", "Ian" ];
  members = $.map(members, function(n,i){ return(i+1+"."+n); });
  $('p').html(members.join("<br />"));
});
```

1. 把名字转换为大写

请看如何使用回调方法操作数组元素。第一个解决方案显示了如何使用 `toUpperCase()` 方法把所有的名字转换为大写：

```
$(document).ready(function() {
  var members = [ "John", "Steve", "Ben", "Damon", "Ian" ];
  members=$.map(members, function(n,i){ return(i+1+"."+n.toUpperCase());});
  $('p').html(members.join("<br/>"));
});
```

2. 使用有序列表

以大写字母显示数组元素及其序号的另一个方法是利用有序列表元素。假设 HTML 文件包含标题元素和空的有序列表元素，如下所示：

```
<body>
<h3> Members of my Group are </h3>
<ol id="list">
</ol>
</body>
```

以大写字母显示数组中的元素，jQuery 代码如下所示：

```
$(document).ready(function() {
  var memlist = $( "#list" );
  var members = [ "John", "Steve", "Ben", "Damon", "Ian" ];
  members=$.map(members, function(n){ return(n.toUpperCase());});
  $.each(members,function( index, value ){
    memlist.append($( "<li>" + value + "</li>" ));
  });
});
```

知其所以然

要理解这个攻略，需要了解 map()方法。map()方法迭代数组的每个元素，并为每个数组元素分别调用一次回调函数。可以把返回的元素分配到另一个数组，如果你愿意，也可以分配到同一个数组。map()方法也可以遍历具有 length 属性、类似数组的对象。下面是 map()方法的语法：

```
map(array, callback);
```

此处的回调函数包含对数组元素执行处理任务的语句。第一个解决方案中，想要连同其序号一起显示存储在数组中的名字。定义 members 数组，包含想要显示的名字。接着把数组传递到 map()方法。map()方法中的回调函数包含两个参数，即 n 和 i，其中 n 指向传递到 map()方法的数组元素，i 是单独的数组元素的索引（索引从 0 开始）。为了使序号从 1（而不是从 0）开始，可以在每次迭代时给 i 加 1。因此回调函数的返回值如下所示：

```
return(i+1+"."+n)
```

以上语句逐个返回数组中的元素，索引从 1 开始。输出如图 2-5 所示。

Members of my Group are

1.John
2.Steve
3.Ben
4.Damon
5.Ian

图 2-5　使用数组映射，为数组元素分配序号

紧接着在回调函数中调用 toUpperCase()方法，把存储在数组中的所有名字转换为大写：

```
members=$.map(members,
function(n,i){ return(i+1+"."+n.toUpperCase()); });
```

回忆一下，在回调函数中参数 n 和 i 分别指向数组元素和索引。应用 toUpperCase()到 n（即以数组元素形式存储在数组中的名字），把名字转换为大写并返回，并在段落元素中显示。

运行结果是转换为大写的名字及其序号，如图 2-6 所示。

有序列表的解决方案把编号自动地应用到列表元素。map()方法把每个数组元素转换为大写，再把结果分配到 members 数组。然后把 members 数组的成员逐个地追加到有序列表（id 为 list，jQuery 代码通过 id 来识别它）。运行结果如图 2-7 所示。

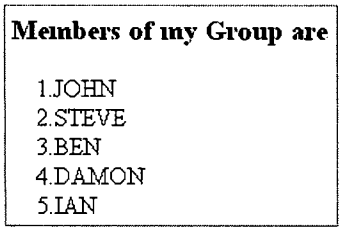


图 2-6 使用数组映射，把数组元素转换为大写

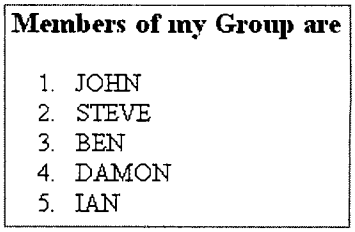


图 2-7 通过列表项显示大写的数组元素

2.3 筛选数组元素，只显示所需的数据

问题描述

假设一个数组包含了一些名字，为了只看到所需的名字，需要筛选数组元素。例如，希望只看到那些长度超过 4 个字符的名字。

解决方案

假设 HTML 文件包含两个标题元素，每个标题元素之后跟着一个段落元素。为第一个段落元素分配类名 allmem，为另一个段落元素分配类名 selected。该 HTML 文件的内容如下所示：

```
<body>
<h3> Members of my Group are </h3>
<p class="allmem"></p>
<h3> Names with more than 4 characters in length are </h3>
<p class="selected"></p>
</body>
```

使用第一个段落（类名为 allmem）来显示整个数组，即列表中所有的名字，使用第二个段落来显示筛选的名字（类名为 selected）。

1. 使用grep()

写一段 jQuery 代码，利用 grep()方法只显示长度超过 4 个字符的名字（在“知其所以然”一节中详述其工作原理）。

```
$(document).ready(function() {
  var members = [ "John", "Steve", "Ben",  "Damon","Ian" ];
  $('p.allmem').html(members.join("<br/>"));
  members = $.grep(members, function(v) { return v.length > 4});
  $('p.selected').html(members.join("<br/>"));
});
```

2. 使用match()

写一段 jQuery 代码，使用 match()方法和正则表达式，只显示数组中以字母 A 到 D 之间的任何字符打头的那些名字：

```
$(document).ready(function() {
```

```
    var members = [ "John", "Steve", "Ben",  "Damon","Ian" ];
    $('p.allmem').html(members.join("<br/>"));
    members = $.grep(members, function(v) { return v.match(/^[A-D]/)});
    $('p.selected').html(members.join("<br/>"));
});
```

知其所以然

第一个解决方案使用 grep()方法对数组元素进行筛选。这个方法分析数组的所有元素，为每个元素分别调用回调函数。在回调函数中编写语句，把不想要的元素过滤掉，也就是说，回调函数只返回被过滤的数组中我们想要的值。语法如下所示：

```
grep(array, callback, boolean)
```

grep()方法的参数如下：

- array 是原始数组，grep()方法对原始数组进行筛选。
- callback 是回调函数，执行筛选任务，返回构成过滤数组的值。回调函数有两个参数，第一个是数组元素，第二个是索引值。
- boolean 通常省略。如果指定该参数并设置为 false（这是默认值），则没有任何效果。如果设置为 true，那么回调函数的运算则是逆向的。也就是说，它将返回与回调函数中提供的条件语句不匹配的值。

利用以上关于 grep()的知识来分析第一个解决方案。可以看到 members 是数组，包含一些名字。以下语句以换行符（
）分隔每个数组元素，并显示每个数组元素，作为第一个段落（类名为 allmem）的文本。因此，将在屏幕上显示原始数组：

```
$('p.allmem').html(members.join("<br />"));
```

以下语句利用了 grep()方法。把 members 数组传递到 grep()方法。回调函数的参数 v 代表每个数组元素：

```
members = $.grep(members, function(v) { return v.length >4});
```

可以看到，回调函数只返回长度大于 4 个字符的名字。返回的数组元素存储在原始数组 members 中（取代原始内容）。以下语句显示筛选数组的元素，作为类名为 selected 的段落的文本。为了分行显示各元素，数组元素之间以换行符（
）作为分隔符：

```
$('p.selected').html(members.join("<br/>"));
```

执行以上 jQuery 代码，运行结果如图 2-8 所示。

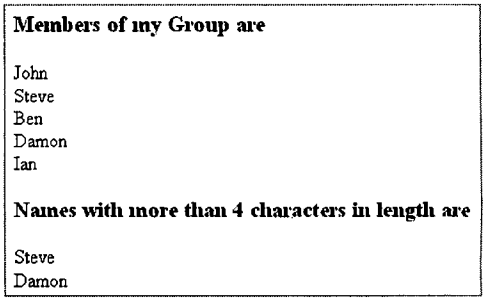


图 2-8　显示长度超过 4 个字符的数组元素

下面在 grep()方法中稍作修改，把布尔参数包括进来。该参数的作用如前所述，设置为 true 时就会反转回调函数的操作，如下所示：

```
members = $.grep(members, function(v) { return v.length >4}, true);
```

以上语句将返回数组中长度小于或等于 4 个字符的名字。

第二个解决方案使用 match()方法来定义正则表达式，在示例中匹配以 A 到 D 打头的字符串，match()是 String 类的一个方法，用于决定指定的字符串对象是否匹配指定的正则表达式。

使用正则表达式

正则表达式是用于指定一个元素中我们想要的字符模式的简明的方法，是分析和验证字符串的强大工具。

以下列出了正则表达式表示法：

字 符	匹 配	字 符	匹 配
\b	单词边界	{n,}	重复 n 次或更多次
\d	从 0 到 9 的任何一个数字	{n.m}	重复 n 次到 m 次（可以是 n 次或 m 次）
\s	单个空白字符	?	0 次或多次
\w	任何字符、数字或下划线	+	1 次或多次
.	换行符以外的任何字符	^	匹配出现在行首或字符串开始位置的空字符串
[…]	方括号中的任何一个字符	$	匹配出现在行末的空字符串
{n}	重复 n 次		

下面的语句把 members 数组传递到 grep()方法：

```
members = $.grep(members, function(v) { return v.match(/^[A-D]/)});
```

再次强调一下，回调函数包含参数 v，逐个获取数组中的元素。回调函数只返回以 A 到 D 之间的任何字符打头的数组元素。运行结果如图 2-9 所示。

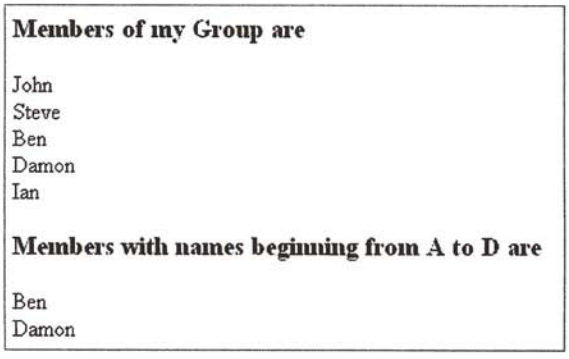


图 2-9 显示以 A 到 D 的任何字符打头的名字

理解了以上新知识点，现在可以修改 match()中的正则表达式，以便只显示以字符 n 结尾的那些名字了，如下所示：

```
members = $.grep(members, function(v) { return v.match(/[n]$/)});
```

回想一下，$用在正则表达式中，指定我们希望看到的内容出现在行尾。运行结果如图 2-10 所示。

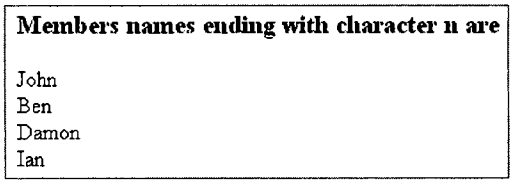


图 2-10　显示以字符 n 结尾的名字

为了查看数组中出现字符 e 一次或多次的所有名字，使用以下正则表达式：

```
members = $.grep(members, function(v) { return v.match(/[e].+/)});
```

只显示出现字符 e 一次或多次的那些名字，如图 2-11 所示。

为了查找长度正好是 5 个字符的名字，使用以下正则表达式：

```
members = $.grep(members, function(v) { return v.match(/\b.{5}\b/)});
```

运行结果如图 2-12 所示。

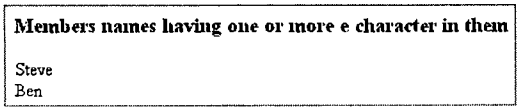


图 2-11　显示包含一个或多个字符 e 的名字

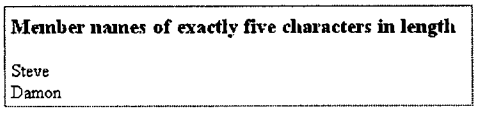


图 2-12　显示恰好是 5 个字符的名字

2.4　字符串数组和数值数组的排序

问题描述

有两个数组，一个是字符串数组，另一个是数值数组。请分别为它们排序。

解决方案

首先为字符串数组排序。

1. 字符串数组的排序

以下 HTML 文件包含两个标题元素，每个标题元素后面是一个段落元素。标题元素用来分别显示原始数组和排序数组的标题。为了区分这两个段落元素，给其分配不同的类名，allmem 和 sorted。HTML 文件的内容如下：

```
<body>
<h3> Members of my Group are </h3>
<p class="allmem"></p>
<h3> Members of my Group in sorted order </h3>
<p class="sorted"></p>
</body>
```

类名为 allmem 的段落元素用于显示原始数组元素。类名为 sorted 的段落元素用于显示排序数组元素。用于字符串数组排序的 jQuery 代码如下所示，稍后解释其工作原理：

```
$(document).ready(function() {
  var members = [ "John", "Steve", "Ben",  "Damon","Ian" ];
  $('p.allmem').html(members.join("<br />"));
  members = members.sort();
  $('p.sorted').html(members.join("<br />"));
});
```


2. 数值数组的排序

在jQuery中，数值数组排序使用不同于字符串数组排序的方式。再次新建一个包含两个标题元素和两个段落元素的HTML文件，如下所示：

```
<body>
<h3>Original numerical array is </h3>
<p class="allmem"></p>
<h3> Array in sorted order </h3>
<p class="sorted"></p>
</body>
```

与字符串的解决方案一样，类名为allmem的段落元素用来显示原始数组，类名为sorted的段落元素用来显示排序数组。

下面是对数值数组进行排序的jQuery代码：

```
$(document).ready(function() {
  var members = [45, 10,3,22,7 ];
  $('p.allmem').html(members.join("<br>"));
  members = members.sort(function(a,b){
    return a-b;
  });
  $('p.sorted').html(members.join("<br>"));
});
```

知其所以然

上一个攻略中，对字符串数组进行实际排序的语句如下所示：

```
members = members.sort();
```

以上sort()方法按字母顺序为字符串数组排序。此方法基于ASCII值进行排序，所以最好统一名字的形式。也就是说，要么都以大写开头，要么都以小写开头，不要混淆。以上语句对字符串数组members进行排序，排序的结果存储在members数组中（取代旧值）。

在调用members字符串数组的sort()方法之前，把该数组分配到类名为allmem的段落元素（进行显示），排序之后分配到类名为sorted的段落元素。运行结果如图2-13所示。

如前所述，sort()方法是基于ASCII值为元素进行字符串排序，因此不能用于数值排序，因为它会认为10小于3（因为1的ASCII值小于3的ASCII值）。如果要使用sort()方法为数值数组排序，jQuery代码如下所示：

```
$(document).ready(function() {
  var members = [45, 10, 3, 22, 7];
  $('p.a').html(members.join("<br/>"));
  members = members.sort();
  $('p.b').html(members.join("<br/>"));
});
```

运行结果如图 2-14 所示。

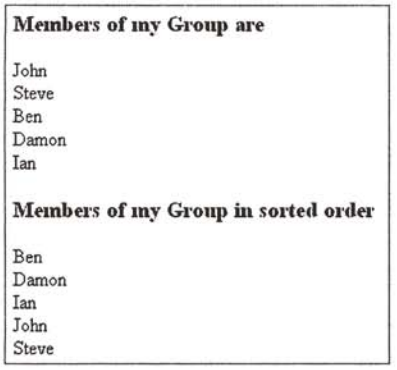


图 2-13　显示排序之后的名字

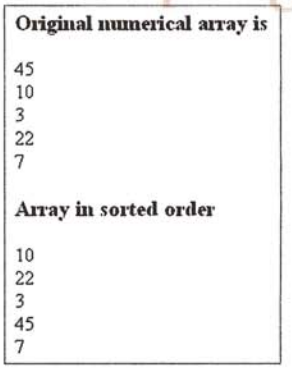


图 2-14　排序不正确的数值数组

正如你所看到的，sort()方法对数值的排序并不正确，因为它按照所有数值的左起第一位数字进行排序。因此，为了正确地对数值排序，必须为 sort()方法定义一个比较函数。

如果定义了比较函数，将把数组中的一对值发送到比较函数进行比较，重复这一操作，直到把数组的所有元素处理完毕。在比较函数中编写语句，对传入的一对值进行比较，然后返回以下 3 个值的任何一个：小于 0、等于 0 或大于 0。

- 当函数返回值小于 0，可知第二个值（发送到函数的数组的一对值）大于第一个值，因此第二个值排序靠后。
- 当函数返回值大于 0，可知第一个值大于第二个值，所以第一个值排序靠后。
- 当函数返回值等于 0，就意味着不必改变排序，因为这两个值相等。

包含了比较函数的 sort()方法，其 jQuery 代码如下所示：

```
members = members.sort(function(a,b){
  return a-b;
});
```

比较函数的以下语句将对数值进行升序排序（从小到大）：

```
return a-b;
```

运行结果如图 2-15 所示。

当然，对数值进行降序排序（从大到小），只需把比较函数的返回值从 a-b 修改为 b-a！

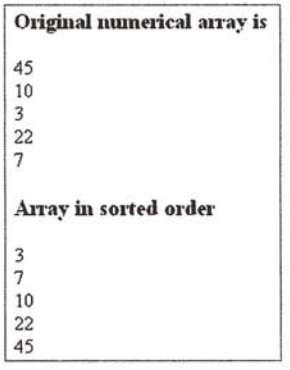


图 2-15　正确排序的数值数组

2.5　拆分数组

问题描述

假设有一个数组，需要把该数组拆分成两部分。

解决方案

以下 HTML 文件包含 3 个标题元素，分别用于为原始数组和拆分后的两部分显示标题信息。还包含 3 个段落元素，类名为 allnum、firstp 和 secondp，分别用于显示原始数组和拆分而成的两个数组的内容：

```
<body>
<h3>Original numerical array is </h3>
<p class="allnum"></p>
<h3> First piece of array </h3>
<p class="firstp"></p>
<h3> Second piece of array </h3>
<p class="secondp"></p>
</body>
```

分拆数组的 jQuery 代码如下：

```
$(document).ready(function() {
  var members = [45, 10, 3, 22, 7];
  $('p.allnum').html(members.join("<br>"));
  memsecond = members.splice(0,3);
  $('p.firstp').html(memsecond.join("<br />"));
  $('p.secondp').html(members.join("<br />"));
});
;
```

知其所以然

在 jQuery 代码中，使用 splice()方法去分拆数组。为了分拆数组，该方法需要两个参数：第一个参数指定索引位置，即从哪里开始分拆，第二个参数指定从原始数组中删除多少个元素。从原来的数组中提取并返回这两个参数所定义范围内的数组元素，可以在另一个数组中保存返回的数组元素。总之，删除并返回数组的一部分（由发送到 splice 的两个参数所定义），其余部分留在原来的数组中。

```
subarray = mainarray.splice (m,n);
```

这里的 subarray 是从 mainarray 的索引 m 开始删除 n 个元素之后所取得的数组[①]。

注意 剩余元素将留在mainarray中。

从上面的示例代码可见，数值数组 members 中定义了 5 个数字。数组中的元素以类名为 allnum 的段落文本形式显示在网页上，元素之间以换行符（
）分隔。也就是说，纵向显示所有的数组元素。

以下语句从索引位置 0 开始提取 3 个元素，将它们返回并存储在 memsecond 数组中：

```
memsecond = members.splice(0,3);
```

① 可以这样理解，subarray 是原始数组 mainarray 的一个“切片”。——译者注

不出所料，剩余部分将留在主数组 members 中。因此，memsecond 数组包含原来的 members 数组的前 3 个元素，最后两个元素留在 members 数组中。存储在 memsecond 的 3 个元素在类名为 firstp 的段落元素中显示，留在 members 数组中的两个元素在类名为 secondp 的段落元素中显示。运行结果如图 2-16 所示。

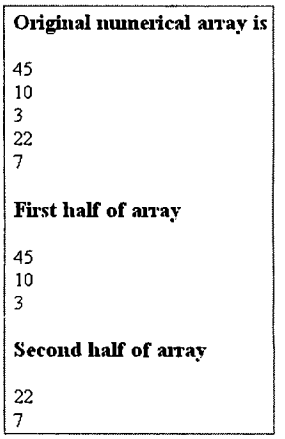


图 2-16　把数组一分为二

2.6　合并数组

问题描述

请把两个数组合二为一。

解决方案

以下 HTML 文件包含 3 个标题元素，用于显示两个数组以及合并数组（由两个数组合并而成）的标题信息。该文件还包含 3 个段落元素，类名是 firstarr、secondarr 和 combinedarr，分别用于显示两个数组的元素以及合并数组的元素：

```
<body>
<h3>First array is </h3>
<p class="firstarr"></p>
<h3> Second array is </h3>
<p class="secondarr"></p>
<h3> Array after combination </h3>
<p class="combinedarr"></p>
 </body>
```

合并两个数组的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var mem1 = [45, 10, 3];
  var mem2 = [22, 7];
  $('p.firstarr').html(mem1.join("<br/>"));
  $('p.secondarr').html(mem2.join("<br/>"));
  members = mem1.concat(mem2);
  $('p.combinedarr').html(members.join("<br/>"));
});
```

知其所以然

concat()是数组对象的方法。把数组 2 作为参数传递到数组 1，在数组 1（被连接的数组）上调用 concat()方法。返回的数组是这两个数组的连接（即两个数组的合并），并保存在第三个数组中，如下所示：

```
combinedarray = array1.concat(array2);
```

array1 调用 concat()方法，array2 作为参数传递到该方法。合并之后的数组如下：array1

的所有元素之后跟着 array2 的所有元素。在解决方案中，首先定义两个数组 mem1 和 mem2。然后，分别以类名为 firstarr 和 secondarr 的段落文本形式显示 array1 和 array2 的元素。也就是说，这两个数组的内容分别在类名为 firstarr 和 secondarr 的段落中显示。我们还看到，mem1 数组调用 concat()方法，传递 mem2 数组作为参数。合并而成的数组存储在 members 数组中。然后，在类名为 combinedarr 的段落中显示 members 数组的内容。

执行以上 jQuery 代码，运行结果如图 2-17 所示。

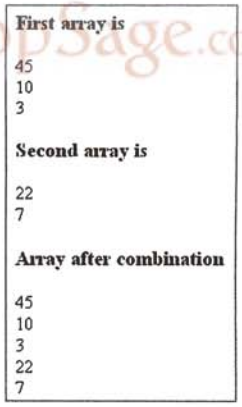


图 2-17 合并两个数组

2.7 把数值数组转换成字符串，并查找其子字符串

问题描述

假设有一个数值数组，请把它转换成一个字符串，以便应用 substr()方法获取子字符串。

解决方案

以下 HTML 文件包含 3 个标题元素，分别用于显示原始数值数组、由该数组转换而成的字符串和该字符串的子字符串的标题。此外，每个标题元素下面是一个段落元素。给这 3 个段落元素分别指定类名 origarr、arrstring 和 partstring。类为名 origarr 的段落用于显示数值数组的元素，类名为 arrstring 的段落用于显示字符串（由数值数组转换而成），而类名为 partstring 的段落用于显示我们要获取的子字符串。该 HTML 文件的内容如下所示：

```
<body>
<h3>Original array is </h3>
<p class="origarr"></p>
<h3> Array in form of string </h3>
<p class="arrstring"></p>
<h3> Substring is </h3>
<p class="partstring"></p>
</body>
```

把数值数组转换成一个字符串，并获取它的子字符串，jQuery 代码如下所示：

```
$(document).ready(function() {
  var members = [45, 10, 3, 22, 7];
  $('p.origarr').html(members.join("<br/>"));
  var str = members.join("");
  $('p.arrstring').text(str);
  var substr = str.substr(0,3);
  $('p.partstring').text(substr);
});
```

知其所以然

我们定义了包含5个元素的数值数组members，并在类名为origarray的段落元素中显示数组的元素，以换行符（
）分隔数组元素，以便纵向显示数组中的元素。

接下来把所有的数组元素逐个地联接到字符串变量str，元素之间没有任何空白，从而把数值数组members转换成为字符串。在类名为arrstring的段落元素中显示str变量。最后，从字符串变量str中获取子字符串，从索引位置0开始，提取3个字符并存储在字符串变量substr中。在类名为partstring的段落元素中显示变量substr的内容。运行结果如图2-18所示。

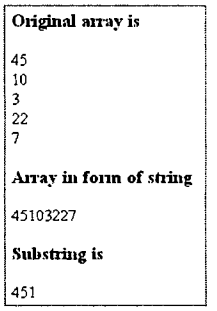


图2-18　把一个数值数组转换为字符串

2.8　创建对象数组

问题描述

请创建一个对象数组，存储关于特定的实体、项目、个人、对象等信息。在此解决方案中，要保存3个学生的资料。假设每个学生对象包含3个属性：role、name和emailId。

解决方案

创建一个HTML文件，包含标题元素和段落元素，分别显示标题和数组内容：

```
<body>
<h3>List of students is </h3>
<p class="listofstud"></p>
</body>
```

创建对象数组students，每个对象包含3个属性role、name和emailId，jQuery代码如下所示：

```
$(document).ready(function() {
  var students=[
  {
  "role": 101,
  "name": "Ben",
  "emailId":"ben@gmail.com"
  },
  {
  "role": 102,
  "name": "Ian",
  "emailId":"ian@gmail.com"
  },
  {
  "role": 103,
  "name": "Caroline",
```

```
    "emailId":"carol@gmail.com"
    }
    ];

    $.each(students,function( index, value ){
      $('p.listofstud').append(value.role+" "+value.name+" "+value.emailId);
    });
});
```

以上代码的运行结果不是那么吸引人，下面计划在表格元素中显示数组内容，属性 role、name 和 emailId 将被对齐，这样才看起来干净整洁。该 HTML 文件的内容如下所示：

```
<body>
<h3>List of students is </h3>
<table class="listofstud"></table>
</body>
```

创建一个对象数组，并分配到表格元素，jQuery 代码如下所示。除了黑体字的行以外，该文件和前面相一致：

```
$(document).ready(function() {
  ...

  $.each(students,function( index, value ){
    $('table.listofstud').append("<tr><td>"+value.role+"</td><td>"+value.name+
        "</td><td>"+value.emailId+"</td></tr>");
  });
});
```

知其所以然

在上述第一个解决方案中，创建了包含 3 个元素的数组 students。之后调用 each()方法，把回调函数应用到每一个数组元素。

在回调函数中，传递两个参数：index 和 value。参数 index 引用数组元素的索引位置，而参数 value 引用数组元素本身。每个数组元素有 3 个属性：role、name 和 emailId。因此，所有的数组元素的 3 个属性 role、name 和 emailId 被追加到类名为 listofstud 的段落元素中，属性之间以空白分隔，在屏幕上显示所有学生的信息。也可以使用 for 循环去遍历数组元素。把前面的 each()方法改为 for 循环，jQuery 代码如下所示：

```
for(var i=0;i<students.length;i++){
    $('p.listofstud').append(students[i].role+" "+students[i].name+"
"+students[i].emailId);
}
```

如你所见，each()方法自动遍历数组中的所有元素，因此使用起来更为方便。这两种方法都得到相同的运行结果，如图 2-19 所示。

List of students is

101 Ben ben@gmail.com102 Ian ian@gmail.com103 Caroline carol@gmail.com

图 2-19 输出对象数组的内容，作为段落元素的文本

从表格显示的示例中可见，创建对象数组 students 之后，使用 each()方法解析每个数组元素并通过回调函数进行处理。在回调函数中，每个数组元素的属性(即 role、name 和 emailId)在<td>和</td>标签之间显示。在表格元素的单独行中显示各个数组元素，在不同的列中显示数组元素的各个属性。最终以表格形式显示对象数组，如图 2-20 所示。

List of students is

101 Ben	ben@gmail.com
102 Ian	ian@gmail.com
103 Caroline	carol@gmail.com

图 2-20　以表格形式显示的一个对象数组

2.9　为对象数组排序

问题描述

假设以对象数组形式来存储学生信息。每个学生对象包含 3 个属性：role、name 和 emailId。请基于属性 role 为该数组排序。

解决方案

创建一个 HTML 文件，显示标题元素和类名为 listofstud 的空表格元素。该表格元素用于显示排序的对象数组。HTML 文件的内容如下所示：

```
<body>
<h3>List of students is </h3>
<table class="listofstud"></table>
</body>
```

编写 jQuery 代码来创建一个对象数组，存储 3 个学生的信息。每个学生对象包含属性 role、name 和 emailId。同时需要编写代码，执行基于学生对象 role 属性的排序。jQuery 代码如下所示。在下一节中解释其工作原理：

```
$(document).ready(function() {
  var students=[
  {
  "role": 101,
  "name": "Ben",
  "emailId":"ben@gmail.com"
  },
  {
  "role": 102,
  "name": "Ian",
  "emailId":"ian@gmail.com"
  },
  {
  "role": 103,
```

```
  "name": "Caroline",
  "emailId":"carol@gmail.com"
  }
  ];

  students = students.sort(function(a,b){
    return b.role-a.role;
  });

  $.each(students,function( index, value ){
  $('table.listofstud').append("<tr><td>"+value.role+"</td><td>"+value.name+"</td>
<td>"+
      value.emailId+"</td></tr>");
  });
});
```

如果要按照属性 name 的字母顺序为数组排序，就必须替换前面的 sort()函数，如下所示：

```
students = students.sort(function(a,b){
  if(a.name<b.name){ return -1 };
  if(a.name>b.name){ return 1 };
  return 0;
});
```

知其所以然

在 sort()方法中，需要添加比较函数，反复从数组中获取一对值，在比较的基础上返回小于 0、等于 0 和大于 0 的值。攻略 2.4 描述了基于什么返回这些值。

在示例的比较函数中，比较学生对象的 role 属性。该比较函数中返回值如下：

```
return b.role-a.role;
```

这意味着该函数将按照 role 属性降序排序。

此后，使用 each()方法来解析每个数组元素，并通过回调函数对它们进行处理。在回调函数中每个数组元素的属性，也就是 role、name 和 emailId，包围在<td>和</td>之间显示。这意味着，在表格的单独行（即表格数据元素）中存储不同的数组元素，而以列的形式显示数组元素的各个属性。其结果是，最终以表格形式显示对象数组，如图 2-21 所示。

如果按照 name 属性排序，则在比较函数中基于 students 对象的 name 属性进行比较。如果第一个元素的 name 属性小于（按照 ASCII 值）第二个元素的 name 属性，则该函数返回-1，反之则返回 1。其结果是基于 name 属性对学生对象数组进行排序。运行结果如图 2-22 所示。

List of students is

103 Caroline carol@gmail.com
102 Ian ian@gmail.com
101 Ben ben@gmail.com

图 2-21 一个学生对象数组，按 role 属性降序排序

List of students is

101 Ben ben@gmail.com
103 Caroline carol@gmail.com
102 Ian ian@gmail.com

图 2-22 学生对象数组基于 name 属性以字母顺序排序

2.10　小结

本章讨论了处理数组的不同攻略。例如，如何在列表中显示名称、如何操作数组中的元素等。还学习了如何执行数组筛选任务。

下一章将学习几个关于事件处理的攻略，涉及如何动态地突出显示文本、如何使图像随着鼠标的移动而获得和失去焦点等。还将看到如何创建图片切换效果、基于事件添加和删除文本等内容。

第3章 事件处理

本章讲述如何运用 jQuery 处理事件。可以是悬停、单击、双击等事件，还会看到同时关联于鼠标和键盘的事件。虽然第 6 章讲述视觉效果、介绍如何用图像创建动画，但有关文本的动画效果放在本章，因为文本的动画效果对事件作出反应时非常有用。本章讲述的攻略如下：

- 查找被点击的按钮；
- 自动触发事件；
- 点击之后禁用按钮；
- 处理鼠标事件；
- 查找被按下的鼠标键；
- 查找按下鼠标键的屏幕坐标；
- 动态突显文本；
- 随着鼠标移动使图像或明或暗；
- 查找何时元素获得和失去焦点；
- 把悬停效果应用到按钮；
- 切换应用不同的 CSS 类；
- 创建基于图像的切换效果；
- 在事件响应中添加和删除文本；
- 在事件响应中应用样式；
- 显示文字气球；
- 创建“返回顶部”链接；
- 提供“更多……”链接；
- 显示文字动画效果；
- 以滑动效果来替换文字；
- 使图像滚动；
- 确定哪个键被按下；
- 防止事件冒泡；
- 链接多个活动。

3.1　查找被点击的按钮

问题描述

网页上有几个按钮，用于完成不同的任务。为了进行正确的操作，需要知道被按下的是哪个按钮。

解决方案

本攻略中假设有两个按钮，想知道用户点击了哪个按钮。定义 HTML 文件，包含文本 Bold 和 Italic，通过应用不同的样式属性，给文本 Bold 和 Italic 分配按钮的形状。HTML 文件的内容如下：

```
<body>
<span class="bold buttons">Bold</span>
<span class="italic buttons">Italic</span>
</body>
```

在外部样式表文件 style.css 中编写 CSS 类，命名为 buttons，以便给文本分配按钮的形状。CSS 类 buttons 具有以下属性：

```
.buttons{
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

编写 jQuery 代码，使用 bind()方法为按钮附加单击事件：

```
$(document).ready(function() {
  $('.bold').bind('click', function(){
    alert('You have clicked the Bold button');
  });

  $('.italic').bind('click', function(){
    alert('You have clicked the Italic button');
  });
});
```

把单击事件应用于两个按钮

通过把单击事件添加到整个 buttons 类，从而把单击事件同时应用到两个按钮，而不是单独地为每个按钮添加点击事件。实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.buttons').bind('click', function(){
    alert('You have clicked the ' +$(this).text()+' button');
  });
});
```

1. 直接附加事件

不使用 bind()方法也可以把事件直接附加到任何指定的元素。请看以下 jQuery 代码，把单击事件附加到 buttons 类的元素：

```
$(document).ready(function() {
  $('.buttons').click(function(){
    alert('You have clicked the ' +$(this).text()+' button');
  });
});
```

2. 利用事件对象的目标属性

事件对象包含事件细节，JavaScript 把事件对象自动发送到事件处理函数。事件对象的其中一个属性称为 target（目标），可以用来查明在哪个元素上发生了事件。

利用事件对象的 target 属性，查明在哪个元素上发生了单击事件，实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.buttons').click(function(event){
    var $target=$(event.target);
    if($target.is('.bold')){
      alert('You have clicked the Bold button');
    }
    if($target.is('.italic')){
      alert('You have clicked the Italic button');
    }
  });
});
```

3. 附加双击事件

稍微修改 jQuery 代码，把双击事件附加到 buttons 类的 HTML 元素：

```
$(document).ready(function() {
  $('.buttons').dblclick(function(){
    alert('You have double-clicked the ' +$(this).text()+' button');
  });
});
```

知其所以然

HTML 文件中的以下语句：

```
<span class="bold buttons">Bold</span>
```

以上语句在 bold 和 buttons 类的 span 元素中定义了文本 Bold。bold 类用于把 jQuery 代码应用到 span 元素，而 buttons 类则把样式表中定义的 CSS 类 buttons 的指定样式应用到 span 元素。

同样，第二个语句在 italic 和 buttons 类的 span 元素中定义了文本 Italic。italic 类是为了应用 jQuery 代码，而 buttons 类则是为了应用 CSS 类。

样式表中定义的样式规则把 width、float、text-align、margin、border 和 font-weight 属性应用到 buttons 类的两个 span 元素上。运行结果是显示两个按钮，如图 3-1 所示。

前面的 jQuery 代码利用了 bind()方法。下面学习 bind()方法的语法，再去理解相关的 jQuery 代码。

1. bind()

此方法把指定的事件附加到指定的元素。

```
bind(eventType, data, handler)
```

- eventType，指定事件类型的字符串，click（单击）、double-click（双击）、focus（聚焦）、blur（失焦）等。
- data，传递到事件处理函数进行处理的数据。如果省略，事件处理函数可作为第二参数。
- handler，事件处理函数，包含在指定事件发生时要执行的语句。

jQuery 代码中，以下语句

```
$('.bold').bind('click', function(){
```

为 bold 类的 HTML 元素绑定内联函数作为单击事件处理程序。也就是说，把单击事件绑定到 bold 类的 span 元素。因此，如果用户单击文本 Bold，就会执行存储在内联函数中的语句。

以下语句

```
alert('You have clicked the Bold button');
```

当文本上发生单击事件时，就显示警报消息。

执行第一段 jQuery 代码，可以看到，文本 Bold 和 Italic 以按钮形式出现。点击 Bold 按钮，就会看到警报消息“You have clicked the Bold button”。如图 3-1 所示。

图 3-1　按钮被点击时显示警报消息

把单击事件应用到两个按钮的 jQuery 代码中，以下语句

```
$('.buttons').bind('click', function(){
```

把单击事件绑定到 buttons 类的 HTML 元素。由于文字 Bold 和 Italic 包含在 buttons 类的 span 元素中，于是单击事件被绑定到文本 Bold 和 Italic（定义于 style.css 文件中的样式规则，把按钮形状指派到文本 Bold 和 Italic）。

以下语句

```
alert('You have clicked the ' +$(this).text()+' button');
```

一旦发生单击事件，$(this).text()将会显示 HTML 元素的文本内容。也就是说，如果单击 Bold 按钮，$(this).text()将显示 Bold。同样，如果单击 Italic 按钮，它会显示 Italic。运行结果如图 3-1 所示。

下面讲述单击事件的运行机制。

2. click()

此方法把单击事件绑定到指定元素。如果将鼠标指针放在元素上，按下和释放鼠标键，则发生单击事件。

```
.click(handler)
.click()
```

这里的 handler 是事件处理函数，用鼠标点击指定元素时，就执行该函数所包含的语句。

利用事件对象的 target 属性，也可以得到相同的结果，如图 3-1 所示。在 jQuery 代码中，事件对象的 target 属性被用来查明元素的目标（已经发生单击事件），并存储在变量$target 中。在此之后，通过条件语句检查存储在$target 变量中的元素是属于 bold 类还是 italic 类，即检查被单击的是 Bold 按钮还是 Italic 按钮，然后在屏幕上显示相应的警报消息。

下面学习双击事件。双击事件类似于单击事件。

3. dblclick()

该方法把双击事件附加到指定元素。所谓双击事件，就是把鼠标移动到元素上双击时发生的事件。

```
.dblclick(handler)
.dblclick()
```

这里的 handler 是事件处理函数，包含双击元素时我们想要执行语句。

在 jQuery 代码中利用双击事件，为文本 Bold 和 Italic 附加 dblclick()事件。双击这两个按钮中的任何一个，输出如图 3-2 所示。

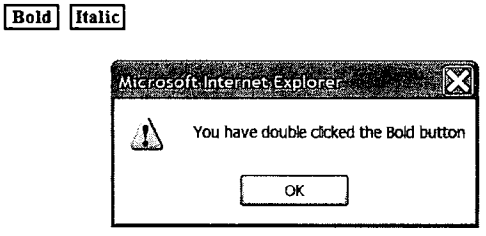


图 3-2 按钮被双击时显示警报消息

3.2 自动触发事件

问题描述

网页上有两个按钮，Bold 和 Italic，要求实现自动触发任一按钮上的单击事件。

解决方案

在此攻略中，想要自动触发两个按钮的单击事件。新建 HTML 文件，包含文本 Bold 和

Italic，分别为文本 Bold 和 Italic 创建按钮。该 HTML 文件的内容如下：

```
<body>
<span class="bold buttons">Bold</span>
<span class="italic buttons">Italic</span>
</body>
```

为了把按钮形状赋予文本，在外部样式表 style.css 中编写 CSS 类 buttons：

```
.buttons{
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

编写 jQuery 代码，自动触发事件，也就是说，由脚本触发事件而不是由用户触发事件。我们可能需要自动触发某些事件，比如按钮的自动点击或表单的自动提交。jQuery 提供的触发事件的方法是 trigger()。在 Italic 按钮上自动触发单击事件的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.buttons').bind('click', function(){
    alert('You have clicked the ' +$(this).text()+' button');
  });
  $('.italic').trigger('click');
});
```

知其所以然

在 HTML 文件中，以下语句

```
<span class="bold buttons">Bold</span>
```

在 bold 类和 buttons 类的 span 元素中定义文本 Bold。bold 类把 jQuery 代码应用到 span 元素，而 buttons 类把在样式表中定义 CSS 类的样式应用到 span 元素。

同样，第二个语句在 italic 类和 buttons 类的 span 元素中定义文本 Italic。italic 类把 jQuery 代码应用到 span 元素上，而 buttons 类把 CSS 类应用到 span 元素上。

在 jQuery 代码中，利用 trigger()方法自动触发事件。

下面简要介绍 trigger()方法。

trigger()

此方法调用指定事件类型的事件处理函数（事件类型作为参数传递到此方法）。

```
trigger(eventType)
```

其中 eventType 是字符串，指定事件的类型，即单击、双击、聚焦，等等。

该方法返回 jQuery 对象。当脚本触发事件时，将执行相应的事件处理函数中的代码。这意味着在任何元素上使用 trigger()方法之前，需要确认已经为该元素定义了事件处理函数。

在 jQuery 代码中，把 click 事件附加到 buttons 类的元素。即把 click 事件附加到 Bold

按钮和 Italic 按钮。此外，已经为这两个按钮定义了内联函数作为事件处理函数。在此之后，以下语句

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

触发 Italic 按钮的 click 事件，导致它的事件处理函数被调用。该事件处理函数显示警报消息，指出该按钮被单击，如图 3-3 所示。

图 3-3 自动触发 Italic 按钮的 click 事件，显示警报消息

3.3 点击之后禁用按钮

问题描述

有时我们希望事件只触发一次，或者想在满足某些条件之后将其禁用。例如，提交按钮被点击一次之后，我们想要禁用该按钮。

解决方案

在此攻略中假设有两个按钮，按钮被点击一次后要禁用之。定义 HTML 文件，分别创建包含文本 Bold 和 Italic 的两个“按钮”。HTML 文件的内容如下：

```
<body>
<span class="bold buttons">Bold</span>
<span class="italic buttons">Italic</span>
</body>
```

在外部样式表 style.css 文件中编写 CSS 类 buttons，以便把按钮形状赋予文本。CSS 类 buttons 具有以下属性：

```
.buttons{
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

按钮被点击一次之后禁用事件处理函数。实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.buttons').bind('click', function(){
    alert('You have clicked the ' +$(this).text()+' button');
    $('.buttons').unbind('click');
  });
});
```

知其所以然

jQuery提供unbind()方法，用于从指定元素中删除某事件类型。jQuery还支持名称空间的事件，这样便可以通过名称空间而触发或取消绑定（unbind）指定组的已绑定处理函数，无需直接引用它们。

unbind()方法从指定的元素中删除先前绑定的事件处理函数：

```
unbind(eventType, handler)
unbind(eventType)
unbind()
```

- eventType引用不同事件类型，如单击、双击等。附加了指定eventType的所有事件处理函数将被停止执行。
- handler是要删除的事件处理函数，应该与传递到bind()方法的事件处理函数相同。
- 如果不传递任何参数，所有的事件将被删除。

在jQuery示例代码中，把单击事件绑定到按钮。点击任一按钮将显示警报消息，显示被选择按钮的文本。所以，如果单击Italic按钮，将得到如图3-2的运行结果。在那之后的以下语句

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

取消绑定单击事件的事件处理函数（内联函数），因此点击任一按钮将不再显示任何消息。

3.4　处理鼠标事件

问题描述

需要处理程序中与各种鼠标相关的事件。

解决方案

在此攻略中，假设有两个按钮，对发生在按钮上的不同鼠标事件进行测试。定义HTML文件，创建分别包含文本Bold和文本Italic的两个按钮：

```
<body>
<span class="bold buttons">Bold</span>
<span class="italic buttons">Italic</span>
</body>
```

在外部样式表style.css文件中编写CSS类buttons，以便把按钮形状赋予文本。CSS类buttons具有以下属性：

```
.buttons{
```

```
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

编写 jQuery 代码，在任一按钮上按下鼠标键时显示消息：

```
$(document).ready(function() {
  $('.buttons').bind('mousedown', function(){
    alert('The mouse button is pressed over ' +$(this).text()+' button');
  });
});
```


1. 响应mouseup事件

响应 mouseup 事件的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.buttons').mouseup(function(){
    alert('The mouse button is released over ' +$(this).text()+' button');
  });
});
```

2. 响应mouseover事件

响应 mouseover 事件的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.buttons').mouseover(function(){
    alert('The mouse is over ' +$(this).text()+' button');
  });
});
```

知其所以然

在前面的 jQuery 示例代码中已经使用 mousedown()方法，首先简要介绍该方法，然后研究 mouseup()、mouseover()和 mouseout()方法。

1. mousedown()

一旦在指定的元素上发生 mousedown 事件，此方法就执行附加的事件处理函数。mousedown 事件意味着鼠标指针在指定的元素上时按下鼠标键。

```
.mousedown(handler)
.mousedown()
```

第一个语法中的 handler 是发生 mousedown 事件时执行的函数。

第二个语法是手动调用(触发)mousedown()事件。在以下示例中，一旦 button1 (button1 类的 HTML 元素)被点击，就调用(触发)button2(button2 类的 HTML 元素)的 mousedown()事件：

```
$(' .button1').click(function(){
  $('.button2').mousedown();
});
```

在示例的 jQuery 代码中，我们为类 buttons 的 HTML 元素绑定 mousedown 事件，因此一旦鼠标键在 Bold 按钮上被按下，我们就会得到输出如图 3-4 所示。

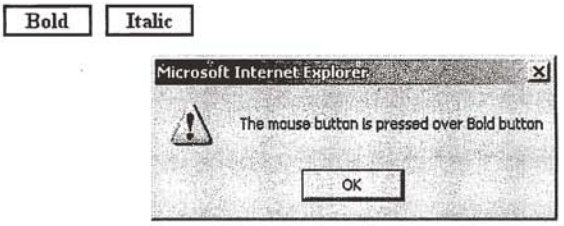


图 3-4　一旦在 Bold 按钮上按下鼠标键，就显示警报消息

下面逐个查看处理鼠标事件的其他方法。

2. mouseup()

此方法把 mouseup 事件绑定到指定的元素。一旦鼠标指针在元素上并且释放鼠标键，则发生 mouseup 事件。

```
.mouseup(handler)
.mouseup()
```

这里的 handler 是事件发生时执行的内联函数，第二个语法用于手动调用 mouseup 事件。

3. mouseover()

此方法为指定的元素绑定 mouseover 事件。一旦鼠标指针进入指定元素的区域，则发生了 mouseover 事件。

```
.mouseover(handler)
.mouseover()
```

第一个语法中的 handler 是事件发生时执行的内联函数，第二个语法用于手动调用（触发）mouseover 事件。

在响应 mouseover 事件的 jQuery 示例代码中，一旦鼠标指针进入 Bold 按钮的区域，就会得到输出如图 3-5 所示。

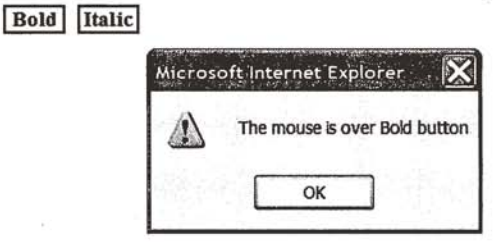


图 3-5　一旦鼠标指针进入 Bold（加粗）按钮区域，就显示警报消息

4. mouseout()

此方法把 mouseout 事件绑定到指定的元素。一旦鼠标指针离开指定的元素，就会发生

mouseout 事件。

```
.mouseout(handler)
.mouseout()
```

第一个语法的 handler 是事件发生时执行的内联函数，第二个语法用于手动调用（触发）mouseout 事件。

3.5 查明哪个鼠标键被按下

问题描述

假设网页上有一个按钮，请检测在这个按钮上哪个鼠标键按下：鼠标左键或鼠标右键。

解决方案

假设 HTML 文件包含文本 Click Me（表现为按钮）和空段落元素，如下所示：

```
<body>
<span class="buttons">Click Me</span><br/><br/>
<p></p>
</body>
```

类选择器 buttons 如下所示：

```
.buttons{
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

检测在 Click Me 按钮上哪个鼠标键被按下的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.buttons').mousedown(function(event){
    if(event.button==1){
      $('p').text('Left mouse button is pressed');
    }
    else
    {
      $('p').text('Right mouse button is pressed');
    }
  });
});
```

知其所以然

在 HTML 文件中，文本 Click Me 包含在 buttons 类的 span 元素中，因此在样式表中定义的 buttons 类选择器的属性可以应用到文本 Click Me，赋予它按钮形状。空段落元素用于显示消息，显示按下了哪个鼠标键。

在jQuery代码中，把mousedown事件附加到buttons类的元素，即附加到文本Click Me（被赋予按钮形状）。此外，当事件发生时，JavaScript把事件对象发送到事件处理函数。所以，一旦在Click Me按钮上按下任一鼠标键，事件对象就会自动传递到事件处理函数。

利用事件对象的button属性来确定哪个鼠标键被按下。对于鼠标左键，button属性的值为1，而对于鼠标右钮，button属性的值为2。如果由事件对象的button属性返回的值是1，就为段落元素分配文本Left mouse button is pressed，如图3-6所示。

如果由事件对象的button属性返回的值不是1，就为段落元素分配文本Right mouse button is pressed，如图3-7所示。

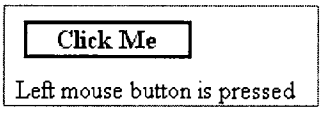


图3-6　一旦在Click Me按钮上鼠标左键被按下，就显示如图的文本消息

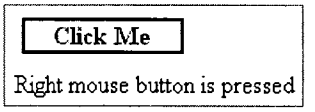


图3-7　一旦在Click Me按钮上鼠标右键被按下，就显示如图的文本信息

3.6　查找鼠标按下时的屏幕坐标

问题描述

假设网页上有图像，一旦在该图像上方按下鼠标键，就显示其在屏幕上的坐标位置。

解决方案

新建HTML文件，显示图像和空段落元素，如下所示：

```
<body>
<img src="cell.jpg"/>
<p></p>
 </body>
```

空段落元素用于显示屏幕坐标。

编写jQuery代码，为该图像附加mousedown事件，监视是否鼠标键在图像的任意之处被按下。该代码如下所示：

```
$(document).ready(function() {
   $('img').mousedown(function(event){
    $('p').text('Mouse is clicked at horizontal coordinate: '+event.screenX+
                ' and at vertical coordinate: '+event.screenY);
  });
});
```

知其所以然

在此攻略中，利用了事件对象的属性。一旦发生了某个事件，JavaScript自动发送事件对象到事件处理函数。事件对象有几个特性（attribute）或属性(property)。这个解决方案所使用的两

个属性如下：

- screenX：指定事件发生时相对于屏幕原点的横坐标
- screenY：指定事件发生时相对于屏幕原点的纵坐标

在 mousedown 的事件处理函数中编写代码，在段落元素中以文本形式显示事件对象的 screenX 和 screenY 属性。执行 jQuery 代码，运行结果如图 3-8 所示。

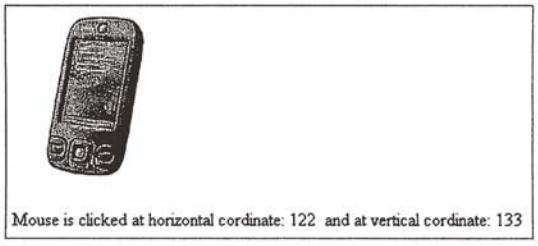


图 3-8 在屏幕上的位置坐标显示鼠标键在图像上的何处被点击

3.7 动态地突出显示文本

问题描述

假设在网页上把按钮形状赋予文本，当鼠标在文本上方移动时，需要突出显示该文本（通过改变文本的前景色和背景色）。

解决方案

新建 HTML 文件，包含 buttons 类的 span 元素（包含文本 Hightlight）和段落元素，内容如下：

```
<body>
<span class="buttons">Highlight</span><br/><br/>
<p>Styles make the formatting job much easier and more efficient. To give an attractive
look to web sites, styles are heavily used. A person must have a good knowledge of HTML
and CSS and a bit of JavaScript.  </p>
</body>
```

把样式表中定义的 buttons 样式规则应用于文本 Hightlight，赋予其按钮形状。样式表具有以下属性：

```
.buttons{
width: 80px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

为了把样式动态地应用到文本（当鼠标移动到文本上方时）而编写 jQuery 代码如下：

```
$(document).ready(function() {
  $('.buttons').mouseover(function(){
    $('p').css({
      'background-color':'cyan',
      'font-weight':'bold',
      'color':'blue'
    });
  });
});
```

知其所以然

在此攻略中使用了一种技术，覆盖在样式表中定义的样式属性，并把 CSS 属性直接应用到指定的元素。jQuery 提供 css()方法，用于把 CSS 属性直接应用到 HTML 元素。此方法为指定的元素直接设置 CSS 属性，覆盖在样式表中定义的样式（如有）。利用 css()方法可以对单个元素以及元素集合更好地控制样式的应用：

```
.css(property, value)
```

这里的 property 是想要设置的 CSS 属性的名称，value 可以是想要为该属性分配的值，也可以是函数，返回用于设置属性的值。

以下是一个示例：

```
$('p').css('color':'blue');
```

该段代码把段落文本的颜色设置为蓝色。下面使用函数返回 img 元素的高度与 30 之和，也就是说，它会使 img 元素的高度增加 30 像素：

```
$('img').css('height',function(){ return $(this).height()+30;});
```

解决方案的运行结果将显示按钮和文本段落，如图 3-9 所示。

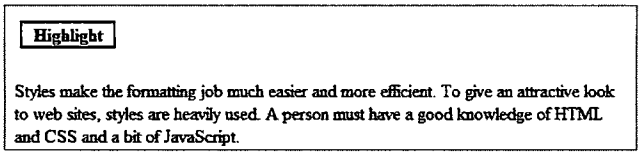


图 3-9　按钮与段落文本

在 jQuery 示例代码中，css()定义了几个属性，比如 background-color 属性为段落文本应用蓝绿色背景，font-weight 使文字显示为粗体，color 使文本的前景色改变为蓝色。当鼠标移动到按钮上方时，在 css()方法中定义的属性将应用于文本，如图 3-10 所示（虽然在书中可能是黑白颜色，但仍可了解大意）。

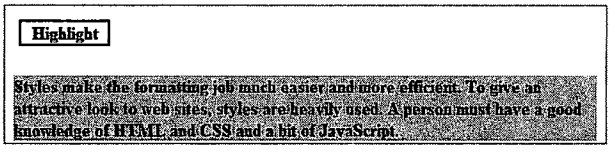


图 3-10　当鼠标移动按钮上方时，突出显示段落文本

3.8 随着鼠标移动使图像明亮或模糊

问题描述

假定网页上有图像和按钮。图像最初是模糊的。当鼠标移动到按钮上方时使图像明亮，当鼠标从按钮上方移走时使图像模糊。此外，一单击按钮，就增加图像的高度和宽度。

解决方案

在本攻略中利用不同的鼠标事件方法，如攻略 3.4 所述。如果你不知道不同的鼠标事件的运作方法，如 mouseover()、mouseout()、mousedown()等，请复习攻略 3.4 之后再学习本攻略。

假设 HTML 文件中包含按钮和图像，内容如下：

```
<body>
<span class="buttons">Bright Image</span>
<img src="cell.jpg"/>
</body>
```

把文本 Bright Image 包含在 buttons 类的 span 元素中，把外部样式表中定义的 buttons 样式规则应用到文本，赋予它按钮形状。

```
.buttons{
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

把效果应用到图像的 jQuery 代码如下：

```
$(document).ready(function() {
  $('img').css('opacity',0.4);

  $('.buttons').bind('mouseover', function(){
    $('img').css('opacity',1.0);
  });

  $('.buttons').bind('mouseout', function(){
    $('img').css('opacity',0.4);
  });

  $('.buttons').bind('mousedown', function(){
    $('img').css('width',function(){ return $(this).width()+50;});
    $('img').css('height',function(){ return $(this).height()+30;});
  });
});
```

知其所以然

在此解决方案中使用 CSS 属性 opacity（不透明度）。opacity 的取值范围从 0（透明）到 1（不透明），或从 0%到 100%。

下面逐行研读代码：

```
$('img').css('opacity',0.4);
```

这一行在最初时以及在鼠标从按钮上移走时使图像模糊。

```
$('img').css('opacity',1.0);
```

当鼠标在 Bright Image 按钮上方时，使图像明亮（不透明）。

```
$('img').css('width',function(){ return $(this).width()+50;});
```

当鼠标在 Bright Image 按钮上方被按下时，使图像的宽度增加 50 像素。

```
$('img').css('height',function(){ return $(this).height()+30;});
```

当鼠标在 Bright Image 按钮上方被按下时，使图像的高度增加 30 像素。

最初时，鼠标远离 Bright Image 按钮，因此图像模糊，如图 3-11 所示。

当鼠标指针移动到 Bright Image 按钮上方时，图像变得明亮（即不透明），如图 3-12 所示。

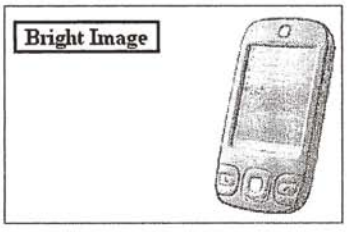


图 3-11　当鼠标离开按钮时，显示模糊的图像

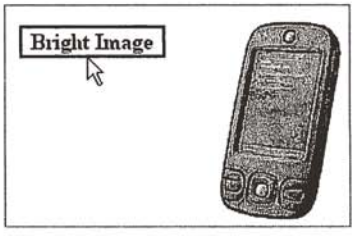


图 3-12　当鼠标移动到 Bright Image 按钮上方时，图像变得明亮（即不透明）

当鼠标在 Bright Image 按钮上被按下时，图像的宽度和高度分别增加 50 和 30 像素，如图 3-13 所示。

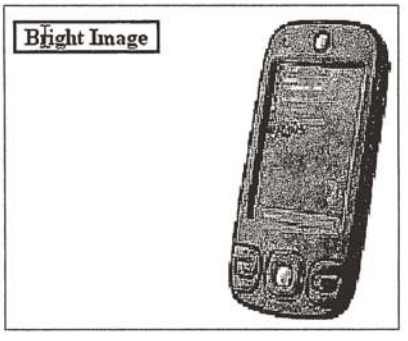


图 3-13　点击按钮时增加图像的宽度和高度

3.9 查明元素何时获得和失去焦点

问题描述

假设网页上有几个文本字段，想知道何时指定的文本字段获得和失去焦点。

解决方案

新建 HTML 文件，包含 name（名字）和 age（年龄）两个文本字段，内容如下：

```
<body>
Name: <input class="name" type="text" /><br>
Age: <input class="age" type="text" />
</body>
```

处理 name 文本字段的焦点事件，实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.name').focus(function(){
    alert('The focus currently is on name field');
  });
});
```

失去焦点时触发事件

当 name 文本字段失去焦点时触发事件，实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.name').blur(function(){
    alert('The focus is lost from name field');
  });
});
```

知其所以然

在 jQuery 示例代码中利用 focus()方法获知元素何时获得焦点。下面考察其工作原理。

1. focus()

此方法把焦点事件绑定到指定的元素。它包含一个函数，当指定的元素得到焦点时触发焦点事件，执行该函数：

```
.focus(handler)
```

这里的 handler 是事件处理函数，包含指定的元素获得焦点时要执行的语句。获得焦点可以通过指针设备（如鼠标）或 Tab 导航（即键盘 Tab 键）。

示例代码中把 focus()方法绑定到 name 类的 HTML 元素，HTML 代码中把 name 类分配到用来输入名字的文本字段。也就是说，当光标进入 name 文本字段时，焦点事件将被触发。focus()方法包含附加的内联函数，显示警报消息 The focus currently is on name field（目前焦点在 name 字段），如图 3-14 所示。

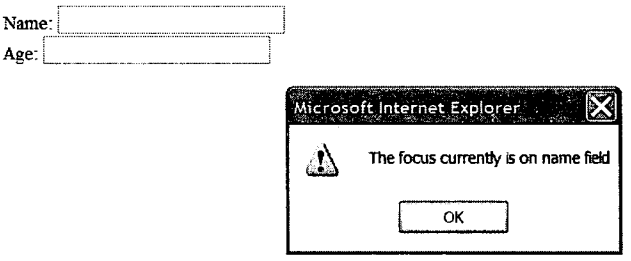


图 3-14　当 name 文本字段获得焦点时显示警报消息

当元素失去焦点时触发事件，这就是 blur()方法。下面学习 blur()方法的工作原理。

2. blur()

当指定的元素失去焦点时触发 blur 事件。当焦点从指定的元素离开时，就执行附加的事件处理程序。当用户按下 Tab 键或在页面上其他地方点击鼠标时，即可使焦点离开某个元素。

```
.blur(handler)
```

这里的 handler 是事件处理函数，包含要执行的语句。当焦点离开指定的元素时就执行该函数。

从 jQuery 示例代码可见，失去焦点时就触发事件。把 blur()方法附加到 name 文本字段，又把内联函数附加到 blur()方法，当焦点离开 name 文本字段时就调用内联函数。内联函数将显示警报消息 The focus is lost from name field（焦点离开了 name 文本字段）如图 3-15 所示。

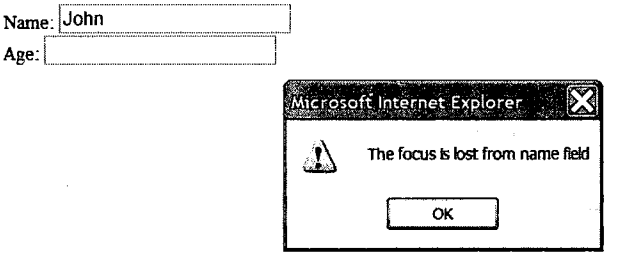


图 3-15　当焦点离开 name 文本字段时显示警报消息

3.10　在按钮上应用悬停效果

问题描述

应用悬停效果，意味着当鼠标经过 HTML 元素时，改变元素的样式（比如按钮或图像的形状或颜色）。请在几个按钮上实现相同的效果。

解决方案

要在按钮上应用悬停效果，就必须在 HTML 文件中把按钮的形状赋予文本。利用以下 HTML

文件：

```
<body>
<span class="bold buttons">Bold</span>
<span class="italic buttons">Italic</span>
</body>
```

在外部样式表中编写 CSS 类 buttons 用于把按钮的形状赋予文本；编写 CSS 类 hover 用于鼠标悬停时（即当鼠标指针在按钮上方时）应用指定的属性。样式表文件的内容如下：

```
.buttons{
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
.hover{
cursor: crosshair;
color: blue;
background-color:cyan
}
```

编写代码为按钮添加悬停事件。当鼠标指针进入按钮区域时，就为按钮应用 hover 样式规则，当鼠标指针离开按钮区域时，就从按钮中删除 hover 样式规则：

```
$(document).ready(function() {
  $('.buttons').hover(
    function(){
      $(this).addClass('hover');
    },
    function(){
      $(this).removeClass('hover');
    }
  );
});
```

知其所以然

hover()方法把两个事件处理程序附加到指定的元素。一个事件处理程序在鼠标指针进入元素时触发，另一个事件处理程序在鼠标指针离开元素时触发。

```
.hover(handler1, handler2)
```

handler1 是事件处理函数，包含鼠标进入指定的元素时要执行的语句，而 handler2 也是事件处理函数，包含鼠标离开指定的元素时要执行的语句。于是两个函数结合起来，提供悬停效果。

在 jQuery 示例代码中，选择器$('.buttons')选择文档中 buttons 类的所有元素，把 hover()方法附加到这些元素。当鼠标进入 buttons 类的元素时，就把 hover 类应用到该元素；当鼠标离开该元素时，就从该元素中删除 hover 类。

最初时，按钮如图 3-16 所示。

图 3-16　为 buttons 类的元素应用悬停样式

当鼠标进入 buttons 类的 HTML 元素时，就把 hover 类应用到该元素，如图 3-17 所示。

图 3-17　当鼠标在上方移动时，按钮改变了颜色

当鼠标离开按钮时，就删除 hover 类，按钮显示常规的样式。

3.11　切换应用一个 CSS 类

问题描述

当某事件发生在元素上时，请为该元素应用定义在 CSS 类中的样式属性，当同一事件发生在同一元素上时，从该元素中删除 CSS 类。也就是说，要在 CSS 类的应用与不应用之间进行切换。

解决方案

假设有 HTML 文件包含 Bold 和 Italic 两段文本。为了应用选择器 buttons（把按钮形状赋予文本），把文本 Bold 和 Italic 分别包含在 buttons 类的两个 span 元素中。HTML 文件的内容如下：

```
<body>
<span class="bold buttons">Bold</span>
<span class="italic buttons">Italic</span>
</body>
```

在以下样式表中定义了 CSS 类 buttons 和 hover，伴随着每次发生的事件，希望把为 buttons 类的元素应用或删除 hover 类所包含的样式属性。

```
.buttons{
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}

.hover{
cursor: crosshair;
color: blue ;
```

```
background-color:cyan
}
```

编写 jQuery 代码，利用 toggleClass()方法来切换 CSS 类：

```
$(document).ready(function() {
  $('.buttons').click(function(){
    $(this).toggleClass('hover');
  });
});
```

知其所以然

利用 toggleClass()方法，为按钮应用 hover 类以及删除 hover 类。首先学习该方法的语法。

1. toggleClass()

如果 CSS 类已经应用到指定元素，则此方法删除 CSS 类；如果尚未应用，则为指定元素应用 CSS 类。

```
.toggleClass(class)
```

这里的 class 是在指定的元素上将要应用的 CSS 类（如果尚未应用），或将要删除的 CSS 类（如果已经应用）。

从上面的 jQuery 代码可知，当单击事件发生时，利用 toggleClass()为按钮应用或删除 CSS 类 hover。也就是说，按钮上第一次发生单击事件时，把 hover 类所提供的样式属性应用到按钮，改变其背景色为青蓝色，前景色为蓝色，如图 3-17 所示。再次单击按钮时，也就是说，在同一个按钮上再次发生单击事件时，从按钮中删除 CSS 类 hover，按钮恢复初始的外观（如图 3-16 所示）。

应用另一个方法，可以得到相同的运行结果，那就是 toggle()方法。

2. toggle()

此方法为指定的元素附加两个事件处理函数。第一个事件处理函数在事件偶数次发生时被执行，而第二个事件处理函数在事件奇数次发生时被执行，从 0 开始计数。换句话说，在指定元素上第一次发生事件时，第一个事件处理函数将被执行（因为计数从 0 开始），在指定元素上第二次发生事件时，第二个事件处理函数将被执行。也就是说，每次发生事件时，两个事件处理函数将被轮流执行。

```
.toggle(handler1, handler2)
```

通过使用 toggle()方法来应用和删除 CSS 类 hover（每当点击事件发生时）可以获得如同上一示例的效果。请看以下 jQuery 代码：

```
$(document).ready(function() {
  $('.buttons').toggle(
    function(){
      $(this).addClass('hover');
    },
    function(){
```

```
        $(this).removeClass('hover');
      }
    );
});
```

从示例代码可见，在 toggle()方法中使用了两个事件处理函数。第一个为 buttons 类的元素（即 Bold 和 Italic 按钮）应用 CSS 类 hover，而第二个则为按钮删除 CSS 类。因此，单击任一按钮时，CSS 类 hover 将应用于该按钮，改变其背景色和前景色，如图 3-17 所示。当在同一按钮上再次发生单击事件时，CSS 类 hover 将被删除，使其显示为简单的按钮（没有应用任何前景色或背景色），如初始时的那样，如图 3-16 所示。

3.12　创建基于图像的变换

问题描述

创建基于图像的变换。图像变换是指当鼠标在图像上移动时图像改变形状、图像指定超链接到某网站。一旦单击图像，图像也发生变化，表示该图像已经被访问过。

解决方案

新建 HTML 文件，其中包含超链接元素：

```
<body>
<a href="abc.com"><span class="roll"></span></a>
</body>
```

在以下样式表中编写样式规则，命名为 link、hover 和 active 以及类选择器 img，无需利用 jQuery 代码，img 元素将自动应用类选择器 img 的属性。样式表文件的内容如下：

```
.link{
display:block;
width:170px;
height:55px;
background-image:url(btn1.bmp);
background-repeat:no-repeat;
background-position: top left;
}

.hover{
display:block;
width:200px;
height:70px;
background-image:url(btn2.bmp);
background-repeat:no-repeat;
background-position: top left;
}

.active{
display:block;
width:170px;
```

```
height:55px;
background-image:url(btn3.bmp);
background-repeat:no-repeat;
background-position: top left;
}

img{
border:0;
}
```

把定义在样式表中的样式规则应用到 roll 类的空 span 元素的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.roll').addClass('link');
  $('.roll').hover(
    function(){
      $(this).addClass('hover');
    },
    function(){
      $(this).removeClass('hover');
    }
  );

  $('.roll').click(function(event){
    $(this).addClass('active');
    event.preventDefault();
  });
});
```

知其所以然

前面的 HTML 文件中包含 roll 类的 span 元素，通过样式规则和 jQuery 代码为该元素填充图像。本解决方案中使用 3 个图像 btn1.bmp、btn2.bmp 和 btn3.bmp，如图 3-18 到图 3-20 所示。

在样式表中，样式规则 link 包含网页加载时应用到图像的属性。它为图像指定宽度为 130 像素，高度为 35 像素。把图像 btn1.bmp 加载为背景，并且把 background-repeat 属性值设置为 no-repeat，以避免重复该图像，使之只出现一次。

因为鼠标指针在按钮上方移动时，我们想让按钮变大，所以使样式规则 hover 包含第二个图像 btn2.bmp（如图 3-18 所示）。此样式规则的其余属性与样式规则 link 相同。

第三个样式规则指定一旦该链接被访问过，则显示什么图像，我们使用名为 btn3.bmp 的图像。图像 btn1.bmp 和 btn3.bmp 两者之间的唯一变化在于按钮文本的颜色不同。在 btn3.bmp 中按钮的文本颜色被设置为红褐色，代表链接已被访问。

该样式表还包含类型选择器 img，包含值为 0 的唯一属性 border，用于删除将要显示的 3 个图像的边框。

在 jQuery 代码中，如下语句

```
$('.roll').addClass('link');
```

把样式规则 link 应用到 span 元素，使 btn1.bmp 图像显示在屏幕上。

如以下语句

```
$('.roll').hover(
  function(){
    $(this).addClass('hover');
  },
  function(){
    $(this).removeClass('hover');
  }
);
```

为 span 元素添加悬停事件，并为悬停事件附加两个内联函数。第一个函数把样式规则 hover 应用到 span 元素（当鼠标指针在图像上方移动时），使 btn2.bmp（扩大的图像）取代 btn1.bmp。图像 btn1.bmp 将被图像 btn2.bmp 隐藏，因为通过 background-position 属性，把图像 btn2.bmp 的位置设置为左上角。第二个内联函数（当鼠标指针离开图像时）从图像中删除样式规则 hover，使图像恢复到初始样式，也就是说，即显示 btn1.bmp。

如以下语句

```
$(this).addClass('active');
```

把 active 样式规则中定义的属性应用到图像，把背景图像 btn3.bmp 设置为可见。这个图像把按钮文本设置为栗色，使它看起来不同于最初的图像，表明它已被访问过。

如以下语句

```
event.preventDefault();
```

防止浏览器导航到超链接所指向的网站。也就是说，它使超链接忽略默认操作（即导航到链接所指向的网站）。因此，即使在单击图像之后，仍然停留在同一个网页。

最初时，图像如图 3-18 所示。

图 3-18　默认状态下显示的常规按钮

当鼠标指针在图像上方移动时，图像变大，如图 3-19 所示。

图 3-19　当鼠标在上方移动时，处于放大状态的按钮

一点击图像链接，图像便发生变化，如图 3-20 所示。

图 3-20　处于活动状态时，按钮上的文字改变了颜色（当按钮被点击时）

3.13 为响应事件而添加和删除文本

问题描述

假设网页上有两个按钮，按钮的文本分别是 Add 和 Remove。当用户选择 Add 按钮时把某文本添加到网页，当用户选择 Remove 按钮时，则删除已添加的文本。

解决方案

首先新建 HTML 文件，包含 buttons 类的两个 span 元素（分别包含文本 Add 和 Remove），因此在样式表中定义的类选择器 buttons 可以应用于文本 Add 和 Remove：

```
<body>
<span class="add buttons">Add</span>
<span class="remove buttons">Remove</span><br><br>
<div></div>
</body>
```

样式表文件的内容如下：

```
.buttons{
width: 80px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

把单击事件及其内联函数添加到两个按钮，以便添加和删除文本，实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.add').click(function(){
    $('div').prepend('<p>Styles make the formatting job much easier and more efficient.
To give
an attractive look to web sites, styles are heavily used. A person must have a good
knowledge of HTML and CSS and a bit of JavaScript.  </p>');
  });

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

知其所以然

HTML 文件中包含空 div 元素。一点击 Add 按钮，就会把段落元素（包含文本）添加到空 div 元素。一点击 Remove 按钮，就会把新添加的段落元素从 div 元素中删除。为了添加段落元素，使用第 1 章中学过的 prepend()方法。为了删除段落元素，使用 remove()方法。

remove()方法从 DOM 中删除指定的元素集合，并返回 jQuery 对象。它也删除所有相关的

事件处理函数和内部缓存数据。不需要传递任何参数到此方法。

在jQuery示例代码中，以下语句

```
$('div').prepend('<p>Styles...</p>');
```

一点击Add按钮，就会把段落元素添加到div元素的开头，而如下语句

```
$('p').remove();
```

则访问HTML文档的段落元素并删除它们。由于HTML文件中只有一个段落元素（通过Add按钮添加的），一点击Remove按钮，就会删除段落元素。

执行以上jQuery代码，最初将看到两个按钮，按钮上的文本分别是Add和Remove，如图3-21所示。

图3-21　用于添加和删除文本的两个按钮的初始化

一点击Add按钮，包含文本的段落元素就会显示，如图3-22所示。

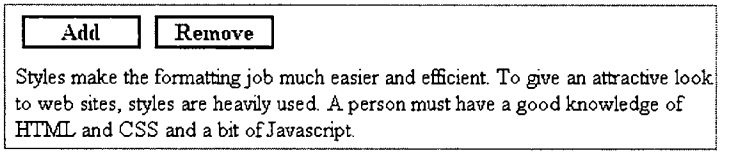


图3-22　一点击Add按钮，就会显示文本

一点击Remove按钮，文本就会消失。

3.14　应用样式作为对事件的响应

问题描述

网页上有两个按钮，按钮上的文本分别是Bold和Italic。网页上也有一些文本。请实现这样的效果：按下Bold按钮时，文本变成粗体；按下Italic按钮时，文本变成斜体。与此同时，还要在按钮上实现悬停效果（当鼠标指针移动到按钮上方时，改变按钮的背景色和前景色）。

解决方案

新建HTML文档，文档上有两个按钮和一些文本，如下所示：

```
<body>
<span class="buttons" id="boldbutton">Bold</span>
<span class="buttons" id="italicbutton">Italic</span><br/><br/>
<div id="info">Creating Rich Internet Applications<br/>
jQuery is an open source project<br/>
Manipulating DOM using jQuery<br/>
</div>
```

```
</body>
```

样式表的内容如下：

```
.buttons{
width: 80px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}

.hover{
cursor: crosshair;
color: blue ;
background-color:cyan
}

.boldmatter
{
  font-weight: bold;
}

.italicmatter
{
  font-style: italic;
}
```

使按钮显示悬停效果的 jQuery 代码如下：

```
${document).ready(function() {
  $('.buttons').hover(
    function(){
      $(this).addClass('hover');
    },
    function(){
      $(this).removeClass('hover');
    }
  );

  $('#boldbutton').bind('click', function(){
    $('#info').removeClass('italicmatter');
    $('#info').addClass('boldmatter');
  });

  $('#italicbutton').bind('click', function(){
    $('#info').removeClass('boldmatter');
    $('#info').addClass('italicmatter');
  });
});
```

知其所以然

点击相应的按钮时，为了能够应用样式把文本转变为粗体或斜体，为两个 span 元素分别分

配 id 为 boldbutton 和 italicbutton。给将要应用样式变为粗体或斜体的文本，分配 id 为 info。div 元素包含 3 行文本。

在 jQuery 代码中，首先选择文档中的所有 buttons 类的元素，并通过 hover()方法对那些元素应用指定的 CSS 类。借助于 addClass()和 removeClass 函数，当鼠标进入元素（包含 buttons 类的元素）时，hover()方法应用 CSS 类 hover，当鼠标离开元素时，hover()方法删除 CSS 类 hover。

bind()方法允许把任何 JavaScript 事件指定到、把函数（应用某 CSS 类）附加到任何元素。使用此技术，选择 id 为 boldbutton 的元素，绑定 click 事件。也就是说，在这样的元素上发生单击事件时，就执行函数：选择 id 为 info 的元素（HTML 文档的 body 而不是按钮），删除 CSS 类类 itaitalicmatter 而应用类 boldmatter，使选定的内容（文本）变为粗体。

最后的代码块选择 id 为 italicbutton 的元素，并使用 bind()方法为它绑定单击事件。一点击此元素，就执行函数：选择文档中 id 为 info 的元素，删除 CSS 类 boldmatter 而应用类 italicmatter。

初始时的运行结果如图 3-23 所示。

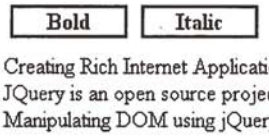


图 3-23　粗体按钮、斜体按钮和一些文本

一点击 Bold 按钮，按钮就会出现悬停效果，HTML 文档 body 的文本变成粗体，如图 3-24 所示。

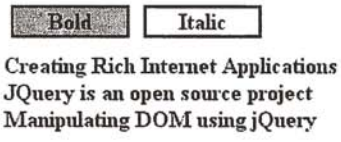


图 3-24　当鼠标移动到 Bold 按钮上方时，悬停样式应用于按钮而粗体样式应用于文档的 body

同样，一点击 Italic 按钮，按钮就出现悬停效果而 HTML 文档 body 的文本变成斜体。

3.15　显示文字气球

问题描述

在网页上有两个按钮，按钮上的文本分别是 Bold 和 Italic。当中任一按钮被单击时，请创建文字气球效果。如果 Bold 按钮被点击，就在文字气球里显示文本 This is Bold menu（如图 3-27 所示），如果 Italic 按钮被点击，就在文字气球里显示文本 This is Italic menu（如图 3-28 所示）。

解决方案

使用以下 HTML 文档来显示 Bold 按钮和 Italic 按钮，如图 3-1 所示：

```
<body>
<span class="bold buttons">Bold</span>
<span class="italic buttons">Italic</span>
</body>
```

创建样式表，包含类选择器 buttons，以便把属性应用到 span 元素，赋予按钮的形状。除此之外，样式表还包含两个样式规则 hover 和 showtip，以便实现悬停效果以及在文字气球中显示文本。样式表文件的内容如下：

```
.buttons{
width: 150px;
float: left;
text-align: center;
color:#000;
background-color:red;
margin: 5px;
font-weight: bold;
}

.hover{
color:red;
background:url(balloon.jpg);
background-repeat:no-repeat;
background-position:bottom;
}

.showtip{
display:block;
margin:25px;
}
```

把样式规则应用到按钮的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.buttons').hover(
    function(event){
      $(this).addClass('hover');
      var $txt=$(this).text();
      $('<span class="showtip"> This is '+$txt+' menu </span>').appendTo($(this));
    },
    function(){
      $(this).removeClass('hover');
      $('.showtip').remove();
    }
  );
});
```

知其所以然

上述样式表所使用的图像文件 balloon.jpg 如图 3-25 所示。

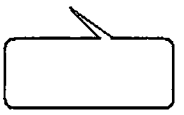


图 3-25　balloon.jpg 文件中的气球图像

样式表中 hover 样式规则包含以下属性。

- color：设置按钮文本的前景色为红色。
- background：显示文字气球图像作为背景。
- background-repeat：设置为 no-repeat，使文字气球图像只显示一次。
- background-position：使得文字气球图像出现在屏幕底部。

样式表文件中，showtip 样式规则包含 display 属性（值设置为 block），以便把文本（在文字气球中显示）作为块元素处理。在块元素的开头和结尾留有空白。showtip 样式规则还包含 margin 属性，值设置为 25 像素，以便设置文本和气球边框之间的间隙。

jQuery 示例代码中把悬停事件分配到 buttons 类，即分配到两个按钮。于是当鼠标指针进入任一按钮，就会首先应用样式规则中所定义的属性，在屏幕上创建空白文字气球，然后把按钮的文本保存到变量$txt 中。

把一个 span 元素追加到按钮的前面。span 元素包含 showtip 类（这样在类选择器 showtip 中定义的属性将自动应用于 span 元素的文本）。把 span 元素的文本内容设置为

```
"This is $txt menu"
```

其中$txt 包含了按钮（鼠标指针在该按钮上悬停）的文本，因此所需的文本将在文字气球中显示。悬停事件的第二个内联函数删除样式规则 hover 和 showtip 中所定义的属性。

初始时，按钮如图 3-26 所示。

图 3-26　没被选中时的 Bold 按钮和 Italic 按钮

鼠标指针一进入 Bold 按钮的范围，就应用样式以改变按钮的前景色和背景色，并在文字气球中显示文本 This is Bold menu，如图 3-27 所示。

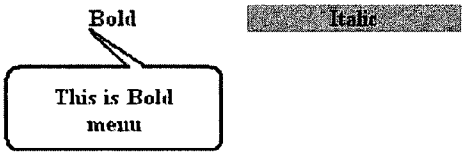


图 3-27　鼠标指针一进入 Bold 按钮的范围，文字气球就显示文本 This is Bold menu

同样，鼠标指针一进入 Italic 按钮的范围，就会改变它的前景色和背景色，文字气球将显示文本 This is Italic menu，如图 3-28 所示。

图 3-28 鼠标指针一进入 Italic 按钮的范围，文字气球就显示文本 This is Italic menu

3.16 创建“返回顶部”链接

问题描述

假设一网页包含大量文本。请在每个文本块之后，显示一个链接，链接上的文字是 Return to Top（返回顶部），为用户导航到网页开头。

解决方案

新建以段落元素形式包含几个文本块的 HTML 文档，如下所示：

```
<body>
<p>Styles make the formatting job much easier and more efficient.
To give an attractive look to web sites, styles are heavily used.
JQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only it is easy to learn, but it's easy to implement too.</p>
<p> A person must have a good knowledge of HTML and CSS and a bit of JavaScript. jQuery
is an open source project that provides a wide range of features with cross-platform
compatibility</p>
</body>
```

在每个段落元素之后创建一个包含文本 Return to top 的链接，以便使文本定位到网页的开头，实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('<a href="#topofpage">Return to top</a>').insertAfter('p');
  $('<a id="topofpage" name="topofpage"></a>').prependTo('body');
});
```

知其所以然

以下语句

```
$('<a href="#topofpage">Return to top</a>').insertAfter('p');
```

在 HTML 文档的每个段落元素之后创建一个包含文本 Return to top 的链接。用户一旦选择该链接，就会导航到 id 为 `topofpage` 的元素。

以下语句

```
$('<a id="topofpage" name="topofpage"></a>').prependTo('body');
```

在 HTML 文档的 body 之前添加了锚点元素（name 和 id 均为 topofpage）。换句话说，在网页的开头创建了 id 为 topofpage 的锚元素。执行前面的 jQuery 代码，运行结果如图 3-29 所示。

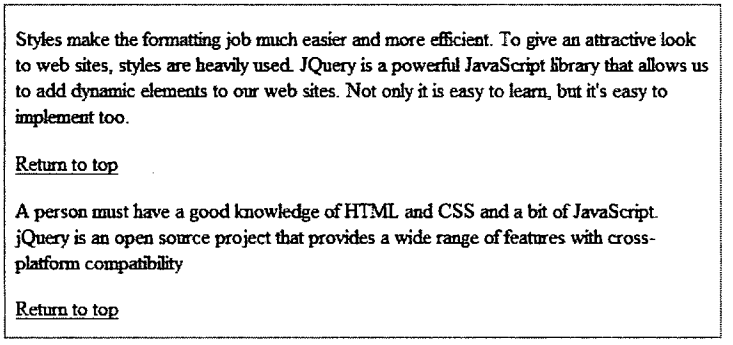


图 3-29　在每个段落之后显示文本 Return to top

点击任何一个“返回顶部”链接，就会导航到网页的开头。

3.17　提供“更多……”链接

问题描述

假设有大块的文本需要显示，为了使屏幕保持整洁，初始时只显示其中的几行文字，连同一个“更多……”链接。当用户选择此链接，才显示文本的其余部分（默认隐藏）。

解决方案

新建 HTML 文档，包含两个 div 元素和在 div 元素之间的 readmore 类的 span 元素，内容如下：

```
<body>
<div>
Styles make the formatting job much easier and more efficient.
</div>
<span class="readmore">Read More...</span>
<div class="message">
To give an attractive look to web sites, styles are heavily used.
JQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only it is easy to learn, but it's easy to implement too. A person must
have a good knowledge of HTML and CSS and a bit of JavaScript. jQuery is an open source
project that provides a wide range of features with cross-platform compatibility.
</div>
</body>
```

编写 jQuery 代码，使下 div 元素不可见，为文本 Read More...添加单击事件和内联函数，使

得单击事件发生时下 div 元素变为可见：

```
$(document).ready(function() {
  $('.message').hide();
  $('span.readmore').click(function(){
    $('.message').show('slow');
    $(this).hide();
  });
});
```

知其所以然

HTML 文档加载之后，最初时显示上 div 元素的文本而隐藏下 div 元素的文本（类名为 message）。与此同时，上 div 元素的文本之后紧跟着的是包含在 readmore 类的 span 元素中（才能把单击事件应用到 Read More...文本）的 Read More...文本。

只有当用户点击 Read More...文本时，下 div 类元素（类名为 message）的文本才变得可见。

jQuery 示例代码中利用 show()和 hide()两个方法，以使文本可见或不可见。

注意 可以使用toggle()方法，使匹配的元素可见（如果原先是隐藏），再一次点击则反过来，使匹配的元素隐藏。

show()使指定的隐藏元素可见。

```
.show(speed, callback)
```

这里的 speed 定义了动画效果的持续时间，即决定元素是否要立即显示或缓慢地显示。速度可被定义为多少毫秒，或预定义的字符串：slow、normal 或 fast（分别代表速度值 600、400 和 200 毫秒）。也就是说，速度定义动画将运行多久。如果省略，则使用默认速度 normal。callback（回调），是动画完成时调用的函数。

hide()使指定的元素不可见：

```
.hide(speed, callback)
```

hide()的 speed 和 callback 与 show()具有相同的含义。以下 jQuery 代码语句

```
$('.message').hide();
```

使 message 类的 div 元素初始时不可见（即只显示上 div 元素的文本），而以下语句

```
$('span.readmore').click(function(){
```

把 click 事件绑定到 readmore 类的 span 元素（即绑定到 Read More... 文本）。

以下代码

```
$('.message').show('slow');
```

当 Read More...文本被点击时，使得 A 类的 div 元素可见，即显示隐藏的文本。以下语句

```
$(this).hide();
```

隐藏 Read More...的文本信息（当不再需要显示时）。

最初只显示一半的文字，底部是一个 Read More...链接，如图 3-30 所示。

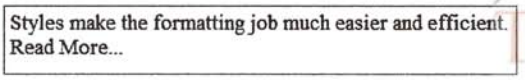


图 3-30　显示上半文本，带着 Read More...链接

当 Read more...文本被选中时，显示隐藏的文本，如图 3-31 所示。

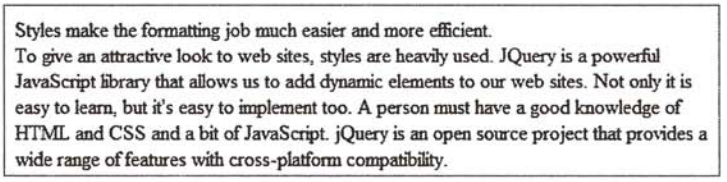


图 3-31　选择 Read More…链接之后，显示全部的文本

1. 更少

稍微修改 jQuery 代码，在显示隐藏的文本时把 Read More...转换为 Read Less...。同时，当 Read Less...被选中时，使显示的文本回到隐藏状态。实现以上功能的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.message').hide();
  $('span.readmore').toggle(function(){
    $('.message').show('slow');
    $(this).text("Read Less…");
  },
  function(){
    $('.message').hide('slow');
    $(this).text("Read More…");
  });
});
```

每次单击 `readmore` 类的 `span` 元素时，以上 jQuery 代码使用 `toggle()` 方法交替应用不同的事件处理函数。第一个事件处理函数将显示隐藏的文本，把 Read More... 文本转换为 Read Less...。第二个事件处理函数隐藏显示的文本，把 Read Less... 文本再次转换为 Read More…。

2. 应用动画效果

在前面的示例中，一选中 Read More... 文本，就显示隐藏的文本。可以使隐藏的文本慢慢显示出来，呈现动画效果，使其更具吸引力。使隐藏的文本以动画效果显示的是 `fadeIn()` 方法，而使显示的文本以动画效果隐藏的是 `fadeOut()` 方法。

注意　`fadeIn()`和`fadeOut()`方法，通过调整所选元素的不透明度而起作用。

`fadeIn()` 方法通过把元素变为不透明而显示选定的元素。

```
.fadeIn(speed, callback)
```

`speed` 指定动画的持续时间。可以指定 `speed` 为预定义的字符串 `fast`、`normal`、`slow` 或毫秒数。毫秒数越大，动画持续的时间越长。默认速度是 `normal`。

callback 是动画完成时调用的函数。

fadeOut()通过淡出使选定的元素变为不可见。

```
.fadeOut(speed, callback)
```

这里的 speed 和 callback 具有和 fadeIn()方法相同的含义。

使隐藏的文本（message 类的 div 元素）慢慢显示的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.message').hide();
  $('span.readmore').click(function(){
    $('.message').fadeIn('slow');
    $(this).hide();
  });
});
```

3.18 以动画效果显示文本

问题描述

网页上有 3 个按钮，即书籍、电影和音乐。点击任一按钮时，以动画效果显示适当的文本。

解决方案

新建 HTML 文件，包含 3 个按钮：书籍、电影和音乐（分配的类名分别是 books、movies 和 music）和 3 个段落。每个按钮被点击时显示对应段落所包含的文本。初始时段落的文本不可见，只有当它的按钮被点击时，才会以动画方式显示对应的文本。HTML 文件的内容如下：

```
<body>
<span class="buttons" id="booksbutton"> Books </span>
<span class="buttons" id="moviesbutton"> Movies  </span>
<span class="buttons" id="musicbutton"> Music </span><br><br>
<p class="books">Books on a range of different subjects available at reasonable prices.
Ranging from web development, programming languages, and text books, all are available
at heavy discount. Shipping is free. Also available in stock are popular magazines,
e-books, and tutorial CDs at affordable prices.</p>
<p class="movies">Find new movie reviews & the latest Hollywood movie news. Includes
new movie trailers, latest Hollywood releases, movie showtimes, entertainment news,
celebrity interviews etc. Also find Hollywood actresses, actors, videos, biographies,
filmography, photos, wallpaper, music, jokes, and live TV channels at your doorstep.</p>
<p class="music">Find music videos, internet radio, music downloads and all the latest
music news and information. We have a large collection of music and songs classified
by type, language, and region. All downloads are streamed through RealAudio. You can
also watch free music videos, tune in to AOL Radio, and search for your favorite music
artists.</p>
</body>
</body>
```

样式表的内容如下：

```
.buttons{
width: 100px;
float: left;
text-align: center;
margin: 5px;
border: 2px solid;
font-weight: bold;
}
```

初始时隐藏文本，当按钮被点击时以动画方式显示对应段落的内容：

```
$(document).ready(function() {
  $('.books').hide();
  $('.movies').hide();
  $('.music').hide();

  $('#booksbutton').click(function(){
    $('.books').show('slow');
    $('.movies').hide();
    $('.music').hide();
  });
  $('#moviesbutton').click(function(){
    $('.movies').show('slow');
    $('.books').hide();
    $('.music').hide();
  });

  $('#musicbutton').click(function(){
    $('.music').show('slow');
    $('.books').hide();
    $('.movies').hide();
  });
});
```

知其所以然

在 HTML 文件中，把文本 Books、Movies 和 Music 包含在 buttons 类的 span 元素中，以便样式表中定义的类选择器 buttons 的属性自动应用到文本。类选择器 buttons 包含的属性，把按钮的形状赋予文本。同时，分别给 3 个按钮分配唯一的 id 即 booksbutton、moviesbutton 和 musicbutton，以便绑定各自的单击事件。在事件处理函数中编写代码隐藏旧文本（来自上一次单击按钮的信息）以及显示当前单击按钮的相关信息。

上面的 jQuery 代码初始时隐藏 books、movies 和 music 类的段落的文本，即加载网页之后初始时只有按钮可见。然后，把单击事件绑定到 id 为 booksbutton 的 HTML 元素（Books 按钮）。一点击 Books 按钮，books 类的段落内容（包含与图书相关的信息）慢慢变为可见，带着动画效果。于此同时，使 movies 和 music 类的段落内容不可见，只显示屏幕上被单击按钮的相关信息。

初始时只显示 3 个按钮 Books，Movies 和 Music，如图 3-32 所示。

图 3-32 3 个按钮：书籍、电影和音乐

一点击 Books 按钮，就会显示与之有关的信息，即显示 books 类的段落内容，如图 3-33 所示。

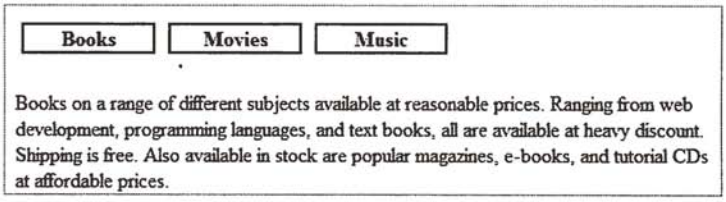


图 3-33 一点击 Books 按钮，就显示有关图书的信息

同样，如果点击 Movies 按钮，就会隐藏与书籍或音乐有关的信息，而在屏幕上以动画效果慢慢显示与电影有关的信息（即 movies 类的段落）如图 3-34 所示。

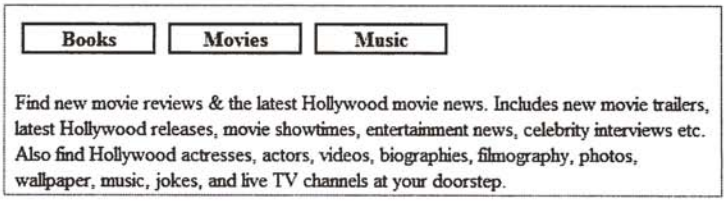


图 3-34 一点击 Movies 按钮，就显示有关电影的信息

添加Mouseover事件

除了为按钮应用单击事件，还可以为按钮附加 mouseover 事件。当鼠标指针进入按钮时（不必单击按钮）mouseover 事件将使适当的信息显示出来。附加 mouseover 事件的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.books').hide();
  $('.movies').hide();
  $('.music').hide();

  $('#booksbutton').mouseover(function(){
    $('.books').show('slow');
    $('.movies').hide();
    $('.music').hide();
  });

  $('#moviesbutton').mouseover(function(){
    $('.movies').show('slow');
    $('.books').hide();
    $('.music').hide();
  });
```

```
  $('#musicbutton').mouseover(function(){
    $('.music').show('slow');
    $('.books').hide();
    $('.movies').hide();
  });
});
```

3.19　以滑动效果来替换文本

问题描述

请以一段文本替换另一段文本，使被取代的文本慢慢变得不可见，与此同时，另一段文本渐渐显示出来。

解决方案

本解决方案使用 css()方法直接把 CSS 样式应用到元素，而不是从样式表中选取样式（关于 css()的细节请看攻略 3.6）。

假设 HTML 文档包含两个段落元素，分别给它们分配 id 为 message1 和 message2，以示区分。

```
<body>
<p id="message1">jQuery is an open source project</p>
<p id="message2">Manipulating DOM using jQuery</p>
</body>
```

以下 jQuery 代码应用了滑动效果：

```
$(document).ready(function() {
  $('p#message1').css({'border': '2px solid', 'text-align':
'center','font-weight':'bold'}).hide();
  $('p#message2').css({'backgroundColor': '#f00','color':'#fff','text-align':
'center',
'font-weight':'bold'}).click(
    function(){
      $(this).slideUp('slow');
      $('p#message1').slideDown('slow');
    }
  );
});
```

知其所以然

给 id 为 message1 的段落分配了宽为 2 像素的实线边框，把段落的文本对齐于浏览器窗口的中心，并设置为以粗体显示。把 id 为 message2 的段落的背景色设置为红色，把文本的前景色设置为白色，也对齐于浏览器窗口的中心。

在 jQuery 示例代码中，利用 slideUp()方法使一段文本逐渐消失，利用 slideDown()方法

使另一段文本逐渐显示，替换消失的那段文本。下面请看这两个方法的工作机制。

slideDown()以滑动效果来逐渐显示选定元素。

```
.slideDown(speed, callback)
```

Speed 指定动画的持续时间。可以指定字符串 fast、normal 或 slow 或毫秒数。毫秒数越大，动画持续的时间越长。

callback，是动画完成后调用的函数。

相反，slideUp()以活动效果逐渐使选定元素消失不可见。

```
.slideUp(speed, callback)
```

slideDown()的 speed 和 callback 具有和 slideUp()的相同的含义。

jQuery 示例代码使 id 为 message1 的段落隐藏，而把 id 为 message2 的段落设置为可见。因此，初始时输出如图 3-35 所示。另外，把单击事件绑定到可见段落元素。在单击事件的内联函数中，编写代码使可见段落元素“上滑”（id 为 message2 的段落）逐渐变得不可见，而使初始时不可见的段落元素“下滑”（id 为 message1 的段落）逐渐变得可见。

图 3-35 应用 slideUp()和 slideDown()

单击可见段落，它会慢慢变为不可见，初始时不可见的段落慢慢变为可见，如图 3-36 所示。

图 3-36 一个段落正处于下滑过程，另一个段落正处于上滑过程

后者完全取代前者，如图 3-37 所示。

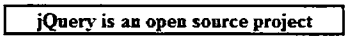


图 3-37 一个段落完全消失，另一个段落显示出来

3.20 使图像滚动

问题描述

点击图像之后，使图像从左到右滚动。

解决方案

创建 HTML 文件，读取 cell.jpg 文件显示一个图像，如下所示：

```
<body>
<img src="cell.jpg" />
```

```
</body>
```

为了使元素从当前位置移动，必须应用 position（位置）属性。因此，对 img 元素应用 position 属性，在样式表文件 style.css 中利用类型选择器把 position 属性值设置为 relative（相对）：

```
img{
position:relative;
}
```

使图像从左到右滚动的 jQuery 代码如下：

```
$(document).ready(function() {
  $('img').click(function(){
    $(this).animate({left:600}, 'slow')
  });
});
```

1. 滚动之后使图像隐藏

希望图像从左到右滚动，一旦抵达浏览器窗口右侧，就调用 slideUp()方法使图像变为不可见。修改之后的 jQuery 代码如下：

```
$(document).ready(function() {
  $('img').click(function(){
    $('img').animate({left: 600}, 'slow')
    $('img').slideUp('slow');
  });
});
```

2. 滚动之后使图像淡出

我们希望图像从左到右滚动，一旦抵达浏览器窗口右侧，就使图像变为不可见。这回使用 fadeTo()方法。

修改之后的 jQuery 示例代码如下：

```
$(document).ready(function() {
  $('img').click(function(){
    $('img').animate({left: 600}, 'slow')
    $('img').fadeTo('slow',0)
  });
});
```

3. 使图像一边向右滚动一边扩大

随着图像从左到右滚动，同时扩大图像的宽和高。实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('img').click(function(){
    $(this).animate({left:600, width:$(this).width()*2,height:$(this).height()*2},'slow');
  });
});
```

4. 使图像滚动到右侧再向左滚动

希望图像滚动到浏览器窗口的右侧。一抵达右侧就使图像淡出，然后再次淡入（即慢慢变为不可见，然后变为可见）。之后使图像滚回左侧起始位置。

实现此效果的 jQuery 代码如下：

```
$(document).ready(function() {
  $('img').click(function(){
    $('img').animate(
      {left: 600},
      'slow',
      function(){
        $('img').fadeTo('slow',0);
        $('img').fadeTo('slow',1);
        $('img').animate({left: 0}, 'slow')
      }
    );
  });
});
```

知其所以然

本解决方案将利用 animate() 方法。首先了解 animate() 方法的工作机制。

1. **animate()**

此方法用于把自定义动画应用到选定的元素。根据指定的 CSS 属性和使用的缓和参数来控制自定义动画。

```
.animate(properties, speed, easing, callback)
```

在此示例代码中

- properties 是指 CSS 属性（终值），即该元素在动画完成后达到的值。

speed 是预定义的字符串或以毫秒为单位的数值，决定效果（或动画）的持续时间。预定义的字符串是 slow、normal 和 fast。如果不指定 speed 参数，则默认速度为 normal。

easing（缓和）是一个函数（可选），控制动画随着时间如何进行。它需要一个插件。有两个缓和函数：linear（线性）和 swing（摆动）。默认值为 swing。

callback，是动画完成后调用的函数。

jQuery 示例代码的以下语句：

```
$(this).animate({left:600}, 'slow')
```

使图像从当前位置缓慢地移动到距离浏览器窗口左侧 600 像素之处。在图像上发生的单击事件，会触发该语句的执行。初始时图像在最左侧显示，如图 3-38 所示。

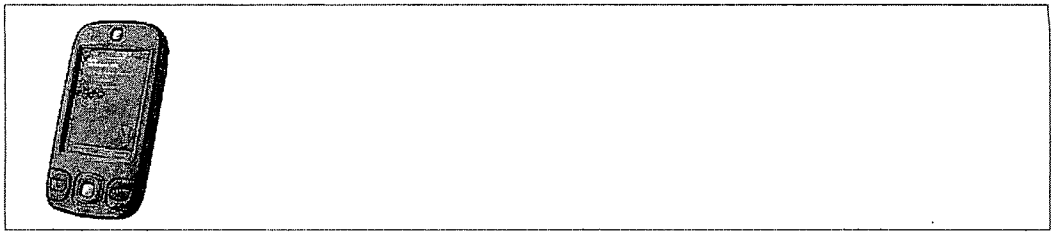


图 3-38 初始时图像在最左侧

当图像被点击时，就会向右移动，并在距离浏览器窗口左侧 600 像素之处停止，如图 3-39 所示。

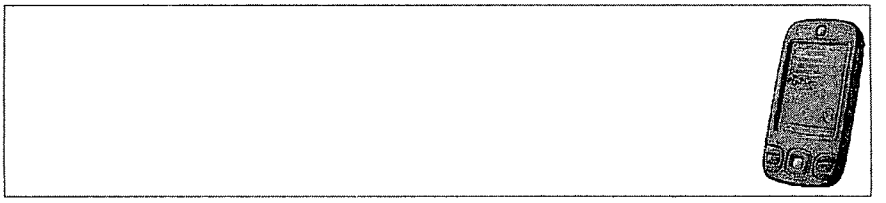


图 3-39　一旦被点击，图像将滚动到右侧

jQuery 示例代码首先使图像滚动，然后使之隐藏。单击图像使图像由左至右滚动，并在距离浏览器窗口左侧 600 像素之处停止。然后调用 slideUp()方法使图像缓慢地隐藏起来。

在 jQuery 示例代码中，使图像滚动到右侧，再折返向左滚动，以下语句

```
$('img').animate({left: 600}, 'slow', function(){
```

使图像缓慢地从左往右滚动，在距离浏览器窗口左侧 600 像素之处停止。

最后，以下语句

```
$('img').animate({left: 0}, 'slow')});
```

使图像回滚（从右往左）到起始位置，在距离浏览器窗口左侧 0 像素之处停止。

2. fadeTo()

此方法调整选定元素的不透明度。

```
.fadeTo(speed, opacity, callback)
```

Speed 指定动画的持续时间。可以指定为预定义的字符串 fast、normal，slow，或以毫秒为单位的数值。毫秒数越大，动画持续的时间越长。

opacity（不透明度）是介于 0 和 1 之间的数值，决定了动画完成时的不透明度值。值为 0 意味着选定元素不可见，1 意味着选定元素完全可见，0.5 意味着选定元素模糊灰暗。

callback（回调），是动画完成后调用的函数。从使图像滚动后淡出的 jQuery 示例代码可见，单击图像之后，图像向右移动，并在距离浏览器窗口左侧 600 像素之处停止。

一抵达右侧，图像就慢慢消失，因为它的不透明度设置为 0。

使用回调函数可以达到同样的效果：

```
$(document).ready(function() {
  $('img').click{function(){
    $('img').animate(
      {left: 600},
      'slow',
      function(){
        $('img').fadeTo('slow',0)
      }
    );
  });
});
```

jQuery 示例代码使图像一边向右滚动一边增大尺寸，在单击事件的事件处理函数中，编写代码使图像从左往右滚动，使它在距离浏览器窗口左侧边框 600 像素之处停止，并使它抵达右侧边框时宽度和高度变为初始值的两倍。这就使得图像缓慢地向右滚动，同时使其宽度和高度逐渐增加。

初始时,图像如图 3-38 所示。图像一旦被单击，就开始向右移动，抵达右侧边框时图像如图 3-40 所示。

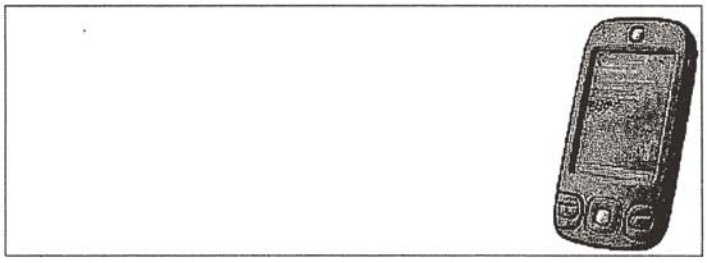


图 3-40 抵达右侧时图像的宽和高变为初始值的两倍

以下语句

```
$('img').fadeTo('slow',0);
```

把图像的不透明度设置为 0，从而使图像慢慢淡出。

同样，以下语句

```
$('img').fadeTo('slow',1);
```

使图像淡入，即把不透明度设置为 1，从而使图像再次慢慢地变为可见。

3.21 确定被按下的键

问题描述

用户往输入字段中输入数据时想要知道哪个键被按下，这种方式可用于合法性检查等。

解决方案

新建 HTML 文件，包含一个输入文本字段和一个空段落元素，内容如下：

```
<body>
<input type="text" class="infobox" />
<p></p>
</body>
```

显示用户按下键的数字代码，实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.infobox').keypress(function(event){
    $('p').text('Character typed is: '+event.keyCode);
  });
```

```
});
```

把按下键的数字代码转换为字符格式，实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.infobox').keypress(function(event){
    $('p').text('Character typed is: '+String.fromCharCode(event.keyCode));
  });
});
```

知其所以然

HTML 文档在空段落中显示用户按下了什么键。调用 keypress()方法去查明用户按下的键。为了进一步应用 keypress()方法，首先了解该方法及其相关方法。

1. keypress()

keypress()把事件处理函数绑定到指定元素，当用户在键盘上按键时就触发事件。keydown()和 keypress()的区别是，如果用户反复按任意键，按下然后保持按下状态，keydown()事件只执行一次，而 keypress()事件则是每个插入一个字符就执行一次(keydown()的持续时间更长一些)。此外，修改键 Shift 键、Ctrl 键等为 keydown()所识别，而这些修改键不会触发 keypress()事件。

```
.keypress(handler)
.keypress()
```

这里的 handler 包含每次某个键盘键被按下时将要执行的语句。

不带参数的 keypress()方法用于手动触发按键事件。以下示例中， button 类的 HTML 元素被单击时，就手动触发类 infobox 的输入文本字段的按键事件：

```
$('.button').click(function(){
  $('.infobox').keypress();
});
```

为了配合以上代码触发的 keypress()，需要编写事件处理函数来完成所需的任务。

为了获知哪个键被按下，需要使用传递到事件处理函数的事件对象。事件对象包含有关事件的信息。事件对象的 keyCode（键码）属性值即为按下键的数字代码。

以下 jQuery 代码

```
$('.infobox').keypress(function(event){
```

把按键事件绑定到输入文本字段，监听任何按下的键。

以下语句

```
$('p').text('Character typed is: '+event.keyCode);
```

利用（传递到事件处理函数的）事件对象的 keyCode 属性，每当（在输入文本字段）发生按键事件时，就在段落元素中显示按下键的数字代码。运行结果如图 3-41 所示。

从图中可见，段落元素中显示着字符 k 的数字代码。

为了把按下键的数字代码转换为字符格式，使用 JavaScript 的 String 对象的 fromCharCode()方法。因此把事件对象的 keyCode 属性传递给 fromCharCode()方法，以字

符格式显示按下的键。输出如图 3-42 所示。

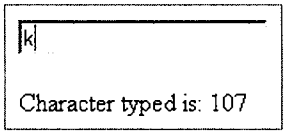


图 3-41 显示文本字段中输入字符所对应的数字代码

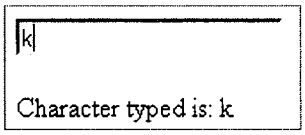


图 3-42 在文本字段中键入的字符，显示为字符格式

2. keydown()

keydown()把事件处理程序绑定到指定的元素，一旦用户在键盘上按下某键，就会触发事件处理程序的执行：

```
.keydown(handler)
.keydown()
```

这里的 handler 包含用户在键盘上按下某键时想要执行的语句。

不带参数的 keydown()方法用来手动触发 keydown 事件。在以下示例中，当 button 类的 HTML 元素被单击时，就手动触发 infobox 类的输入文本字段的 keydown 事件：

```
$(' .button').click(function(){
  $('.infobox').keydown();
});
```

为了配合以上代码触发的 keydown()事件，需要编写事件处理函数来完成所需的任务。

3. keyup()

keyup()把事件处理函数绑定到指定的元素，一旦用户在键盘上释放某键，就会触发事件处理函数的执行。

```
.keyup(handler)
.keyup()
```

这里的 handler 包含用户在键盘上释放某键时想要执行的语句。

不带参数的 keyup()方法被用来手动触发 keyup 事件。在以下示例中，当 button 类的 HTML 元素被单击时，就手动触发 infobox 类的输入文本字段的 keyup 事件：

```
$(' .button').click(function(){
  $('.infobox').keyup();
});
```

为了配合以上代码触发的 keyup()事件，需要编写事件处理函数来完成所需的任务。

3.22 防止事件冒泡

问题描述

假设面临事件冒泡的问题，如何阻止意外结果的发生？

解决方案

为了理解事件冒泡，假设 HTML 文档包含层层嵌套的几个元素：

```
<body>
<div>Div Element
<p>Paragraph Element<br/>
<span>Span Element</span>
</p>
</div>
</body>
```

为了确切地理解事件冒泡的概念，请看以下 jQuery 代码，把单击事件附加到 DOM 的所有 3 个元素：

```
$(document).ready(function() {
  $('div').click(function(event){
    alert('You have clicked the div element');
  });

  $('p').click(function(event){
    alert('You have clicked the paragraph element');
  });

  $('span').click(function(event){
    alert('You have clicked the span element');
  });
});
```

为了阻止事件冒泡，并控制事件应该发生在何处，修改 jQuery 代码如下：

```
$(document).ready(function() {
  $('div').click(function(event){
    var $target=$(event.target);
    if($target.is('div')){
      alert('You have clicked the div element');
    }
    if($target.is('p')){
      alert('You have clicked the paragraph element');
    }
    if($target.is('span')){
      alert('You have clicked the span element');
    }
  });
});
```

知其所以然

以上 HTML 文档中，DOM 最上层元素（即根元素）是 `div` 元素，随后是段落元素（作为 `div` 元素的子元素），最后是 `span` 元素。

当任何元素上发生事件时，事件处理机制首先检查该元素是否附加了事件方法（以及事件处理函数）。如果是，就执行（附加的事件方法的）事件处理函数的语句。在此之后，事件处理机

制继续检查该元素的亲节点，看是否附加了事件方法，如果是，也会执行其事件处理函数。继续检查其亲元素（即亲元素的亲元素），如此类推。换句话说，事件冒泡指的是被触发的事件在 DOM 中向上传播直到 DOM 根的过程。

第一个 jQuery 示例中，把单击事件附加到所有 3 个元素。请单击示例中的 span 元素（在 DOM 的最下层节点）。span 元素的事件处理函数会被执行。但事件处理机制并不就此打住，而是将其传播到亲元素，即段落元素将被检查，因为它也有附加的单击事件，段落元素的事件处理函数也会被执行。在这之后，DOM 根（即 div 元素）的事件处理函数，也会被执行。

现在执行示例的 jQuery 代码，单击 span 元素，首先看到因执行 span 元素的事件处理函数而出现的警报消息，如图 3-43 所示。

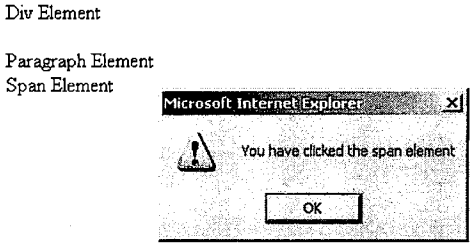


图 3-43 单击 span 元素，显示 span 元素的警报消息

在这之后，span 元素的亲元素的事件处理函数（即段落元素）也会被执行，显示的警报消息如图 3-44 所示。

因为事件冒泡在 DOM 根处停止，根（即 div 元素）的事件处理函数也被执行，输出如图 3-45 所示。

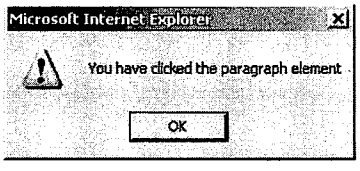


图 3-44 由于事件冒泡而执行段落元素的事件处理函数，显示警报消息

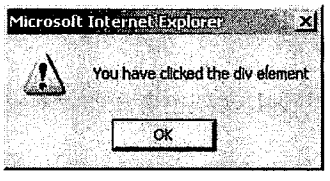


图 3-45 div 元素的警报消息，因事件冒泡而被触发执行

假设单击段落元素，就会看到段落元素的警报消息，紧跟着 div 元素的警报信息。如果点击 div 元素，就只显示 div 元素的警报信息，因为 div 元素没有亲元素可供向其传播。

现在运行停止事件冒泡的 jQuery 代码。利用事件对象的 target（目标）属性，控制事件传播，并停止事件冒泡。利用 target 属性，可以确定哪个 DOM 元素首先接收到事件，并确保该事件不会自动传播到其亲元素。

从修改后的 jQuery 代码可见，确定目标元素（在该元素上发生单击事件）并将其存储在变量 $target 中。然后使用 if 语句逐个检查哪个元素接收到了单击事件，据此发出警报消息。

3.23　链接多个活动

问题描述

链接是一个过程，逐个地编写 jQuery 包装器方法，以便在一个语句中执行几个活动。设法利用 jQuery 的链接功能。

解决方案

假设 HTML 文档中包含几个 HTML 元素，内容如下：

```
<body>
<div>Div Element
<p>Paragraph Element<br>
<span>Span Element</span>
</p>
</div>
</body>
```

此外还需要样式表，以便应用几个 CSS 样式到 HTML 元素，为此在样式表中编写一个样式规则如下：

```
.hover{
color: blue ;
background-color:cyan
}
```

链接的示例如下：

```
$(document).ready(function() {
  $('div').children().clone().prependTo('div').addClass('hover');
});
```

知其所以然

以上 HTML 文档包含 3 个元素，如图 3-46 所示。

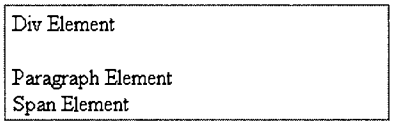


图 3-46　3 个元素：分区（div）、段落（paragraph）和跨度（span）

前面的样式表中包含的样式规则 hover 分别把 HTML 元素的前景色和背景色改变为蓝色和青色。

前面的示例代码选择 div 元素的子元素（即段落元素和 span 元素）并克隆（复制）它们。然后在 div 元素前插入段落元素和 span 元素的副本。此后把 hover 样式规则应用到插入的段落元素和 span 元素。由此可见 jQuery 的链接功能，可以在一个语句中完成多个活动。前面的 jQuery 代码的运行结果如图 3-47 所示。

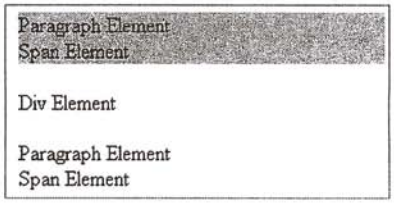


图 3-47 把 div 元素的子元素的副本插入到 div 元素之前，并为它们应用样式

删除链接副作用

链接处理最新的包装集。比如前面的 jQuery 代码创建 div 元素的子元素的副本。副作用是现在有两个包装集，一个是 div 元素的原始的子元素，另一个是它们的副本。在这种情况下，子元素的副本被插入到 div 元素之前，并只为克隆的包装集(子元素的副本)应用 CSS 类 hover，不为原来的包装集应用 CSS 类 hover。为了针对原包装集，也就是说，如果只想为 div 元素的子元素（而不是其副本）应用 CSS 样式规则，则应该调用 end()方法。

在 jQuery 链接中使用 end()方法，可以退回到上一个包装集合，并把上一个包装集作为返回值，以便应用随后的操作。

修改前面的 jQuery 代码，把 hover 样式规则应用到 div 元素的实际的子元素（而不是其副本）：

```
$(document).ready(function() {
  $('div').children().clone().prependTo('div').end().addClass('hover');
});
```

以上 jQuery 代码中创建 div 元素的子元素的副本（即段落元素和 span 元素的副本），并插入到 div 元素之前。然而，由于 end()命令把原来的包装集（div 元素的实际的子元素）作为返回值（而不是把其副本作为返回值），hover 样式被应用到原来的包装集，而不是应用到其副本。运行结果如图 3-48 所示。

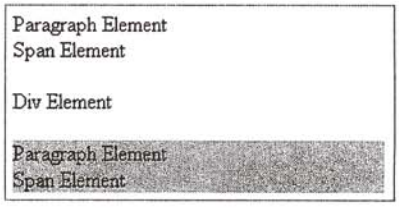


图 3-48 把 hover 样式规则应用到 div 元素的实际子元素，而不是其副本

3.24 小结

本章中讲述了各种不同的鼠标事件和键盘事件，比如单击、双击、按键、keydown 和 keyup，为图像应用各种效果（如淡入、淡出、上滑、下滑等）。下一章将学习表单验证攻略，学习如何验证不同的表单字段以及查明表单的哪个 HTML 元素被选中（例如是否被用户勾选）。最后将学习如何禁用或启用不同的表单元素。

第4章

表单验证

本章讲述表单验证的不同方法，从基本的空白字段检查到更为复杂的密码和字段禁用技术，包括以下攻略：

- 确认必需字段不留空；
- 验证数字字段；
- 验证电话号码；
- 验证用户 ID；
- 验证日期；
- 验证电子邮件地址；
- 检查复选框是否被选中；
- 检查单选按钮是否被选中；
- 检查 select 元素的选项是否被选中；
- 把样式应用到选项和表单按钮；
- 一步勾选或取消所有的复选框；
- 验证两个字段；
- 验证密码和确认密码字段是否匹配；
- 禁用某些字段；
- 验证整个表单；
- 表单数据序列化。

4.1 确认必需字段不留空

问题描述

假设表单中有一个文本输入字段，请确保该字段不留空白。如果用户没有往必须字段中输入数据，就显示错误消息。

解决方法

新建 HTML 文档，显示包含标签、文本字段、错误消息和提交按钮的表单，如下所示：

```
<body>
<form id="signup" method="post" action="">
<div><span class="label">User Id *</span><input type="text"  class="infobox"
name="userid" /><span class="error"> This field cannot be blank</span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

因为本 HTML 表单的目的是验证文本输入字段，而不是把输入的数据发送到其他页面处理，所以将表单的 action 属性留白。给表单指定 id 为 signup，把 method 属性设置为 post（虽然设置 method 属性对验证过程不起什么效果）。

把标签消息 User Id 包含在 label 类的 span 元素中。为文本输入字段指定 infobox 类，把错误消息（此字段不能为空）存储在 error 类的 span 元素中。最后，为提交按钮指定 submit 类。

之所以为表单中的所有 4 个项目指定类名，是为了给表单的 4 个项目分别自动应用定义在类选择器 label、infobox、error 和 submit 中的属性（在 style.css 样式表中定义）。包含各个类选择器的样式表如下：

```
style.css
.label {float: left; width: 120px; }
.infobox {width: 200px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 125px; margin-top: 10px;}
```

确认文本输入字段不是空白，如果留空就显示错误消息，实现的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var data=$('.infobox').val();
    var len=data.length;
    if(len<1)
    {
      $('.error').show();
      event.preventDefault();
    }
    else
    {
      $('.error').hide();
    }
   });
});
```

知其所以然

类选择器 label（标签）具有值为 left 的 float 属性，使下一个项目（文本输入字段）出现在它的右侧，把它的 width（宽度）属性设置为 120 像素，给予足够的空间来显示标签。把类选择器 infobox 的 width 属性（文本输入字段的大小）设置为 200 像素。把类选择器 error 的颜色属性设置为红色，以突出显示错误消息，左侧填充设置为 10 像素，以便与文本输入字段

保持适当的距离。把类选择器.submit 的 margin-left 属性和 margin-top 属性分别设置为125像素和10像素，设置了距浏览器窗口左侧的距离和距文本输入字段的距离，因为希望提交按钮显示在文本输入字段的下方。

前面的jQuery代码中，初始时错误消息是隐藏的，而且把单击事件附加到提交按钮。在单击事件的事件处理函数中，提取用户在文本输入字段中输入的数据（指定类名 infobox）并存储在变量 data 中。计算数据的长度，如果小于1（即用户在文本输入字段中没有输入任何数据），就在屏幕上显示错误消息。事件对象的 preventDefault()方法用于防止提交按钮把用户输入的数据发送到服务器。

执行jQuery代码，就会显示包含文本标签、文本输入字段和提交按钮的表单。如果不在文本输入字段中输入任何数据就单击提交按钮，就会在文本输入字段旁边显示红色文字的错误消息This field cannot be blank（此字段不能为空），如图4-1所示。

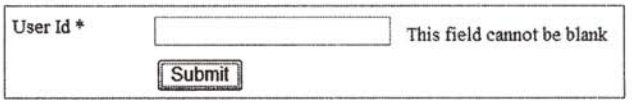


图4-1　留空文本输入字段时显示错误消息

如果在文本字段中输入名字之后单击提交按钮，就不会看到任何错误消息，如图4-2所示：

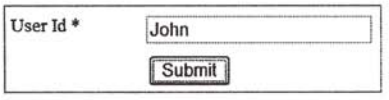


图4-2　如果在文本字段中输入数据，就不会显示错误消息

4.2　验证数字字段

问题描述

假设有一个文本字段用于输入人的年龄，请确认输入的是数字而不是字符或符号。

解决方案

新建HTML文档，包含标签、文本输入字段、错误消息和提交按钮，内容如下：

```
<body>
<form id="signup" method="post" action="">
 <div><span class="label">Enter Age </span><input type="text"  class="infobox"
name="age" /><span class="error"> Only numericals allowed</span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

因为HTML表单的目的只是验证文本输入字段，而不是把输入的数据发送到其他页面进行处理，所以将表单的 action 属性留白。把表单ID指定为 signup，method 属性设置为 post

(虽然这对验证过程不起什么效果)。

把标签消息 User Id 包含在 label 类的 span 元素中。为文本输入字段指定类名 infobox，把错误消息（此字段不能为空）存储在 error 类的 span 元素。最后，为提交按钮指定类名 submit。

之所以为表单中的所有 4 个项目指定类名，是为了给表单的 4 个项目分别自动应用定义在类选择器 label、infobox、error 和 submit 中的属性（在 style.css 样式表中定义）。包含各个类选择器的样式表如下：

```
style.css
.label {float: left; width: 120px; }
.infobox {width: 200px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 125px; margin-top: 10px;}
```

1. 只允许输入数字

检查在文本输入字段中输入的年龄是否仅包含数字，而没有文字或符号，jQuery 代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var data=$('.infobox').val();
    var len=data.length;
    var c;
    for(var i=0;i<len;i++)
    {
      c=data.charAt(i).charCodeAt(0);
      if(c <48 || c >57)
      {
        $('.error').show();
        event.preventDefault();
        break;
      }
      else
      {
        $('.error').hide();
      }
    }
  });
});
```

2. 允许负值

有时，在输入数字时需要输入负值。在前面的 jQuery 代码中，不允许输入任何符号，当然也不能使用连字符或减号。因此，前面的代码让我们无法在文本输入字段中输入负值。下面 jQuery 代码，允许输入负值：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var data=$('.infobox').val();
    var len=data.length;
    var c;
```

```
    for(var i=0;i<len;i++)
    {
      c=data.charAt(i).charCodeAt(0);
      if(c==45 && i==0)
      {
        continue;
      }
      if(c <48 || c >57)
      {
        $('.error').show();
        event.preventDefault();
        break;
      }
      else
      {
        $('.error').hide();
      }
    }
   });
});
```

3. 限定值的范围

假设表单中的年龄字段允许输入值的范围是5到99，也就是说，如果输入值大于或小于指定的年龄范围，就在屏幕上显示错误消息。

限定输入数字5到99的jQuery代码如下所示：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var data=$('.infobox').val();
    var len=data.length;
    var c=0;
    var age=0;
    var flag=0;
    for(var i=0;i<len;i++)
    {
      c=data.charAt(i).charCodeAt(0);
      if(c <48 || c >57)
      {
        $('.error').show();
        flag=1;
        event.preventDefault();
        break;
      }
      else
      {
        $('.error').hide();
      }
    }

    if(flag==0)
    {
      age=parseInt(data);
```

```
        if(age<5 || age>99)
        {
          $('.error').show();
          $('.error').text('Invalid Age. Please enter the age within the range 5 to 99');
          event.preventDefault();
        }
      }
    });
  });
```

知其所以然

在 HTML 文档中，把标签设置为显示文本 Enter Age（请输入年龄）而把错误消息指定为文本 Only numerals allowed（只允许输入数字）。此外，为标签、文本输入字段、错误消息和提交按钮指定不同的类名（label、infobox、error 和 submit），以便为它们应用类选择器中定义的属性（定义在样式表 style.css 中）。本攻略使用与攻略 4.1 相同的外部样式表 style.css。

上面的 jQuery 示例代码“只允许输入数字”一节中把错误消息设置为初始时不可见。 把单击事件附加到提交按钮，如果单击提交按钮，就执行事件处理函数。从事件处理函数中可见，检索在文本输入字段中输入的数据（文本输入字段包含在 infobox 类的 span 元素中）并储存到名为 data 的变量中。计算输入数据的长度，执行 for 循环（执行次数由输入数据的长度决定）逐个分析输入数据的每个字符。

在 for 循环中，一次处理（输入数据的）一个字符，利用 charCodeAt()方法查找其 ASCII 值。如果字符的 ASCII 值小于数字 0（ASCII 值 48）或大于数字 9（ASCII 码值 57），就意味着不是数字，那么就使错误消息在屏幕上可见并退出循环。调用 event 对象的 preventDefault()方法防止把用户输入的数据发送到服务器或把用户导航到目标表单。

执行该程序，在年龄字段中输入一些字符，就会看到错误消息 Only numerals allowed，如图 4-3 所示。

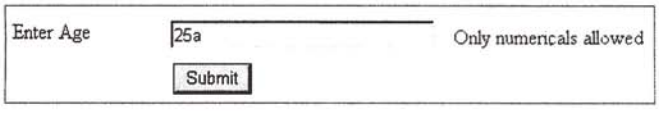


图 4-3 如果在年龄字段中输入字母，就显示错误消息

即使把字符夹在数字之间，仍然会显示错误消息，如图 4-4 所示。

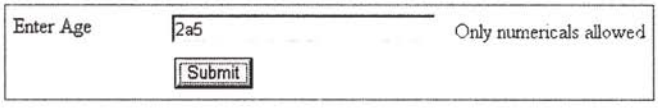


图 4-4 如果数字之间出现其他字符，也显示错误消息

此外，如果添加任何一个符号，比如减号或下划线，也会显示错误消息，如图 4-5 所示。

图 4-5　如果输入非数字符号，也显示错误消息

上面的 jQuery 示例代码“允许负值”一节中把错误消息设置为初始时隐藏。把单击事件的事件处理函数附加到提交按钮，以便从文本输入字段中提取数据并存储到变量 data 中。利用 for 循环逐个解析存储在变量 data 中的每个字符。在循环中把变量 data 中的每个字符转换为 ASCII 码值，并检查第一个字符的 ASCII 码值是否为 45（负号的 ASCII 码值），继续检查其余字符而不显示任何错误消息。for 循环中的第二个条件语句如上所述：检查在变量 data 中的字符是否是数字，如果不是，就显示错误消息。

上面的 jQuery 示例代码“限定值的范围”一节中把错误消息设置为初始时不可见。把单击事件附加到提交按钮。在单击事件的事件处理函数中完成几件事情：提取在文本输入字段（指定类名为 infobox）中输入的数字，并存储到变量 data 中。计算 data 的长度，执行循环，逐个解析每个字符。如果 data 的内容之中有任何一个字符的 ASCII 码值小于数字 0（ASCII 码值为 48）或大于数字 9（ASCII 码值为 57），就意味着变量 data 包含非数字，于是使错误消息可见。另外，把变量 flag 的值设置为 1（表明只允许数字）并退出循环。如果变量 flag 的值被设置为 1，就意味着不必检查数字的范围，因为它已是无效的数据。

如果输入数据是数字，就需要进一步检查数字的范围，也就是说，是否在 5 到 99 之间。因此，如果变量 flag 值为 0（执行 for 循环之后，即在检查变量 data 的所有字符之后），就意味着输入的数据有效（只包含数字）。然后执行第二个条件语句，如果数字小于 5 或大于 99，则使错误消息可见并把错误消息设置为 Invalid Age（无效年龄）。调用事件对象的 preventDefault()方法（在输入数据无效的情况下）防止输入的数据被提交。如果值不在限定范围 5 到 99 之内，就显示错误消息，如图 4-6 所示。

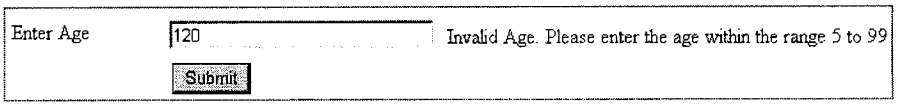


图 4-6　如果值不在 5 至 99 的范围内，就显示错误消息

4.3　验证电话号码

问题描述

假设在电话号码字段中用户只能输入数字和+或-号，不能输入其他字符。

解决方案

新建 HTML 文档，显示包含文本输入字段、错误消息和提交按钮的表单。HTML 文档的内

容如下：

```
<body>
<form id="signup" method="post" action="">
<div><span class="label">Enter Phone number </span><input type="text" class="infobox"
name="phone" /><span class="error"> Phone number can contain only numbers, + and
-</span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

为HTML文档中的所有4个项目分别指定类名label、infobox、error和submit，因此在类选择器（在外部样式表style.css中定义）中定义的样式属性，可以自动应用到这些项目。

在样式表文件中定义的类选择器如下：

```
style.css
.label {float: left; width: 150px; }
.infobox {width: 200px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 150px; margin-top: 10px;}
```

确认文本输入字段只接受数字连同-或+而不接受其他字符的jQuery代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var data=$('.infobox').val();
    if(validate_phoneno(data))
    {
      $('.error').hide();
    }
    else
    {
      $('.error').show();
      event.preventDefault();
    }
   });
 });

function validate_phoneno(ph)
{
  var pattern=new RegExp(/^[0-9-+]+$/);
  return pattern.test(ph);
}
```

知其所以然

类选择器label具有float属性，设置为left，因此使下一个项目（文本输入字段）显示在它的右侧，而width（宽度）属性设置为150像素，为显示标签消息Enter Phone number（请输入电话号码）提供足够的空间。类选择器infobox的width属性被设置为200像素，最终成为文本输入字段的宽度。类选择器error把颜色属性设置为红色以便突出显示，左侧填充设置为10像素，与文本输入字段保持一定距离。类选择器.submit具有margin-left和

margin-top 属性，分别设置到浏览器窗口左侧的距离为 150 像素，到文本输入框的距离为 10 像素，因为想要提交按钮出现在文本输入字段的正下方。

前面的 jQuery 代码中，把错误消息设置为初始时不可见，把单击事件附加到提交按钮。在单击事件的事件处理函数中，检索用户在文本字段（即 infobox 类的 span 元素）中输入的电话号码并存储到变量 data 中，再把变量 data 传递到 validate_phoneno()方法进行验证。

把变量 data 的内容分配到 validate_phoneno()方法的 ph 参数，在 validate_phoneno()中创建 RegExp 类的实例。把正则表达式/^[0-9-+]+$/传递到 RegExp 类的构造函数，这意味着在文本输入字段输入的数据能以 0 到 9 开头或结尾（^表示开始而$表示结束），可以带有-号或+号。右方括号（]）后面的+号意味着这种模式可以重复 1 次或多次，也就是说，可以在文本输入字段（电话号码）中输入数字、-号或+号 1 次或多次。

validate_phoneno()以正则表达式来测试 ph 参数的内容，如果 ph 变量的内容匹配指定的正则表达式就返回 true，否则返回 false。由 validate_phoneno()返回的布尔值决定错误消息可见或不可见。如果在输入电话号码时使用+或-以外的符号，就会显示错误消息 Phone number can contain only numbers, + and -（电话号码只能包含数字、+或-），如图 4-7 所示。

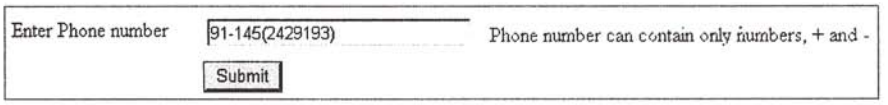


图 4-7　输入数字、+或-以外的数据时，显示错误消息

如果电话号码由数字和+或-组成，将被视为有效的电话号码而接受，而不显示任何错误消息，如图 4-8 所示。

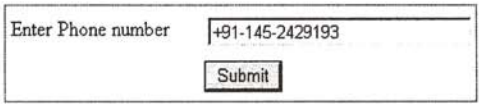


图 4-8　如果电话号码由数字和+或-号组成，则被视为有效

4.4　验证用户 ID

问题描述

验证用户 ID 由字母、数字和下划线组成，而不包含其他字符。

解决方案

新建 HTML 文档，显示包含标签、文本输入字段、错误消息和提交按钮的表单。把标签的文本设置为 Enter User id（输入用户 ID），错误消息设置为 User id can contain only letters, numbers or underscore（用户 ID 只能包含字母、数字或下划线）。分别为这 4 个项目分配类名 label、

infobox、error 和 submit。在外部样式表中分别为这些 CSS 类编写各自的类选择器。该 HTML 文档的内容如下：

```
<body>
<form id="signup" method="post" action="">
<div><span class="label">Enter User id </span><input type="text"  class="infobox"
name="userid" /><span class="error"> User id can contain only letters, numbers or
underscore</span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

定义在样式表 style.css 文件中的类选择器所包含的样式属性，会自动应用到 HTML 元素。本攻略使用与攻略 4.3 相同的样式表文件 style.css。

只接受由数字、字母或下划线组成的用户 ID 的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var data=$('.infobox').val();
    if(validate_userid(data))
    {
      $('.error').hide();
    }
    else
    {
      $('.error').show();
      event.preventDefault();
    }
  });
});

function validate_userid(uid)
{
  var pattern= new RegExp(/^[a-z0-9_]+$/);
  return pattern.test(uid);
}
```

知其所以然

以上 jQuery 代码把错误消息设置为初始时不可见，把单击事件附加到提交按钮。在单击事件的事件处理函数中，用户在文本输入字段中（即 infobox 类的 span 元素）输入的用户 ID 被检索并存储到 data 变量中。把 data 变量传递到 validate_userid()函数进行验证。把 data 变量的内容分配到 validate_userid()方法的 uid 参数。

在 validate_userid()中创建 RegExp 类的实例。把正则表达式/^[a-z0-9_]+$/传递到 RegExp 的构造函数，这意味着在文本输入字段中输入的用户 ID 能以字母 a 到 z、数字 0 到 9 或下划线（_）开始或结束（^表示开始，$表示结束）。右方括号（]）后跟+号，意味着这个模式可以重复一次或多次，也就是说，可以在文本输入字段中输入字符、数字或下划线一次或多次作为用户 ID。

validate_userid()以正则表达式来测试 uid 参数的内容。如果 uid 变量的内容匹配指定的正则表达式就返回 true，否则返回 false。由 validate_userid()函数返回的布尔值决定错误消息可见或不可见。

如果输入用户 ID 时使用下划线以外的符号，就会显示错误消息 User id can contain only letters, numbers or underscore（用户 ID 只能包含字母、数字或下划线），如图 4-9 所示。

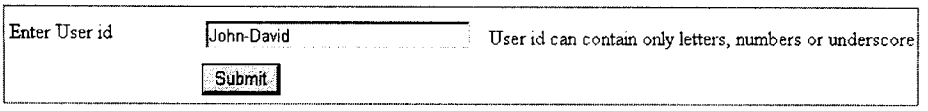


图 4-9　如果在用户 ID 字段中输入下划线以外的字符，就显示错误消息

如果用户 ID 由字母、数字或下划线组成，就被认为是有效的用户 ID 而接受，而不显示任何错误消息，如图 4-10 所示。

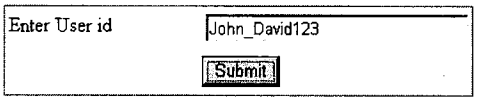


图 4-10　如果用户 ID 由字母、数字和下划线组成，就被认为是有效的

4.5　验证日期

问题描述

验证日期的格式为 mm/dd/yyyy 或 mm-dd-yyyy。如果输入的日期不匹配任何一种格式，就显示错误消息。

解决方案

新建 HTML 文档，显示包含标签、文本输入字段、错误消息和提交按钮的表单。把标签的文本设置为 Enter Date of Birth（输入出生日期），把错误消息设置为 Invalid Date（无效的日期）。分别为这 4 个项目分配类名 label、infobox、error 和 submit。在外部样式表中为这些 CSS 类编写各自的类选择器。HTML 文档的内容如下所示：

```
<body>
<form id="signup" method="post" action="">
 <div><span class="label">Enter Date of Birth </span><input type="text"
class="infobox" name="dob" /><span class="error"> Invalid Date. Correct format is
mm/dd/yyyy or mm-dd-yyyy </span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

定义在样式表 style.css 中的类选择器由样式属性组成，它们会自动应用到 HTML 元素。本攻略使用与攻略 4.3 相同的样式表文件 style.css。

接受日期格式为 mm/dd/yyyy 或 mm-dd-yyyy 的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var data=$('.infobox').val();
    if(validate_date(data))
    {
      $('.error').hide();
    }
    else
    {
      $('.error').show();
      event.preventDefault();
    }
  });
});

function validate_date(date)
{
  var pattern= new RegExp(/\b\d{1,2}[\/-]\d{1,2}[\/-]\d{4}\b/);
  return pattern.test(date);
}
```

知其所以然

把错误消息设置为初始时不可见。然后把单击事件附加到提交按钮。在单击事件的事件处理函数中，检索用户在文本输入字段中（infobox 类的 span 元素）输入的日期并存储到 data 变量中。把 data 变量传递到 validate_date()函数进行验证。把 data 变量的内容分配到 validate_date()函数的 date 参数。在 validate_date()函数中创建 RegExp 类的实例。传递到 RegExp 构造器的正则表达式为/\b\d{1,2}[\/-]\d{1,2}[\/-]\d{4}\b/。

关于该正则表达式的解释如下：

- \b 在正则表达式的开头和结尾，表示单词边界，也就是说，单词必须与模式完全匹配
- \d{1,2} 意味着出现 1 到 2 位数字
- [\/-] 意味着出现符号/或-
- \d{4} 意味着正好出现 4 位数字

因此在文本字段中输入的日期必须以 1 到 2 位数字（某月）开头，后跟符号/或-。然后再次出现 1 到 2 位数字（某日），后跟符号/或-，最后必须正好是 4 位数字（某年）。validate_date()函数以正则表达式来测试 date 参数的内容，如果 date 变量匹配指定的正则表达式就返回 true，否则返回 false。由 validate_date()函数返回的布尔值决定错误消息可见或不可见。

假设输入日期时犯了错误，比如输入了两位数字的年份（而不是 4 位数字），就会显示错误消息 Invalid Date（无效日期），如图 4-11 所示。

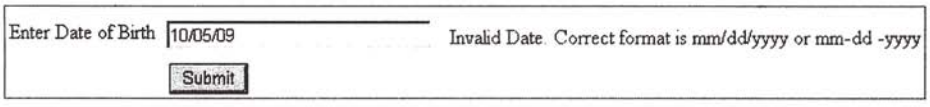


图 4-11 如果输入错误的日期，就会显示错误消息 Invalid Date

如果输入正确的日期，以/或-作为日、月、年的分隔符，就会被接受而不显示任何错误消息，如图 4-12 所示。

图 4-12　如果遵循指定的模式输入日期，就会被接受而不显示任何错误消息

4.6　验证电子邮件地址

问题描述

验证电子邮件地址，也就是说，需要确认电子邮件地址由字母、数字、@和.（半角句号）组成。

解决方案

新建 HTML 文档，显示包含标签、文本输入字段、错误消息和提交按钮的表单。把标签文本设置为 Enter Email Id（输入电子邮件地址），而把错误消息设置为 Invalid Email Address（无效的电子邮件地址）。分别为这 4 个项目分配类名 label、infobox、error 和 submit。在外部样式表中，分别定义了这些 CSS 类名对应的类选择器。HTML 文档的内容如下：

```
<body>
<form id="signup" method="post" action="">
 <div><span class="label">Enter Email Id </span><input type="text"  class="infobox"
name="email" /><span class="error"> Invalid Email Address. Correct email address is
the one that essentially has an @ sign and a . (period) after  @ sign. It should not
have any other special symbol except hyphen or underscore and must be terminated by
any letter only </span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

定义在样式表 style.css 文件中的类选择器由样式属性组成，它们会自动应用到 HTML 元素。本攻略使用与攻略 4.3 相同的样式表文件 style.css。

接受电子邮件地址并加以验证的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var data=$('.infobox').val();
    if(valid_email(data))
    {
      $('.error').hide();
    }
    else
```

```
    {
      $('.error').show();
    event.preventDefault();
    }
  });
});

function valid_email(email)
{
  var pattern= new RegExp(/^[\w-]+(\.[\w-]+)*@([\w-]+\.)+[a-zA-Z]+$/);
  return pattern.test(email);
}
```

知其所以然

把错误消息设置为初始时不可见，然后把单击事件附加到提交按钮。在单击事件的事件处理函数中，用户在文本输入字段中输入的电子邮件地址（包含在 infobox 类的 span 元素内）检索并存储到 data 变量中。把 data 变量传递到 validate_email()函数进行验证。把 data 变量的内容分配到 validate_email()函数的 email 参数。在 validate_email()函数中，创建 RegExp 类的实例。传递到 RegExp 构造器的正则表达式为/^[\w-]+(\.[\w-]+)*@([\w-]+\.)+[a-zA-Z]+$/。

关于该正则表达式的解释如下。

- /^[\w-]+ 意味着电子邮件地址的开头可以是字母、数字、下划线或连字符。^表示开始。\w 表示字母、数字和下划线。右方括号（]）之后的+号表示 1 次或多次。
- (\.[\w-]+)* 表示模式由.（半角句点）后跟 1 个或多个字母、数字、下划线和连字符组成，可以出现 0 或多次。该模式之后的*号表示 0 次或多次。
- @ 表示@必须准确出现在所定义的电子邮件地址的某处：在前面的表达式之后，在后面的表达式之前。
- ([\w-]+\.)+ 表示字母、数字、下划线和连字符可以出现 1 次或多次，后跟 1 个.（半角句点）。合并的序列（字母、数字、下划线、连字符和句点）可以出现 1 次或多次。
- [a-zA-Z]+$/ 定义电子邮件地址的结尾，即电子邮件地址必须以大写或小写字母（可以出现 1 次或多次）结尾。$表示结束。

终上所述，在文本输入字段中输入的电子邮件地址必须以字母、数字、下划线或连字符开头，后跟.（半角句点），后跟字母、数字、下划线或连字符 1 次或多次。在这之后，必须有个@符号，后跟字母、数字、下划线或连字符，后跟.（半角句点）。最后，电子邮件地址必须以任意大写或小写字母结尾。

validate_email()函数以正则表达式测试 email 参数的内容，如果 email 参数的内容匹配指定的正规表达式，则返回 true，否则返回 false。由 validate_email()函数返回的布尔值决定错误消息可见或不可见。假设在电子邮件地址中没有输入.（半角句点），就显示错误消息 Invalid Email Address（无效的电子邮件地址），如图 4-13 所示。

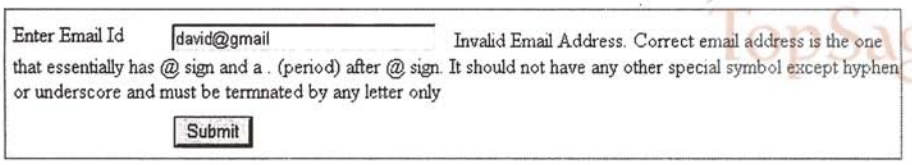


图 4-13　如果输入无效的电子邮件地址，就会显示错误消息

如果输入正确的电子邮件地址，包含@和.（半角句点），将被接受而不显示任何错误消息，如图 4-14 所示。

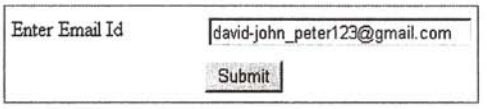


图 4-14　包含.和@符号的电子邮件地址将被接受

4.7　检查复选框是否被选中

问题描述

假设有几个复选框，分别指定美食广场正在出售的一种食品。单击提交按钮时请确认至少有一个复选框被选中。如果没有任何复选框被选中，则显示错误消息。

解决方案

新建 HTML 文档，显示 4 个复选框、错误消息和用于显示结果（选定食品项的总金额）的空段落。HTML 文件的内容如下：

```
<body>
<form>
<input type="checkbox"  id="pizza" name="pizza" value=5  class="infobox">Pizza $5 <br>
<input type="checkbox"   id="hotdog" name="hotdog" value=2   class="infobox">HotDog
$2<br>
<input type="checkbox"   id="coke" name="coke" value=1   class="infobox">Coke $1<br>
<input type="checkbox"  id="fries" name="fries" value=3  class="infobox">French Fries
$3<br>
<p class="error">Select at least one checkbox </p>
<p class="result"></p>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

在外部样式表文件 style.css 中所定义的类选择符把样式属性自动应用到 HTML 元素，其内容如下：

```
style.css
.infobox { margin-top: 15px; }
.error { color: red; }
```

检查复选框是否被选中的jQuery代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var count=0;
    var amt=0;
    $('form').find(':checkbox').each(function(){
      if($(this).is(':checked'))
      {
                count++;
                amt=amt+parseInt($(this).val());
      }
    });
    if(count==0)
    {
              $('p.result').hide();
              $('.error').show();
     }
    else
    {
              $('.error').hide();
              $('p.result').show();
              $('p.result').text('Your bill is $ '+amt);
    }
    event.preventDefault();
  });
});
```

利用length方法检查复选框

以下 jQuery 代码首先确认是否有复选框被选中，如果有就使用循环来检查选中的各个复选框：

```
$(document).ready(function() {
          $('.error').hide();
          $('.submit').click(function(event){
            var amt=0;
            var count=$('input:checked').length;
            if(count==0)
            {
              $('p.result').hide();
              $('.error').show();
            }
            else
            {
              $('form').find(':checkbox').each(function(){
                if($(this).is(':checked'))
                {
                          amt=amt+parseInt($(this).val());
                }
              });
              $('.error').hide();
              $('p.result').show();
```

```
        $('p.result').text('Your bill is $ '+amt);
      }
      event.preventDefault();
    });
});
```

知其所以然

HTML文件中，为4个复选框指定类名infobox，代表在美食广场出售的4种食品：比萨饼、热狗、可乐和薯条以及它们的价格。为显示错误消息的段落元素指定类名error，错误消息的文本是Select at least one checkbox（选择至少一个复选框）。还有一个段落元素显示通过复选框选中的食品项的支出，为它分配类名result，目前是空的。将通过jQuery代码为它指定文本，显示金额为多少。

样式表中定义了两个类选择器：应用于复选框的infobox和应用于显示错误消息的段落元素的error。infobox类选择器包含属性margin-top，设置为15像素，为显示的复选框之间创建足够的垂直距离。类选择器error包含样式属性color，值设置为red，以便以红色文本来突出显示错误消息。

在jQuery代码中，首先把错误消息设置为不可见。然后把单击事件附加到提交按钮。一单击提交按钮，就会执行它的事件处理函数，完成几个任务。在事件处理函数中，首先初始化计数器count，值设置为0（用于计算选中的复选框的数目），然后初始化另一个变量amt，值设置为0，用于总计选定食品项目的金额。利用.each()函数，逐个检查表单中的所有复选框，如果有任一复选框被选中，计数器变量count的值就加1，而该食品项目的支出则被加到变量amt中。也就是说，把变量count的值设置为选中的复选框的数目，把变量amt的值设置为选中的复选框的总金额。检查所有复选框之后，如果发现变量count的价值为0，就意味着没有一个复选框被选中，就使显示错误消息的段落元素可见，显示Select at least one checkbox on the screen（在屏幕上选择至少一个复选框）。如果变量count的值不为0（意味着至少有一个复选框被选中），就使result类的段落元素可见（并使error类的段落元素隐藏）并指定以下文本：Your bill is $ +amt（你的账单是$ +amt），其中amt的值为选中的食品项目的总金额。

如果没有选择任何复选框而点击提交按钮，屏幕上就会显示错误消息 Select at least one checkbox（选择至少一个复选框），如图4-15所示。

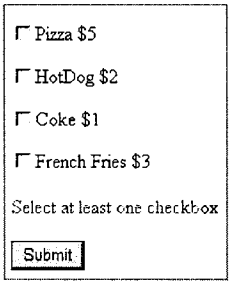


图4-15　如果没有复选框被选中，就显示错误消息

选中一个复选框，错误消息将变为不可见，所选食物项目的金额将被显示，如图 4-16 所示。如果选中多个复选框，所选食物项目的总金额将被显示，如图 4-17 所示。

图 4-16 显示选中的食品项目的金额

图 4-17 显示选中的 3 个食品项目的总金额

4.8 检查单选按钮是否被选中

问题描述

假设有几个单选按钮，分别指定用户可能用来付款的信用卡品牌。在单击提交按钮时，请确认其中一个单选按钮被用户选中。如果没有一个单选按钮被选中，就显示错误消息。

解决方案

以下 HTML 文件显示 3 个单选按钮。为每个单选按钮分配类名 `infobox`，用于验证检查和应用样式。为显示错误消息的段落元素分配类名 `error`，为提交按钮分配类名 `submit`。HTML 代码如下：

```
<body>
<form>
<input type="radio"  name="paymode" class="infobox" value="MasterCard">MasterCard
<br>
<input type="radio"  name="paymode" class="infobox" value="ANZ Grindlay Card">ANZ
Grindlay Card<br>
<input type="radio"  name="paymode" class="infobox" value="Visa Card">Visa Card<br>
<p class="error">Select at least one Option </p>
<p class="result"></p>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

外部样式表包含以下类选择器：

```
style.css
.infobox { margin-top: 15px; }
.error { color: red; }
```

确认至少选中了 1 个单选按钮的 jQuery 代码如下所示：

```
1.  $(document).ready(function() {
2.          $('.error').hide();
3.          $('.submit').click(function(event){
4.                      var amt=0;
5.                      var count=$('input:checked').length;
6.                      if(count==0)
7.                      {
8.                              $('p.result').hide();
9.                              $('.error').show();
10.         }
11.         else
12.         {
13.                     $('.error').hide();
14.                     $('p.result').show();
15.                     $('p.result').text('You have selected
                        '+$('input:checked').attr("value"));
16.         }
17.         event.preventDefault();
18.     });
19. });
```

知其所以然

在样式表文件中，类选择器 infobox 包含 margin-top 属性，设置为 15 像素，以使单选按钮之间保持适当的距离，类选择器 error 则为错误消息指定以红色显示。

下面逐行解释 jQuery 代码。

- 第 2 行，把 error 类的段落元素的错误消息设置为隐藏。
- 第 3 行，把单击事件附加到提交按钮。
- 第 5 行，查明选中的单选按钮的数目（如有）并存储到变量 count 中。
- 第 8 行，如果没有任何单选按钮被选中，就隐藏 result 类的段落元素（用来显示选定的信用卡名称）。
- 第 9 行，如果没有任何单选按钮被选中，就在 error 类的段落元素中显示错误消息。
- 第 15 行，指定文本 You have selected n（你选择了 n），其中 n 是选中的单选按钮的值。

如果不选择任何单选按钮而单击提交按钮，就会看到错误消息 Select at least one Option（至少选择一个选项），如图 4-18 所示。

选中一个单选按钮并单击提交按钮，就显示确认选中的信用卡名称的消息，如图 4-19 所示。

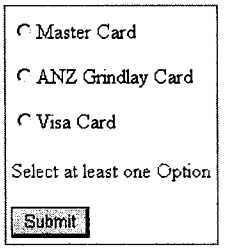


图 4-18　如果没有选择任何单选按钮，就显示错误消息

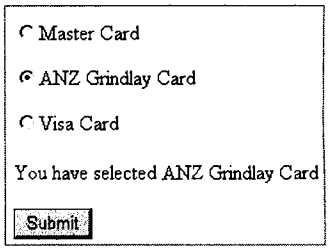


图 4-19　确认选中的单选按钮的消息

4.9 检查 select 元素中的选项是否被选中

问题描述

假设有一个下拉列表（select 元素）显示食品项目。单击提交按钮时，请确认用户从下拉列表中选择了某个选项。如果没有选项被选中，就显示错误消息。

解决方案

以下 HTML 文件显示包含一些食品项目的下拉列表。在 select 元素中显示下拉列表。为 select 元素指定类名 infobox，以便应用定义在类选择器 infobox 中（定义在外部样式表文件 style.css 中）的样式属性。通过 label 类的 span 元素来显示标签消息 Select the Food Item（请选择食品项目）。通过 error 类的段落元素来显示错误消息，而为提交按钮指定类名 submit。HTML 代码如下：

```
<body>
<form>
<span class="label">Select the Food Item </span>
<select id="food" class="infobox">
<option value="0">Select a Food</option>
<option value="Pizza $5">Pizza $5</option>
<option value="HotDog $2">HotDog $2</option>
<option value="Coke $1">Coke $1</option>
<option value="French Fries $3">French Fries $3</option>
</select>
<p class="error">You have not selected any Option</p>
<p class="result"></p>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

外部样式表包含以下类选择器，把样式自动应用到指定类的各个 HTML 元素：

```
style.css
.label {float: left; width: 150px; }
.infobox {width: 150px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 150px; margin-top: 10px;}
```

确认从下拉列表中选择了选项的 jQuery 代码如下：

```
$(document).ready(function() {
        $('.error').hide();
        $('.submit').click(function(event){
                var count=$('select option:selected').val();
                if(count==0)
                {
                        $('p.result').hide();
                        $('.error').show();
                }
                else
```

```
                {
                        $('.error').hide();
                        $('p.result').show();
                        $('p.result').text('You have selected '+$('select
                          option:selected').text());
                }
                event.preventDefault();
        });
});
```

知其所以然

在样式表文件中，类选择器 label 包含 float 属性，设置为 left（左浮动），从而使标签在左侧显示（为下拉列表在右侧显示而创造空间），把 width 属性设置为 150 像素，定义标签占用的宽度。类选择器 infobox 包含 width 属性，设置为 150 像素，指定下拉列表的宽度，类选择器 error 使错误消息以红色显示，并且距离浏览器窗口左侧 10 像素。类选择器 submit 包含 margin-left 属性，设置为 150 像素，使提交按钮显示在距离浏览器窗口左侧 150 像素之处（以便在下拉列表的下面显示），把 margin-top 属性设置为 10 像素，以便和上面可能显示的元素（错误消息或结果消息）保持适当的距离。

在 jQuery 代码中，使显示错误消息的段落元素初始时不可见，然后把点击事件附加到提交按钮。以下语句

```
var count=$(select option:selected).val();
```

检索从 select 元素中选择的选项值，存储到变量 count 中。如果 count 变量的值是 0（意味着用户没有选择任何选项），就使 error 类的段落元素变为可见，从而在屏幕上显示错误消息，如果 count 变量的值不为 0，就用以下语句通过 result 类的段落元素显示结果：

```
You have selected +$(select option:selected).text());
```

以上语句显示从 select 元素中选定的选项的文本。

执行示例代码，如果不从 select 元素中选择任何选项而按下提交按钮，就会显示错误消息 You have not selected any Option（你没有选择任何选项），如图 4-20 所示。

如果从列表中选择任何选项，并按下提交按钮，就会看到显示选定的选项的消息，如图 4-21 所示。

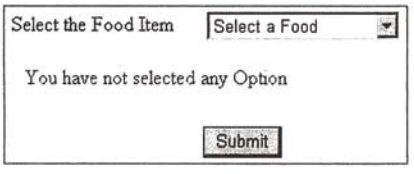


图 4-20　如果不从 select 元素中选择任何选项，就显示错误消息

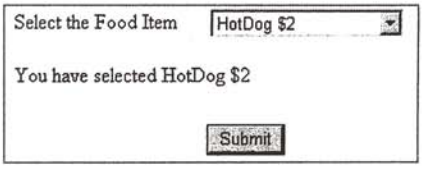


图 4-21　显示所选选项的文本

选择多个选项

修改前面的解决方案，让用户可以从 select 元素中选择多个选项。想要从 select 元素中

选择一个以上的选项，需要利用 select 元素的 MULTIPLE 属性，如下所示：

```
<body>
<form>
<span class="label">Select the Food Item </span>
<select id="food" class="infobox" MULTIPLE>
<option value="0" selected="0">Select a Food</option>
<option value="Pizza $5">Pizza $5</option>
<option value="HotDog $2">HotDog $2</option>
<option value="Coke $1">Coke $1</option>
<option value="French Fries $3">French Fries $3</option>
</select>
<p class="error">You have not selected any Option </p>
<p class="result"></p>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

在 select 元素中附加 MULTIPLE 属性，使用户能够选择多个选项（利用 Ctrl 或 Shift 键）。使用相同的 jQuery 代码和样式表，将会看到消息，显示用户从 select 元素中选择的所有选项，如图 4-22 所示。

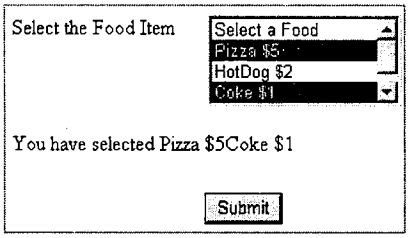


图 4-22 显示所有被选项的文本，选项文本之间没有插入空格

从图 4-22 可见，显示的被选项之间没有插入任何空格。想要以逗号分隔被选项，修改 jQuery 示例代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    var selectedopts="";
    var count=$('select option:selected').val();
    if(count==0)
    {
      $('p.result').hide();
      $('.error').show();
    }
    else
    {
      $('select option:selected').each(function(){
        selectedopts+=$(this).text()+",";
      });
      $('.error').hide();
      $('p.result').show();
```

```
        $('p.result').text('You have selected '+ selectedopts);
      }
      event.preventDefault();
    });
});
```

以上jQuery代码中定义字符串变量selectedopts,调用.each()方法从select元素中提取所有被选项。把所有被选项以逗号分隔，连接到selectedopts变量。当所有被选项的文本添加到selectedopts变量之后,通过result类的段落元素显示selectedopts变量的内容。运行结果如图4-23所示。

如果不从select元素中选择任何选项而按下提交按钮,就会显示错误消息,如图4-24所示。

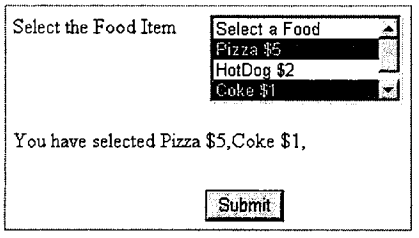


图4-23　显示所有被选项的文本，选项文本之间以逗号分隔

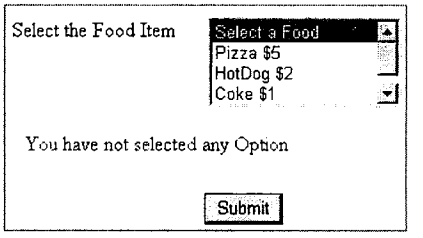


图4-24　如果没有选项被选中，就显示错误消息

4.10　把样式应用到选项和表格按钮

问题描述

有一个下拉列表（select元素）显示不同的食品项，需要把样式应用到select元素的选项。

解决方案

本攻略使用与攻略4.9相同的HTML文件。需要往样式表添加类型选择器option（以便把option的样式属性自动应用到select元素中的所有选项）和CSS类.meal（以便把.meal的样式应用到select元素的奇数选项)。样式表的内容如下：

```
style.css
.label {float: left; width: 150px; }
.infobox {width: 150px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 150px; margin-top: 10px;}

option{
background-color:red;
color:white;
}
```

```
.meal{
background-color:cyan;
color:blue;
}
```

为了使选项显示不同的颜色，把不同的样式应用到 select 元素的偶数和奇数选项。类型选择器 option 把样式属性应用到 select 的所有选项，把背景色设置为红色，前景色为白色。利用 jQuery 代码只是把样式规则 meal 应用到 select 元素的奇数选项，把背景色设置为蓝绿色，前景色设置为蓝色。

为了把样式规则 meal 应用到 select 元素的奇数选项，需要往 jQuery 示例代码添加以下语句：

```
$('option:odd').addClass('meal');
```

完整的 jQuery 代码如下（关于验证逻辑的细节，请参阅攻略 4.9）：

```
$(document).ready(function() {
  $('.error').hide();
  $('option:odd').addClass('meal');
  $('.submit').click(function(event){
    var selectedopts="";
    var count=$('select option:selected').val();
    if(count==0)
    {
      $('p.result').hide();
      $('.error').show();
    }
    else
    {
      $('select option:selected').each(function(){
        selectedopts+=$(this).text()+",";
      });
      $('.error').hide();
      $('p.result').show();
      $('p.result').text('You have selected '+ selectedopts);
    }
    event.preventDefault();
  });
});
```

知其所以然

执行以上 jQuery 示例代码，可见 select 元素的偶数和奇数的选项以不同的前景色和背景色显示，如图 4-25 所示。

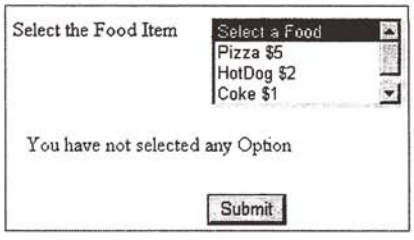


图 4-25 把样式应用到 select 元素的选项

1. 把样式应用到表单按钮

把样式应用到提交按钮使其更具吸引力。为此，需要修改类选择器 .submit 的样式属性（定义在样式表文件 style.css 中），如下所示：

```
style.css
.label {float: left; width: 150px; }
.infobox {width: 150px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 150px; margin-top:
10px;font-size:1.5em;background-color:green;color:blue;}

option{
background-color:red;
color:white;
}

.meal{
background-color:cyan;
color:blue;
}
```

类选择器.submit 包含以下几个属性。

- 把 margin-left 属性设置为 150 像素，使提交按钮在距离浏览器窗口左侧 150 像素之处显示（在 select 元素之下）。
- 把 margin-top 属性设置为 10 像素，在 select 元素之上创建一定的空间。
- 把 font-size 属性设置为 1.5em，使字体大小增大到默认字体大小的 150%。
- 把 background-color 和 color 属性分别设置为 green 和 blue，使文本颜色与绿色背景形成对比。

提交按钮的显示效果如图 4-26 所示。

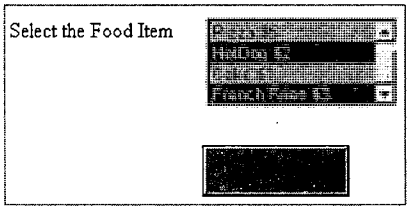


图 4-26　把样式应用到提交按钮

2. 创建图像式提交按钮

下面以图像替换提交按钮，使图像发挥提交按钮的作用。在继续这个解决方案之前，需要有一个图像文件 submit.jpg，如图 4-27 所示。还需要在以下样式表文件 style.css 中修改类选择器.submit。

```
style.css
.label {float: left; width: 150px; }
.infobox {width: 150px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 150px; margin-top: 10px;width:150px;height:40px;}

option{
background-color:red;
color:white;
}

.meal{
background-color:cyan;
color:blue;
}
```

类选择器.submit 分别定义图像（提交按钮）距离浏览器窗口左侧 150 像素，距离顶部元素 10 像素，图像本身的宽度和高度分别是 150 像素和 40 像素。

为了用 submit.jpg 图像替换提交按钮，需要修改 jQuery 代码如下：

```
<body>
<form>
<span class="label">Select the Food Item </span>
<select id="food" class="infobox" MULTIPLE>
<option value="0">Select a Food</option>
<option value="Pizza $5">Pizza $5</option>
<option value="HotDog $2">HotDog $2</option>
<option value="Coke $1">Coke $1</option>
<option value="French Fries $3">French Fries $3</option>
</select>
<p class="error">You have not selected any Option </p>
<p class="result"></p>
<input class="submit" type="image" value="Submit" src="submit.jpg">
</form>
</body>
```

可见通过把 type 属性指定为 image，把 src 属性指定为图像文件名，从而使提交按钮以图像形式显示，如图 4-27 所示。

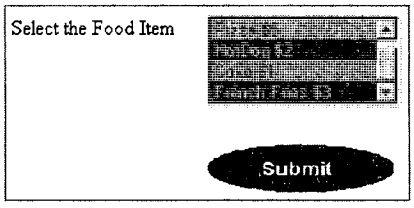


图 4-27 把图像应用到提交按钮

4.11 一步选择或取消所有的复选框

问题描述

有几个复选框，分别指定美食广场正在出售的一个食品项目。需要 Check All（选择全部）复选框，如果勾选了 Check All 复选框，就会自动勾选所有其他的复选框，如果取消了 Check All 复选框，就会自动取消所有其他的复选框。

解决方案

新建 HTML 文件，显示 5 个复选框（其中 4 个用于食品项目，1 个用于 Check All 复选框）。HTML 文件的内容如下：

```
<body>
<form>
<div class="infobox"><input type="checkbox" id="checkall">Check/Uncheck all
```

```
Checkboxes</div>
<div class="infobox"><input type="checkbox"  id="pizza" name="pizza" value=5>Pizza
$5</div>
<div class="infobox"> <input type="checkbox" id="hotdog" name="hotdog" value=2>HotDog
$2</div>
<div class="infobox"><input type="checkbox"  id="coke" name="coke" value=1>Coke
$1</div>
<div class="infobox"><input type="checkbox"  id="fries" name="fries" value=3>French
Fries $3</div>
</form>
</body>
```

在样式表中定义类选择器 infobox 如下：

```
.infobox{ padding: 5px; }
```

把填充属性设置为 5 像素，创建复选框之间的间距。

通过选择或取消 Check All 复选框从而选择或取消所有其他的复选框，以及显示选定食品项目的金额的 jQuery 代码如下：

```
$(document).ready(function() {
  $('#checkall').click(function(){
    $("input[type='checkbox']").attr('checked', $('#checkall').is(':checked'));
  });
  $('form').find(':checkbox').click(function(){
    var amt=0;
    $('div').filter(':gt(0)').find(':checkbox').each(function(){
      if($('div:gt(0)'))
      {
        if($(this).is(':checked'))
        {
                  amt=amt+parseInt($(this).val());
        }
      }
    });
    $('p').remove();
    $('<p>').insertAfter('div:eq(4)');
    $('p').text('Your bill is $ '+amt);
  });
});
```

知其所以然

在 HTML 文件中，把类名 infobox 分配到 5 个复选框，其中 4 复选框代表美食广场出售的 4 个项目：比萨饼、热狗、可乐、薯条，以及它们的价格。

在 jQuery 代码中，把单击事件附加到 Check All 复选框（id 为 checkall）：

```
$("input[type='checkbox']").attr('checked', $('#checkall').is(':checked'));
```

为了理解以上语句，先要理解以上语句中用到的.attr()和.is()方法：

```
.attr()
```

.attr()方法用于设置选定的元素属性。

语法：

```
.attr(attribute, value)
.is()
```

.is()方法以选择器检查选定的元素，如果选定的任何元素与选择器匹配就返回 true，否则返回 false。

语法：

```
.is(selector)
```

前面语句中的$(#checkall).is(:checked)部分，检查 id 为 checkall 的复选框是否被选中。如果被选中，.is()方法将返回 true，否则返回 false。如果.is()方法返回 true，就把所有的 type 为 checkbox 的元素（即所有的复选框）都设置为被选中模式，如果.is()方法返回 false，就把所有的 type 为 checkbox 的元素都设置为未选中模式。因为用户可以选择任何一个复选框，所以检查索引值大于 0 的每个复选框的状态（因为索引值为 0 的复选框是 Check All 复选框）。把所有被选中的复选框的值添加到 amt 变量。为了显示金额，创建段落元素，并添加文本 Your bill is amt（其中 amt表示 amt 变量所包含的数值），并在索引值为 4 的 div 元素（即最后的复选框）之后插入这个段落元素。

如果选中 Check All 复选框，所有下面的复选框将被选中，金额的总和以账单形式显示，如图 4-28 所示。

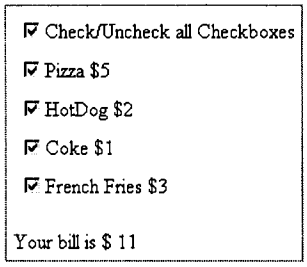


图 4-28 选中 Check/Uncheck all Checkboxes 复选框，所有其他的复选框也被自动选中

如果取消 Check All 复选框，所有下面的复选框将被取消，因此显示账单的金额为 0 美元，如图 4-29 所示。

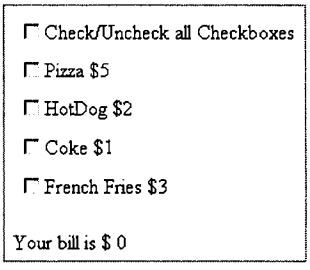


图 4-29 取消 Check/Uncheck all Checkboxes 复选框，所有其他的复选框也被自动取消

用户还可以选择任何单独的复选框。选中的复选框的总金额会被显示，如图4-30所示。

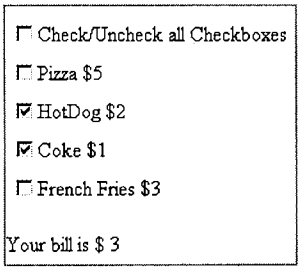


图4-30　单独选中的食品项目的金额

4.12　验证两个字段

问题描述

有用户名和密码两个字段，需要确认每个字段都不留空白。如果其中任何一个被留空，就在屏幕上显示错误消息。

解决方案

新建HTML文件，显示两个标签和两个文本字段，如下所示：

```
<body>
<div><span class="label">User Id *</span><input type="text"  class="infobox"
name="userid" /><span class="error"> This field cannot be blank</span></div>
<div><span class="label">Password *</span><input type="password" class="infobox"
name="password" /><span class="error"> This field cannot be blank</span></div>
</body>
```

包含各个类选择器的样式表的内容如下：

```
style.css
.label {float: left; width: 120px; }
.infobox {width: 200px; }
.error { color: red; padding-left: 10px; }
div{padding: 5px; }
```

测试字段不被留空的jQuery代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.infobox').each(function(){
    $(this).blur(function(){
      var data=$(this).val();
      var len=data.length;
      if(len<1)
      {
        $(this).parent().find('.error').show();
      }
```

```
        else
        {
          $(this).parent().find('.error').hide();
        }
      });
    });
});
```

知其所以然

HTML 文件中把标签消息 User Id*和 Password *包含在 label 类的 span 元素中。把类名 infobox 分配到文本输入字段，把错误消息（该字段不能为空）存储作为 error 类的 span 元素。之所以把 CSS 类分配到所有 3 个项目——标签、文本输入字段和错误消息，是为了自动应用类选择器 label、infobox 和 error 中（定义在样式表文本 style.css 中）所定义的属性。

在样式表文件中，类选择器 label 包含 float 属性，设置为 left，使标签在左边显示（为在右边显示文本输入字段而创造空间），width 属性设置为 120 像素，定义标签可以占用的宽度。类选择器 infobox 包含 width 属性，设置为 200 像素，为下拉列表指定宽度，类选择器 error 把红色赋予错误消息，并且使错误消息显示距离元素左侧 10 像素之处。类型选择器 div 包含 padding 属性，设置为 5 像素，在两个 div 元素之间创造一定的空间，其中每个 div 元素都包含标签、文本输入字段和错误消息的组合。

在 jQuery 代码中，初始时隐藏所有的错误消息（error 类的 span 元素），接着调用.each()方法，检测在任何一个文本输入字段上是否发生了 blur 事件（即用户对该字段失去焦点），如果是，就从文本输入字段中检索值并存储到变量 data 中。如果 data 变量为空，也就是说，如果其长度小于 1，就显示有关该字段的错误消息。

如果使焦点离开第一个文本字段（用户 ID）移动到下一个字段，就会显示有关第一个字段的错误消息，如图 4-31 所示。

图 4-31 如果留空第一个字段，就显示错误消息

如果留空第二个文本字段并使其失去焦点，就会显示有关第二个字段的错误消息，如图 4-32 所示。

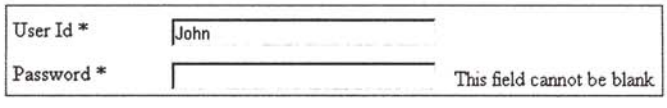


图 4-32 如果留空第二个字段，就显示错误消息

如果在两个文本输入字段中输入数据，就不显示任何错误消息，如图 4-33 所示。

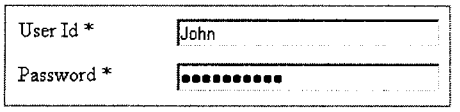


图 4-33　如果在两个文本输入字段中输入数据，就不显示任何错误消息

添加提交按钮

在前面的解决方案中没有添加提交按钮。下面添加提交按钮，当用户单击提交按钮时，逐个验证文本输入字段。为前面的 HTML 文件添加提交按钮如下：

```
<body>
<form>
<div><span class="label">User Id *</span><input type="text"  class="infobox"
name="userid" /><span class="error"> This field cannot be blank</span></div>
<div><span class="label">Password *</span><input type="password" class="infobox"
name="password" /><span class="error"> This field cannot be blank</span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

需要往样式表文件 style.css 中添加类选择器 .submit，把样式属性自动应用到提交按钮，使其在文本输入字段之下显示，与正上方的文本输入字段保持一点距离。现在 style.css 文件的内容如下：

```
style.css
.label {float: left; width: 120px; }
.infobox {width: 200px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 125px; margin-top: 10px;}
div{padding: 5px; }
```

在提交按钮上发生单击事件时执行验证。修改后的 jQuery 示例代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    $('.infobox').each(function(){
      var data=$(this).val();
      var len=data.length;
      if(len<1)
      {
      $(this).parent().find('.error').show();
      }
      else
      {
        $(this).parent().find('.error').hide();
      }
    });
    event.preventDefault();
  });
});
```

因为我们的目的是单击提交按钮时执行验证，而不是把（在文本输入字段中输入的）数据发送到服务器，所以调用 event 对象的 .preventDefault() 方法，中止发送数据到服务器。event

对象由 JavaScript 自动发送到单击事件的事件处理函数。

如果留空两个字段并单击提交按钮，就会显示两个错误消息，如图 4-34 所示。

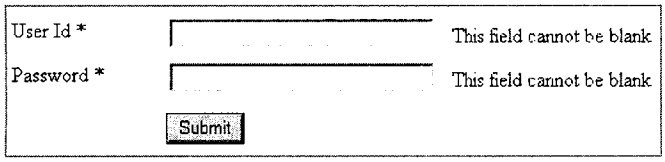


图 4-34 如果留空两个字段，就会显示两个错误消息

如果留空两个字段中的一个而单击提交按钮，就会显示留空字段的错误消息，如图 4-35 所示。

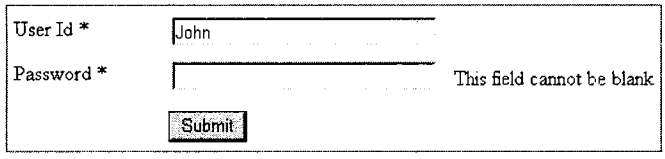


图 4-35 如果留空其中一个字段，就会显示相应的错误消息

4.13 验证密码和确认密码字段是否匹配

问题描述

需要保证在密码字段和确认密码字段中输入的密码完全匹配。

解决方案

新建 HTML 文件，显示 3 个标签和用户 ID、密码和确认密码的 3 个文本字段，HTML 文件的内容如下：

```
<body>
<form>
<div ><span class="label">User Id </span><input type="text"  class="userid"
name="userid" /></div>
<div ><span class="label">Password </span><input type="password" class="password"
name="password" /><span class="error"> Password cannot be blank</span></div>
<div ><span class="label">Confirm Password </span><input type="password"
class="confpass" name="confpass" /><span class="error"> Password and Confirm Password
don't match</span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

包含各个类选择器的样式表的内容如下：

```
style.css
```

```
.label {float: left; width: 120px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 125px; margin-top: 10px;}
div{padding: 5px; }
```

检测在密码字段和确认密码字段中输入的数据是否完全一致的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    data=$('.password').val();
    var len=data.length;
    if(len<1)
    {
      $('.password').next().show();
    }
    else
    {
      $('.password').next().hide();
    }
    if($('.password').val() !=$('.confpass').val())
    {
      $('.confpass').next().show();
    }
    else
    {
      $('.confpass').next().hide();
    }
    event.preventDefault();
  });
});
```

知其所以然

在 HTML 文件中把标签消息 User Id、Password 和 Confirm Password 包含在 label 类的 span 元素中。为 3 个文本输入字段赋予各自的类名 userid，password 和 confpass，以便通过 jQuery 检索在文本输入字段中输入的数据。最后，所有的文本输入字段之后是错误消息，内嵌在 error 类的 span 元素中。

之所以为所有的元素即标签、错误消息和提交按钮分配类名，是为了自动应用类选择器 label、error 和.submit 中（定义在样式表文本 style.css 中）所定义的属性。此外，标签、文本输入字段和错误消息的组合被嵌套在 div 元素中，以便将样式属性应用到 div 元素，创建每个组合（即标签、文本输入字段和错误消息的组合）之间的间距。

在样式表文件中，类选择器 label 包含 float 属性，设置为 left，使标签显示在浏览器窗口的左侧（为显示在标签的右边的文本输入字段创造空间），width 属性设置为 120 像素，定义标签可以占用的宽度。类选择器 error 把红色分配到错误消息，并使错误消息显示在距离其左边元素 10 像素之处。类选择器.submit 包含 margin-left 属性，设置为 125 像素，margin-top 属性，设置为 10 像素，使提交按钮显示在距离浏览器窗口左侧 125 像素之处，与其正上方的元素保持 10 像素的间距（显示在文本输入字段之下）。类型选择器 div 包含 padding

属性，设置为 5 像素，以便在 div 元素之间创造一定的空间（每个 div 元素都包含标签、文本输入字段和错误消息的组合）。

为了理解前面的 jQuery 示例代码，必须先了解其中用到的.next()方法的用法，下面学习.next()方法的用法。

.next()

此方法返回的每个元素的下一个兄弟元素，而不像 nextAll()方法那样，返回所有的下一个兄弟元素。

语法：

```
.next(selector)
```

这里的 selector 选择器是可选的参数，用于指定选择器表达式（匹配指定的元素）。

在 jQuery 代码中，初始时隐藏错误消息（内嵌在 error 类的 span 元素中）。此后，把点击事件附加到提交按钮。在单击事件的事件处理函数中检索密码字段（password 类的文本输入字段）中的数据并存储到变量 data 中。如果发现变量 data 的内容长度小于 1，也就是说，如果用户没有往密码字段中输入任何数据，就使密码字段的下一个元素（error 类的 span 元素）在屏幕上显示，也就是说，在屏幕上显示错误消息 Password cannot be blank（密码不能为空）。

如果用户没有把密码留空，就检查在密码字段和确认密码字段中输入的数据（.password 类和.confpass 类的元素中的数据）是否完全一致。如果不完全一致，就显示错误消息（即确认密码字段的下一个元素）Password and Confirm Password don't match（密码和确认密码不匹配）。还调用事件对象的 preventDefault()方法，以防止用户输入的数据发送到服务器，因为我们的目的主要在于确认在密码字段和确认密码字段中输入的数据相互匹配。

执行 jQuery 示例代码，如果将密码字段留空并单击提交按钮，就会显示错误消息 Password cannot be blank（密码不能为空），如图 4-36 所示。

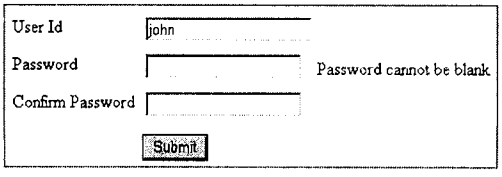


图 4-36 如果没有输入密码，就显示以上错误消息

如果密码字段和确认密码字段的内容不匹配，就显示错误消息 Password and Confirm Password don't match（密码和确认密码不匹配），如图 4-37 所示。

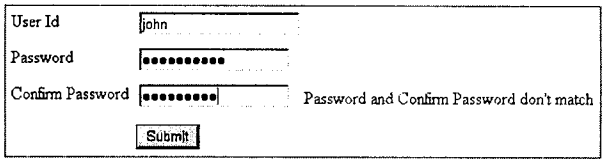


图 4-37 如果密码和确认密码不匹配，就显示以上错误消息

4.14　禁用某些字段

问题描述

要求用户填写用户 ID、密码和确认密码。如果用户输入无效数据或留空任何字段，不只是显示错误消息，而且禁用其余字段直到错误得到纠正。

解决方案

使用与攻略 4.13 相同的 HTML 文件和样式表文件 style.css。唯一不同的是从 HTML 文件中删除提交按钮，因为本示例利用 `blur()`事件验证字段，而不是 `click()`事件。提供 HTML 代码以供参考：

```
<body>
<form>
<div ><span class="label">User Id </span><input type="text"  class="userid"
name="userid" /><span class="error"> User id cannot be blank</span></div>
<div ><span class="label">Password </span><input type="password" class="password"
name="password" /><span class="error"> Password cannot be blank</span></div>
<div ><span class="label">Confirm Password </span><input type="password"
class="confpass" name="confpass" /><span class="error"> Password and Confirm Password
don't match</span></div>
</form>
</body>
```

样式表文件 style.css 与攻略 4.13 所使用的完全一样。

以下 jQuery 代码完成下列任务：

- 验证用户输入的数据；
- 显示错误消息；
- 如果留空字段或输入无效的数据，就禁用其余的字段。

```
$(document).ready(function() {
  $('.error').hide();
  $('.userid').blur(function(){
    data=$('.userid').val();
    var len=data.length;
    if(len<1)
    {
      $('.userid').next().show();
      $('.password').attr('disabled',true);
      $('.confpass').attr('disabled',true);
    }
    else
    {
      $('.userid').next().hide();
      $('.password').removeAttr('disabled');
      $('.confpass').removeAttr{'disabled');
    }
  });
```

```
$('.password').blur(function(){
  data=$('.password').val();
  var len=data.length;
  if(len<1)
  {
    $('.password').next().show();
    $('.confpass').attr('disabled',true);
  }
  else
  {
    $('.password').next().hide();
    $('.confpass').removeAttr('disabled');
  }
});

$('.confpass').blur(function(){
  if($('.password').val() !=$('.confpass').val())
  {
    $('.confpass').next().show();
  }
  else
  {
    $('.confpass').next().hide();
  }
});
});
```

知其所以然

初始时隐藏错误消息（内嵌在 error 类的 span 元素中）。此后，检查 userid 类的文本输入字段上是否发生了 blur()事件，也就是说，用户 ID 字段是否失去了焦点。如果是，检索在字段中用户输入的数据并将其存储到变量 data 中。如果发现变量 data 的内容长度小于 1，即用户 ID 字段留空，就使用户 ID 的下一个元素（error 类的 span 元素）显示在屏幕上，即在屏幕上显示错误消息 User id cannot be blank（用户 ID 不能为空）。此外，还使用下面的两个语句：

```
$('.password').attr('disabled',true);
$('.confpass').attr('disabled',true);
```

禁用密码字段和确认密码字段（password 类和 confpass 类的字段）。也就是说，如果用户不输入用户 ID，就禁用密码字段和确认密码字段；如有用户在用户 ID 字段输入了数据，禁用的密码字段和确认密码字段将被启用（使用以下两个语句）：

```
$('.password').removeAttr('disabled');
$('.confpass').removeAttr('disabled');
```

同样，检测密码字段是否留空。如果是的话，就再次显示错误消息 Password cannot be blank（密码不能为空）和禁用确认密码字段，直到用户在密码字段中输入数据。

最后，检查在密码字段和确认密码字段中输入的数据（.password 类和.confpass 类的元素中的数据）是否完全相同。如果不匹配，就显示错误消息（确认密码字段的下一个元素）Password

and Confirm Password don't match（密码和确认密码不匹配）。

执行 jQuery 示例代码，如果留空用户 ID 字段并使其失去焦点，就会显示错误消息 User id cannot be blank（用户 ID 不能为空），如图 4-38 所示。此外，其余的字段将被禁用。如果用户不在用户 ID 字段中输入数据，就无法在其余的字段中输入任何数据：

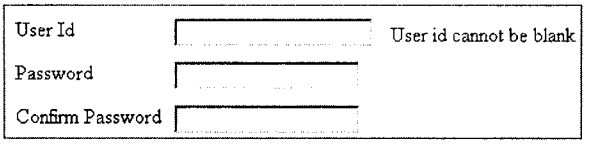


图 4-38　如果不输入用户 ID，就显示错误消息，并且禁用其余的字段

如果留空密码字段，并使其失去焦点，就会在屏幕上显示错误消息 Password cannot be blank（密码不能为空），并且禁用确认密码字段，直到用户在密码字段中输入数据，如图 4-39 所示。

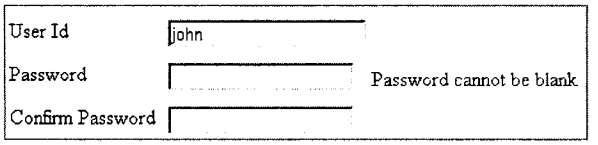


图 4-39　如果不输入密码，就显示错误消息并且禁用确认密码字段

4.15　验证整个表单

问题描述

假设用户在表单中输入用户 ID、密码和电子邮件地址，选择想要购买的食品项目，选择付款方式（指定信用卡），选择居住国。验证表单中的每个字段如下。

- 用户 ID 应该只包含字母、数字和下划线。
- 密码不能为空。
- 电子邮件地址必须包含一个句点和@符号。
- 至少选中一个复选框（食品项目）。
- 必须选择一种付款方式。
- 必须选择居住国。

解决方案

新建 HTML 文件，显示标签和字段，如图 4-40 所示，有 6 个标签、3 个文本输入字段、4 个复选框、3 个单选按钮和 1 个 `select` 元素。HTML 代码如下：

```
<body>
<form>
<div>    <span class="label">User Id </span><input type="text"  class="userid"
```

```
name="userid" /><span class="error">User id can contain only numeral, character or
_(underscore)</span></div>
<div><span class="label">Password </span><input type="password" class="password"
name="password" /><span class="error"> Password cannot be blank</span></div>
<div><span class="label">Email Address </span><input type="text" class="emailadd"
name="emailid" /><span class="error"> Invalid email address</span></div>
<div><span class="label">Select Food items</span><br><input type="checkbox"
id="pizza" name="pizza" value=5  class="chkb">Pizza $5 <br>
<input type="checkbox"  id="hotdog" name="hotdog" value=2  class="chkb">HotDog $2<br>
<input type="checkbox"  id="coke" name="coke" value=1  class="chkb">Coke $1<br>
<input type="checkbox"  id="fries" name="fries" value=3  class="chkb">French Fries
$3<br>
<span class="fooderror">You have not selected any food item</span></div>
<div><span class="label">Mode of  Payment</span><br><input type="radio"
name="paymode" class="radiobtn" value="MasterCard">MasterCard <br>
<input type="radio"  name="paymode" class="radiobtn" value="ANZ Grindlay Card">ANZ
Grindlay Card<br>
<input type="radio"  name="paymode" class="radiobtn" value="Visa Card">Visa Card<br>
<span class="payerror">You have not selected any payment method</span></div>
<div><span class="label">Country</span><select id="country" class="infobox">
<option value="0">Select a Country</option>
<option value="USA">USA</option>
<option value="United Kingdom">United Kingdom</option>
<option value="India">India</option>
<option value="China">China</option>
</select>
<span class="error"> Please select the country</span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
</html>
```

把样式属性应用到 HTML 元素的样式表文件的内容如下：

```
style.css
.label {float: left; width: 120px; }
.infobox {width: 120px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 125px; margin-top: 10px;}
div{padding: 5px; }
.chkb { margin-left: 125px; margin-top: 10px;}
.radiobtn { margin-left: 125px; margin-top: 10px;}
```

验证所有字段的 jQuery 示例代码如下：

```
$(document).ready(function() {
  $('.error').hide();
  $('.fooderror').addClass('error');
  $('.fooderror').hide();
  $('.payerror').addClass('error');
  $('.payerror').hide();

  $('.submit').click(function(event){
    var data=$('.userid').val();
    if(validate_userid(data))
```

```
{
  $('.userid').next().hide();
}
else
{
  $('.userid').next().show();
}

data=$('.password').val();
var len=data.length;
if(len<1)
{
  $('.password').next().show();
}
else
{
  $('.password').next().hide();
}

data=$('.emailadd').val();
if(valid_email(data))
{
  $('.emailadd').next().hide();
}
else
{
  $('.emailadd').next().show();
}

var count=0;
$('div').find(':checkbox').each(function(){
  if($(this).is(':checked'))
  {
    count++;
  }
});
if(count==0)
{
  $('.fooderror').css({'margin-left':250}).show();
}
else
{
  $('.fooderror').hide();
}

count=0;
$('div').find(':radio').each(function(){
  if($(this).is(':checked'))
  {
    count++;
  }
});
if(count==0)
{
```

```
      $('.payerror').css({'margin-left':250}).show();
    }
    else
    {
      $('.payerror').hide();
    }

    count=$('select option:selected').val();
    if(count==0)
    {
      $('.infobox').next().show();
    }
    else
    {
      $('.infobox').next().hide();
    }

    event.preventDefault();
  });
});

function valid_email(email)
{
  var pattern= new RegExp(/^[\w-]+(\.[\w-]+)*@([\w-]+\.)+[a-zA-Z]+$/);
  return pattern.test(email);
}

function validate_userid(uid)
{
  var pattern= new RegExp(/^[a-z0-9_]+$/);
  return pattern.test(uid);
}
```

知其所以然

逐行解释以上 Query 示例代码的含义如下。

- 第 2 行，隐藏所有有关用户 ID、密码、电子邮件地址和国家的错误消息。把所有这些错误消息内嵌在 error 类的 span 元素中。
- 第 3-4 行，把类选择器 error 中所包含的样式属性应用到有关复选框（显示不同的食品项目）的错误消息。类选择器 error 中所包含的样式属性把红色赋予错误消息。目前错误消息是隐藏的。
- 第 5-6 行，把类选择器 error 中所包含的样式属性应用到有关单选按钮（显示不同的付款方式）的错误消息。错误消息显示为红色，初始时不可见。
- 第 7 行，把单击事件附加到提交按钮。
- 第 8 行，检索用户 ID 字段（userid 类的文本输入字段）中的数据并存储到变量 data 中。
- 第 9-16 行，把变量 data 中的用户 ID 传递到 validate_userid() 函数进行验证，与正则表达式比较，检测用户 ID 是否只是由字母、数字或下划线组成而不包含其他字符。如

果用户 ID 与指定的正则表达式相匹配，则不显示错误消息，否则就在屏幕上显示错误消息（用户 ID 字段的下一个元素）User id can contain only numeral, character, or _（用户名只能包含数字、字母或下划线）。

- 第 17-26 行，检索在密码字段中输入的数据（password 类的文本输入字段），并将其存储到变量 data 中。查名变量 data 的内容的长度，如果长度小于 1，就意味着密码字段留空，则在屏幕上显示错误消息（密码字段的下一个元素）Password cannot be blank（密码不能为空），否则错误消息保持隐藏状态。
- 第 27-35 行，检索数在电子邮件地址字段（emailadd 类的文本输入字段）中输入的数据，并将其存储到变量 data 中。然后把变量 data 中的电子邮件地址传递到 validate_email()函数进行验证，与正则表达式比较，检测电子邮件地址是否以字母数字开头，以及包含一个半角句点和@符号。如果电子邮件地址与指定的正则表达式相匹配，就不显示错误消息，否则在屏幕上显示错误消息（电子邮件地址字段的下一个元素）Password cannot be blank（无效的电子邮件地址）。
- 第 36-42 行，把变量 count 初始化为 0。利用.each()函数查找 div 元素中的所有选中的复选框。每找到一个选中的复选框，变量 count 的值就加 1。换句话说，在计算选中的复选框的数目。
- 第 43-50 行，如果变量 count 的值为 0，意味着没有复选框被选中，就显示错误消息（fooderror 类的 span 元素），调用.css()方法把值为 250 像素的 margin-left 属性应用到错误消息，使其显示在距离浏览器窗口左侧 250 像素之处（如果有其他错误消息，同时也显示在这些错误消息之下）。如果变量 count 的值不为 0，则错误消息保持隐藏状态。
- 第 51-57 行，把变量 count 初始化为 0。利用.each()方法查找 div 元素中所有选中的单选按钮。每找到一个选中的单选按钮，就使变量 count 的值加 1。
- 第 58-65 行，如果变量 count 的值为 0，意味着没有单选按钮被选中，就显示错误消息（payerror 类的 span 元素），调用.css()方法把值为 250 像素的 margin-left 属性应用到错误消息，使其显示在距离浏览器窗口左侧 250 像素之处（如果有其他错误消息，同时也显示在其他错误消息之下）。如果变量 count 的值不为 0，则错误消息保持隐藏状态。
- 第 66-74 行，查明 select 元素中选中的选项数目。把选中的选项数目存储到变量 count 中。如果变量 count 的值为 0，也就是说，如果没有从 select 元素中选择任何选项，就在屏幕上显示错误消息（infobox 类的 select 元素的下一个元素）Please select the country（请选择居住国），否则错误消息保持隐藏状态。
- 第 75 行，调用事件对象的 preventDefault()方法，以防止把用户输入或选择的数据发送到服务器，因为本示例只关注数据的验证。
- 第 78-82 行，验证电子邮件地址。
- 第 83-87 行，验证用户 ID。

执行 jQuery 示例代码，如果将所有的字段留白并单击提交按钮，就会看到错误消息，如图

4-40 所示。

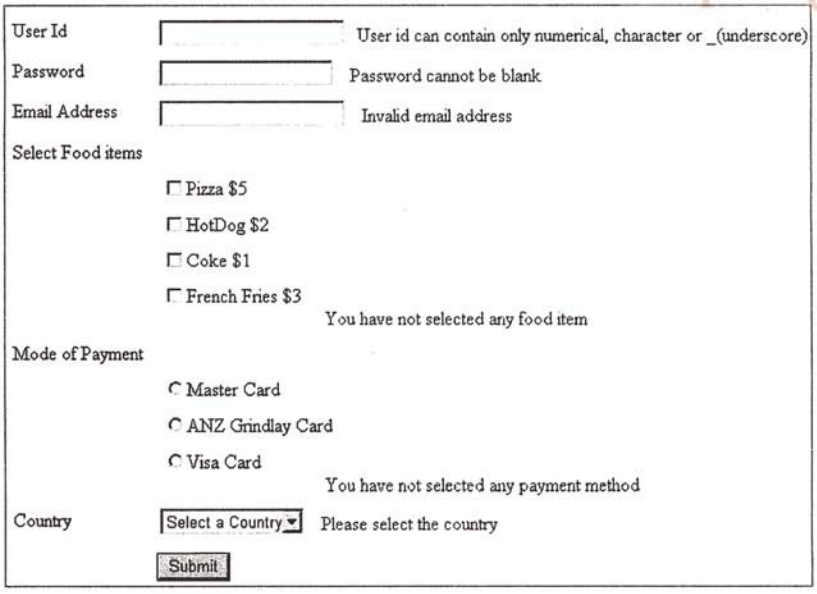


图 4-40 如果没有在任何字段中输入数据，就显示以上错误消息

如果输入无效的用户 ID 或电子邮件地址，而不选择任何复选框或 `select` 元素中的任何选项，就会显示错误消息，如图 4-41 所示。

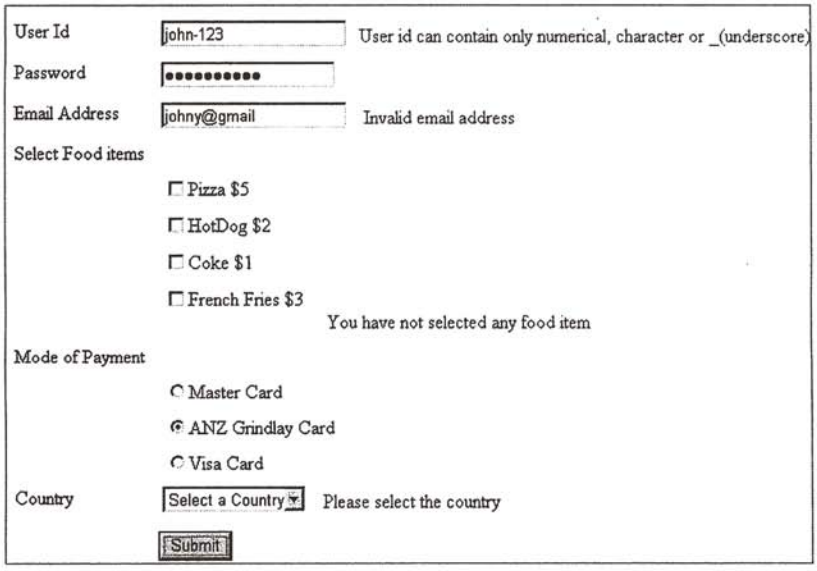


图 4-41 如果没有数据或输入无效数据，则显示以上错误消息

在所有的字段中输入合法数据，并选择所有必需的选项，输入数据将被接受而不显示任何错误消息，如图 4-42 所示。

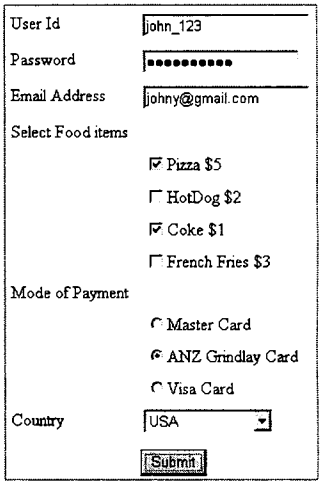


图 4-42　接受有效数据

突出显示输入字段以及通用表单元素分组

下面对表单的相关元素进行分组，同时突出显示输入字段(输入字段得到焦点时)。对表单元素进行分组和应用标题，需要使用两个 HTML 元素：Fieldset 和 Legend。

- <fieldset>标签用于对几个表单元素分组。它在分组元素周围画上线框。
- <legend>标签用于为表单元素（表单元素由 fieldset 元素进行分组）定义标题。

下面应用 fieldset 元素，为前面表单的元素创建 3 个分组，并且应用 legend 元素为表单添加标题 Enter Your Information(请输入你的消息)。修改后的 HTML 表单的内容如下：

```
<body>
<form>
<fieldset>
<legend>Enter Your Information</legend>
<div id="u">   <span class="label">User Id </span><input type="text"  class="userid"
name="userid" /><span class="error">User id can contain only numeral, character, or
_(underscore)</span></div>
<div  id="p"><span class="label">Password </span><input type="password"
class="password" name="password" /><span class="error"> Password cannot be
blank</span></div>
<div><span class="label">Email Address </span><input type="text" class="emailadd"
name="emailid" /><span class="error"> Invalid email address</span></div>
</fieldset>
<fieldset>
<div><span class="label">Select Food items</span><br><input type="checkbox"
id="pizza" name="pizza" value=5  class="chkb">Pizza $5 <br>
<input type="checkbox"  id="hotdog" name="hotdog" value=2  class="chkb">HotDog $2<br>
<input type="checkbox"  id="coke" name="coke" value=1  class="chkb">Coke $1<br>
<input type="checkbox"  id="fries" name="fries" value=3  class="chkb">French Fries
$3<br>
<span class="fooderror">You have not selected any food item</span></div>
<div><span class="label">Mode of  Payment</span><br><input type="radio"
name="paymode" class="radiobtn" value="Master Card">Master Card <br>
<input type="radio"  name="paymode" class="radiobtn" value="ANZ Grindlay Card">ANZ
Grindlay Card<br>
<input type="radio"  name="paymode" class="radiobtn" value="Visa Card">Visa Card<br>
<span class="payerror">You have not selected any payment method</span></div>
</fieldset>
<fieldset>
<div><span class="label">Country</span><select id="country" class="infobox">
<option value="0">Select a Country</option>
<option value="USA">USA</option>
<option value="United Kingdom">United Kingdom</option>
<option value="India">India</option>
```

```
<option value="China">China</option>
</select>
</fieldset>
<span class="error"> Please select the country</span></div>
<input class="submit" type="submit" value="Submit">
</form>
</body>
```

<fieldset>标志着分组的开始而</fieldset>标志着分组的结束。

为了给分组元素定义边框，需要为类型选择器 fieldset 定义样式属性，为了把边框、前景色和背景色应用到标题并使其显示为粗体，需要为类型选择器 legend 定义样式属性。为了突出显示聚焦的文本输入字段，需要在样式表中定义样式规则.inputs。样式表的内容如下：

```
style.css
.submit { margin-left: 125px; margin-top: 10px;}
.label {float: left; width: 120px; }
.infobox {width: 120px; }
.error { color: red; padding-left: 10px; }
div{padding: 5px; }
.chkb { margin-left: 125px; margin-top: 10px;}
.radiobtn { margin-left: 125px; margin-top: 10px;}
.inputs{background-color:cyan}

fieldset{
border:1px solid #888;
}

legend{
border:1px solid #888;
background-color:cyan;
color:blue;
font-weight:bold;
padding:.5em
}
```

为了把样式规则.inputs 中定义的样式属性应用到用户 ID、密码和电子邮件地址这 3 个文本输入字段，需要添加一些语句到前面的 jQuery 示例代码，新添加的语句在以下 jQuery 代码中以粗体显示，其余的代码完全相同。

```
$(document).ready(function() {
  $('.error').hide();
  $('.userid').focus(function(){
    $(this).addClass('inputs');
  });

  $('.password').focus(function(){
    $(this).addClass('inputs');
  });

  $('.emailadd').focus(function(){
    $(this).addClass('inputs');
  });

  $('.fooderror').addClass('error');
  $('.fooderror').hide();
```

```
$('.payerror').addClass('error');
$('.payerror').hide();

$('.submit').click(function(event){
  var data=$('.userid').val();
  if(validate_userid(data))
  {
    $('.userid').next().hide();
  }
  else
  {
    $('.userid').next().show();
  }

  data=$('.password').val();
  var len=data.length;
  if(len<1)
  {
    $('.password').next().show();
  }
  else
  {
    $('.password').next().hide();
  }

  data=$('.emailadd').val();
  if(valid_email(data))
  {
    $('.emailadd').next().hide();
  }
  else
  {
    $('.emailadd').next().show();
  }

  var count=0;
  $('div').find(':checkbox').each(function(){
    if($(this).is(':checked'))
    {
      count++;
    }
  });
  if(count==0)
  {
    $('.fooderror').css({'margin-left':250}).show();
  }
  else
  {
  $('.fooderror').hide();
  }

  count=0;
  $('div').find(':radio').each(function(){
    if($(this).is(':checked'))
    {
      count++;
    }
```

```
    });
    if(count==0)
    {
      $('.payerror').css({'margin-left':250}).show();
    }
    else
    {
      $('.payerror').hide();
    }

    count=$('select option:selected').val();
    if(count==0)
    {
      $('.infobox').next().show();
    }
    else
    {
      $('.infobox').next().hide();
    }

    event.preventDefault();
  });
});

function valid_email(email)
{
  var pattern= new RegExp(/^[\w-]+(\.[\w-]+)*@([\w-]+\.)+[a-zA-Z]+$/);
  return pattern.test(email);
}

function validate_userid(uid)
{
  var pattern= new RegExp(/^[a-z0-9_]+$/);
  return pattern.test(uid);
}
```

执行以上 jQuery 示例代，运行结果如图 4-43 所示。

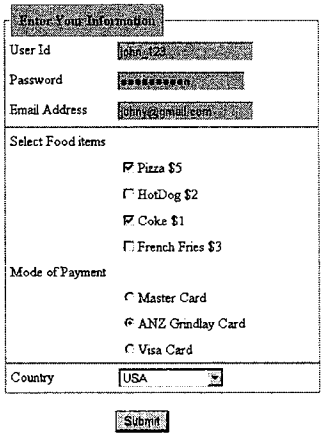


图 4-43 通用表单元素分组

4.16　表单数据序列化

问题描述

需要了解在项目中实际上表单元素如何编码。

解决方案

本攻略利用以下 HTML 文件和样式表文件。

```
<body>
    <form>
        <span class="label">Enter user id</span>
        <input type="text"  name="userid" class="userid"/>  <span class="usrerror">
</span><br/>
        <span class="label">Enter email address</span>
        <input type="text"  name="emailadd" class="emailadd"/> <span class="emerror">
</span><br/>
             <input type="submit" id="submit"/>
         </form>
<div id="message"></div>
</body>
```

可见以上表单中包含两个标签、两个输入文本字段、两个 span 元素、一个提交按钮和一个空 div 元素。我们将揭示在两个输入文本输入字段中输入的数据是如何编码的。编码的数据将显示在 div 元素 message 中。为了提供这些 HTML 元素的细节——宽度、位置、间距和边距，需要利用样式表文件。样式表文件包含的样式规则如下：

```
style.css
.label {float: left; width: 120px; }
.userid {width: 200px; }
.emailadd {width: 200px; }
.usrerror { color: red; padding-left: 10px; }
.emerror { color: red; padding-left: 10px; }
#submit { margin-left: 125px; margin-top: 10px;}
```

调用 serialize()方法、显示编码的数据的 jQuery 代码如下：

```
$(document).ready(function() {
$('.usrerror').hide();
$('.emerror').hide();
    $('#submit').click(function () {
   var info = $("form").serialize();
$('#message').text('The format when input elements are serialized:  '+info);
return false;
});
});
```

知其所以然

在 jQuery 示例代码中，首先隐藏有关 userid 和 emailadd 的错误消息，把单击事件附加

到提交按钮。在单击事件的事件处理函数中，对所有的表单元素进行序列化（以查询字符串形式进行编码）并存储到变量 info 中。然后把变量 info 中的查询字符串分配到 div 元素用于显示。在单击事件中返回 false，因为不想浏览器把文本输入字段中输入的信息发送到服务器。当然，我们想要运行分配到事件处理函数中的代码。

输入用户名和电子邮件地址之后生成的查询字符串应该类似于图 4-44。

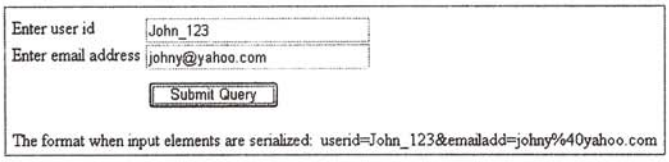


图 4-44 对输入文本框的输出进行序列化

生成的查询字符串如下：

```
userid=John_123&emailadd=johny%40yahoo.com
```

可见 userid 和 emailadd 是分配到文本输入字段的名称，而 John_123 和 johny%40yahoo.com 是由用户输入的值。电子邮件地址中的值%40 代表@符号。

复选框、单选按钮和select元素的序列化

从上面的解决方案中，我们看到了输入文本字段如何编码。下面了解复选框、单选按钮和 select 元素中被选中的选项如何编码。 显示复选框、单选按钮和 select 元素的 HTML 文件的内容如下：

```
<body>
    <form>
       <div><span class="label">Select Food items</span><br><input type="checkbox"
id="pizza" name="pizza" value=5  class="chkb">Pizza $5 <br/>
<input type="checkbox"  id="hotdog" name="hotdog" value="2"  class="chkb">HotDog
$2<br/>
<input type="checkbox"  id="coke" name="coke" value="1"  class="chkb">Coke $1<br/>
<input type="checkbox"  id="fries" name="fries" value="3"  class="chkb">French Fries
$3<br/>
</div>h
<div><span class="label">Mode of  Payment</span><br/><input type="radio"
name="paymode" class="radiobtn" value="Master Card">Master Card <br/>
<input type="radio"  name="paymode" class="radiobtn" value="ANZ Grindlay Card">ANZ
Grindlay Card<br/>
<input type="radio"  name="paymode" class="radiobtn" value="Visa Card">Visa Card<br/>
</div>
<div><span class="label">Country</span><select id="country" name="country">
<option value="0">Select a Country</option>
<option value="USA">USA</option>
<option value="United Kingdom">United Kingdom</option>
<option value="India">India</option>
<option value="China">China</option>
</select>
<br/>
        <input type="submit" id="submit"/>
```

```
        </form>
<div id="message"></div>
</body>
```

浏览下面的图 4-45，以上代码产生的布局一目了然。在样式表文件 style.css 中定义的类和 ID 选择器如下所示：

```
style.css
.label {float: left; width: 120px; }
#submit { margin-left: 125px; margin-top: 10px;}
div{padding: 5px; }
.chkb { margin-left: 125px; margin-top: 10px;}
.radiobtn { margin-left: 125px; margin-top: 10px;}
```

显示复选框、单选按钮和 select 元素中的选定选项的序列化信息，实现的 jQuery 代码如下：

```
$(document).ready(function() {
    $('#submit').click(function () {
   var info = $("form").serialize();
$('#message').text('The format when input elements are serialized:  '+info);
return false;
});
});
```

知其所以然

示例代码的关键部分执行单击事件（附加到提交按钮）的事件处理函数：单击提交按钮时，访问所有的表单元素，调用 serialize()方法以查询字符串形式对所有的表单元素进行编码，并把查询字符串存储到变量 info 中。然后把变量 info 分配到 div 元素 message 显示给用户。在单击函数中返回 false，因为不希望浏览器把选中的选项发送到服务器，只需运行分配到事件处理函数的代码。

编码的查询字符串如图 4-45 所示（当然，如果选择的选项不同，编码的查询字符串也有所不同）。

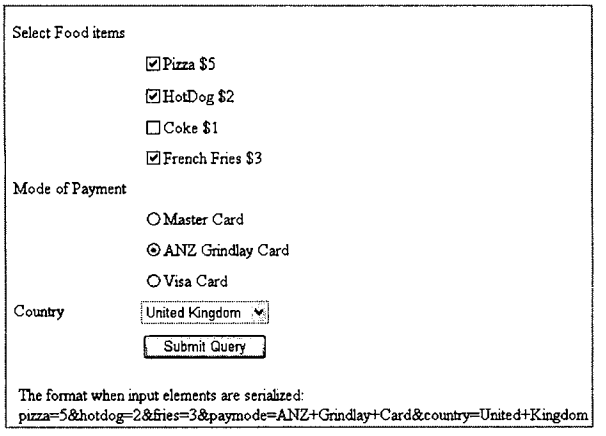


图 4-45　复选框、单选按钮和选择元素的序列化输出

可见返回的查询字符串中包含所选择的选项的名值对。

使用serializeArray()方法

在上面的示例中，调用.serialize()方法来访问表单元素。下面调用.serializeArray()方法，以便访问表单元素并以选中元素的对象（包含名/值对）数组形式返回。对象数组看起来有点像JSON数据。JSON数据采用以下名值对的格式，如下所示：

```
[
     {name: 'pizza', value: '5'},
     {name: 'hotdog', value: '2'},
     {name: 'paymode', value: 'ANZ Grindlay Card'}
  ]
```

采用与上例相同的 HTML 文件和样式表，以对象数组形式返回选中的选项，并显示选中选项的值，实现的jQuery代码如下：

```
$(document).ready(function() {
$('#submit').click(function () {
var selectedopts="";
var info = $("form").serializeArray();

$.each(info, function(i, d){
selectedopts+=d.value+" ";
});

$('#message').text('The options chosen are:  '+selectedopts);
return false;
});
});
```

下面逐行研读以上示例代码。

- 第2行，把单击事件附加到提交按钮。
- 第3行，初始化变量selectedopts，用于存储选中的选项值。
- 第4行，调用serializeArray()方法收集表单中所有选中的选项作为对象数组，并存储到变量data中。对象数据中的每个元素都具有name和value两个属性。
- 第5行，调用each()函数，分析info数组中的每个元素。在回到函数中使用两个参数i和d。i指向元素的索引位置而d指向元素（数据）。数据的格式是名/值对。例如：{ name: 'paymode', value: 'ANZ Grindlay Card'}。
- 第6行，把业已存储在selectedopts变量中的数据（即选中的复选框、单选按钮或选项的数据）的value属性串联起来。
- 第8行，在div元素message中显示选中的选项值（在selectedopts变量中）。
- 第9行，在单击事件中返回false，因为不希望浏览器把选中的选项发送到服务器，而只是要执行分配到事件处理函数的代码而已。

选中的选项值如图4-46所示。

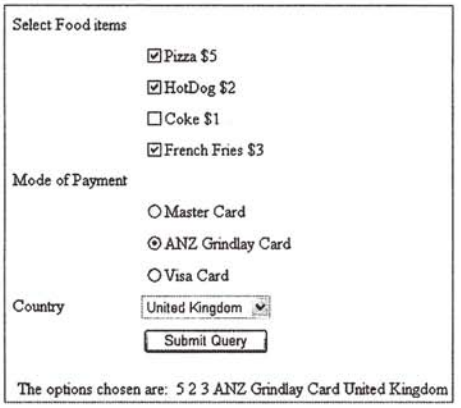


图 4-46　利用 serializeArray()方法显示复选框、单选按钮和选择元素的值

4.17　小结

本章既学习了简单的有效数据验证，比如确认字段不留空、输入一定范围内的数字等，也学习了如何确认有效的电话号码、日期、电子邮件地址等，还学习了通过 jQuery 检测复选框或单选按钮是否选中的技巧，最后学习了如何验证整个表单以及如何把表单数据序列化。下一章将学习不同的导航技巧，包括如何创建上下文菜单、折叠菜单、动态可视化菜单等等。

第5章

页面导航

本章学习如何创建不同类型的菜单，在用户的网站导航方面非常有用。本章介绍的攻略如下：

- 编写面包屑菜单；
- 为菜单项添加悬停效果；
- 创建上下文菜单；
- 创建带有快捷键的导航菜单；
- 创建右键单击上下文菜单；
- 创建具有各自菜单项的两个菜单；
- 创建具有子菜单项的两个菜单；
- 创建折叠式菜单；
- 创建动态可视化菜单。

5.1 编写面包屑菜单

问题描述

想要以面包屑[①]形式表示链接式菜单。

解决方案

下面以无序列表形式显示 Books 菜单的几个菜单项：网页开发、编程和关系数据库管理系统。HTML 文件的内容如下：

```
<body>
        <ul id="menu">
                <li><a href="http://example.com">Books</a>
        <ul>
        <li><a href="http://example.com">Web Development</a></li>
        <li><a href="http://example.com">Programming</a></li>
```

① 面包屑通常水平地出现在页面顶部，一般会位于标题或页头的下方。它们提供给用户返回之前任何一个页面的链接（这些链接也是能到达当前页面的路径），在层级架构中通常是这个页面的父级页面。面包屑提供给用户回溯到网站首页或入口页面的一条快速路径，它们绝大部分看起来就像这样：首页→分类页→次级分类页。——译者注

```
            <li><a href="http://example.com">RDBMS</a></li>
                                </ul>
                        </li>
            </ul>
    </body>
```

为了把面包屑形状赋予列表项，在样式表文件 style.css 中定义两个样式规则 liststyle 和 ulistyle，各自包含一组样式属性，内容如下：

```
style.css
.liststyle {
background-image:url(arrow.jpg);
background-repeat:no-repeat;
background-position:left;
padding-left:30px;
display: inline;
}

.uliststyle {
list-style:none;
margin:0;
padding:0;
display: inline;
}
```

把.uliststyle 和.liststyle 这两个样式规则应用到无序列表及其元素的 jQuery 代码如下：

```
$(document).ready(function() {
  $('ul').addClass('uliststyle');
  $('ul li ul li').addClass('liststyle');
});
```

知其所以然

在 HTML 文件中，为无序列表分配 ID 为 menu，由单独的列表项 Books 组成，而 Books 本身又是包含 Web Development、Programming 和 RDBMS 这 3 个元素的无序列表。此外，所有的菜单项指向假设的网站 example.com。用户单击面包屑的任何链接，就被导航到这个目标网站。

在样式表文件中，样式规则.liststyle 包含 background-image 属性，设置为 url(arrow.jpg)，显示箭头图像（参照图 5-1）。把 background-repeat 属性设置为 no-repeat，使图像仅显示一次。把 background-position 属性设置为 left，使图像显示在应用样式规则 liststyle 的元素的左侧。把 padding-left 属性设置为 30 像素，在左侧创造 30 像素的间距，把 display 属性设置为 inline，删除块元素中的任何空格，使它们显示在一行中（不带空格）。

样式规则 uliststyle 包含 list-style 属性的值设置为 none，从无序列表中删除传统的项目符号。把 margin 和 padding 属性的值设置为 0，删除传统的空格。把 display 属性设置为 inline，使块元素显示在同一行中。

在 jQuery 示例代码中，把样式规则.ulistyle 应用到无序列表，把.liststyle 应用到无序列表的列表项（内嵌在无序列表的第 1 个列表项中）。

执行 jQuery 代码，运行结果如图 5-1 所示。

Books ▸▸ Web Development ▸▸ Programming ▸▸ RDBMS

图 5-1 面包屑形式的链接元素

5.2 把悬停效果添加到菜单项

问题描述

显示包含几个菜单项的菜单，同时把悬停效果应用到菜单和菜单项。

解决方案

创建 HTML 文件，利用两个无序列表（其中一个嵌套另一个）来显示菜单标题和菜单项。HTML 文件的内容如下：

```
<body>
        <ul>
                <li><a href="http://example.com">Books</a>
                        <ul>
                           <li><a href="http://example.com">Web Development</a></li>
        <li><a href="http://example.com">Programming</a></li>
        <li><a href="http://example.com">RDBMS</a></li>
        </ul>
                    </li>
        </ul>
</body>
```

从以上 HTML 文件可见，一个无序列表元素包含列表项 Books，Books 本身又是包含 3 个列表项（显示超级链接 Web Development、Programming 和 RDBMS）的无序列表。这些超链接指向假想的网站 http://example.com，只要点击任何一个菜单项，就会把用户导航到 http://example.com 网站。

为了把菜单外观赋予无序列表元素，需要把某些样式应用到所有 3 个元素<u>、<li>和<a>。在样式表文件中为它们编写类型选择器，类型选择器中的属性会自动应用到这 3 个元素。样式表文件的内容如下：

```
style.css
ul {
 width: 200px;
}

ul li ul {
 list-style-type:none;
 margin: 5;
 width: 200px;
}
```

```
a {
  display:block;
 border-bottom: 1px solid #fff;
 text-decoration: none;
  background: #00f;
 color: #fff;
           padding: 0.5em;
}

li {        .
           display:inline;
}

.hover {
 background: #000;
}
```

把悬停事件应用到链接元素的jQuery代码如下：

```
$(document).ready(function() {
  $('a').hover(
    function(event){
      $(this).addClass('hover');
    },
    function(){
      $(this).removeClass('hover');
    }
  );
});
```

知其所以然

类型选择器ul包含width属性，设置为200像素，定义菜单标题Books的宽度。把类型选择器ul li ul应用到菜单项。它包含list-style-type属性，设置为none，从无序列表元素中删除传统的项目符号。把margin属性设置为5，使菜单项相对于菜单标题稍微缩进显示。把width属性设置为200像素，定义容纳菜单项的宽度。

类型选择器a包含display属性，设置为block，使链接元素显示为块元素而不是单个元素。把border-bottom属性设置为1px solid #fff，创建厚度为1像素的白色实线，显示在每个链接元素之下（作为分隔符）。把text-decoration属性设置为none，删除显示在超链接之下的传统的下划线。把padding属性设置为.5em（即默认大小的50%），定义超链接文本及其边框的间距。

把类型选择器li设置为inline，是为了删除列表项之间的空格。CSS类.hover包含background属性，把菜单项的背景色设置为黑色（当用户在任何超链接元素上悬停时）。

从jQuery示例代码中可见，把悬停事件应用到超链接元素。回忆悬停事件包含两个事件处理函数，一个在鼠标指针移到超链接元素之上时被调用，另一个在鼠标指针离开超链接元素时被调用。在鼠标指针移到超链接元素之上时被调用的事件处理函数中，应用CSS类hover（定义在样式表文件中），使超链接元素的背景色变黑。在鼠标指针离开超链接元素时被调用的事件处理

函数中，从超链接元素中删除 CSS 类 hover，使超链接元素回复初始时的样式。

执行 jQuery 代码，显示的菜单如图 5-2 所示。

鼠标指针在任何菜单项上悬停时，它的背景色变为黑色，如图 5-3 所示。

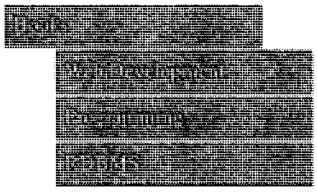


图 5-2 有 3 个菜单项的书籍菜单

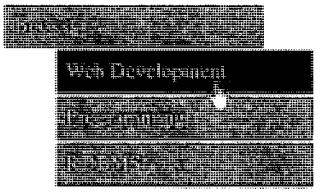


图 5-3 鼠标指针悬停时菜单项变为黑色

5.3 创建上下文菜单

问题描述

显示包含几个菜单项的菜单。当任何菜单项被悬停时（鼠标指针在其之上移动），显示有关该菜单项的信息，同时突出显示该菜单项。一旦单击菜单项，用户就被导航到相关网站。

解决方案

创建 HTML 文件，显示包含 3 个菜单项的菜单标题 Books。利用两个无序列表来创建菜单及其 3 个菜单项，其中一个无序列表嵌套另一个无序列表。列表项包含超链接元素，表示菜单项并且指向目标网站http://example.com，用户一旦选中任何菜单项，就被导航到目标网站。同时，在 3 个段落中显示有关 3 个菜单项的信息。HTML 文件的内容如下：

```
<body>
  <table>
    <td>
      <ul>
        <li><a href="http://example.com">Books</a>
          <ul>
            <li><a href="http://example.com" id="webd">Web Development</a></li>
            <li><a href="http://example.com" id="pgmng">Programming</a></li>
            <li><a href="http://example.com" id="datab">RDBMS</a></li>
          </ul>
        </li>
      </ul>
    </td>
    <td valign="top">
      <p class="web" >The wide range of books that includes how Web development can
be done
        with ASP.NET, PHP, JSP etc.</p>
      <p class="prog" >The wide range of books that includes developing Programming
skills
        in C, C++, Java etc.</p>
```

```
      <p class="rdbms" >The wide range of books that includes how Data Base Management
is
        done via Oracle, MySQL, SQL Server etc.</p>
    </td>
  </table>
</body>
```

为了使菜单显示在左侧而内容显示在右侧，创建表格，把菜单放在第一列，把包含有关菜单项信息的段落放在第二列。

为了把菜单的外观赋予无序列表元素，需要把某些样式应用到所有<u>、<li>和<a>元素。在样式表中编写它们的类型选择器，以便其中的属性可以自动应用于这3个元素。样式表文件的内容如下：

```
style.css
ul {
            width: 200px;
}

ul li ul {
            list-style-type:none;
            margin: 5;
            width: 200px;
}

a {
            display:block;
            border-bottom: 1px solid #fff;
            text-decoration: none;
            background: #00f;
            color: #fff;
            padding: 0.5em;
}

li {
            display:inline;
}

.hover {
            background: #000;
}
```

显示被鼠标指针悬停的菜单项的相关信息的jQuery代码如下：

```
$(document).ready(function() {
  $('.web').hide();
  $('.prog').hide();
  $('.rdbms').hide();

  $('#webd').hover(function(event){
    $('.web').show();
    $('.prog').hide();
    $('.rdbms').hide();
    $('#webd').addClass('hover');
```

```
    }, function(){
      $('#webd').removeClass('hover');
  });

  $('#pgmng').hover(function(event){
    $('.web').hide();
    $('.prog').show();
    $('.rdbms').hide();
    $('#pgmng').addClass('hover');
    }, function(){
      $('#pgmng').removeClass('hover');
  });

  $('#datab').hover(function(event){
    $('.web').hide();
    $('.prog').hide();
    $('.rdbms').show();
    $('#datab').addClass('hover');
    }, function(){
      $('#datab').removeClass('hover');
  });
});
```

知其所以然

在样式表文件中，类型选择器 ul 包含 width 属性，设置为 200 像素，定义菜单标题 Books 的宽度。把类型选择器 ul li ul 应用到菜单项。类型选择器 ul li ul 把 list-style-type 属性设置为 none，从无序列表元素中删除传统项目符号，把 margin 属性设置为 5，使菜单项相对于菜单标题稍微缩进显示，把 width 属性设置为 200 像素，定义用于容纳菜单项的宽度。类型选择器 a 包含 display 属性，把值设置为 block，使得超链接元素显示为块元素而不是单个元素，把 border-bottom 属性设置为 1px solid #fff，生成厚度为 1 像素的白色实线，显示在每个超链接元素之下（作为分隔符），把 text-decoration 属性设置为 none，删除超链接之下的传统下划线，把 padding 属性设置为.5em（默认字体大小的 50%），定义超链接文本及其边框的间距。

把类型选择器 li 设置为 inline，删除列表项之间的空格。

CSS 类.hover 包含 background 属性，用户点击时把菜单项（超链接元素）的背景色设置为黑色。

下面探讨 jQuery 代码语句的含义：初始时，隐藏所有 3 个段落中存储的信息（对应不同的菜单项）。也就是说，隐藏 web、prog 和 rdbms 类的 3 个段落中存储的信息，仅当相关菜单项被鼠标悬停时才显示对应的段落中的信息。

然后把悬停事件附加到第一个菜单项 Web Development，即 ID 为 webd 的超链接元素。在悬停事件的第一个事件处理函数中（此菜单项被属性悬停时执行），把 web 类的段落元素（包含 Web Development 元素的信息）设置为可见模式，显示有关网页开发的书籍信息，而其他段落元素保持隐藏状态。也就是说，prog 类和 rdbms 类的段落元素会保持隐藏状态。同时，把样式规

则.hover 中定义的属性应用到鼠标悬停的菜单项以便突出显示，当鼠标指针离开菜单项时，就执行悬停事件的第二个事件处理函数，从菜单项中删除悬停样式规则。

把悬停事件附加到第二个菜单项 Programming，即 ID 为 pgmng 的超链接元素。在悬停事件的第 1 个事件处理函数中（当此菜单项被鼠标悬停时执行），把 prog 类的段落元素（包含有关编程这一主题的信息）设置为可见模式，显示有关编程的数据信息，而使其他段落元素保持隐藏状态。同时，把样式规则.hover 中定义的属性应用到被鼠标悬停的菜单项以便突出显示。当鼠标指针离开菜单项时，就执行悬停事件的第二个事件处理函数，从菜单项中删除悬停样式规则。

最后，把悬停事件附加到第三个菜单项 rdbms，即 ID 为 datab 的超链接元素。在悬停事件的第 1 个事件处理函数中（当此菜单项被鼠标悬停时执行），把 rdbms 类的段落元素（包含数据库这一主题的信息）设置为可见模式，而使 web 类、prog 类的段落元素保持隐藏状态。同时，把样式规则.hover 中定义的样式规则应用到菜单项以便突出显示。当鼠标指针离开菜单项时，就执行悬停事件的第二个事件处理函数，从菜单项中删除悬停样式。

执行 jQuery 示例代码，会看到包含 3 个菜单项的菜单。如果在任何菜单项之上悬停鼠标，该菜单项就会突出显示，与之有关的信息将会显示，如图 5-4 所示。

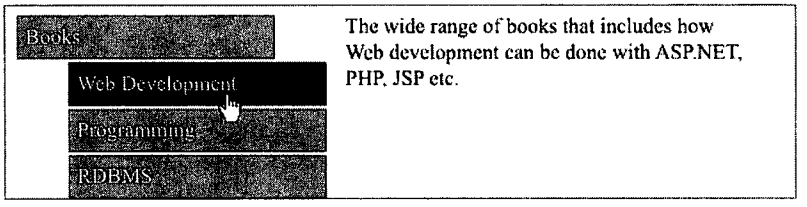


图 5-4　鼠标悬停时菜单项保持突出显示，同时显示相关信息

5.4　创建具有快捷键的导航菜单

问题描述

显示包含几个菜单项的菜单。显示菜单项的访问键。访问键即快捷键[①]，表示相应的菜单项。此外，需要确保任何菜单项被鼠标悬停时，显示与之有关的信息。在这两种情况下都必须显示有关信息：当菜单项被鼠标悬停时，以及任何菜单项的快捷键被按下时。如果用户单击任何菜单项，他就会被导航到相关网站。

解决方案

新建 HTML 文件，显示菜单标题 Books 以及 3 个菜单项。利用序列列表创建菜单和 3 个菜单项。列表项包含显示为菜单项的超链接元素。此外，在 3 个段落元素中编写有关 3 个菜单项的信息。为了使菜单项的文字显示下划线效果，把菜单项的文字内嵌于 hot 类的 span 元素。HTML

① 下文通称快捷键。——译者注

文件的内容如下：

```
<body>
  <table>
    <td>
      <ul>
         <li><a href="http://example.com">Books</a>
          <ul>
             <li><a href="http://example.com" id="webd"><span class="hot">W</span>eb
              Development</a></li>
             <li><a href="http://example.com" id="pgmng"><span class="hot">P</span>
               rogramming</a></li>
             <li><a href="http://example.com" id="datab" ><span
class="hot">R</span>DBMS
              </a></li>
          </ul>
        </li>
       </ul>
    </td>
    <td valign=top>
      <p class='web' >The wide range of books that includes how Web development can
be
        done with ASP.NET, PHP, JSP etc.</p>
      <p class='prog' >The wide range of books that includes developing Programming
        skills in C, C++, Java etc.</p>
      <p class='rdbms' >The wide range of books that includes how Data Base Management
        is done via Oracle, MySQL, Sql Server etc.</p>
    </td>
  </table>
</body>
```

为了把菜单的外观赋予无序列表元素，需要把某些样式应用到所有 3 个元素：<u>、<li>和<a>。在样式表文件中编写它们的类型选择器，以便把其中的特性自动应用到这 3 个元素。样式表文件的内容如下：

```
style.css
ul {
 width: 200px;
}

ul li ul {
 list-style-type:none;
 margin: 5;
 width: 200px;
}
a {
  display:block;
 border-bottom: 1px solid #fff;
 text-decoration: none;
  background: #00f;
 color: #fff;
  padding: 0.5em;
}
```

```
li {
  display:inline;
}

.hover {
 background: #000;
}

.hot{
  text-decoration:underline;
}
```

只要把鼠标悬停在菜单项之上，或按下菜单项的快捷键，下面的jQuery代码就会显示菜单项的相关信息。此外，应用某些样式规则突出显示被鼠标悬停的菜单项：

```
$(document).ready(function() {
  $('.web').hide();
  $('.prog').hide();
  $('.rdbms').hide();

$('body').keypress(function(event){
  if(String.fromCharCode(event.keyCode)=="w" ||
String.fromCharCode(event.keyCode)=="W")
  {
    $('#webd').hover();
  }
  if(String.fromCharCode(event.keyCode)=="p" ||
String.fromCharCode(event.keyCode)=="P")
  {
    $('#pgmng').hover();
  }
  if(String.fromCharCode(event.keyCode)=="r" ||
String.fromCharCode(event.keyCode)=="R")
  {
    $('#datab').hover();
  }
});

  $('#webd').hover(function(event){
    $('.web').show();
    $('.prog').hide();
    $('.rdbms').hide();
    $('#webd').addClass('hover');
    }, function(){
      $('#webd').removeClass('hover');
  });

  $('#pgmng').hover(function(event){
    $('.web').hide();
    $('.prog').show();
    $('.rdbms').hide();
    $('#pgmng').addClass('hover');
    }, function(){
      $('#pgmng').removeClass('hover');
```

```
  });

  $('#datab').hover(function(event){
    $('.web').hide();
    $('.prog').hide();
    $('.rdbms').show();
    $('#datab').addClass('hover');
    }, function(){
      $('#datab').removeClass('hover');
  });
});
```

知其所以然

在 HTML 文件中，请注意到超链接元素的第一个字符将被突出显示并且作为快捷键。比如超链接元素 Web Development 的快捷键被设置为字符 w，因此只要按下 w 键（或 W 键）就能够访问此菜单。为了让用户知道 w 是快捷键，需要给 w 添加下划线。为了给菜单项 Web Development 的第一个字符 W 加上下划线，把字符 W 内嵌在 span 中，并且把类名 hot 分配到 span 元素，以便在样式表文件中进行标识。同样，把所有想要显示为快捷键的字符都内嵌在 hot 类的 span 元素中。

此外，为了使菜单显示在左边，内容显示在右边，创建表格，把菜单放在第一列，把包含信息的段落放在第二列。

在样式表文件中，类型选择器 ul 包含 width 属性，设置为 200 像素，定义菜单标题 Books 的宽度。把类型选择器 ul li ul 应用到菜单项。它包含 list-style-type 属性，设置为 none，以便从无序列表元素中删除传统项目符号。把 margin 属性设置为 5，使菜单项相对于菜单标题稍微缩进显示，把 width 属性设置为 200 像素，定义容纳菜单项的宽度。

类型选择器 a 包含 display 属性，设置为 block，使超链接元素显示为块元素而不是单个元素。把 border-bottom 属性设置为 1px solid #fff，生成厚度为 1 像素的白色实线，显示在超链接元素之下（作为分隔符）。把 text-decoration 属性设置为 none，从超链接元素之下删除传统的下划线。把所有超链接元素的背景色设置为蓝色，前景色设置为白色。把 padding 属性设置为 .5em（默认字体大小的 50%），定义超链接文本和边框的间距。

把类型选择器 li 设置为 inline，删除列表项之间的任何空格。

CSS 类 .hover 包含 background 属性，用户点击菜单项（超链接元素）时把菜单项的背景色设置为黑色。

CSS 类 .hot 包含 text-decorate 属性，设置为 underline，使每个菜单项（内嵌在 hot 类的 span 元素中）的快捷键（字母）显示下划线。

在 jQuery 示例代码中，初始时 3 个段落元素是不可见的。只有在任何快捷键被按下或任何菜单项被鼠标悬停时，才显示有关信息。把 keypress 事件附加到 HTML 文件的 body 部分，监视任何按键事件。一旦发生按键事件，就使用条件语句检查按键是否为下列字母中的任何一个：w、W、 p、P、r 或 R。一旦上述任何字母被按下，就调用对应超链接元素的悬停事件。例如，

如果字母 w 或 W 被按下，就调用 ID 为 webd 的超链接元素 Web Development 的悬停事件，以便显示有关信息。所有 3 个菜单项都附加了悬停事件。我们知道悬停事件包括两个事件处理函数，在悬停事件的第一个事件处理函数中（当此菜单项被鼠标悬停时执行），把包含有关信息的段落设置为可见模式，显示需要的信息，同时使其余段落元素隐藏。此外，把样式规则.hover 中定义的属性应用到鼠标悬停的菜单项使其突出显示。当鼠标指针离开菜单项时，执行悬停事件的第二个事件处理函数，从菜单项中删除样式规则.hover。

执行示例代码，显示带有快捷键（有下划线的字母）的 3 个菜单项，如图 5-5 所示。按下快捷键或在菜单项上悬停鼠标，就会显示与之相关的信息，如图 5-4 所示。

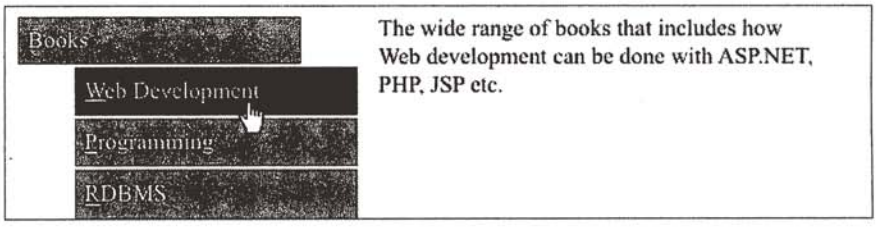


图 5-5　带有下划线快捷键的菜单项

5.5　创建一个右键单击上下文菜单

问题描述

显示一段文字，右键单击时，在屏幕上显示上下文菜单。此外，上下文菜单的菜单项具有悬停效果（即鼠标指针在菜单项之上移动时，使菜单项突出显示）。按下 Esc 键，就使上下文菜单消失。

解决方案

新建 HTML 文件，其中包含段落元素和无序列表形式的菜单。以无序列表的列表项表示菜单标题和菜单项。以内嵌于列表项的超链接元素形式来编写菜单项。超链接元素指向某假定的网站 http://example.com。点击任何菜单项，就把用户导航到该网站。HTML 文件的内容如下：

```
<body oncontextmenu="return false">
  <p class="info">
    Books are the world of information. Books are our best friends. A wise man
    always has a library of several books</p>
  <ul id="contextmenu">
    <li><a href="http://example.com">Books</a>
      <ul>
        <li><a href="http://example.com">Web Development</a></li>
     <li><a href="http://example.com">Programming</a></li>
     <li><a href="http://example.com">RDBMS</a></li>
     </ul>
    </li>
```

```
  </ul>
</body>
```

需要在样式表中定义几个样式规则，把菜单的外观赋予无序列表，此外，鼠标悬停在菜单项上时，使菜单项突出显示。样式表中的样式规则如下所示：

```
style.css
ul {
 width: 200px;
}

ul li ul {
 list-style-type:none;
 margin: 5;
 width: 200px;
}

a {
   display:block;
 border-bottom: 1px solid #fff;
 text-decoration: none;
   background: #00f;
 color: #fff;
   padding: 0.5em;
}

li {
   display:inline;
}

.hover {
 background: #000;
}
```

初始时隐藏上下文菜单，当用户在段落元素上单击右键时显示上下文菜单，按下 Esc 键时使上下文菜单消失。实现的 jQuery 代码如下：

```
$(document).ready(function() {
   $('#contextmenu').hide();
   $('.info').mousedown(function(event){
      if(event.button==2){
        $('#contextmenu').show();
       $('#contextmenu').css({'position': 'absolute', 'left':event.screenX,
          'top':event.screenY-70});
      }
   });
  $('a').hover(function(event){
     $(this).addClass('hover');
     },function(){
       $(this).removeClass('hover');
   });
   $('body').keypress(function(event){
     if(event.keyCode==27)
     {
```

```
        $('#contextmenu').hide();
        }
    });
});
```

知其所以然

显示上下文菜单时通常遇到的问题是，在段落元素上单击右键显示上下文菜单时，浏览器的上下文菜单也会一起显示。为了禁用浏览器的默认的上下文菜单，在 body 元素中设置 oncontextmenu="return false"。把类名 info 赋予段落元素，以便在 jQuery 代码中利用选择器来访问段落元素。从示例中可见，无序列表的第 1 个列表项表现为文本 Books（作为菜单标题）。此列表项本身又包含一个无序列表（表现为菜单项）。

在样式表文件中，类型选择器 ul 包含 width 属性，设置为 200 像素，定义菜单标题 Books 的宽度。把类型选择器 ul li ul 应用到菜单项。类型选择器 ul li ul 包含 list-style-type 属性，设置为 none，以便从无序列表元素中删除传统项目符号。把 margin 属性的值设置为 5，使菜单项相对于菜单标题稍微缩进显示。把 width 属性设置为 200 像素，定义容纳菜单项的宽度。

类型选择器 a 包含 display 属性，设置为 block，使超链接元素显示为块元素，而不是单个元素。把 border-bottom 属性设置为 1px solid #fff，生成厚度为 1 像素的白色实线，显示在每个超链接元素之下（作为分隔符）。把 text-decoration 属性设置为 none，删除显示在超链接元素之下的传统下划线。把所有超链接元素的背景色设置为蓝色，前景色设置为白色。把 padding 属性设置为 .5em（即默认字体大小的 50%），定义超链接文本及其边框之间的间距。类型选择器 li 设置为 inline，从列表项之间删除任何的空格。CSS 类 .hover 包含 background 属性，用户单击菜单项时把菜单项的背景色设置为黑色。

在 jQuery 示例代码中，初始时使表现为无序列表（ID 为 contextmenu）的上下文菜单不可见。然后检查 info 类的段落元素上是否发生了鼠标单击事件。回忆一下，在 HTML 文件中已经把类名 info 赋予段落元素。如果发生了鼠标单击事件，再检查被单击的是否为鼠标右键。如果单击的是鼠标左键，则事件对象的 button 属性的值为 1，如果单击的是鼠标右键，则事件对象的 button 属性的值为 2。如果单击的是鼠标右键，就使上下文菜单（表现为 ID 为 contextmenu 的无序列表元素）在屏幕上可见。

调用 css() 方法，使上下文菜单显示在事件对象的 screenX 和 screenY 属性（表示鼠标键被按下的位置）所指定之处。从 screenY 属性中存储的坐标值减去 70，使显示的上下文菜单距离段落更近，即缩减菜单和段落之间的空白。

同时，把 hover() 事件附加到超链接元素（菜单和菜单项）。当鼠标指针在任何菜单项上移动，就把样式规则 .hover 中定义的样式属性应用到超链接元素（使菜单项的背景色变黑）。当鼠标指针离开菜单项时，就从超链接元素中删除样式规则 .hover 的样式属性，使菜单项的样式复原。

最后，把按键事件附加到 body 元素，监测是否发生按键事件。如果有任何键被按下，就检

查被按下的键是否为 Esc 键（Esc 键的键码是 27）。如果 Esc 键被按下，就隐藏上下文菜单。

在段落文本上单击鼠标右键，就在屏幕上显示上下文菜单。菜单中的菜单项具有悬停效果，如图 5-6 所示。

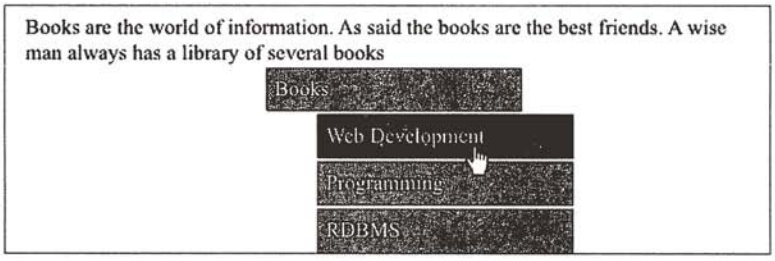


图 5-6 在段落文本上单击右键，就会显示上下文菜单

5.6 创建具有独立菜单项的两个菜单

问题描述

显示具有各自的菜单项的两个菜单。需要在菜单及其菜单项上实现悬停效果。

解决方案

新建 HTML 文件，显示两个菜单标题及其菜单项。利用无序列表来实现，一个无序列表嵌套另一个无序列表。HTML 文件的内容如下：

```
<body>
<ul id="dropdownmenu">
<li class="mainmenu">
          <a href="example.com">Books</a>
          <ul>
          <li><a href="example.com">Web Development</a></li>
          <li><a href="example.com">Programming</a></li>
          <li><a href="example.com">RDBMS</a></li>
          </ul>
          </li>
          <li class="mainmenu">
          <a href="example.com">Movies</a>
      <ul>
          <li><a href="example.com">Latest Movie Trailers</a></li>
          <li><a href="example.com">Movie Reviews</a></li>
          <li><a href="example.com">Celebrity Interviews</a></li>
          </ul>
 </li>
</ul>
</body>
```

从以上代码可见，有 ID 为 dropdownmenu 的无序列表包含 mainmenu 类的两个列表项。这

两个列表项被分配给类的名称主菜单项目。这两个列表项表示菜单 Books 和 Movies。每个列表项依次由包含 3 个元素的无序列表组成。Books 菜单包含 3 个列表项：Web Development、Programming 和 RDBMS。同样，Movies 菜单也包含 3 个列表项：Latest Movie Trailers、Movie Reviews 和 Celebrity Interviews。

把样式应用到上述无序列表，使其显示为包含各自菜单项的两个菜单。需要用到的 CSS 样式如下：

```
.mainmenu {float:left; width:200px; list-style-type:none; margin-right:5px;}
li.mainmenu ul {margin: 0;  }
a  {width: 200px;display:block; text-decoration: none; background: #00f;  color:
#fff;padding: 0.5em;   border-bottom: 1px solid #fff; }
ul#dropdownmenu li a:hover {  background: #000;}
```

当鼠标指针在菜单标题上移动时，显示其中一组菜单项而隐藏另一组菜单项，实现的 jQuery 代码如下：

```
$(document).ready(function(){
  $('li.mainmenu').hover(
    function() {
      $('ul', this).show();
    },
    function() {
      $('ul', this).hide();
    }
  );
});
```

知其所以然

在样式表文件中，类选择器.mainmenu 包含的属性自动应用到 2 个菜单标题 Books 和 Movies。类选择器.mainmenu 包含 float 属性，设置为 left，使第 1 个菜单标题显示在浏览器窗口的左侧（为第 2 个菜单标题在其右边显示创造空间）。把 width 属性设置为 200 像素，使菜单标题宽为 200 像素，把 margin-left 属性设置为 5 像素，使两个菜单标题的间距为 5 像素。

类型选择器 li.mainmenu ul 包含的样式属性自动应用到无序列表（内嵌在.mainmenu 类的列表项中，作为列表项 Books 和 Movies 的菜单项）。把 margin 属性设置为 0，使无序列表的列表项（菜单标题 Books 和 Movies 的菜单项，比如 Web development、Programming 等）与菜单标题左对齐，消除不同层次之间的缩进。

类型选择器 a 包含的属性自动应用到所有的超链接元素，即菜单以及所有的菜单项。把 width 属性设置为 200 像素，指定每个菜单项的宽度。把 display 属性设置为 block，使超链接元素显示为块元素。把 text-decoration 属性设置为 none，从超链接中删除传统下划线。background 属性把菜单标题和菜单项的背景色设置为蓝色，color 属性把菜单标题和菜单项的文本颜色设置为白色。把 padding 属性设置为.5em，在菜单文本及其边框之间创建相当于默认字体大小的 50%的间距。把 border-bottom 属性设置为 1px solid #fff，在每个超链接元素之下创建厚度为 1 像素的白色实线，作为菜单项之间的分隔符。

类型选择器 `ul#dropdownmenu li a:hover` 包含的样式属性在鼠标悬停时自动应用到菜单标题以及菜单项。当鼠标指针离开菜单标题以及菜单项时，`background` 属性把菜单标题以及菜单项的背景色变为黑色。

从 jQuery 示例代码可见，当鼠标指针在 `mainmenu` 类的列表项（即菜单标题）之上移动时，就显示内嵌在该列表项中的无序列表（包含菜单项）。当鼠标指针离开菜单标题时，就把菜单项变为不可见，即把内嵌在该列表项中的无序列表设置为隐藏模式。

初始时，显示两个菜单标题，如图 5-7 所示。

图 5-7 两个菜单标题：书籍和电影

在菜单标题 Books 上移动鼠标指针，就显示其菜单项（即内嵌在包含文本 Books 的列表项中的无序列表），如图 5-8 所示。

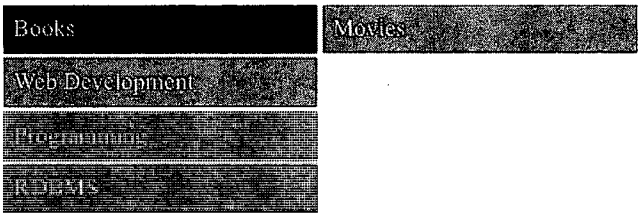


图 5-8 以悬停效果来显示书籍菜单的菜单项

在菜单标题 Movies 上移动鼠标指针时，就显示其菜单项，同时隐藏菜单标题 Books 的菜单项，如图 5-9 所示。

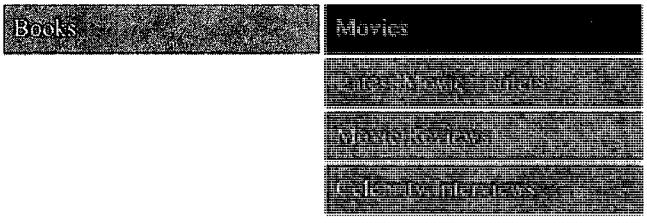


图 5-9 以悬停效果来显示电影菜单的菜单项

5.7 建立包含子菜单项的两个菜单

问题描述

显示包含菜单项的两个菜单，其中有些菜单项附加有子菜单项。需要在菜单、菜单项及其子菜单项上实现悬停效果。

解决方案

新建 HTML 文件，用于两个菜单标题及其菜单项。同时定义子菜单项。实现的方法是利用无序列表，使一个无序列表嵌套另一个无序列表。HTML 文件的内容如下：

```
<body>
  <ul class="dropdown">
    <li><a href="http://example.com">Books</a>
      <ul>
        <li><a href="http://example.com">Programming</a></li>
        <li><a href="http://example.com">Web Development</a>
          <ul>
            <li><a href="http://example.com">.Net</a></li>
            <li><a href="http://example.com">JSP</a></li>
          </ul>
        </li>
        <li><a href="http://example.com">RDBMS</a></li>
        <li><a href="http://example.com">Web Services</a></li>
        <li><a href="http://example.com">Open Source</a></li>
      </ul>
    </li>
    <li><a href="http://example.com">Movies</a>
      <ul>
        <li><a href="http://example.com">Movie Reviews</a></li>
        <li><a href="http://example.com">Celebrity Interviews</a></li>
        <li><a href="http://example.com">Latest Hollywood Movies</a>
          <ul>
            <li><a href="http://example.com">Arnold Schwarzenegger</a></li>
            <li><a href="http://example.com">Sylvester Stallone</a></li>
            <li><a href="http://example.com">Bruce Willis</a></li>
          </ul>
        </li>
        <li><a href="http://example.com">Action Movies</a>
          <ul>
            <li><a href="http://example.com">Casino Royale</a></li>
            <li><a href="http://example.com">Rambo III</a></li>
          </ul>
        </li>
        <li><a href="http://example.com">Comedy Movies</a></li>
      </ul>
    </li>
  </ul>
</body>
```

样式属性被分配给无序列表，使它们以菜单标题、菜单项和子菜单项的形式出现。样式属性如下所示：

```
style.css
a{ text-decoration: none; color: #000;}
ul{ margin: 0; list-style: none; }
ul.dropdown li {  float: left;  background: cyan; }
ul.dropdown a:hover {background: #0f0; color: #00f; }
ul.dropdown li a { display: block; padding: 4px; border-right: 1px solid #000; }
```

```
ul.dropdown ul { width: 200px; visibility: hidden; position: absolute;  }
ul.dropdown ul li {background: yellow; border-bottom: 1px solid #000; }
ul.dropdown ul li a { border-right: none; width: 100%; }
ul.dropdown ul ul { left: 100%; top: 0; }
.hover {position: relative; }
```

使菜单项和子菜单项显示在屏幕上的jQuery代码如下：

```
$(document).ready(function(){
        $("ul.dropdown li").hover(function(){
        $(this).addClass("hover");
                        $('ul:first',this).css('visibility', 'visible');
                }, function(){
                        $(this).removeClass("hover");
                        $('ul:first',this).css('visibility', 'hidden');
        });
        $("ul.dropdown li ul li:has(ul)").find("a:first").append("  >");
});
```

知其所以然

在HTML文件中创建无序列表，分配类名dropdown。它包含两个列表项，列表项的文本分别是Books和Movies。这两个列表项又各自包含无序列表（分配类名submenu）。列表项Books的无序列表包含5个列表项：Programming、Web Development、rdbms、Web Services和Open Source。至此之外，列表项Web Development又包含无序列表，表现为两个子菜单项（分别包含文本.Net和JSP）。

同样，列表项Movies中所包含的无序列表又包含4个列表项：Movies Review、Celebrity Interviews、Latest Hollywood Movies和Action Movies。除此之外，列表项Latest Hollywood Movies又包含无序列表，表现为3个子菜单项：Arnold Schwarzenegger、Sylvester Stallone和Bruce Willis。同时，列表项Action Movies也包含无序列表，表现为3个子菜单项：Casino Royale、Rambo III和Comedy Movies。

在样式表文件中，类型选择器a包含text-decoration属性，设置为none，从所有超链接元素中（即菜单标题、菜单项和子菜单项）删除传统下划线。把color属性设置为黑色，使所有菜单上的文本显示为黑色。

类型选择器ul包含margin属性，值设置为0，从列表项的左边删除层次性的缩进，使菜单项左对齐。把list-style属性设置为none，从无序列表中删除传统项目符号。

类型选择器ul.dropdown li中定义的样式自动应用到dropdown类的无序列表的列表项，即菜单标题Books和Movies。把float属性设置为left，使一个菜单标题显示在浏览器窗口的左部，为在其右边显示的另一个菜单标题创造空间。把菜单标题的背景色设置为蓝绿色。

属性选择器ul.dropdown a:hover包含了background和color属性，设置超链接元素（所有菜单项）的背景色和前景色，悬停时分别为绿色和蓝色。

类型选择器ul.dropdown li a包含的样式属性被自动应用到超链接元素，表现为菜单标题Books和Movies。它包含display属性，设置为block，使超链接元素作为独立的块元素。

把 padding 属性设置为 4 像素，在菜单文本及其边框之间创造间距。把 border-right 属性设置为 1px solid #000，在每个菜单标题的右边创建厚度为 1 像素的黑色边框，使菜单标题之间有所分隔。

类型选择器 ul.dropdown ul 包含的样式被应用到包含菜单项的无序列表。把 width 属性设置为 200 像素，使每个菜单项宽为 200 像素。把 visibility 属性设置为 hidden，使整个菜单项块保持隐藏状态，只有鼠标悬停在菜单标题上时才变为可见。把 position 属性设置为 absolute，使菜单项块显示在各自的菜单标题之下。

类型选择器 ul.dropdown ul li 包含的属性被自动应用到所有的列表项（表现为菜单项）。background 属性把所有菜单项和子菜单项的背景色设置为黄色。把 border-bottom 属性设置为 1px solid #000，创建厚度为 1 像素的黑色边框显示在每个菜单项之下作为分隔。

类型选择器 ul.dropdown ul li a 包含的属性被应用到所有的超链接元素（表现为菜单项和子菜单项）。把 border-right 属性设置为 none，从菜单项的右边删除边框，因为要在菜单项的右边显示子菜单项。把 width 属性设置为 100%，使超链接元素完全占用分配的 200 个像素的宽度。

类型选择器 ul.dropdown ul ul 被应用到表现为子菜单项的无序列表。把 left 属性设置为 100%，以菜单项的 100%的宽度来分隔子菜单项（否则菜单项就会重叠）。把 top 属性设置为 0，使子菜单项到浏览器顶边框的距离与菜单项(它的子菜单项正被显示)到浏览器顶边框的距离一致。

通过 jQuery 代码把样式规则 .hover 应用到列表项，使子菜单项显示在相对于菜单项的某个位置。

下面准备仔细探讨实际的 jQuery 代码。在此之前，首先建立有关:first 的概念，因为在 jQuery 代码中用到了:first。

:first

这是自定义选择器，返回指定的元素的第一个实例。

```
Example:
$('p:first)
```

将返回第一个段落元素。

下面探讨实际解决方案中的 jQuery 代码。当鼠标指针在菜单标题 Books 或 Movies 上移动时(ID 为 dropdown 的无序列表的列表项)，在 CSS 类 .hover 中（定义在样式表文件中）定义的属性就被应用到菜单标题以及处于可见模式的第 1 个无序列表（即菜单标题的相应的菜单项）。同样，当任何菜单项（包含子菜单项形式的无序列表）被鼠标悬停时，就会显示无序列表（即子菜单项）。当鼠标指针离开菜单标题或菜单项时，就隐藏菜单项或子菜单项。另外，对于具有内嵌的无序列表的所有列表项，追加符号>到超链接元素，表示已附加子菜单项。

初始时，显示两个菜单标题 Books 和 Movies，如图 5-10 所示。

图 5-10　菜单标题书籍和电影

在菜单标题 Books 上移动鼠标指针，就会显示它的菜单项，如图 5-11 所示。

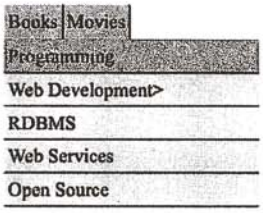


图 5-11 书籍菜单的菜单项（具有子菜单项的菜单项后显示>符号）

在具有子菜单项的菜单项上移动鼠标指针，就会显示它的子菜单项，如图 5-12 所示。

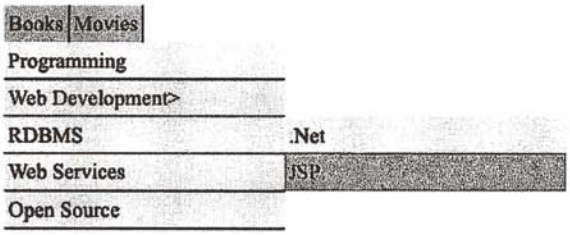


图 5-12 突出显示书籍菜单的菜单项的子菜单项

把同样的效果应用到第 2 个菜单标题 Movies。当鼠标指针悬停在菜单标题 Movies 上时，就显示它的菜单项。此外，当鼠标指针悬停在菜单项上，就显示相应的子菜单项，如图 5-13 所示。

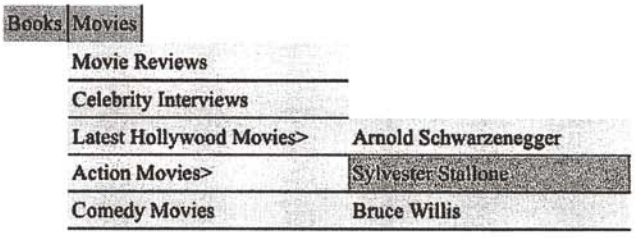


图 5-13 突出显示电影菜单的菜单项的子菜单项

5.8 创建折叠式菜单

问题描述

以折叠式菜单形式显示两个菜单，也就是说，运用上滑和下滑技巧，使鼠标悬停的菜单的子菜单项可见，而使其他菜单标题的子菜单项不可见。当鼠标指针离开两个菜单时，使菜单项消失不可见。

解决方案

新建 HTML 文件，显示两个菜单标题以及它们的菜单项。利用无序列表来实现（一个无序列表嵌套另一个无序列表）。HTML 文件的内容如下：

```
<body>
  <p class="menus">Books</p>
    <div class="menuitems">
   <ul>
    <li><a href="example.com">Web Development</a></li>
    <li><a href="example.com">Programming</a></li>
    <li><a href="example.com">RDBMS</a></li>
   </ul>
    </div>
<p class="menus">Movies</p>
    <div class="menuitems">
   <ul>
    <li><a href="example.com">Latest Movie Trailers</a></li>
    <li><a href="example.com">Movie Reviews</a></li>
    <li><a href="example.com">Celebrity Interviews</a></li>
   </ul>
</div>
</body>
```

应用样式把折叠式菜单的外观赋予以上无序列表，样式表文件的内容如下：

```
style.css
.menus{
  width: 200px;
  padding:5px;
  margin:1px;
  font-weight:bold;
  background-color: #0ff;
}

.menuitems{
  display:none;
}
a{
  display:block;
  border-bottom: 1px solid #fff;
  text-decoration: none;
  background: #00f;
  color: #fff;
  padding:10px;
  font-weight:bold;
          width: 190px;
}

.menuitems a:hover {
  background: #000;
}

li {
  display:inline;
```

```
}
ul{display:inline;}
```

以滑动效果来显示鼠标悬停的菜单标题的菜单项，同时隐藏其他菜单标题（鼠标指针从其上方离开）的菜单项，实现此效果的 jQuery 的代码如下：

```
$(document).ready(function() {
  $('p.menus').mouseout(function(){
    $("div.menuitems'").slideUp("slow");
    $('p').css({backgroundImage:""});
  });

  $('p.menus').mouseover(function(){
    $(this).css({'background-image':"url(down.png)", 'background-repeat':"no-repeat",
      'background-position':"right"}).next("div.menuitems").slideDown(500)
      .siblings("div.menuitems").slideUp("slow");
    $(this).siblings().css({backgroundImage:""});
  });
});
```

知其所以然

从 HTML 文件中可见，menu 类的两个段落元素（分别包含文本 Books 和 Movies）表现为菜单标题。每个段落元素后跟 menuitems 类的 div 元素，每个 div 元素包含具有 3 个列表项的无序列表，每个列表项表现为菜单标题的子菜单项。段落元素 Books 之下的无序列表具有 3 个列表项：Web Development、Programming 和 RDBMS。同样，段落元素 Movies 之下的无序列表具有 3 个列表项：Latest Movie Trailers、Movie Reiews 和 Celebrity Interviews。

在样式表文件中，类选择器.menus 中定义的属性被自动应用到.menu 类的段落元素，赋予它们菜单标题的外观。把 width 属性设置为 200 像素，定义菜单标题的宽度为 200 像素，把 padding 属性设置为 5 像素，使边框和菜单文本之间保持一定的间距。把 margin 属性设置为 1 像素，使两个菜单标题之间保持 1 像素的间距。把 font-weight 属性设置为 bold，使菜单标题显示为粗体，把 background-color 属性设置为#0ff，从而把菜单标题的背景色设置为蓝绿色。

类选择器.menuitems 中的属性被自动应用到 menuitems 类的 div 元素。它包含 display 属性，设置为 none，初始时使菜单项隐藏。

类型选择器 a 中定义的属性被应用到所有的超链接元素（即所有的菜单项）。把 display 属性设置为 block，使超链接元素显示为块。把 border-bottom 属性的值设置为 1px solid #fff，在每个超链接元素之下创建厚度为 1 像素的白色实线作为菜单项之间的分隔。把 text-decoration 属性设置为 none，从超链接元素中删除传统下划线。利用 background 和 color 属性分别把菜单项的背景色和前景色设置为蓝色和白色。把 padding 元素设置为 10 像素，定义菜单项文本及其边框之间的间距。把 font-weight 属性设置为 bold，使菜单项显示为粗体，把 width 属性设置为 190 像素，使菜单项宽为 190 像素。

当鼠标指针悬停在菜单项之上时，属性选择器.menuitems a:hover 中定义的属性被自动应用到内嵌在 menuitems 类的 div 元素中的超链接元素。它包含 background 属性，使鼠标悬

停的菜单项的背景色变为黑色。

类型选择器 li 包含 display 属性，值设置为 inline，删除列表项之间的任何间距。同样，类型选择器 ul 也包含 display 属性，值设置为 inline，删除无序列表之上和之下的间距。

下面探讨实际的 jQuery 代码。把 mouseout 事件附加到 menus 类的段落元素，即菜单标题 Books 和 Movies。之所以这样做是因为当鼠标指针从两个菜单上移走时，需要使菜单隐藏。

然后把 mouseover 事件附加到 menus 类的段落元素，即菜单标题 Books 和 Movies。下一步调用.css()方法，对鼠标悬停的菜单标题显示向下指针的图像，表示当前菜单标题处于扩展模式。把 background-repeat 属性设置为 no-repeat，使图像只出现一次，把 background-position 属性设置为 right，使向下指针显示在菜单标题的右端。

下一个元素（与选择器相匹配）的内容仅是 menuitems 类的 div 元素，包含被鼠标悬停的段落元素的菜单项。通过下滑效果使其可见。利用上滑效果使它的兄弟元素（另一个菜单标题即 menuitems 类的 div 元素的菜单项）不可见。最后，从业已失去焦点的菜单项中删除背景图像。

执行上述 jQuery 代码，菜单标题将纵向排列，如图 5-14 所示。

图 5-14　两个菜单标题：书籍和电影

在第 1 个菜单标题 Books 上悬停鼠标，就会以下滑效果显示它的菜单项，如图 5-15 所示。可见菜单标题具有附加的向下指针，表示菜单标题正处于扩展模式。同样，这些菜单项也具有悬停效果（即鼠标指针在它们之上移动时，它们就被突出显示）。

同样，当鼠标悬停在另一个菜单标题 Movies 之上时，就以下滑效果显示它的菜单项，而以上滑效果使菜单标题 Books 的菜单项消失，如图 5-16 所示。

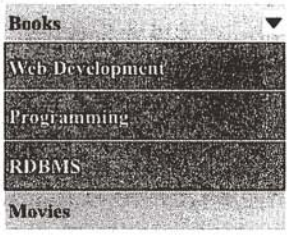


图 5-15　以下滑效果来显示书籍菜单的菜单项

图 5-16　以上滑效果来隐藏书籍菜单的菜单项而以下滑效果来显示电影菜单的菜单项

5.9　创建动态可视化菜单

问题描述

创建包含 3 个菜单 Books、Movies 和 Music 的圆角选项卡导航菜单。菜单选项卡要有悬停

效果（鼠标悬停时突出显示）。当鼠标悬停在选项卡上时，显示与该菜单选项卡有关的信息。

解决方案

新建 HTML 文件，定义文本为 Books、Movies 和 Music（内嵌在 buttons 类的 span 元素之中）的超链接元素。为了便于通过 jQuery 代码访问，分别为超链接元素分配 ID 为 booksbutton、moviebutton 和 musicbutton，使其指向某个假想的网站http://example.com。如果用户选择菜单项，就会被导航到该网站。HTML 文件的内容如下：

```
<body>
<span class="buttons"><a href="example.com" id="booksbutton"> Books </a></span>
<span class="buttons"><a href="example.com" id="moviesbutton"> Movies </a> </span>
<span class="buttons"><a href="example.com" id="musicbutton"> Music
</a></span><br><br>
<p class="books">Books of different subjects available at reasonable prices. Ranging
from web development to programming languages and text books, all are available at heavy
discount. Shipping is free. Also available in stock are popular Magazines, E-books and
Tutorial CDs at affordable prices.</p>
<p class="movies">Find new movie reviews & latest hollywood movie news. Includes new
movie trailers, latest hollywood releases, movie showtimes, entertainment news,
celebrity interviews etc. Also find Hollywood actresses, actors, videos, biographies,
filmographies, photos, wallpapers, music, jokes, and live tv channels at your
doorstep</p>
<p class="music">Find music videos, internet radio, music downloads and all the latest
music news and information. We have a large collection of music and songs classified
by type, language and region. All downloads are streamed through RealAudio. You can
also watch free music videos, tune in to AOL Radio, and search for your favorite music
artists.</p>
</body>
```

在 span 元素之下是 3 个段落元素，分配不同的类名：Books、Movies 和 Music。这 3 个段落包含与 3 个菜单选项卡有关的信息。本解决方案需要 2 个选项卡图像。一个选项卡图像用于菜单选项卡的左侧，赋予菜单选项卡一个圆角，如图 5-17 所示。

用于菜单选项卡左侧的图像保存为 tabl.jpg，而用于菜单选项卡右侧的图像保存为 tabr.jpg，如图 5-18 所示。

图 5-17 用于菜单选项卡左侧的图像

图 5-18 用于菜单选项卡右侧的图像

图 5-17 和图 5-18 中的两个图像是黑色的。还需要与前两个图像相同的绿色的两个图像（用于鼠标悬停时的菜单选项卡）。绿色的两幅图像保存为 tablselect.jpg 和 tabrselect.jpg。

样式表文件 style.css 包含几个样式规则，使 span 元素显示为选项卡导航菜单。style.css 文件的内容如下：

```
style.css
.buttons{
background-image:url(tabl.jpg);
```

```
background-repeat:no-repeat;
background-position: left;
background-color:#000;
width: 80px;
float: left;
text-align: center;
}

a{
display:block;
background-image:url(tabr.jpg);
background-repeat:no-repeat;
background-position: right;
padding:3px;
text-decoration:none;
font-weight:bold;
color:#fff;
}

.rightselectfig{
display:block;
background-image:url(tabrselect.jpg);
background-repeat:no-repeat;
background-position: right;
padding:3px;
text-decoration:none;
font-weight:bold;
color:#fff;
}

.leftselectfig{
background-image:url(tablselect.jpg);
background-repeat:no-repeat;
background-position: left;
background-color:#0f0;
width: 80px;
float: left;
text-align: center;
}
```

把悬停效果应用到菜单选项卡以及显示与菜单选项卡相关信息的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.books').hide();
  $('.movies').hide();
  $('.music').hide();

  $('a').hover(
    function(event){
      $(this).addClass('rightselectfig');
      $(this).parent().addClass('leftselectfig');
    },
    function(){
      $(this).removeClass('rightselectfig');
      $(this).parent().removeClass('leftselectfig');
```

```
    }
  );

  $('#booksbutton').click(function(event){
    event.preventDefault();
    $('.books').show('slow');
    $('.movies').hide();
    $('.music').hide();
  });

  $('#moviesbutton').click(function(event){
    event.preventDefault();
    $('.movies').show('slow');
    $('.books').hide();
    $('.music').hide();
  });

  $('#musicbutton').click(function(event){
    event.preventDefault();
    $('.music').show('slow');
    $('.books').hide();
    $('.movies').hide();
  });
});
```

知其所以然

类选择器.buttons 包含的样式属性被自动应用到 buttons 类的 span 元素，即 Books、Movies 和 Music。把背景图像属性设置为 url（tabl.jpg），使图 5-17 中的图像与菜单文本一起显示。把 background-repeat 属性设置为 no-repeat，使图像只出现一次。把 background-position 属性设置为 left，使图像显示在菜单文本的左侧，形成选项卡的左圆角。

把菜单选项卡的背景颜色设置为黑色，把菜单选项卡的宽度设置 80 像素。把 float 属性设置为 left，使菜单选项卡显示在浏览器窗口的左侧，从而创造了右侧的空间（其他菜单选项卡显示在它的右边）。把 text-align 属性设置为 center，使菜单文本显示在所定义的 80 像素宽度的中心。

类型选择器 a 中定义的属性被自动应用到超链接元素。把 display 属性设置为 block，使超链接元素作为块元素而不是单个元素。把 background-image 属性设置为 tabr.jpg，从而把图 5-18 中所示的图像应用到菜单选项卡的右侧。把 background-repeat 属性设置为 no-repeat，使图像只出现一次。把 background-position 属性设置为 right，使图像显示在右侧，形成选项卡的右圆角。把 padding 属性设置为 3 像素，创建菜单文本及其边框之间 3 像素的间距。把 text-decoration 属性设置为 none，删除显示在超链接元素之下传统下划线。把 font-weight 属性设置为 bold，使菜单文本显示为粗体，把 color 设置为 white，使菜单文本显示为白色。

当鼠标指针悬停在超链接元素上时，样式规则.rightselectfig 包含的属性将被应用到超

链接元素。它包含 display 属性，值设置为 block，使超链接元素作为块元素而不是单个元素。把 background-image 属性设置为 tabrselect.jpg，把图 5-18 中的绿色图像应用到菜单选项卡的右侧。把 background-position 设置为 right，是图像显示在右侧，将右圆角赋予菜单选项卡。把 padding 属性设置为 3 像素，使菜单文本及其边框之间的间距为 3 像素。把 text-decoration 属性设置为 none，删除显示在超链接元素之下的传统下划线。把 font-weight 属性设置为 bold，使菜单文本显示为粗体，把 color 设置为 white，使菜单文本显示为白色。

当鼠标指针悬停在菜单选项卡上时，样式规则 .leftselectfig 包含的属性被应用到菜单选项卡。它包含背景图像属性，设置为 url（tablselect.jpg）使图 5-17 中的绿色图像与菜单文本一起显示。把 background-repeat 属性设置为 no-repeat，使图像只出现一次。把 background-position 属性设置为 left，使图像显示在菜单文本的左侧，从而把左圆角赋予菜单选项卡。把菜单选项卡的背景色设置为绿色，赋予悬停效果。为菜单选项卡分配宽度为 80 像素。把 float 属性设置为 left，使菜单选项卡显示在浏览器窗口的左侧，从而创造出右侧空间（其他菜单选项卡显示在它的右边）。把 text-align 属性设置为 center，使菜单文本显示在所定义的 80 像素宽度的中心。

在 jQuery 示例代码中，初始时把 Books、Movies 和 Music 类的所有段落元素设置为不可见，只有当属性悬停在菜单选项卡上时，才显示相应的段落元素。把悬停事件附加到超链接元素，把样式规则 .rightselectfig 中定义的属性应用到鼠标悬停的菜单选项卡，使其变绿。同时把图像 tabrselect.jpg 添加到右侧，生成菜单选项卡的右圆角。用类似的方法生成选项卡的左圆角。

当鼠标不再悬停在选项卡上时，就删除样式规则 .rightselectfig 和 .leftselectfig 的样式属性，使菜单选项卡回复初始时的样式（初始时鼠标远离菜单选项卡）。

接下来把单击事件附加到选项卡。在本示例中阻止把表单提交到服务器，单击菜单选项卡时也阻止把用户导航到目标网站。显示与被单击选项卡相关的信息，而隐藏与其他选项卡相关的段落元素的内容。

当鼠标悬停在 Books 菜单选项卡上时，菜单选项卡的背景色变为绿色，与书籍相关的信息业被显示出来，如图 5-19 所示。

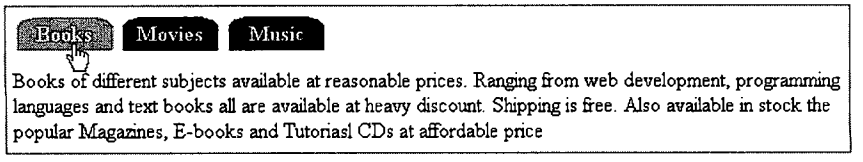


图 5-19　鼠标悬停时突出显示菜单选项卡，显示相关信息

为了将动画效果赋予正在显示的文本，可以使用 slideDown()和 slideUp()方法，而不是简单的 show()和 hide()方法，如以下 jQuery 代码所示：

```
$(document).ready(function() {
  $('.books').hide();
  $('.movies').hide();
  $('.music').hide();
```

```
    $('#booksbutton').mouseover(function(){
      $('.books').slideDown('slow');
      $('.movies').slideUp('slow');
      $('.music').slideUp('slow');
    });

    $('#moviesbutton').mouseover(function(){
      $('.movies').slideDown('slow');
      $('.books').slideUp('slow');
      $('.music').slideUp('slow');
    });

    $('#musicbutton').mouseover(function(){
      $('.music').slideDown('slow');
      $('.books').slideUp('slow');
      $('.movies').slideUp('slow');
    });
  });
```

5.10 小结

本章学习了如何创建不同类型的菜单，如面包屑菜单、上下文菜单、折叠式菜单和动态可视化菜单，也学习了如何使用快捷键来访问菜单项，以及如何创建具有悬停效果的菜单和菜单项。

下一章包括关于动画效果的各种示例。学习如何创建图像滑块、水平和垂直图像滚轮、图像切换器，学习如何创建新滚轮以及如何在网页上显示图像等。

第6章 视觉特效

本章中，我们会一起探究多种用来实现不同动画特效的攻略。具体来说，包括如下解决方案：

- 水平和垂直显示图片；
- 创建水平滑动的图片浏览器；
- 显示一幅图片，当点击时向左滚动并消失；
- 创建一幅图片，使其向左滚动，消失，然后在右侧重新出现；
- 在浏览器窗口的中间滚动图片；
- 在鼠标悬停的时候依次显示图片；
- 垂直滚动图片；
- 水平滚动图片；
- 创建新闻滚动浏览器；
- 在鼠标悬停时显示放大的图片；
- 按页显示图片；
- 在任意两个方向上切换图片；
- 编写钟摆式滚动浏览器；
- 使用数组来滚动图片。

6.1 水平和垂直显示图片

问题描述

现有5张图片，让它们依次水平显示，彼此之间保持很小的距离。

解决方案

首先我们会创建一个HTML文件，将这5张图片以超链接的形式显示。这样，如果访问者点击图片，他就会被导航到目标站点（假设为任意的虚拟站点）。这个HTML文件如下所示：

```
<body>
<div id="images">
<a href="http://example.com" ><img src="image1.jpg" width=150px height=150px /></a>
<a href="http://example.com"><img src="image2.jpg" width=150px height=150px /></a>
<a href="http://example.com"><img src="image3.jpg" width=150px height=150px /></a>
<a href="http://example.com" ><img src="image4.jpg" width=150px height=150px /></a>
<a href="http://example.com" ><img src="image5.jpg" width=150px height=150px /></a>
</div>
</body>
```

所有 anchor 元素中图片都嵌入在 ID 为 images 的 div 元素中。我们可以看到，在上面的 HTML 文件中，所有图片的宽度和高度都被设置为 150px，这会统一它们的显示样式。

在一行中依次显示这些图片的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var $pic = $('#images a');
  $pic.hide();
  var imgs = $pic.length;
  var next;
  for (i=0;i<imgs;i++){
    next=$pic.eq(i);
    next.css({'position': 'absolute','left':160*i});
    next.show();
  }
});
```

知其所以然

在查看代码之前，首先让我们一起研究一下在上述代码中所使用的两个方法：.eq()和.length。

1. .eq()

.eq()是一种过滤方法，它会从指定的集合中取得位于指定索引位置的元素，然后以 jQuery 对象的形式返回。

语法

```
.eq(index)
```

这里的 index 指定的是集合中的位置（从 0 到长度 - 1）. 如果 index 值超出了边界范围，就会返回空的 jQuery 对象。

2. .length

这个方法会返回符合条件的元素的数量。

语法

```
.length
```

现在让我们来看一下实际的 jQuery 解决方案。首先我们会收集 ID 为 images 的 div 元素中所有的 anchor 元素（a），并将其存储在数组$pic 中。由于所有图片都是 anchor 元素的组成部分，因此这 5 张图片的信息都会被存储在$pic 数组中。我们还使用$pic.hide()将所有图片

隐藏起来。稍后我们会借助 for 循环来依次显示这些图片。

然后我们会计算$pic 数组的长度，并将其存储在变量 imgs 中。由于这 5 幅图片都存储在$pic 数组中，因此存储在 imgs 变量中的值是 5。然后我们每次取得$pic 数组中的一张图片，将其存储在 next 变量中并显示出来。

最终，借助于.css()属性，next 变量所指向的图片的位置被设定在从浏览器窗口左侧边界计算的绝对位置上。第一幅图片的位置被设置为距左侧边界 0px，第二幅图片的位置距左侧边界 160px，第三幅图片距左侧边界 320px，依此类推。这样我们就为所有图片都指定了位置，彼此之间有 10px 的间隔。

执行上述 jQuery 代码，我们会得到如图 6-1 所示的效果。

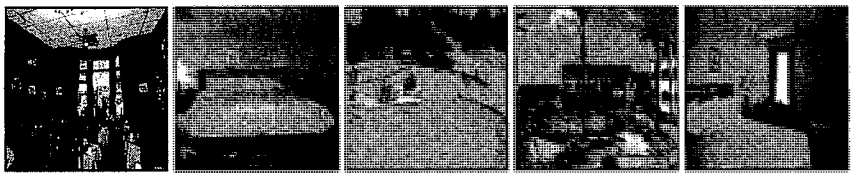


图 6-1　所有图片显示在一行中

垂直显示图片

只需要改变一条语句，我们就可以让上面的 jQuery 代码以垂直的方式依次显示图片。如果我们将这条语句：

```
next.css({'position': 'absolute','left':160*i});
```

改为下面这样：

```
next.css({'position': 'absolute','top':160*i});
```

那么图片就会在距离浏览器窗口顶端边界指定的位置，而不是距离左侧边界的位置显示。根据 i 的值，上面的语句会使图片在距离浏览器窗口顶端边界 0px、160px、320px 等位置显示。

完整代码如下所示：

```
${document).ready(function() {
  var $pic = $('#images a');
  $pic.hide();
  var imgs = $pic.length;
  var next;
  for (i=0;i<imgs;i++){
    next=$pic.eq(i);
    next.css({'position': 'absolute','top':160*i});
    next.show();
  }
});
```

通过这段代码你所得到的显示结果如图 6-2 所示。

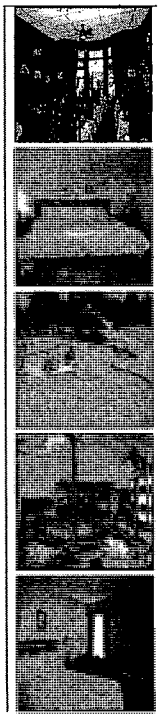


图 6-2　所有图片以垂直的方式依次显示

6.2 创建水平滑动的图片浏览器

问题描述

创建一条图片画廊（其中包含 5 幅图片），在初始的时候显示 3 幅图片。使用水平滑动条，可以看到其余图片。

解决方案

首先让我们来创建一个 HTML 文件，并以超链接的形式来定义图片画廊中的这 5 幅图片。如果我们点击图片，就会被导航到目标站点，以显示图片的完整信息。在这里我们只是将目标站点假设为虚拟的站点。HTML 代码会像下面这样：

```
<body>
<div id="scroller">
<div id="images">
<a href="http://example.com" ><img src="image1.jpg" width=150px height=150px /></a>
<a href="http://example.com"><img src="image2.jpg"  width=150px height=150px /></a>
<a href="http://example.com"><img src="image3.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image4.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image5.jpg"  width=150px height=150px /></a>
</div>
</div>
</body>
```

我们看到这里有两个 div 元素，ID 为 images 的元素嵌入在 ID 为 scroller 的元素之中。

在 ID 为 images 的 div 元素中，我们定义了 5 个 anchor 元素，每个都包含了一幅图片。所有图片的宽度和高度都被统一设定为 150px。

对 ID 为 scroller 的 div 元素，我们应用了.imageslider 样式规则，对 ID 为 images 的 div 元素。我们会应用.pics 样式规则。.imageslider 样式规则中的 style 属性，定义了指定宽度的窗口，使其最多显示 3 幅图片和一个滚动条（这会让我们查看隐藏的图片）。.pics 样式规则中的 width 属性被设定为显示所有五幅图片所需要的总宽度。在这个样式规则中定义的 width 属性决定了我们可以滑动的空间。这两个样式规则都定义在外部样式表文件 style.css 中。将 style.css 属性应用到 ID 为 scroller 和 images 的 div 元素的 jQuery 代码如下所示：

```
$(document).ready(function() {
$('#scroller').css({'margin':auto, 'width':'490px', 'height':'170px',
'overflow':'scroll'});
$('#images').css({'width':'790px'});
});
```

知其所以然

在应用给 ID 为 scroller 的 div 元素的 style 属性中，我们会看到 margin 属性的值被设置为 auto，这会根据浏览器窗口的大小来自动调整图片滑动窗口边界的空白；这样，窗口就会显示在浏览器窗口的正中央。width 属性被设置为 490px，这样可以放置宽度为 150px 的 3 幅图

片，包括它们之间的距离。height 属性被设置为 170px，overflow 属性的值被设定为 scroll，这使得窗口总会出现水平的滚动条。

类似地，应用到 ID 为 images 的 div 元素上的 style 属性将 width 属性设置为 790px（定义所有 5 幅图片的宽度总和）。也就是说，将要应用到外边的 div 元素（ID 为 scroller）上的样式规则是为了指定窗口的宽度（定义我们想在其中看到的图片的数量），而里面的 div 元素（ID 为 images）的样式规则是为了定义看到所有图片所需要的宽度。

这样，我们就得到了一个位于浏览器窗口中间的窗口，其中显示有 3 幅图片，还带有水平的滚动条，滚动它可以看到隐藏的图片，如图 6-3 所示。

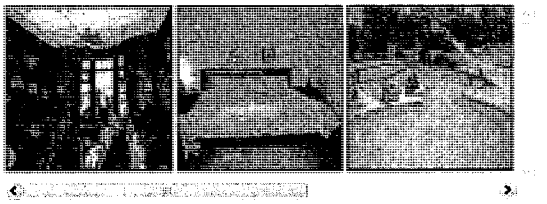


图 6-3　显示在带有水平滚动条的窗口中的图片

我们可以修改样式规则中的属性值，使得窗口垂直滚动而不是水平滚动。对 jQuery 代码的修改会是下面这样：

```
$(document).ready(function() {
$('#scroller').css({'margin':auto, 'width':'170px', 'height':'490px',
'overflow':'scroll'});
$('#images').css({'height':'790px'});
});
```

我们可以看到，在上面的样式属性中，对 width 和 height 属性的值都进行了修改，width 值减小了，而 height 值增大了。图片滚动器的效果如图 6-4 所示。

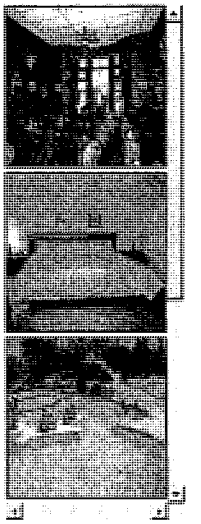


图 6-4　所有图片依次垂直显示，带有垂直滚动条

6.3　显示一幅图片，点击时向左滚动并消失

问题描述

现有一幅显示在浏览器窗口左侧的图片。被点击的时候，图片向左滚动并消失。

解决方案

让我们创建一个 HTML 文件来定义显示图片的 image 元素，我们想要它显示在屏幕上。HTML 文件会像下面这样：

```
<body>
<img src="image1.jpg" width=150px height=150px class="image"/>
```

```
</body>
```

我们会看到，在上面的 HTML 代码中，img 元素的 class 被设为 image，这会被用于在图片上添加点击事件。图片的宽度和高度被设置为 150px。图片的滚动效果是通过设置 position 属性达到的。通过将 position 属性值设置为 relative，我们就可以让图片滚动。

现在让我们在外部样式表文件 style.css 中编写类选择器.image，从而自动地为 class 为 image 的 HTML 元素——也就是 img 元素——设置 position 属性值。样式表可能会像下面这样：

```
style.css
.image{
position:relative;
}
```

我们可以看到，类选择器.image 将 position 属性设置为 relative。让我们编写 jQuery 代码来将 click 事件添加到 class 为 image 的 HTML 元素，也就是 img 元素上。我们还需要在 click 事件的事件处理函数中定义动画的代码。代码可能像下面这样：

```
$(document).ready(function() {
  $('.image').click(function (event){
    $(this).animate({'left': -160}, 'slow' );
  });
});
```

知其所以然

我们可以看到，在（针对 click 事件的）事件处理函数中定义的 animate()方法使图片向左侧滚动，并且在离浏览器左侧边界 160px 的位置停下，这样可以使这幅宽度为 150px 的图片完全消失。这样赋给 left 属性的值代表的是图片会移动到哪一端，然后图片将会从当前状态转换到属性值所指定的状态。初始的时候，图片会完全显示，如图 6-5 所示。

点击图片的时候，它会向左滚动并慢慢消失，如图 6-6 所示。

图 6-5 载入页面的时候显示图片

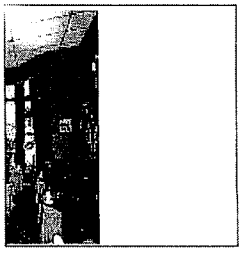


图 6-6 当图片向左滚动的时候，只有部分可见

让滚动的图片显示在浏览器窗口的中间

在上一个解决方案中，我们看到，图片在初始的时候会显示在浏览器窗口的左侧。现在我们想要让它在浏览器的中间显示。我们还想要将图片放置在不可见的窗口中，从而当我们点击图片

的时候，它会向左滚动（在不可见的窗口的边界之内）并逐渐消失。

我们可以把 HTML 文件修改为下面这样：

```
<body>
<div id="scroller">
<img src="image1.jpg" width=150px height=150px class="image"/>
</div>
</body>
```

在 HTML 代码中，你可以看到，img 元素被包含在 div 元素之中，它的 ID 是 scroller。使用 div 元素的原因是为了给图片所在的窗口（我们想让图片在其中滚动）的宽度和高度赋值。

我们在样式表中编写了 ID 选择器#scroller，从而在其中定义的 style 属性能够自动地应用给 ID 为 scroller 的 div 元素，而不需要任何 jQuery 代码。样式表中还会包含类选择器 .image，它会将 img 元素的 position 属性设置为 relative，这对于使图片滚动是必要的。

样式表文件 style.css 如下所示：

```
style.css
#scroller{
width:155px;
height:155px;
margin:auto;
overflow:hidden;
position:relative;
}
.image{
position:relative;
}
```

jQuery 代码会为图片添加点击事件，而 animate 方法（定义在事件处理函数中）会使图片向左滚动，并停在距包含它的窗口的左侧边界 160px 的位置（从而完全消失），代码如下所示：

```
$(document).ready(function() {
  $('.image').click(function (event){
    $(this).animate({'left': -160}, 'slow');
  });
 });
```

知其所以然——第二部分

我们可以看到 ID 选择器#scroller 中将 width 和 height 属性都设置为 155px，这为图片定义了不可见的窗口的宽度和高度。margin 属性被设置为 auto，从而窗口会使用浏览器窗口的空白值，让窗口自动在浏览器窗口左右位置的中间显示。overflow 属性的值被设置为 hidden，使得滚动到窗口边界之外的图片区域不可见。position 属性被设置为 relative，这使得图片相对于包含它的窗口滚动。

类选择器 .image 中将 position 属性设置为 relative，使得图片从当前位置开始滚动。初始的时候，图片的显示效果如图 6-5 所示，然而，它当然会显示在浏览器窗口的中间而不是左侧。点击图片的时候，它会向左滚动，并慢慢消失，如图 6-6 所示。

6.4 创建图片，使它向左滚动消失，然后从右侧重新出现

问题描述

现有一幅显示在浏览器窗口中间的图片（包含在不可见的窗口中）。点击图片的时候，它会（在包含它的窗口中）向左滚动并消失。当图片完全消失的时候，它会从窗口的右侧重新出现并停在开始时的位置上。

解决方案

我们需要编写 HTML 代码来显示图片。我们还需要将 image 元素包含在 div 元素之中，如下所示：

```
<body>
<div id="scroller">
<img src="image1.jpg" width=150px height=150px class="image"/>
</div>
</body>
```

在 HTML 代码中，我们会看到 img 元素被包含在 div 元素之中，它的 ID 被设置为 scroller。使用 div 元素的原因是为了给包含图片的不可见窗口（我们想要图片在其中滚动）的 width 和 height 属性赋值。

我们在样式表中编写了 ID 选择器#scroller，从而在其中定义的 style 属性能够自动应用到 ID 为 scroller 的 div 元素上，而不需要编写 jQuery 代码。样式表还会包含类选择器.image，它会将 img 元素的 position 属性赋值为 relative，这对于让图片滚动是必要的。

样式表文件 style.css 可能会像下面这样：

```
style.css
#scroller{
width:155px;
height:155px;
margin:auto;
overflow:hidden;
position:relative;
}
.image{
position:relative;
}
```

使得图片滚动的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.image').click(function (event){
    ${this).animate(
      {'left': -160}, 'slow',
      function(){
        $(this).css('left',150);
        $(this).animate({'left': 0}, 'slow');
      }
```

```
        );
    });
});
```

知其所以然

我们可以看到ID选择器#scroller中将width和height属性设置为155px，这为包含图片的不可见窗口定义了宽度和高度。margin属性被设置为auto，从而窗口会使用浏览器窗口的边缘空白值。overflow属性被设置为hidden，这使得滚动到窗口边界之外的图片区域变得不可见。position属性被设置为relative，这使得图片相对于包含它的窗口滚动。类选择器.image中将position属性设置为relative，这使得图片从当前位置开始滚动。

在上面的jQuery代码中我们可以看到，在图片上添加了click事件。在click事件的处理函数中，我们使用animate方法让图片慢慢地向左滚动（在包含该图片的窗口的边界之内），并停在距离窗口左侧160px的位置，这样，图片就完全消失了。在animate方法的回调函数中（它会在animate方法完成的时候触发，也就是在图片完全消失的时候），我们使用.css()方法让图片显示在距离不可见窗口左侧150px的位置，使得图片左边的很小一部分显示。在同一个回调函数中，我们再次使用animate方法使得图片向左滚动，并停在距离不可见窗口左侧边界0px的位置；也就是让图片停在开始时所处的位置上。

初始的时候，图片会像图6-7这样显示。

图6-7　载入页面时的图片

当点击图片的时候，它会向左滚动，并慢慢消失，如图6-8所示。

当图片完全消失之后，它会再从窗口右侧出现，并向左滚动，如图6-9所示：

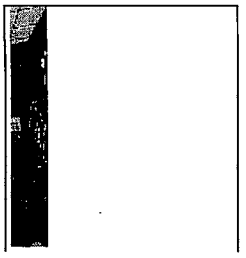


图6-8　图片向左滚动

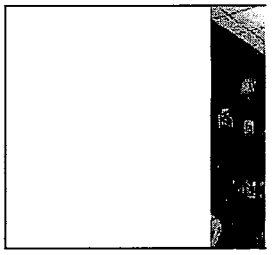


图6-9　图片从不可见窗口的右侧边界出现

6.5 使图片在浏览器窗口中间滚动

问题描述

现有一幅图片显示在浏览器窗口的中间（包含在一个不可见窗口中）。你希望该图片一直滚动；也就是它会向窗口的左边界滚动并消失，然后从窗口的右侧出现，再次滚向左侧，如此循环。

解决方案

我们需要编写 HTML 代码来显示图片。还需要在这里将 image 元素包含在 div 中，如下所示：

```
<body>
<div id="scroller">
<img src="image1.jpg" width=150px height=150px class="image"/>
</div>
</body>
```

在 HTML 代码中，我们可以看到 img 元素被包含在 div 元素中，它的 ID 被赋值为 scroller。使用 div 元素的原因是要给包含图片的不可见窗口（我们想要图片在其中滚动）设定宽度和高度。

我们在样式表中编写了 ID 选择器#scroller，从而在其中定义的 style 属性能够自动应用给 ID 为 scroller 的 div 元素上，而不需要使用 jQuery 代码。样式表中还包含了类选择器 .image，它将 img 元素的 position 属性赋值为 relative，这对于让图片滚动是必需的。

样式表文件 style.css 可能会像下面这样：

```
style.css
#scroller{
width:155px;
height:155px;
margin:auto;
overflow:hidden;
position:relative;
}
.image{
position:relative;
}
```

使得图片滚动的 jQuery 代码如下所示：

```
$(document).ready(function() {
  scroll();
});

function scroll()
{
  $('.image').animate{
    {'left': -160}, 'slow',
    function(){
      $('.image').css('left',150);
      $('.image').animate({'left': 0}, 'slow');
    }
  );
```

```
    setTimeout(scroll, 1200);
}
```

知其所以然

我们会看到，ID选择器#scroller中将width和height属性都设置为155px，这定义了放置图片的不可见窗口的宽度和高度。margin属性被设置为auto，这样窗口会采用浏览器窗口中定义的边缘空白，自动地使窗口显示在浏览器窗口宽度的中间位置。overflow属性的值被设置为hidden，这使得滚动到窗口边界外部的图片部分不可见。position属性被赋值为relative，这使得图片相对于包含它的窗口滚动。类选择器.image中将position属性设置为relative，这使得图片从当前位置开始滚动。

在探究上面的jQuery代码的工作原理之前，让我们首先查看setTimeout()方法，它将在下一节中被用到。

setTimeout()

这个方法被用于使事件在特定的时间之后发生。

语法：

```
setTimeout(action, time)
```

setTimeout()方法有两个参数：我们将要采取的动作以及在采取指定的动作之前所需要等待的时间，这个时间以毫秒计算。

在上面的jQuery代码中，我们可以看到对scroll函数的定义，它包含animate方法，会让图片慢慢地（在包含图片的窗口的边界之内）向左滚动，并停在离窗口左侧160px的位置；这样，图片就完全消失了。在animate方法的回调函数中（它会在animate方法完成的时候被触发，也就是图片完全消失的时候），我们使用.css()方法使得图片在离不可见窗口左侧150px的位置出现，这样会显示图片左侧边缘的一小部分。在同一个回调函数中，我们再次使用了animate方法，使得图片向左滚动，并停在0px的位置（从不可见窗口的左侧计算）；这样，图片会停在它开始时所处的位置。

当载入页面的时候，完成所有上述功能的scroll函数就会被触发，使得图片在窗口中完整地滚动一次。在此之后，在scroll函数中，每隔1200毫秒我们就会再次调用。也就是说，图片一直会每隔1200毫秒滚动一次。开始的时候，图片会像图6-7中这样显示。然后会向左滚动并慢慢消失，如图6-8所示。

当图片完全消失的时候，它会从窗口的右侧重新出现，并向左滚动，如图6-9所示，过程会这样继续。

6.6　在鼠标悬停时依次显示图片

问题描述

现有几幅图片，让它们依次显示。把鼠标悬停在上面时，会显示第一幅图片，然后它会淡出，而另一幅图片淡入。然后第二幅图片在鼠标悬停的时候淡出，而第三幅图片淡入，依此类推。在

显示了最后一幅图片之后，第一幅图片重新显示。

解决方案

让我们创建一个 HTML 文件来以超链接的形式显示所有 5 幅图片，这样如果访问者点击图片，就会被导航到显示图片所代表的对象完整信息的目标网站上。HTML 文件应该像下面这样：

```
<body>
<a class="imge" href="http://example.com" ><img src="image1.jpg" width=300px
height=300px></a>
<a class="imge" href="http://example.com"><img src="image2.jpg" width=300px
height=300px></a>
<a class="imge" href="http://example.com"><img src="image3.jpg" width=300px
height=300px></a>
<a class="imge" href="http://example.com" ><img src="image4.jpg" width=300px
height=300px></a>
<a class="imge" href="http://example.com" ><img src="image5.jpg" width=300px
height=300px></a>
</body>
```

所有 anchor 元素的 class 属性都被赋值为 image，这样就会为它们应用在外部样式表（style.css）中定义的类选择器.image 中所定义的样式属性。

class 名还被用于识别我们想要应用 jQuery 代码的元素。所有图片的 width 和 height 属性都被赋予相同的 300px，这样可以使其有统一的显示风格。

样式表文件 style.css 可能像下面这样：

```
style.css
.imge{
position:absolute;
top:10px;
left:10px;
}
```

下面的 jQuery 代码会使图片慢慢淡出（变为不可见），并被慢慢淡入的另一幅图片所替换（变为可见）：

```
$(document).ready(function() {
  $(".imge").hide();
  $('.imge:first').fadeIn('slow');
  $('.imge').hover(
    function(){
      $(this).fadeIn('slow');
    },
    function(){
      var next = ($(this).next().length) ? $(this).next() :$('.imge:first');
      $(this).fadeOut('slow');
      next.fadeIn('slow');
    }
  );
});
```

知其所以然

我想要的效果是，当我将鼠标悬停在一幅图片上时，它会在原地被另一幅图片所替换，因此我将类选择器.image 中的 position 属性设置为 absolute，这样就定义了图片要显示在网页上的准确位置。另外，我还将 top 和 left 属性设置为 10px，从而确保图片的位置与浏览器的左边界和上边界之间的距离都是 10px。

现在来看一下 jQuery 代码本身，开始时我们让所有图片都消失，因为我们想要它们依次显示。然后我们让 HTML 元素中 class 为 imge 的第一个元素慢慢地显示在网页上。那样，在打开网页的时候，第一幅图片（所有图片中的）会慢慢显示。这是第一个会显示在网页上的图片。一旦完成，我们就会向所有图片（所有 class 为 imge 的 HTML 元素）添加 hover 事件。

在 hover 事件中，我们让当前图片在鼠标悬停的时候慢慢显示。当鼠标指针从当前显示的图片上移出时，我们会看到下一幅要显示的图片。首先我们会检查我们显示的是不是（所有 class 为 imge 的元素中）最后一幅图片，然后我们会将序列中的下一幅图片赋给变量 next。如果我们显示的是最后一幅图片，那么（class 为 imge 的 HTML 元素中）第一幅图片就会被赋给变量 next。也就是说，我们会设定变量 next，使它指向序列中的下一幅图片，或者指向第一幅图片（如果我们显示的是最后一幅图片）。

一旦我们决定要显示 next 变量中的图片，那么当前可见的图片就会慢慢地淡出。最终，变量 next 中的图片（指向序列中下一幅图片）会慢慢地显示在网页上（淡入）。

最初，我们会在屏幕上看到 5 幅图片中的第一幅，如图 6-10 所示。

当鼠标移过图片时（也就是将鼠标移到图片上然后再移走），序列中的下一幅图片就会慢慢显示，如图 6-11 所示。类似地，所有图片会依次显示。当达到最后一幅图片的时候，就会重新显示第一幅。

图 6-10　载入网页的时候显示第一幅图片

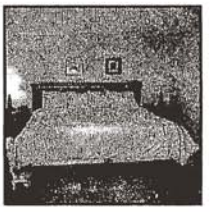


图 6-11　当鼠标移过时，第一幅图片会被下一幅图片所替换

制作幻灯效果

现在让我们来做个幻灯效果。我们会显示一些图片，当点击的时候，当前图片会被序列中的其他图片依次替代。唯一需要对之前的 jQuery 代码所做的修改就是要替换 hover 事件，我们会向图片添加 click 事件。jQuery 代码会像下面这样：

```
$(document).ready(function() {
```

```
    $(".imge").hide();
    var next;
    $('.imge:first').fadeIn('slow');
    $('.imge').click(function(event){
      $(this).fadeIn('slow');
      next = ($(this).next().length) ? $(this).next() :$('.imge:first');
      $(this).fadeOut('slow');
      next.fadeIn('slow');
      event.preventDefault();
    });
});
```

我们能够看到，在初始的时候，所有的图片都不可见。第一幅图片通过淡出效果消失。我们在图片上附加了 click 事件，如果在任意图片上发生了点击事件，那么就会取得序列中的下一幅图片，并存储在变量 next 中。之前可见的图片会慢慢地使用淡出效果消失，而序列中的下一幅图片（被取到 next 变量中）使用淡入的效果显示。为了在点击图片的时候不再转移到目标站点，我们使用了事件对象的 preventDefault 方法（自动通过 JavaScript 传递给事件处理函数）。

6.7 垂直滚动图片

问题描述

现有多幅图片，上下依次排列，让它们慢慢地在浏览器窗口中向下滚动。当所有图片消失的时候，让它们在浏览器窗口的顶端重新显示，并且继续向下滚动。

解决方案

让我们创建一个 HTML 文件，其中以超链接的形式定义了将要滚动的 5 幅图片。如果我们点击任一幅图片，它就会将我们导航到显示图片完整信息的目标站点。在此我们将目标站点设置为虚拟的站点。HTML 文件应该像下面这样：

```
<body>
<div id="scroller">
<a href="http://example.com" ><img src="image1.jpg" width=150px height=150px /></a>
<a href="http://example.com"><img src="image2.jpg"  width=150px height=150px /></a>
<a href="http://example.com"><img src="image3.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image4.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image5.jpg"  width=150px height=150px /></a>
</div>
</body>
```

所有 anchor 元素都被包含在 ID 为 scroller 的 div 元素中，这样我们可以将 ID 选择器 #scroller 中定义的样式属性应用到所有的 anchor 元素和图片上。ID 选择器是在外部样式表文件 style.css 中定义的。另外，所有图片的宽度和高度都被设置为 150px，这样可以统一显示的样式。样式表文件 style.css 可能像下面这样：

```
style.css
#scroller {
```

```
  position: relative;
  height: 760px;
  width: 150px;
}

#scroller a img { border:0; }
```

使得图片滚动的jQuery代码如下所示：

```
$(document).ready(function() {
  var $wrapper=$('#scroller');
  $wrapper.css({overflow: 'hidden'});
  var animator = function(imgblock) {
    imgblock.animate(
      {bottom:"-" + imgblock.height() + "px"}, 5000,
      function() {
        $(this).css({ bottom:imgblock.height()});
        animator($(this));
      }
    );
  }
  animator($wrapper);
});
```

知其所以然

我们会看到，在style.css文件中，ID选择器#scroller中将position属性设置为relative，这对于让图片滚动是必要的（这是通过相对于当前的位置来设置图片位置来实现滚动的）。height和width属性分别被设置为760px和150px。

height被设置为760px是为了显示所有五幅高度为150px的图片（上下依次排列），并且还考虑了它们之间的距离。

另外，样式表中还包含类型选择器#scroller a img，这会应用给嵌入在anchor元素中的img元素，它还依次包含在ID为scroller的HTML元素中。也就是说，样式属性的值为0，它会应用该img元素，使得图片的边框不可见。

在迄今为止的jQuery代码中，我们看到已经取得了所有位于ID为scroller的div元素中的内容，并存储在变量wrapper中。即所有的图片现在都位于wrapper变量中。使用.css()方法，我们将overflow属性设置为hidden，并应用给wrapper变量（即那组图片），从而使得滚动到浏览器窗口下面的图片不可见。此外，我们还定义了名为animator的函数，它会被语句animator($wrapper)所调用。

这条语句会调用animator函数，而包含图片组的变量wrapper会传递给它，作为参数imgblock的值。在animator函数本身中，代码会将图片组移到浏览器窗口的底部，并停在离浏览器窗口底部760px的位置。由于760px是图片组的总高度，所以整个图片组都会消失。图片组会缓慢地向下滚动，因为我们将动作的时间间隔设置为5000毫秒。在animate方法的回调函数中（它会在animate方法结束的时候触发），我们使用了.css()方法来使得图片组出现在离浏览器窗口底部760px的距离处，从而使整组图片都可见。当animator函数被递归调用以触发自

身，使得图片组再次向浏览器底部滚动的时候，这个过程就会循环执行。

在滚动的时候，部分图片组会是可见的，如图 6-12 所示。

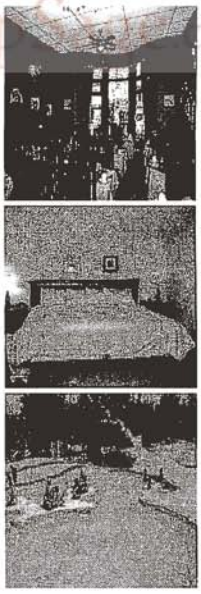


图 6-12 图片组垂直滚动

使图片在浏览器中间的小窗口中滚动

我们刚刚看到的解决方案能够让图片在整个浏览器窗口中滚动。我们还可以让图片显示在较小的不可见窗口中显示，从而减少所用到的空间。此外，我们还可以让它们显示在浏览器窗口的中间而不是左侧。

我们会使用在这个秘诀开始时创建的 HTML 文件。只是需要向 ID 选择器#scroller 和类型选择器#scroller a img 添加更多的属性。

如下所示：

```
style.css
#scroller {
  height: 460px;
  width: 150px;
  overflow:hidden;
  position: relative;
  margin:auto;
}

#scroller a img { border:0; position:relative;}
```

使得垂直滚动器在宽度和高度分别为 150px 和 460px 的窗口中滚动的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var $wrapper=$('#scroller a img');
  $wrapper.css({bottom:750});
  var animator = function(imgblock) {
    imgblock.animate(
      {bottom:-460}, 5000,
      function() {
        imgblock.css({bottom:750});
        animator($(this));
      }
    );
  }
  animator($wrapper);
});
```

知其所以然

ID 选择器#scroller 中将 height 属性设置为 460px，这足以按上下的位置依次显示 3 幅高度为 150px 的图片（我们想要每次最多看到 3 幅图片）。width 属性被设置为 150px（这也是图片的宽度）。overflow 属性被设置为 hidden，使得跨越这个窗口边界的那些图片不可见。position

属性的值被设置为 relative，使得图片可以滚动，而 margin 属性被设置为 auto，使得不可见的窗口（显示垂直滚动器）显示在浏览器窗口的中间。

定义在类型选择器#scroller a img 中的样式规则会自动应用给 img 元素（HTML 文件中的图片）。其中将 border 属性设置为 0（使得图片的边界不可见），并将 position 属性设置为 relative，使得图片可以滚动。

现在让我们看下 jQuery 代码本身，首先我们取得了 HTML 文件中所有 img 元素——它们包含在位于 ID 为 scroller 的 div 元素中的 anchor 元素中——并存储在变量 wrapper 中。然后存储在 wrapper 中的图片组被设定显示在距离不可见窗口的底部 750px 的位置（这使得图片组中的最后一幅图片的底边显示在不可见窗口之内）。

接下来，在 animator 方法中，图片组被设定向不可见窗口的底部滚动，并停在距离底部 460px 的位置，即图片组消失的位置。图片组会慢慢地滚动，因为动画的时间间隔被设置为 5000 毫秒。

在 animator 方法的回调函数中（它会在动画结束时执行），我们设定图片组显示在距离不可见窗口底部 750px 的位置，使得最后一幅图片的底边显示出来。然后 animator 方法会被递归调用，使得垂直滚动器一直滚动。

最终，animator 方法会被调用，所有存储在变量 wrapper 中的图片都会被传递给它。

使滚动器向上滚动

上面的滚动器是向下滚动的。我们可以修改 jQuery 代码使其向上滚动；
也就是向不可见窗口的顶部滚动。达到这个效果的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var $wrapper=$('#scroller a img');
  $wrapper.css({top:0});
  var animator = function(imgblock) {
    imgblock.animate(
      {top:-770}, 5000,
      function() {
        imgblock.css({top:450});
        animator($(this));
      }
    );
  }
  animator($wrapper);
});
```

我们先取得所有图片（嵌套在 anchor 元素中的 img 元素，并依次嵌套在 ID 为 scroller 的 div 元素中），并将其存储在变量$wrapper 中。我们会使用.css()方法设置图片位于离顶部边界距离为 0px 的位置，也就是初始的时候，3 幅高度为 150px 的图片都会上下排列依次显示。然后我们会调用 Animator 函数，所有存储在$wrapper 变量中的图片都会传递给它，作为 imgblock 参数的值。然后图片组会被设置为向不可见窗口的顶部滚动。滚动会停在 770px 的距离处，也就是当整个图片组消失的时候，滚动就会停止。滚动很慢，因为我们将延迟设置为 5000 毫秒。在滚动效果之后，图片组会被设置为显示在离顶部 450px 的位置处。也就是图片组中的第

一幅图片的上边缘会重新显示在不可见窗口的底部。animator 函数会被递归调用，从而使其持续滚动。

6.8 水平滚动图片

问题描述

现有几幅图片，让它们以滚动器的形式显示。即让图片在不可见窗口中从右向左依次滚动。当所有的图片都滚动到左边之后，让第一幅图片重新出现在窗口的右侧，并继续向左滚动，其他图片依次排在它的后面。

解决方案

首先让我们创建一个 HTML 文件，其中以超链接的形式定义了我们想要滚动的 5 幅图片。如果我们点击任一幅图片，它就会将我们导向显示图片完整信息的目标站点。在此我们假设目标站点是一些虚拟的站点。HTML 文件会像下面这样：

```
<div id="scroller">
<div id="images">
<a href="http://example.com" ><img src="image1.jpg" width=150px height=150px /></a>
<a href="http://example.com"><img src="image2.jpg"  width=150px height=150px /></a>
<a href="http://example.com"><img src="image3.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image4.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image5.jpg"  width=150px height=150px /></a>
</div>
</div>
</body>
```

我们可以看到包含 img 元素的 anchor 元素被包含在两个 div 元素之中，而其中的一个 div 元素有嵌入在另一个之中。外面的 div 元素的 ID 被设置为 scroller，而里面的 div 元素的 ID 被设置为 images。我们会为外面的 div 元素应用样式属性，以定义不可见窗口的宽度，它确定了我们每次想要看到的图片的数量。对于里面的 div 元素，我们会应用样式属性，以确定整个图片组的总宽度。所有的图片的宽度和高度都被统一设定为 150px，从而获得一致的显示效果。我们为 div 元素和 img 元素在样式表 style.css 中定义了样式属性，如下所示：

```
style.css
#scroller {
  position: relative;
  height:150px;
  width: 460px;
  overflow:hidden;
  margin:auto;
}

#images{
  width: 770px;
}
```

```
#images a img {border:0; position:relative;}
```

使得水平滚动器显示在浏览器窗口的中间的jQuery代码如下所示：

```
$(document).ready(function() {
  var $wrapper=$('#scroller a img');
  $wrapper.css({left:0});
  var animator = function(imgblock) {
    imgblock.animate(
      {left:-770}, 5000,
      function() {
        imgblock.css({left:450});
        animator($(this));
      }
    );
  }
  animator($wrapper);
});
```

知其所以然

我们会看到，在style.css文件中，ID选择器#scroller中将position属性设置为relative，这对于让图片滚动是必需的（当我们将图片的位置设为与当前位置相对的位置时，图片就会滚动）。height和width属性分别被设置为460px和150px。

将width设置为460px是一次最多显示3幅图片所需要（该宽度包括3幅宽度为150px的图片，以及它们之间的距离）。overflow属性被设置为hidden，使得图片组落在不可见窗口的宽度之外时，就会消失。margin属性被设置为auto，使得水平滚动器显示在浏览器窗口中间位置。

ID选择器#images中包含有width属性，它会被应用给里面的div元素。width属性被设置为770px，这是所有我们要在滚动器中显示的图片宽度的总和（包括图片之间的距离）。这里的width属性确定了我们要在水平滚动器中所要看到的图片的数量。另外，样式表中还包含类型选择器#images a img，这会应用给嵌入在anchor元素中的img元素，该元素依次包含在ID为images的HTML元素中。其中将border属性设置为0（使得图片的边界不可见），并将position属性设置为relative，使得图片可以滚动。

现在来看下jQuery代码本身，首先我们会取得HTML文件中所有img元素，它们包含在位于ID为scroller的div元素中的anchor元素中，并存储在变量wrapper中。然后，存储在变量wrapper中的图片组会被设定显示在离不可见窗口的左边界距离为0px的位置上（这使得图片组中的最前面3幅图片在初始时会显示在不可见窗口中）。

然后，在animator方法中，图片组被设定向不可见窗口的左侧滚动，并停在距离为-770px的位置，即图片组中的5幅图片全部都滚动到左侧。图片组会慢慢地滚动，因为动画的时间间隔被设置为5000毫秒。

在animator方法的回调函数中（它会在动画结束后执行），我们设定图片组显示在离不可见窗口的左边界450px的位置，使得（图片组中）第一幅图片的左边界显示在不可见窗口的右边界处。然后animator方法会被递归调用，使得水平滚动器持续滚动。当所有图片都滚动到左侧

时，第一幅图片会重新出现在不可见窗口的右边界处。

最终，我们会调用 animator 方法，所有存储在变量 wrapper 中的图片都会被传递给它。执行了上述的 jQuery 代码之后，页面上会显示 3 幅图片，如图 6-13 所示。

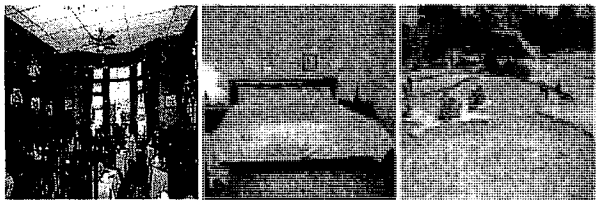


图 6-13 载入页面时图片的显示效果

这些图片开始向左滚动，使得其余的图片（隐藏在不可见窗口之下）开始显示，如图 6-14 所示。

图 6-14 图片向左滚动，隐藏的图片从右侧显示

在鼠标悬停的时候暂停滚动

假设我们希望图片滚动器在鼠标指针悬停在任一幅图片的时候暂停，当鼠标指针从图片上移出时再继续滚动。我们可以将 jQuery 代码修改如下：

```
$(document).ready(function() {
  var $wrapper=$('#scroller a img');
  $wrapper.css({left:0});
  var animator = function(imgblock) {
    imgblock.animate(
      {left:-770}, 5000,
      function() {
        imgblock.css({left:450});
        animator($(this));
      }
    );
  }
  animator($wrapper);

  $wrapper.hover(
    function(){
      $wrapper.stop();
    },
    function(){
```

```
        animator($wrapper);
      }
    );
});
```

在探究 jQuery 代码做了些什么之前，请注意我们在代码中使用的是 stop()方法，它使得当前运行在所有指定元素上的动画停止。

而在 jQuery 代码本身中，我们会取得所有图片（内嵌在 anchor 元素中的 img 元素，并依次内嵌在 ID 为 scroller 的 div 元素中），并存储在变量$wrapper 中。使用.css()方法，图片被设定在距离左边界 0px 的位置处；也就是初始时会有三幅宽度为 150px 的图片依次显示在不可见窗口中（不可见窗口的宽度在样式表中被设置为 360px）。

我们会调用 animator 函数，所有存储在$wrapper 变量中的图片都会传递给它，作为 imgblock 参数的值。然后图片组会被设置为向不可见窗口的左侧滚动。滚动会停在 770px 的距离处，也就是当整个图片组消失在窗口的左侧边界时，滚动就会停止。滚动很慢，因为我们将延迟设置为 5000 毫秒。在执行滚动效果之后，图片组会被设置为显示在离左侧边界 450px 的位置处。也就是图片组中的第一幅图片的左侧边缘会重新显示在不可见窗口的右侧。animator 函数会被递归调用，从而使图片持续滚动。

对于图片组，也就是对于变量$wrapper，我们为其添加了 hover 事件。之前我们提过，hover 事件中有两个事件处理函数：一个会在鼠标指针移到选定的元素上时被调用，另一个会在鼠标指针移出选定元素时执行。在鼠标指针移到图片组上触发的事件处理函数中，它调用了.stop()方法来将滚动停止；而在鼠标指针移出图片组时触发的事件处理函数中，animator 函数会被调用，从而使滚动继续。

向右滚动

上述的滚动器会向左滚动。为了让滚动器向右滚动，我们需要对 jQuery 代码做如下的修改：

```
$(document).ready(function() {
  var $wrapper=$('#scroller a img');
  $wrapper.css({right:0});
  var animator = function(imgblock) {
    imgblock.animate({
      right:-460}, 5000,
      function() {
        imgblock.css({ right:770});
        animator($(this));
      }
    );
  }
  animator($wrapper);
});
```

我们会取得所有图片（内嵌在 anchor 元素中的 img 元素，并依次内嵌在 ID 为 scroller 的 div 元素中），并存储在变量$wrapper 中。使用.css()方法，图片被设定在距离右边界 0px 的位置处；最前面的 3 幅宽度为 150px 的图片依次显示在不可见窗口中（不可见窗口的宽度是

360px)。

我们会调用 animator 函数，所有存储在$wrapper 变量中的图片都会传递给它，作为 imgblock 参数的值。然后图片组会被设置为向不可见窗口的右侧滚动。滚动会停在-460px 的距离处，也就是当整个图片组消失在窗口的右侧边界时，滚动就会停止。滚动会很慢，因为我们将延迟设置为 5000 毫秒。

在执行滚动效果之后，图片组会被设置为显示在离右侧边界 770px 的位置处。也就是图片组中的最后幅图片的右侧边缘会重新显示在不可见窗口的左侧。Animator 函数会被递归调用，从而使滚动继续。

6.9 创建新闻滚动浏览器

问题描述

有一些文本形式的新闻，要显示给访问站点的人看。让新闻在一个不可见窗口中向上滚动。当新闻全部滚动到上面之后，让它在窗口的底部重新出现，并继续向上滚动。我们会使用两个方法来编写针对这个问题的解决方案：首先我们会使用.animate()方法，然后是.css()方法。

解决方案

首先让我们创建一个 HTML 文件，其中以段落元素的形式显示新闻。段落元素嵌入在 ID 为 scroller 的 div 元素中。HTML 文件会像下面这样：

```
<body>
<div id="scroller">
<p>Styles make the formatting job much easier and more more efficient. To give an
attractive look to web sites, styles are heavily used. A person must have a good knowledge
of HTML and CSS and a bit of JavaScript.
jQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only is it easy to learn, but it's easy to implement too.
jQuery is an open source project that provides a wide range of features with
cross-platform compatibility. jQuery has hundreds of plug-ins to extend its features.
jQuery helps in increasing interactions with a web site.
</div>
</body>
```

之所以定义 div 元素，我们的想法是为了通过 ID 选择器#scroller 自动向其应用特定的样式属性，从而定义新闻在其中滚动的窗口的大小。ID 选择器#scroller 是在样式表文件 style.css 中定义的。另外，样式表中还包含了类型选择器#scroller p，其中包含将要应用给表示新闻文本的段落元素的文本的样式属性。样式表文件 style.css 可能像下面这样：

```
style.css
#scroller {
height: 250px;
width: 230px;
overflow:hidden;
position: relative;
```

```
margin:auto;
border:2px solid ;
padding:10px;
}

#scroller p {font-weight:bold;position:relative;}
```

使得新闻滚动的jQuery代码如下所示：

```
$(document).ready(function() {
  var $wrapper=$('#scroller p');
  $wrapper.css({top:0});
  var animator = function(imgblock) {
    imgblock.animate(
      {top:-350}, 5000,
      function() {
        imgblock.css({top:250});
        animator($(this));
      }
    };
  }
  animator($wrapper);
});
```

知其所以然

ID选择器#scroller中将属性height和width分别设置为250px和230px，从而定义了新闻窗口的高度和宽度（我们希望新闻文本在其中滚动）。通过将overflow属性的值设置为hidden，我们使得落在新闻窗口之外的文本变得不可见。position属性的值被设置为relative，这对于任何想要滚动的文本或者图片是必需的，当相对于元素当前的位置对其进行重新定位的时候，它就会滚动。margin属性被设置为auto，使得新闻滚动器显示在浏览器窗口中间位置。border属性会在新闻文本周围创建粗细为2px的实线边框，而padding属性被设置为10px，从而在文本和新闻窗口的边框之间留出一些空白。类型选择器#scroller p中将font-weight属性设置为bold，这使得段落文本加粗显示，position属性被设置为relative，使得段落文本可以滚动。

在jQuery代码中，你可以看到我们取得了所有包含在ID为scroller的div元素中的段落文本，并将其存储在变量$wrapper中。初始时文本被设定显示在距离窗口的边界0px的位置处，即文本的上半部分是可见的。而多出来的文本（超出新闻窗口的高度之外）是不可见的。我们定义了一个animator函数，所有的段落文本（存储在变量$wrapper中）都会传递给这个函数，并作为imgblock参数的值。Imgblock，也就是所有的段落文本，会被设定为向窗口的上边界滚动，并停止在距离新闻窗口的上边界-350px的位置处，也就是所有段落文本都滚动到上面（不可见）的时候。滚动会慢慢地进行，因为动画的时间间隔被设置为5000毫秒。在animate方法的回调函数中（会在动画结束的时候执行），我们使得段落文本在距离窗口的上边界250px的位置显示，使得只有段落文本的第一行显示在窗口的下边界。在此之后，animator函数会被递归调用，从而使段落文本持续滚动。

初始时，符合窗口的高度和宽度限制的段落文本是可见的，如图 6-15 所示。

然后新闻会开始向新闻窗口的上边界慢慢滚动，并在超过上边界的时候消失，如图 6-16 所示。

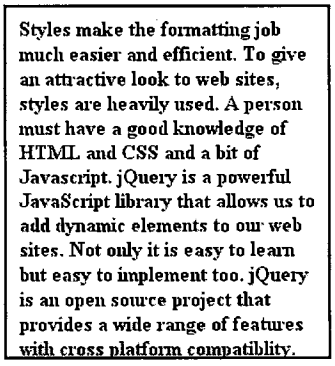


图 6-15 新闻文本向新闻窗口的上边界滚动

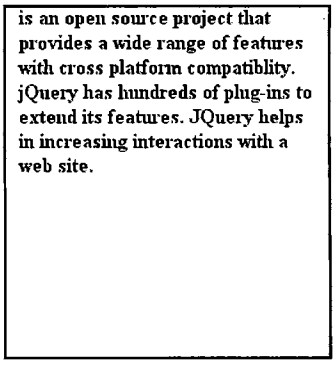


图 6-16 向上滚动的新闻文本消失

当所有的新闻文本都滚动到上边的时候，新闻的第一行会显示在窗口的底部边界，并继续向上滚动。

使用.css()方法创建的新闻滚动浏览器

在我们的解决方案的第二种方法中，我们会在循环中使用.css()方法来创建动画效果。我们会创建新的 HTML 文件来试验这个方法。因此，首先让我们创建一个 HTML 文件，其中以段落元素的形式显示新闻。段落元素被包含在 ID 为 news_scroller 的 div 元素中，从而定义我们希望新闻在其中滚动的窗口的宽度和高度。HTML 文件会像下面这样：

```
<body>
<div id="news_scroller">
<p>Styles make the formatting job much easier and more efficient. To give an attractive
look to web sites, styles are heavily used. A person must have a good knowledge of HTML
and CSS and a bit of JavaScript.
jQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only it is easy to learn, but it's easy to implement too.
jQuery is an open source project that provides a wide range of features with
cross-platform compatiblity. jQuery has hundreds of plug-ins to extend its features.
jQuery helps in increasing interactions with a web site.
</div>
</body>
```

我们会编写一个 ID 选择器#news_scroller 以及一个类型选择器 p，从而自动将样式属性应用给 div 元素和段落元素的文本。外部样式表文件 style.css 的内容如下所示：

```
style.css
#news_scroller{position:relative; height:200px; width:200px; overflow:hidden;
border:2px solid ; padding:10px;}
p{position:relative;}
```

使得文本滚动的 jQuery 代码如下所示：

```
$(document).ready(function() {
  ticker_height = $('#news_scroller').height();
  news_height=$('#news_scroller p').height();
  no_oflines = 0;
  scroll();
});

function scroll() {
  no_oflines -= 2;
  $('#news_scroller p').css( 'top', ''+no_oflines+"px" );
  if( no_oflines<-1*news_height ) no_oflines = ticker_height;
  setTimeout( scroll, 50 );
}
```

知其所以然

ID选择器#news_scroller中将position属性设置为relative，使得其中的文本滚动。height和width属性都被设置为200px，以定义新闻窗口的高度和宽度；overflow属性的值被设置为hidden，使得超出新闻窗口指定的宽度和高度范围之外的文本会消失。border属性被设置为显示粗细为2px的实线边框。padding属性被设置为10px，从而在段落文本和新闻窗口的边界之间保留空白。类型选择器p中将position属性设置为relative，这是让文本（位于新闻窗口中的）滚动的必要条件。

在jQuery代码中，首先我们计算出news_scroller（div元素）的总高度，以及文本（段落）元素的高度，并将它们分别存储在变量ticker_height和news_height中。在这个解决方案中，我们会使用.css()方法而不是animate()方法使文本向新闻窗口的顶部滚动。也就是说，我们在初始时会设置一行滚动，然后逐渐地增加滚动的行数；为了做到这一点，我们需要借助于变量no_oflines。我们会将其初始化为0，然后调用scroll函数。

在scroll函数中，我们将变量no_oflines的值减2（每次调用这个函数的时候）。假设no_oflines的值等于-2，我们就会使用.css()方法将段落文本（包含在ID为news_scroller的div元素中）设定在距离新闻窗口的上边界-2px的位置处；也就是说，段落文本会向上边界滚动2px，从而消失（因为overflow属性被设置为hidden）。在下次调用scroll函数的时候，no_oflines变量的值会变成-4（每次都减2），使得文本超出上边界4px（也就是说，又有2px的段落文本向上滚动而消失），这个过程会不断重复。当变量no_oflines的值与news_height的值相等的时候，也就是当所有的新闻文本都滚动上去的时候，我们会设置no_oflines变量的值等于ticker_height，也就是news_scroller窗口的高度，使得文本再次出现在新闻窗口的底部边界。

scroll函数被设置为每50毫秒就递归调用自身一次，这使得文本持续滚动。

当鼠标悬停的时候暂停新闻滚动浏览器

现在让我们添加更多的jQuery代码，让鼠标移到文本上的时候使新闻滚动器暂停，当鼠标从新闻滚动器移出的时候再继续滚动。我们需要增加一个布尔型变量rotate，它会决定什么时候

停止滚动以及什么时候开始继续滚动。初始的时候变量 rotate 的值会被设置为 true，如下 jQuery 代码所示：

```
$(document).ready(function() {
  ticker_height = $('#news_scroller').height();
  news_height=$('#news_scroller p').height();
  no_oflines = 0;
  rotate = true;

  $('#news_scroller').hover(
    function(){
      rotate = false;
    },
    function(){
      rotate = true;
    }
  );

  scroll();
});

function scroll() {
  no_oflines += rotate ? -2 : 0;
  $('#news_scroller p').css( 'top', ''+no_oflines+"px" );
  if( no_oflines<-1*news_height ) no_oflines = ticker_height;
  setTimeout( scroll, 50 );
}
```

在这段 jQuery 代码中，和之前所做的一样，首先我们会计算出 news_scroller（div 元素）的总高度，以及文本（段落）元素的高度，并将它们分别存储在变量 ticker_height 和 news_height 中。我们还将变量 no_oflines 的值初始化为 0（我们会使用这个变量来滚动文本）。我们会逐渐减少 no_oflines 变量的值，使得新闻文本滚动。我们还使用了布尔型变量 rotate，初始时它被设置为 true。当变量 rotate 被设置为 false 的时候，我们会让新闻文本停止滚动，当它的值被设置为 true 的时候，会继续滚动。此外，我们还对 ID 为 news_scroller 的 div 元素应用了 hover()事件，从而当鼠标悬停在新闻窗口上的时候，我们会将 rotate 变量的值设置为 false，从而让新闻滚动器暂停；当鼠标指针从新闻窗口上移出的时候，我们将布尔型变量 rotate 设置为 true，使得文本重新滚动。然后我们会调用 scroll 函数。

现在，在 scroll 函数中，如果变量 rotate 的值是 true，那么我们就会将 no_oflines 的值减 2；否则，我们不会改变它的值（在这种情况下，新闻文本会保持在原来的位置，而不会滚动）。假设 no_oflines 的值等于-2，我们会使用.css()方法将段落文本（包含在 ID 为 news_scroller 的 div 元素中）设定在距离新闻窗口的上边界-2px 的位置处；也就是说，段落文本会向上边界滚动 2px 并消失（因为 overflow 属性被设置为 hidden）。

在下次调用 scroll 函数的时候，no_oflines 变量的值会变成-4（每次都减 2），使得文本超出上边界 4px（也就是说，又有 2px 的段落文本向上滚动并消失），这个过程会不断重复。当变量 no_oflines 的值与 news_height 的值相等的时候，也就是当所有的新闻文本都滚动上去的时候，我们会设置 no_oflines 变量的值等于 ticker_height，也就是 news_scroller 窗口

的高度，使得文本再次出现在新闻窗口的底部边界。

scroll 函数被设置为每 50 毫秒就递归调用自身一次，这使得文本持续滚动。

6.10　在鼠标悬停时显示放大的图片

问题描述

在网页上显示几种要销售的商品，并有效地利用空间，因此要以图标的形式来显示这些商品的图片。当鼠标指针移动到这些图标上的时候，就会显示该商品放大的图片，从而让访问者看得更清楚。

解决方案

首先让我们创建一个 HTML 文件，来显示我们想要出售的产品的图标。和通常一样，这个图片会通过 img 元素显示，并包含在 anchor 元素中。anchor 元素指向图标放大的图片。也就是说，如果访问者点击任一图标，那么他就会被导向到显示图片的放大图片的页面上。此外，所有 anchor 元素都写在 class 为 small 的 div 元素中，从而我们可以通过 jQuery 代码或者样式表将样式应用给整个集合。HTML 文件会像下面这样：

```
<body>
  <div class="small">
            <a href="a1.jpg"><img src="a1.jpg"  /></a>
            <a href="a2.jpg"><img src="a2.jpg"  /></a>
            <a href="a3.jpg"><img src="a3.jpg"  /></a>
            <a href="a4.jpg"><img src="a4.jpg"  /></a>
 </div>
          <img src="a1.jpg" class="large" />
</body>
```

在上面的 HTML 文件中，我们还会看到 anchor 元素和 img 元素都指向相同的图片文件，也就是说，anchor 元素和 img 元素会显示相同的图片，其中我们希望 img 元素显示小图片（以图标的形式），而 anchor 元素显示放大的图片。虽然 anchor 元素和 img 元素中的图片大小实际上是相同的，但是通过像 img 元素（位于 class 为 small 的 div 元素中）应用样式属性，我们会缩小图片的宽度和高度，使其显示为图标。通过 class 为 large 的 img 元素显示的图片会以实际的尺寸显示，从而被用于显示放大的图片。默认情况下它会显示第一幅图片的放大效果。

应用给包含在 class 为 small 的 div 元素中的 img 元素的样式属性可以编写在外部样式表文件 style.css 中，如下所示：

```
style.css
.small img { border:none;margin:10px;width:60px; height:60px; }
```

当鼠标移到图标上的时候，显示放大视图的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $(".small a").hover(
    function(){
```

```
        var imgname = $(this).attr('href');
        $(".large").fadeTo(
          "slow", 0,
          function() {
            $('.large').attr('src',imgname);
          }
        ).fadeTo("slow", 1);
      }
    );
  });
```

知其所以然

你会看到，定义在类型选择器.small img 中的属性会自动应用给包含在 class 为 small 的 div 元素中的 img 元素。border 属性被设置为 none，这会使图片的边框不可见。margin 属性被设置为 10px，这会在图标之间留出 10px 的距离。width 和 height 属性被设置为 60px，从而减少了图片的宽度和高度，使其看起来像是图标。

现在来看下 jQuery 代码，我们首先向嵌入在 class 为 small 的 div 元素中的 anchor 元素添加 hover 事件。然后在 hover 事件中，我们取得了鼠标在上面悬停的 anchor 元素的 href 特性，并将其存储在变量 imgname 中。即图标（鼠标在上面悬停）的图片文件名（为 a1.jpg、a2.jpg 等等）分别被存储在变量 imgname 中。然后借助于.fadeTo()方法，当前被放大的图片（赋给 class 为 large 的 img 元素）会慢慢消失。

在.fadeTo()方法的回调函数（当动画效果完成的时候会被调用）中，我们将 src 特性的值（class 为 large 的 img 元素的源图片的值）设置为与存储在变量 imgname 中相等的值。然后存储在变量 imgname 中的图片文件名（鼠标悬停其上的）被赋予 class 为 large 的 img 元素，从而显示图标的放大视图。在第二个 fadeTo()调用中，图片的放大视图慢慢地显示在屏幕上。最终，hover 事件的第二个函数（当鼠标移出图标时会被调用）被保留为空，因为我们不想做任何动作，只是让图标的放大视图显示在屏幕上。

在执行 jQuery 代码的过程中，如果我们将鼠标移到第一个图标上，那么它的放大视图就会显示，如图 6-17 所示。

类似地，如果我们将鼠标移到其他图标上，那么它的放大视图就会显示，如图 6-18 所示。

图 6-17 图标列表，第一个图片被放大显示

图 6-18 当鼠标移到图标上时显示它的放大图片

当鼠标悬停的时候放大图标本身

如果我们只想放大图标本身会怎么样呢？让我们修改上面的解决方案，当鼠标指针移动到图标上的时候，我们会放大图标本身。当鼠标指针从图标上移开的时候，我们希望它还原到原来的大小。我们会对 HTML 文件作出修改，使其只显示图标和 div 部分，以显示放大的图片并移除原来的小图标，如下：

```
<body>
  <div class="small">
            <a href="a1.jpg"><img src="a1.jpg"  /></a>
            <a href="a2.jpg"><img src="a2.jpg"  /></a>
            <a href="a3.jpg"><img src="a3.jpg"  /></a>
            <a href="a4.jpg"><img src="a4.jpg"  /></a>
 </div>
</body>
```

你会看到我们使用 img 元素显示了 4 幅图标图片，它们都嵌入在 class 为 small 的 div 元素中。为了定义 img 元素的 width、height 和 border 属性，我们在样式表文件 style.css 中定义了一个类型选择器，如下所示：

```
style.css
.small img { border:none;margin:10px;width:60px; height:60px; }
```

当鼠标移到图标上的时候，放大图标的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $(".small a").hover(
    function(){
      $(this).find('img').css({'width':200,'height': 200});
    },
    function(){
      $(this).find('img').css({'width':60,'height': 60});
    }
  );
});
```

知其所以然——放大图标自身

我们会看到在上面的样式表文件中包含了类选择器 img，它的属性会自动应用给 class 为 small 的元素。类选择器中 border 属性被设置为 none，这会使图片的边框不可见。margin 属性被设置为 10px，这会在图标之间留出 10px 的距离。width 和 height 属性被设置为 60px，这会缩小图片的宽度和高度，使其看起来像是图标。

在 jQuery 代码中，我们向所有嵌入在 class 为 small 的 div 元素中的 anchor 元素添加了 hover 事件。在事件处理函数中（它会在鼠标指针移动到图片上的时候执行），我们使用了 .css() 方法来将图标的宽度和高度放大到 200px，而在另一个事件处理函数中（它会在鼠标移出图片区域的时候执行），我们会再使用 .css 方法来将图标的大小还原到原来的尺寸，也就是宽度和高度都是 60px。

初始时，图标的显示效果如图 6-19 所示。

图 6-19 当载入网页时所有图标的显示效果

当鼠标移到任一图标上时，它就会被突出显示，如图 6-20 所示。当鼠标指针从上面移走的时候，图标会恢复原来的大小。

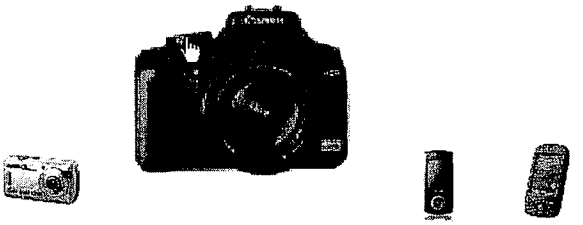


图 6-20 当鼠标悬停在图标上的时候，它会成为放大的图片

创建内容滑动浏览器

现在让我们进一步修改上面的解决方案来使用内容滑动浏览器。也就是说，它会以图标的形式按上下的顺序依次显示一些项目，当鼠标指针移动到任一图标上的时候，放大的视图会显示在右侧，如图 6-21 所示：

```
<body>
<table border="1">
<tr><td width=150 align=center> <a href="a1.jpg" class="small"><img src="a1.jpg"
/></a><p>
"Model: M1001. Price 100$"</td><td rowspan="4" width=350 align=center><img
src="a1.jpg" class="large" /></td></tr>
<td width=150 align=center><a href="a2.jpg" class="small"><img src="a2.jpg"/></a><p>
"Model: M1002. Price 150$"</p></td></tr>
 <td width=150 align=center><a href="a3.jpg" class="small"><img src="a3.jpg"/></a><p>
"Model: M1003. Price 90$"</td></tr>
<td width=150 align=center> <a href="a4.jpg" class="small"><img src="a4.jpg"/></a><p>
"Model: M1004. Price 200$"</td></tr>
</table>
</body>
```

在上面的 HTML 文件中，我们看到 anchor 元素和 img 元素（都指向相同的图片文件）都嵌入在<td>元素中，使得他们以表格元素的列的形式显示。此外，在列中，图标之下，我们包含了段落元素以显示它所代表的商品的编号和价格。表格的第二列的高度被设置为与其他 4 列高度的总和相等；也就是说，图片会显示在表格的第二列中，它的大小会是所有 4 行的总和。表格元素的边框被设置为 1，使得它的显示效果如图 6-21 所示。通过 class 为 large 的 img 元素显示的图片会显示鼠标在上面悬停的图片的放大图片。

我们在外部样式表文件 style.css 中编写了样式属性，它会应用给嵌入在 class 为 small 的 div

元素中的 img 元素。另外，样式表文件中还包含了我们想要应用给放大的图片的 CSS 类型，以及应用给当鼠标指针悬停时的图标的 CSS 类型。样式表文件可能会像下面这样：

```
style.css
.small img { border:none;margin:10px;width:60px; height:60px; }
.large{width:200px; height:200px;}
.hover{color: blue ; background-color:cyan }
```

当鼠标移到图标上的时候，应用 hover 样式并显示放大视图的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('td').find('p').css({'font-size':12, 'font-weight':'bold'});
  $(".small").hover(
    function(){
      $(this).parent().addClass('hover');
      var imgname = $(this).attr('href');
      $(".large").fadeTo(
        "slow", 0,
        function() {
          $('.large').attr('src',imgname);
        }
      ).fadeTo("slow", 1);
    },
    function(){
      $(this).parent().removeClass('hover');
    }
  );
});
```

知其所以然——内容滑动浏览器

类型选择器.small img（它将会自动应用给嵌入在 class 为 small 的 div 元素中的 img 元素）中将 border 属性设置为 none，以删除图片的边框。margin 属性被设置为 10px，从而在图标之间留出 10px 的距离，而 width 和 height 属性分别被设置为 60px，从而缩小图片的宽度和高度，使其显示为图标。类选择器.large 中将 width 和 height 属性设置为 200px，从而通过 class 为 large 的 img 元素显示图片的放大视图。CSS 类 hover 中将 background-color 和 color 属性分别设置为 cyan 和 blue，从而向鼠标悬停在上面的元素应用背景色和前景色。

现在让我们来看下 jQuery 代码。首先我们会找到位于 td 元素中的段落元素，并借助于.css()方法将它的 font-size 属性设置为 12px，font-weight 属性设置为 bold。然后我们会向所有 class 为 small 的元素添加 hover 事件，也就是向所有 anchor 元素（并以此对内嵌的 img 元素）添加事件。

在 hover 事件处理器中，将会对鼠标悬停的元素应用 CSS 类 hover；也就是会改变图片元素和段落元素的背景色和前景色。然后 anchor 元素的 href 特性的值会被取得，并存储在变量 imgname 中。即图标（鼠标在上面悬停的）的图片文件名（为 a1.jpg、a2.jpg 等等）分别被存储在变量 imgname 中。这些完成了之后，借助于.fadeTo()方法，当前被放大的图片（赋给 class 为 large 的 img 元素）会慢慢消失。

在.fadeTo()方法的回调函数中(将会在动画效果结束的时候调用),我们将 src 特性(class 为 large 的 img 元素的图片源文件的值)的值设置为与变量 imgname 中存储的相同。也就是说,存储在变量 imgname 中的图片文件名(鼠标悬停其上的)被赋予 class 为 large 的 img 元素,从而显示图标的放大视图。第二个 fadeTo()方法会使得图片的放大视图慢慢地显示在屏幕上。

hover 事件中第二个函数,会在鼠标指针从图标上移出的时候被触发,它被用于从图片和段落元素上移除 hover 样式,使得它们和鼠标移到上面之前一样显示。在执行 jQuery 代码的过程中,如果我们将鼠标移到第一个图标上,那么它的背景和前景色会发生改变,并且会显示放大视图,如图 6-21 所示。

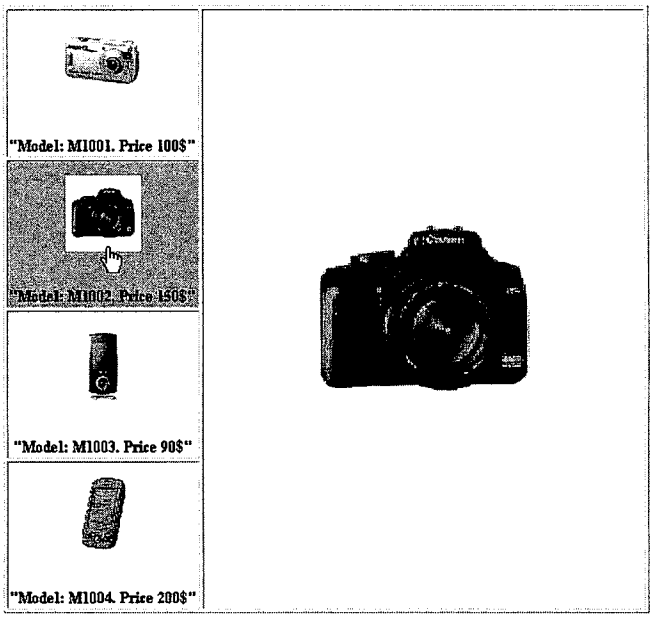


图 6-21 鼠标悬停在上面的商品的放大图片

6.11 按页显示图片

问题描述

现有网页上的几幅图片,你想要对其分页显示,其中每页包含一幅或多幅图片(这与网页上的空间有关)。在图片的顶部,你希望显示代表页码的数字,并希望在屏幕上显示选定页码的图片。

解决方案

让我们创建 HTML 文件,来定义想要显示的 5 幅图片。另外,因为我们希望图片显示为超链接,它会将访问者导航到目标网页(其中会显示选中图片更多的细节信息),我们需要将图片元

素嵌入在 anchor 元素中。HTML 文件会像下面这样：

```
<body>
<div id="images">
<a href="http://example.com" ><img src="image1.jpg" width=150px height=150px /></a>
<a href="http://example.com"><img src="image2.jpg"  width=150px height=150px /></a>
<a href="http://example.com"><img src="image3.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image4.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image5.jpg"  width=150px height=150px /></a>
</div>
</body>
```

在 HTML 页面中我们可以看到，所有的 img 元素都被赋予相同的宽度和高度 150px，从而得到统一的显示效果，而 anchor 元素的目标是一些虚拟的站点，当访问者选择任一图片的时候会被导向那里。

现在我们来定义一下外部样式表文件 sty/e.css 中的 CSS 类，代码如下：

```
style.css
.page{
margin:5px;
}
.hover{
color: blue ;
background-color:cyan
}
```

将图片分页并且显示选定页中的图片的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var $pic = $('#images a');
  $pic.hide();
  var imgs = $pic.length;
  var next=$pic.eq(0);
  next.css({'position': 'absolute','left':10});
  next.show();
  var $pagenumbers=$('<div id="pages"></div>');
  for(i=0;i<imgs;i++)
  {
    $('<span class="page">'+(i+1)+'</span>').appendTo($pagenumbers);
  }
  $pagenumbers.insertBefore(next);

  $('.page').hover(
    function(){
      $(this).addClass('hover');
    },
    function(){
      $(this).removeClass('hover');
    }
  );

  $('span').click(function(event){
    $pic.hide();
    next=$pic.eq($(this).text()-1);
```

```
        next.show();
      });
    });
```

知其所以然

我们会发现，CSS 类 page 中将样式属性 margin 设置为 5px，从而定义了页码之间的空间；而类 hover 中包含两个属性 color 和 background-color，它们分别被设置为 blue 和 cyan，从而当鼠标指针移到上面的时候会改变页码的背景色和前景色。

现在让我们来看下 jQuery 代码本身。首先，我们会取得所有内嵌在 ID 为 images 的 div 元素中的 anchor 元素（也就是所有包含在 anchor 元素中的图片），并将其存储在变量$pic 中。对象$pic 会包含所有图片。然后我们会隐藏所有图片，并设置一些变量；图片的数量被存储在变量 imgs 中，而$pic 对象中的第一幅图片被存储在变量 next 中。

对于存储在变量 next 中的第一幅图片，我们会使用.css()方法向它应用一些样式属性。position 属性被设置为 absolute，left 属性被设置为 10px，使得存储在变量 next 中的图片显示在距离浏览器窗口左侧边界 10px 的位置上。这些完成了之后，next 变量中的图片会显示在屏幕上，然后我们会定义变量$pagenumbers，并将其赋值为 ID 为 pages 的 div 元素。

然后我们会使用 for 循环来创建一些 class 为 pages 的 span 元素（数量与图片的数量相等）。span 元素中的文本会是 1，2…… （作为页码）。span 元素的 class 被设置为 pages，从而定义在类选择器.pages（位于 style.css 文件中）中的属性会自动应用给它们。span 元素会被附加到 ID 为 pages 的 div 元素上（我们已经把它赋给变量$pagenumbers）。

整个包含了 span 元素的 ID 为 pages 的 div 元素被插入在使用变量 next 显示的第一幅图片之前，并且当鼠标指针悬停在页码上的时候，我们会向页码（class 为 page 的 span 元素）应用 CSS 类 hover。接下来我们需要向 span 元素（也就是页码）添加 click 事件。

在此我们隐藏了所有图片，包括如果用户选择任一页码所会显示的当前的那些图片。最终，我们从$pic 对象（包含图片的数组）中取得图片，这是根据选择的页码的值而完成的；我们会将其存储在变量 next 中，并显示从变量 next 中取得的图片。

在 jQuery 代码的执行过程中，初始时会显示第一幅图片，顶部是页码，如图 6-22 所示。

当选择页码的时候，该页中的图片就会显示，如图 6-23 所示。

图 6-22 第一幅图片，上面显示有页码

图 6-23 当点击页码 4 的时候所显示的图片

6.12　在任意两个方向上切换图片

问题描述

在网页上的不可见窗口中显示一些图片，在下面带有左右箭头。箭头可用，按下左箭头的时候，图片会向左侧切换（使得隐藏的图片向左滚动），选择右箭头的时候，所有图片会向右滚动，以显示隐藏的图片。

解决方案

让我们创建HTML文件，来定义想要显示的图片。图片都嵌入在ID为images的div元素中（以通过jQuery代码来应用样式），它依次嵌入在ID为scroller的div元素中。要创建的HTML文件如下所示：

```
<body>
<div id="scroller">
<div id="images">
<a href="http://example.com" ><img src="image1.jpg" width=150px height=150px /></a>
<a href="http://example.com"><img src="image2.jpg"  width=150px height=150px /></a>
<a href="http://example.com"><img src="image3.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image4.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image5.jpg"  width=150px height=150px /></a>
</div>
</div>
<div id="direction">
<img src="leftarrow.jpg" class="leftarrow"/>
<img src="rightarrow.jpg" class="rightarrow"/>
</div>
</body>
```

图片被包含在anchor元素中，从而可以将访问者导航到目标网页上，以显示选定图片的详细信息。此时，我们假设目标站点是一些虚拟的站点。包含img元素的anchor元素被包含在两个div元素之中，而其中的一个div元素嵌入在另一个之中。外面的div元素的ID被设置为scroller，而里面的div元素的ID被设置为images。我们会为外面的div元素应用样式属性，以定义不可见窗口的宽度，它确定了我们每次想要看到的图片的数量。对于里面的div元素，我们会应用样式属性，以确定整个图片组的总宽度。所有的图片的宽度和高度都被统一设定为150px，从而获得一致的显示效果。

我们还使用img元素来显示左右箭头按钮。这两个img元素的class分别被赋予leftarrow和rightarrow，从而定义在类选择器里面的样式属性能够分别应用其上。这两个箭头被嵌入在ID为direction的div元素中。我们为div元素和img元素在样式表style.css中定义了样式属性，如下所示：

```
style.css
#scroller {
  position: relative;
  height:150px;
```

```
  width: 460px;
overflow:hidden;
margin:auto;
}

#images{
  width: 770px;
}

#images a img { border:0; position:relative;}
#direction
{
width: 460px;
margin:auto;
}

.leftarrow{margin-top:10px;}
.rightarrow{margin-left:390px;margin-top:10px;}
```

使得图片在选择左右箭头图片时滚动的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var $wrapper=$('#scroller a img');

  var leftanimator = function(imgblock) {
    imgblock.animate({left:-310 }, 2000);
  }

  var rightanimator = function(imgblock) {
    imgblock.animate({left:0 }, 2000);
  }

  $('.leftarrow').click(function(event){
    leftanimator($wrapper);
    event.preventDefault();
  });

  $('.rightarrow').click(function(event){
    rightanimator($wrapper);
    event.preventDefault();
  });
});
```

知其所以然

你会看到，在 style.css 文件中，ID 选择器#scroller 中将 position 属性设置为 relative，这对于让图片滚动是必要的（当你将图片的位置设为与当前位置相对的位置时，图片就会滚动）。height 和 width 属性分别被设置为 460px 和 150px。

将 width 设置为 460px，这是一次最多显示 3 幅图片所需要的宽度（该宽度包括 3 幅宽度为 150px 的图片，以及它们之间的距离）。overflow 属性被设置为 hidden，使得图片组落在不可见窗口的宽度之外时，就会消失。margin 属性被设置为 auto，使得水平滚动器显示在浏览器窗

口中间位置。

ID选择器#images中包含有width属性，它会被应用给里面的div元素。

width属性被设置为770px，这是我们要在滚动器中显示的所有图片宽度的总和（包括图片之间的距离）。这里的width属性确定了我们要在水平滚动器中所要看到的图片的数量。另外，样式表中还包含类型选择器#images a img，这会应用给嵌入在anchor元素中的img元素，它依次包含在ID为images的HTML元素中。正如你所看到的，其中将border属性设置为0（使得图片的边界不可见），并将position属性设置为relative，使得图片可以滚动。

ID选择器#direction中包含了将要应用给左右箭头图片的样式属性。其中width属性被设置为460px，这是区域的最大宽度（包括两个箭头图片），而margin属性的值被设置为auto，使得区域（有两幅图片的）显示在浏览器窗口的中间，就在图片区域的下面。

类选择器.leftarrow中的属性会自动应用给左箭头图片。其中将margin-top属性设置为10px，使得左箭头与图片区域的顶部保持10px的距离。同时，类选择器.rightarrow中的属性会自动应用给右箭头图片。其中将margin-left属性设置为390px，使得右箭头图片在父元素的460px的宽度中以右对齐的方式排列；margin-top属性被设置为10px，从而与图片区域的顶部保持10px的距离。

现在让我们来查看解决方案中的jQuery代码，你会发现所有的图片都嵌入在ID为scroller的div元素中的anchor元素中，它们会被取得并存储在变量$wrapper中。

接下来我们定义了一个函数leftanimator，它有一个参数imgblock，其功能是让图片组向左慢慢移动，并停在距离左侧-310px的位置（进入到左边界310px）。这样，图片组左侧的两幅图片就会消失在不可见窗口的左边界之中，使得右侧的两幅图片（之前是隐藏的）显示出来。

rightanimator方法会接受参数imgblock，它会使图片组慢慢向右移动，并停在距离左边界0px的位置。也就是说，它会使得图片组向右边界滚动，并在图片组的第一幅图片显示的时候停止——当前面的3幅图片显示在不可见窗口中的时候，滚动就会停止。我会让右侧的两幅图片显示出来。

接下来我们会为左箭头图片添加click事件，当该事件被触发的时候，就会调用leftanimator函数，而包含所有5幅图片的变量$wrapper被发送给它，作为参数imgblock的值。该函数会使得图片组向左滚动，显示最后的3幅图片。我们还会调用event对象的preventDefault方法，以防止在点击时将我们导向图片（嵌入在anchor元素中）所指向的目标站点。

最终，我们会为右箭头图片添加click事件，当该事件被触发的时候，就会调用rightanimator函数，而包含所有5幅图片的变量$wrapper被发送给它，作为参数imgblock的值。函数会让图片组向右滚动，使最前面的3幅图片显示在屏幕上。我们还会调用event对象的preventDefault方法，以防止点击时将我们导向图片（嵌入在anchor元素中）所指向的目标站点。

那么这些看起来会是什么样子呢？初始的时候，前面的3幅图片会显示，下面带有左右箭头，如图6-24所示。

图 6-24 初始时显示 3 幅图片，底部带有左右箭头

当选择左箭头的时候，图片会向左滚动，我们能够看到最后的 3 幅图片（包含 5 幅图片的图片组中），如图 6-25 所示。

图 6-25 当选择左箭头的时候图片向左滚动

类似地，如果我们选择右箭头按钮，图片会向右滚动，使得前面的 3 幅图片重新显示出来。

6.13 编写钟摆式滚动器

问题描述

现有一组 5 幅图片，要达到这样的效果，初始的时候在不可见窗口中显示 5 幅图片中的 3 幅。然后图片会向左滚动，并消失在窗口之外，就像我们摇晃钟摆一样。当最后一幅图片消失的时候，让图片从左边界出现，并向右边界滚动（使得最后一幅图片首先显示，然后是第 4 幅，以此类推）。所有图片都会向右边界滚动，并从窗口中消失。当第一幅图片消失的时候，让图片再次向左边界滚动，这个过程会继续。

解决方案

让我们创建 HTML 文件，来定义想要显示的图片。图片都嵌入在 ID 为 `images` 的 `div` 元素中（以便通过 jQuery 代码来应用样式），它依次嵌入在 ID 为 `scroller` 的 `div` 元素中。HTML 文件会像下面这样：

```
<body>
<div id="scroller">
<div id="images">
```

```
<a href="http://example.com" ><img src="image1.jpg" width=150px height=150px /></a>
<a href="http://example.com"><img src="image2.jpg"  width=150px height=150px /></a>
<a href="http://example.com"><img src="image3.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image4.jpg"  width=150px height=150px /></a>
<a href="http://example.com" ><img src="image5.jpg"  width=150px height=150px /></a>
</div>
</div>
</body>
```

图片被包含在 anchor 元素中，从而可以将访问者导航到目标网页，以显示选定图片的详细信息。此时，我们假设目标站点是一些虚拟的站点。

包含 img 元素的 anchor 元素被包含在两个 div 元素之中，而其中的一个 div 元素嵌入在另一个之中。外面的 div 元素的 ID 被设置为 scroller，而里面的 div 元素的 ID 被设置为 images。我们会为外面的 div 元素应用样式属性，以定义不可见窗口的宽度，它确定了我们每次想要看到的图片的数量。对于里面的 div 元素，我们会应用样式属性，以确定整个图片组的总宽度。所有图片的宽度和高度都被统一设定为 150px，从而获得一致的显示效果。

我们为 div 元素和 img 元素在样式表 style.css 中定义了样式属性，如下所示：

```
style.css
#scroller {
  position: relative;
  height:150px;
  width: 460px;
overflow:hidden;
margin:auto;
}

#images{
  width: 770px;
}

#images a img { border:0; position:relative;}
```

使得图片像钟摆一样滚动的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var $wrapper=$('#scroller a img');

  var left_rightanimator = function() {
    $wrapper.animate(
      {left:-770}, 5000,
      function() {
        $wrapper.animate({left:465 }, 5000);
        left_rightanimator();
      }
    );
  }

  left_rightanimator();
});
```

知其所以然

你会看到，在 style.css 文件中，我们将 ID 选择器#scroller 的 position 属性设置为 relative，想让图片滚动这是必需的（当你将图片的位置设为与当前位置相对时，图片才会滚动）。height 和 width 属性分别被设置为 460px 和 150px。

将 width 设置为 460px，这是一次最多显示 3 幅图片所需要的宽度（该宽度包括 3 幅宽度为 150px 的图片，以及它们之间的距离）。overflow 属性被设置为 hidden，使得图片组落在不可见窗口的宽度之外时就会消失。margin 属性被设置为 auto，使得水平滚动器显示在浏览器窗口中间位置。

ID 选择器#images 中包含有 width 属性，它会被应用给里面的 div 元素。width 属性被设置为 770px，这是我们要在滚动器中显示的所有图片宽度的总和（包括图片之间的距离）。这里的 width 属性确定了我们要在水平滚动器中看到的图片的数量。

另外，样式表中还包含类型选择器#images a img，这会应用给嵌入在 anchor 元素中的 img 元素，它依次包含在 ID 为 images 的 HTML 元素中。

其中将 border 属性设置为 0（使得图片的边界不可见），并将 position 属性设置为 relative，使得图片可以滚动。

现在查看一下 jQuery 代码，我们会取得所有内嵌在 anchor 元素和 ID 为 scroller 的 div 元素中的图片，并将其存储在变量$wrapper 中。即变量$wrapper 中会包含整组的 5 幅图片。

接下来我们定义了函数 left_rightanimator，它会让图片组向（不可见窗口的）左边界移动，停在距离左边界-770px 的距离处，也就是左边界之外 770px 处，这会使得整组的 5 幅图片消失（回想一下，每幅图片的宽度都是 150px，并且在彼此之间有些距离）。然后，图片组再次被设置从左侧边界显示，并向右移动（使得最后一幅图片首先显示），当第一幅图片也（从右侧边界）消失在窗口之外距离左边界 465px 的位置的时候，滚动就会停止，这也是第一幅图片消失在不可见窗口的右侧边界的位置。

这个方法中最后的动作是递归调用 left_rightanimator()函数，从而使图片组一直向左或向右滚动。最终，我们会调用 left_rightanimator()函数来让过程持续。

图片会向左滚动、消失，从左侧边界重新显示，然后再向右边界滚动。最后两幅图片在向左滚动时的效果如图 6-26 所示。

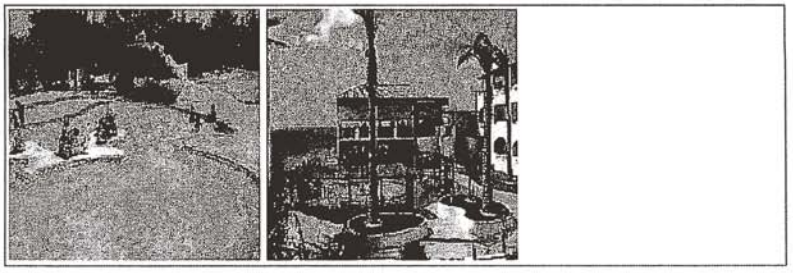


图 6-26 图片向左滚动并消失

6.14　使用数组来滚动图片

问题描述

现有一组5幅图片，要达到这样的效果，初始的时候在不可见窗口中显示5幅图片中的3幅。然后这些图片会向左侧边界滚动（使得最后两幅隐藏的图片显示在不可见窗口中）。所有图片都从左侧边界消失。

借助数组来做到这一点。

解决方案

我们会使用与攻略6.13中相同的HTML和样式表文件（style.css）。

使得每幅图片向左边界滚动并依此消失的jQuery代码如下：

```
$(document).ready(function() {
  var $pic = $('#scroller a img');
  var imgs = $pic.length;
  var next;
  for (var i=0;i<imgs;i++){
    next=$pic.eq(i);
    scroll(next);
  };
});

function scroll(im)
{
  im.animate({'left': -770}, 5000);
};
```

在此，我们首先会取得所有嵌入在anchor元素中的图片，它们依次嵌入在ID为scroller的div元素中，并将其存储在变量$pic中。现在$pic是存有5幅图片的数组。然后我们会找出数组$pic中图片的数量，并将其存储在变量imgs中。

在for循环中，我们会从$pic数组中取得一幅图片，并将其存储在变量next中。也就是，所有的图片会依次赋给变量next。为了让图片滚动，我们会调用函数scroll，并将存储在变量next中的图片传递给它（变量next被赋予参数im）。

在scroll()方法中，图片被设置为向左移动，并停在距离左侧边界-770px的位置处——也就是左侧边界之外770px——使得最后一幅（第5幅）图片也消失。

滚动时的图片如图6-27所示。

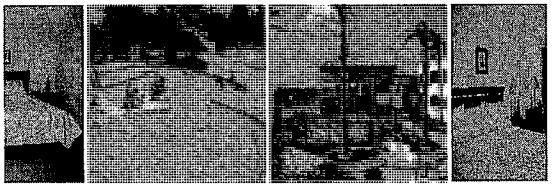


图6-27　所有图片向左滚动

使一幅图片在另一幅之上滚动

假设你想要修改上面的解决方案，因为你希望3幅图片保持固定，而一幅图片（第4幅）在这3幅图片之上滚动。针对这个解决方案的jQuery代码如下：

```
$(document).ready(function() {
  var $pic = $('#scroller a img');
  var next;
  next=$pic.eq(3);
  scroll(next);
});

function scroll(im)
{
  im.animate({'left': -770}, 5000);
};
```

我们会看到，变量next被赋值为第4幅图片（索引值是3），它位于$pic数组的第4个位置，并且被传递给scroll函数以在图片之上滚动。初始的时候，我们在不可见窗口中显示有3幅图片，如图6-28所示。

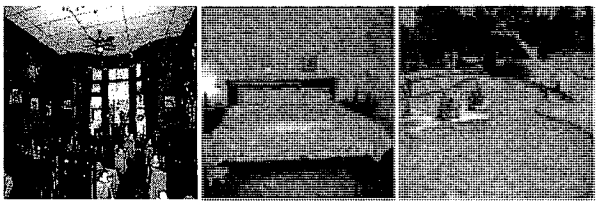


图6-28 载入网页时的初始效果

第4幅图片会从右侧边界开始向（不可见窗口的）左侧边界在3幅图片上滚动，如图6-29所示，其中第4幅图片位于第2和第3幅图片之上。

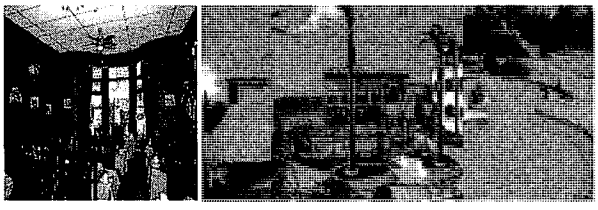


图6-29 一幅图片在其他图片之上向左滚动

只让鼠标悬停其上的图片滚动

你有3幅图片，初始时显示在不可见窗口中，你想实现这样的功能，任一幅鼠标在上面悬停的图片会向左侧边界滚动并消失。jQuery代码可能像下面这样：

```
$(document).ready(function() {
```

```
  var $pic = $('#scroller a img');
  $pic.hover(
    function(){
      $(this).animate({'left': -770}, 5000);
    },
    function(){
    }
  );
});
```

我们会看到，第3条语句向图片组（存储在$pic数组中）添加hover事件，并且在它的事件处理函数中，我们让鼠标悬停其上的图片向左侧边界移动，并停在距离左侧边界-770px的位置处，也就是在左侧边界之外770px（使得它完全消失）。如果我们将鼠标悬停在中间的图片上，它会开始向左滚动，如图6-30所示。

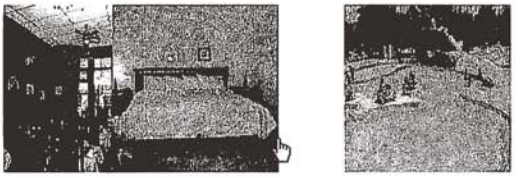


图6-30　鼠标悬停其上的图片开始向左滚动

当中间的图片完全滚动到左边界之外时，我们会发现在中间位置为其保留了空间，如图6-31所示。

图6-31　中间的图片向左滚动，留下空白位置

淡出并替换图片

你有3幅图片，初始时显示在不可见窗口中，你希望鼠标悬停其上的图片使用淡出的效果渐渐消失，而它的空间会被下一幅图片填充（使得隐藏的图片显示）。jQuery代码可能像下面这样：

```
$(document).ready(function() {
  var $pic = $('#scroller a img');
  $pic.hover(
    function(){
      $(this).fadeOut(5000);
    },
    function(){
    }
  );
});
```

我们为$pic 变量添加了 hover 事件，而$pic 只是包含 5 幅图片的数组。在 hover 事件的处理函数中，我们让图片在鼠标悬停的时候在 500 毫秒之内淡出。图片完全消失的时候，$pic 数组中下一幅图片会填充空出来的位置。

初始时，我们在窗口中显示有 3 幅图片，如图 6-28 所示，当鼠标指针在第一幅图片上悬停的时候，它会慢慢地淡出，它的位置会被序列中的下一幅图片所占据。

让一幅图片向左滚动，一幅向右滚动，淡出中间的图片

在这组攻略的最后，我们来对当前的攻略来做个富有想象力的修改——让你可以更好地了解当前关于滚动和淡出的核心解决方案，并激发出更好的想法。

现在让我们假设你有 3 幅图片，初始时显示在不可见窗口中，你希望第一幅图片向左滚动消失，第 3 幅图片向不可见窗口的右侧滚动并消失，而中间的图片保持原位置不动，慢慢淡出。这次，jQuery 代码可能像下面这样：

```
$(document).ready(function() {
  var $pic = $('#scroller a img');
  $pic.eq(0).animate(
    {'left': -155}, 5000,
    function(){
      $pic.eq(2).animate(
        {'right': -155}, 5000,
        function(){
          $pic.fadeOut(5000);
        }
      );
    }
  );
});
```

我们会取得所有内嵌在 ID 为 scroller 的 anchor 元素中的图片元素，并将其存储在变量 $pic 中。第一幅图片（数组$pic 中索引为 0 的图片）会被设置向左边界移动，并停在距离左边界-155px 的位置处，也就是左边界之外 155px 处（这对于让宽度为 150px 的图片消失已经足够了）。类似地，第 3 幅图片被设置向右边界移动，并停在距离右侧边界-155px 的位置处，也就是在右侧边界之外 155px 处（使得图片消失）。中间的图片留在原处，并慢慢地在 5000 毫秒之内淡出。如果你已经看过前面的秘诀，那么现在的代码对于你来说已经很熟悉了。但是你的最终目的是要吸引用户的注意，并让他们继续访问你的站点。

6.15 小结

本章中我们讲述了关于向站点应用视觉特效的攻略，包括创建图片滑动浏览器、水平和垂直的图片滚动器和新闻滚动器等等。我们还探究了分页显示图片以及在图标上悬停时放大图片的技术。

下一章中，我们会一起学习处理表格的攻略，像突出显示表格的行列、过滤出需要的行、删除选定列的内容以及展开和折叠表格的行等等。你还会看到表格如何根据选定列进行排序，以及如何对表格的行进行分页。

第 7 章

处理表格

在本章中，我们会探究如下攻略，在处理表格的时候，它们会非常有用：

- 在鼠标悬停时突出显示一行；
- 交替突出显示相邻的列；
- 过滤行；
- 隐藏选定行；
- 分页显示表格；
- 展开和折叠列表项；
- 展开和折叠行；
- 对列表项目排序；
- 对表格排序；
- 过滤表格中的行。

7.1 在鼠标悬停时突出显示行

问题描述

现有一表格，其中包含一些行和列。当鼠标指针移动到其中一行上时，使其能够突出显示。

解决方案

让我们先创建一个 HTML 文件；其中定义了包含带有一些行列元素（th、td、tr）的表格元素。HTML 文件会像下面这样：

```
<body>
<table border="1">
<thead>
<tr><th>Roll</th><th>Name</th><th>Marks</th></tr>
</thead>
<tbody>
<tr><td>101</td><td>John</td><td>87</td></tr>
<tr><td>102</td><td>Naman</td><td>90</td></tr>
<tr><td>103</td><td>Chirag</td><td>85</td></tr>
</tbody>
```

```
</table>
</body>
```

让我们在样式表文件 style.css 中定义样式规则 .hover，从而为鼠标悬停其上的行应用样式属性。样式规则可能会像下面这样：

```
style.css
.hover { background-color: #00f; color: #fff; }
```

将悬停效果应用于表格行的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('tbody tr').hover(
    function(){
      $(this).find('td').addClass('hover');
    },
    function(){
      $(this).find('td').removeClass('hover');
    }
  );
});
```

知其所以然

在上面的 HTML 文件中我们可以看到，在表格的定义中，边框被设置为 1px，并拥有 3 个列标题（使用 th 元素定义）：Roll，Name 和 Marks。另外它还包含 3 行学生的记录。表格标题嵌入在 thead 元素中，而表格的主体（包含信息的行）嵌入在 tbody 元素中。

在样式表文件中，.hover 样式规则将 background-color 属性设置为#00f，将 color 属性设置为#fff，从而分别将鼠标悬停其上的行的背景变为蓝色，前景变为白色。

在 jQuery 代码中，我们为嵌入在 tbody 元素中的 tr（行元素）添加了 hover()事件，因为我们只希望包含学生信息的行有鼠标悬停的效果，而不希望包含列标题的行有那样的效果。在 hover 事件的处理函数中，我们会搜索鼠标悬停行的 td 元素，并对其应用样式规则 .hover（位于样式表文件 style.css 中），分别将它的背景色变为蓝色，前景色变为白色，从而突出显示它。

初始时，表格的显示效果如图 7-1 所示。

当鼠标移到任一行上时，该行会被突出显示，如图 7-2 所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-1 包含一些行和列的表格

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-2 当鼠标悬停时突出显示的行

7.2 交替突出显示相邻列

问题描述

现有一表格，其中包含一些行和列。你希望相邻的列可以突出显示。

解决方案

对于这个解决方案，我们会使用与攻略 7.1 中同样的 HTML 和样式表文件。

应用样式规则.hover 定义的样式属性来改变列的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('td:nth-child(odd)').addClass('hover');
});
```

知其所以然

既然我们在 jQuery 代码中使用了:nth-child()方法，那么就让我们先来快速浏览一下它能做到什么，以及你应如何使用它。

- **:nth-child()**

这个方法是用来取得父元素的第 *n* 个子元素的。它从 1 而不是 0 开始计数。该方法与:eq()方法的不同体现在以下两个方面。

- :eq()方法只匹配一个元素，而:nth-child()会为每个父元素的指定索引匹配一个元素。
- :eq()方法是从 0 开始的，即它从 0 开始计数。而:nth-child()是从 1 开始的（它从 1 开始计数）。

例如：

```
$('tr:nth-child(3)');
```

这会选择父元素（可能是 tbody 或者 table 元素）的第 3 个子元素中的所有 tr 元素，即它会选择表格的第 3 行。

类似地，下面这个语句：

```
$('tr:nth-child(even)');
```

会选择表格中的所有偶数行。

在上面的 jQuery 代码中，语句

```
$('td:nth-child(odd)').addClass('hover');
```

会选择表格中所有的奇数列（也就是说，父元素中索引为奇数的 td 子元素），并将样式规则.hover 定义的样式属性应用给它，将其背景色变为蓝色，前景色变为白色，如图 7-3 所示：

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-3　奇数列被突出显示的表格

1. 交替突出显示相邻行

为了交替地突出显示相邻行，我们要修改 jQuery 代码，如下所示：

```
$(document).ready(function() {
  $('table tr:odd').addClass('hover');
});
```

上面的 jQuery 代码会选择表格所有的奇数行，并将定义

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-4　交替突出显示相邻行的表格

在样式规则.hover 中的样式属性应用给它们，从而将其突出显示，如图 7-4 所示。

2. 突出显示鼠标悬停的列

你只想突出显示鼠标悬停在上面的标题所在列，也就是当鼠标指针移到任一列标题上时，你想要该列被突出显示。jQuery 代码如下所示：

```
$(document).ready(function() {
  $('th').hover(
    function(){
      var colindex=$(this).parent().children().index(this);
      $('table td:nth-child('+(colindex+1)+')').addClass('hover');
    },
    function(){
      $('table tr').children().removeClass('hover');
    }
  );
});
```

在研究 jQuery 代码之前，让我们先来看一下其中使用的.index()方法：

- **.index()**

该方法会在作为参数传递过来的元素中寻找所有匹配的元素，并返回传递过来的元素的索引，如果找到的话，那么从 0 开始。如果在匹配集中没有找到传递过来的元素，那么 Index()函数就会返回-1。如果 jQuery 对象被作为参数传递给 index()方法，那么它只会检查对象中的第一个元素。

语法：

```
.index(element)
```

现在让我们来看下 jQuery 代码本身。我们首先将 hover 事件附加给表格的标题（th）元素。当用户在表格标题上悬停鼠标指针时，我们就会找到该列的索引（列编号）并将它存储在变量 colindex 中。.index()方法从 0 开始计数。因此 index 为 0 意味着鼠标在第一列标题上悬停，index 为 1 意味着鼠标在第二列标题上悬停，依此类推。

然后我们可以将样式规则.hover 中定义的样式属性应用给索引存储在变量 colindex 中的列。由于:nth-child 方法从 1 开始计数（这与.index()方法从 0 开始不同），在 hover 事件的处理函数中，我们首先需要对存储在 colindex 中的值加 1，然后才将样式规则.hover 应用给我们需要 colindex 所代表的列。

当鼠标指针从列标题栏移出时，就会触发事件处理函数，其中我们将样式规则.hover 中定义的样式属性从表格的所有行上移除。

鼠标悬停的列的内容会被突出显示，如图 7-5 所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-5 鼠标悬停的列的内容被突出显示

3. 当鼠标悬停时突出显示列标题

在上面的例子中，我们看到，当鼠标经过列标题的时候，列的内容会突出显示，但是标题并没有被突出显示，它还和初始的时候一样。为了将列标题也突出显示，我们需要再添加一条语句（在 jQuery 代码中以黑体字显示）：

```
$(document).ready(function() {
  $('th').hover(
    function(){
      var colindex=$(this).parent().children().index(this);
      $(this).addClass('hover');
      $('table td:nth-child('+(colindex+1)+')').addClass('hover');
    },
    function(){
      $('table tr').children().removeClass('hover');
    }
  );
});
```

语句

```
$(this).addClass('hover');
```

当鼠标在列标题上悬停的时候，会把样式规则.hover 中定义的样式属性也应用给列标题，如图 7-6 所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-6　当鼠标悬停在标题处，列的内容和标题都突出显示

4. 当鼠标悬停时突出显示表格中单独的单元格

当鼠标经过表格中任一不是标题的单元格时，你想要突出显示它。达到这个效果的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('td').hover(
    function(){
      $(this).addClass('hover');
    },
    function(){
      $('table tr').children().removeClass('hover');
    }
  );
});
```

这次我们会看到 hover 事件被附加给 td 元素，这样，如果鼠标经过任一单元格（除了列标题）时，样式规则.hover 中定义的属性都会应用给它，使其突出显示，如图 7-7 所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-7 鼠标悬停的时候突出显示单元格

7.3 过滤行

问题描述

现有一表格，其中包含一些行和列。你想要的效果是，当鼠标指针移过任一行时，整行都会突出显示，当用户点击任一行时，所有其他行都会消失。

解决方案

对于这个解决方案，我们会使用与攻略 7.1 中同样的 HTML 和样式表文件。

当鼠标经过时突出显示行以及使所有行消失的 jQuery 代码如下所示：

```
$(document).ready(function() {

  $('tbody tr').hover(
    function(){
      $(this).find('td').addClass('hover');
    },
    function(){
      $(this).find('td').removeClass('hover');
    }
  );

  $('tbody tr').click(function(){
    $('table').find('tbody tr').hide();
    $(this).show();
  });
});
```

知其所以然

在上半部分 jQuery 代码中，我们为嵌入在 tbody 元素中的 tr（行元素）添加了 hover()事件，因为我们只希望包含学生信息的行在鼠标悬停时显示效果，而不是标题行。在 hover 事件的处理函数中，我们会搜索鼠标悬停行的 td 元素，并对其应用样式规则.hover（位于样式表文件 style.css 中），分别将其背景色变为蓝色，前景色变为白色，从而使其突出显示。

在下半部分 jQuery 代码中，我们向所有 tr 元素（嵌入在 tbody 元素中）添加了 click 事件，在它的事件处理函数中，我们会搜索所有 tr 元素（嵌入在 tbody 元素中），并让它们全部消失，也就是除了列标题之外的所有行都会消失。然后，我们会让被点击的行可见，从而使得只

有被点击的行显示在表格中。

当鼠标移过表格中任意一行时，效果会如图 7-8 所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-8　当鼠标移过一行时，该行被突出显示

当点击任意一行的时候，除了被点击的行之外的所有行都会消失，如图 7-9 所示。

Roll	Name	Marks
103	Chirag	85

图 7-9　表格中只留下被选中的行

隐藏选中行

我们可以修改上面的 jQuery 代码来取得相反的效果，也就是不保留选中的行，而是将其隐藏。jQuery 代码如下所示：

```
$(document).ready(function() {

  $('tbody tr').hover(
    function(){
      $(this).find('td').addClass('hover');
    },
    function(){
      $(this).find('td').removeClass('hover');
    }
  );

  $('tbody tr').click(function(){
    $(this).hide();
  });
});
```

我们可以看到，在 jQuery 代码中，我们使用 `hide()` 方法隐藏选中的行。当鼠标经过最后一行的时候，表格的显示可能会像图 7-8 这样。当点击那行的时候，它会被删除，剩下的表格如图 7-10 所示。

Roll	Name	Marks
101	John	87
102	Naman	90

图 7-10　表格中选中的行被删除

7.4 隐藏选定列

问题描述

现有一个表格，其中包含一些行和列。当鼠标指针移动到任一列的标题上时，你希望该列（包括列标题）可以突出显示。另外，当用户点击任一列标题的时候，整列——包括标题——都会隐藏起来。

解决方案

对于这个解决方案，我们会使用与攻略 7.1 中相同的 HTML 和样式表文件。

突出显示列（当鼠标经过列标题时）以及在点击时使其内容消失的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('th').hover(
    function(){
      var colindex=$(this).parent().children().index(this);
      $(this).addClass('hover');
      $('table td:nth-child('+(colindex+1)+')').addClass('hover');
    },
    function(){
      $('table tr').children().removeClass('hover');
    }
  );

  $('th').click(function(){
    $(this).hide();
    colindex=$(this).parent().children().index(this);
    $('table td:nth-child('+(colindex+1)+')').hide();
  });
});
```

知其所以然

在 jQuery 代码中，首先我们会向表格中所有列标题添加 hover 事件。在事件处理函数中，我们会找到鼠标悬停的列标题的索引位置，然后将其存储在变量 colindex 中。index()方法是从 0 开始的，即它从 0 开始计数。

然后我们会将样式规则.hover 中定义的样式属性应用给要突出显示的列标题和索引位置为存储在 colindex 变量中值的列（鼠标悬停的那个），以突出显示它们。由于:nth-child()方法从 1 开始计数，因此在突出显示它之前将 colindex 加 1。

当鼠标指针从列标题栏移出时，我们会将样式规则.hover 中定义的样式属性从表格的所有列上移除。在 click 事件处理器中，我们会隐藏被点击的行标题，并找到它的索引值，将其存储在变量 colindex 中。

最终，我们将索引值存储在变量 colindex（列标题被点击的索引位置）中的列的内容隐藏。这样，当点击任一列标题的时候，列标题和完整的列内容都会消失。当鼠标悬停在列标题上时，

它就会和列的内容一起突出显示，如图 7-11 所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-11　当鼠标悬停在标题上时，列的内容和标题都突出显示

当点击列标题的时候，它和列的内容会一起被删除，如图 7-12 所示。

Name	Marks
John	87
Naman	90
Chirag	85

图 7-12　表格中选中的行被删除

过滤列

你想要让表格中除了被点击的列标题之外的所有列都不可见。也就是你想要保留被点击的列（和标题），而使其他列消失。

想要突出显示列（当鼠标经过列标题时）并使被点击的列之外的列都消失，我们需要修改 jQuery 代码，如下所示：

```
$(document).ready(function() {
  $('th').hover(
    function(){
      var colindex=$(this).parent().children().index(this);
      $(this).addClass('hover');
      $('table td:nth-child('+(colindex+1)+')').addClass('hover');
    },
    function(){
      $('table tr').children().removeClass('hover');
    }
  );

  $('th').click(function(){
    colindex=$(this).parent().children().index(this);
    $('table th:not(:nth-child('+(colindex+1)+'))').hide();
    $('table td:not(:nth-child('+(colindex+1)+'))').hide();
  });
});
```

知其所以然

让我们看一下 jQuery 代码。首先要注意的是:not()选择器会选择所有不符合指定选择器的元素。比方说，它的工作方式是这样的，它会选择表格中所有不属于类型 student 的元素：

`$('table :not(.student)')`。

在 hover 事件处理函数中，我们会找到鼠标悬停的列标题的索引位置，然后将其存储在变量 colindex 中。.index()方法从 0 开始计数。然后我们会将样式规则.hover 中定义的样式属性应用给突出显示的列标题和索引位置为存储在 colindex 变量中值的列（鼠标悬停的那个），以突出显示它。由于:nth-child()方法从 1 开始计数，因此在突出显示它之前需要将 colindex 加 1。

当鼠标指针从列标题栏移出时，我们会将样式规则.hover 中定义的样式属性从表格的所有行上移除。在 click 事件处理函数中，我们会找到被点击的列标题的索引值，然后将其存储在变量 colindex 中。然后我们会使用 not 选择器来隐藏所有未被点击的列标题；即我们会保留索引位置存储在变量 colindex 中的列标题，而使其他列标题都不可见。

最终，我们将索引值不等于变量 colindex（列标题被点击的索引位置）的列的内容隐藏。唯一保留为可见的列是列标题被点击的那个。当鼠标悬停在列标题上时，它就会和列的内容一起突出显示，如图 7-13 所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-13 当鼠标悬停在标题上时，列的内容和标题都突出显示

当点击任一列标题的时候，那个标题（和该列的内容）会被过滤出来，使得表格中其他所有列都不可见，如图 7-14 所示。

Roll
101
102
103

图 7-14 选定的列被过滤出来

7.5 分页显示表格

问题描述

现有一个表格，其中包含一些行和列。你想要对该表格的行记录分页显示，即希望在表格的顶端显示页码，当点击任一页码的时候，属于该页的行就会显示出来。

解决方案

在这个解决方案中，我们会使用与攻略 7.1 中相同的 HTML 文件。我们需要定义样式规则，

从而当鼠标在页码上悬停时，页码就会突出显示，并在页码之间保持一定的距离。我们定义的两个样式规则是.hover 和.page，如下样式表文件所示：

```
style.css
.hover { background-color: #00f; color: #fff; }
.page{ margin:5px; }
```

下面的 jQuery 代码会将表格的行分页（根据我们想要在每页上显示的行数），并在我们点击页码的时候分别显示相应的行：

```
$(document).ready(function() {
  var rows=$('table').find('tbody tr').length;
  var no_rec_per_page=1;
  var no_pages= Math.ceil(rows/no_rec_per_page);
  var $pagenumbers=$('<div id="pages"></div>');
  for(i=0;i<no_pages;i++)
  {
    $('<span class="page">'+(i+1)+'</span>').appendTo($pagenumbers);
  }
  $pagenumbers.insertBefore('table');

  $('.page').hover(
    function(){
      $(this).addClass('hover');
    },
    function(){
      $(this).removeClass('hover');
    }
  );

  $('table').find('tbody tr').hide();
  var tr=$('table tbody tr');

  $('span').click(function(event){
    $('table').find('tbody tr').hide();
    for(var i=($(this).text()-1)*no_rec_per_page;
        i<=$(this).text()*no_rec_per_page-1;
        i++)
    {
      $(tr[i]).show();
    }
  });
});
```

知其所以然

样式规则 hover 中将 background-color 和 color 属性分别设置为#00f 和#fff，从而将鼠标悬停的页码的背景色变为蓝色，前景色变为白色。样式规则.page 中将 margin 属性设置为 5px，从而在页码之间保持 5px 的距离。

我们的 jQuery 代码首先会计算行（嵌入在 tbody 元素中的 tr 元素）的数目，并将其存储在变量 rows 中。例如，假设我们想要每页只看到一条记录，因此我们将变量 no_rec_per_page

的值初始化为 1。然后我们需要用总行数除以每行想要看到的行数，得到总的页数。页数被赋给 no_pages 变量。在开始处理事件之前，我们最后要做的是设置页码的显示效果。首先我们会定义一个 ID 为 pages 的 div 元素，并把它赋给变量$pagenumbers。

借助 for 循环，我们创建了一些 span 元素（数量和页数一样），其中包含有一系列的页码 1、2……，并且 span 元素的 class 属性被赋值为 page，从而定义在类选择器.page 中的样式属性会自动应用给所有的页码。最终，所有包含页码的 span 元素都被附加给 ID 为 pages 的 div 元素。为了完成目标，我们需要将存储在变量$pagenumbers 中 ID 为 pages 的 div 元素插入到 table 元素之前，这会使页码显示在表格之上。

接下来我们会为页码（class 为 page 的 span 元素）增加 hover()事件。在事件处理器中，当鼠标悬停在页码上的时候，就会使其突出显示。（在样式规则.hover 中定义的属性被应用给页码，分别将背景和前景的颜色变为蓝色和白色。）相反，当鼠标指针从上面移出时，我们会去掉样式规则.hover 中的样式属性。

在 hover 事件处理器之后，因为我们不想在首次载入页面的时候看到任何数据，所以隐藏了表格中所有行（嵌入在 tbody 元素中的 tr 元素），只显示列标题。只有在用户选择页码的时候，属于该页的行才会显示。然后我们会取得表格中的所有行，并将其存储在变量 tr 中，即现在 tr 是一个包含表格所有行的数组。

在页码的 click 事件处理器中，我们隐藏了表格的所有行（只保持该列可见）。然后我们会使用 tr 数组来显示点击的页码范围之内的行。

初始的时候，我们看到的是空白的表格，其中只有列标题和顶部的页码，如图 7-15 所示。

当鼠标悬停在任一页码上时，该数字会被突出显示；当点击页码的时候，属于该页的行就会显示出来，如图 7-16 所示。

1 2 3

Roll	Name	Marks

图 7-15 带有列标题和页码的表格

1 2 3

Roll	Name	Marks
101	John	87

图 7-16 显示第一页的内容（假设一页有一行）

在此我们看到在表格中只显示了一行，这是因为我们将变量 no_rec_per_page（它决定了每页所要显示的行数）的值设置为 1（参见代码的第 3 行）。下面让我们将这个值设置为 5。假设我们已经在 HTML 文件中向该表格元素中添加了更多行，现在当选择页码的时候，我们会看到每页有 5 行，如图 7-17 所示。

1 2 3

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85
104	David	92
105	Kelly	81

图 7-17 显示第一页的内容（假设一页有 5 行）

我们还可以修改 jQuery 代码，在开始时就显示属于第一页的内容（而不是像图 7-15 中那样只显示列标题）。默认在第一页中显示内容的 jQuery 代码如下：

```
$(document).ready(function() {
  var rows=$('table').find('tbody tr').length;
  var no_rec_per_page=1;
  var no_pages= Math.ceil(rows/no_rec_per_page);
  var $pagenumbers=$('<div id="pages"></div>');
  for(i=0;i<no_pages;i++)
  {
    $('<span class="page">'+(i+1)+'</span>').appendTo($pagenumbers);
  }
  $pagenumbers.insertBefore('table');

  $('.page').hover(
    function(){
      $(this).addClass('hover');
    },
    function(){
      $(this).removeClass('hover');
    }
  );

  $('table').find('tbody tr').hide();
  var tr=$('table tbody tr');

  for(var i=0;i<=no_rec_per_page-1;i++)
  {
    $(tr[i]).show();
  }

  $('span').click(function(event){
    $('table').find('tbody tr').hide();

for(i=($(this).text()-1)*no_rec_per_page;i<=$(this).text()*no_rec_per_page-1;i++)
    {
      $(tr[i]).show();
    }
  });
});
```

我们会看到在代码中添加了一个 for 循环，从而在屏幕上显示第一页（这依赖于赋给变量 no_rec_per_page 的值）内容。

7.6 展开和折叠列表项

问题描述

现有一个未排序的列表，其中有两个列表项，每个都包含未排序的内嵌列表。在两个列表项左边显示加号（+）图标（表示其中隐藏有其他项目）。当用户选择加号图标的时候，展开列表项，显示所有隐藏的元素，并且在展开模式的时候，加号图片会被替换为减号（-）图标。

解决方案

首先让我们创建一个 HTML 文件，其中包含带有两个列表项的未排序列表：Tea 和 Coffee。这两个列表项的 class 名都被赋值为 drink，从而可以通过 jQuery 代码来识别并访问它们。第一个列表项（Tea）是包含 3 个未排序项目 Darjeeling、Assam 和 Kerala 的列表。列表项 Assam 的 class 名为 drink，并且包含未排序的列表，其中有 2 个列表项：Green Leaves 和 Herbal。

第二个列表项 Coffee，它是包含两个列表项 Cochin 和 Kerala 的未排序列表。HTML 文件会像下面这样：

```
<body>
<ul>
  <li class="drink">Tea
    <ul>
      <li>Darjeeling</li>
      <li class="drink">Assam
        <ul>
          <li>Green Leaves</li>
          <li>Herbal</li>
        </ul>
      </li>
      <li>Kerala</li>
    </ul>
  </li>
  <li class="drink">Coffee
    <ul>
      <li>Cochin</li>
      <li>Kerala</li>
    </ul>
  </li>
</ul>
</body>
```

如果我们不应用任何样式或者 jQuery 代码的话，那么 HTML 文件在执行的时候会显示分别带有列表项的未排序列表，如图 7-18 所示。

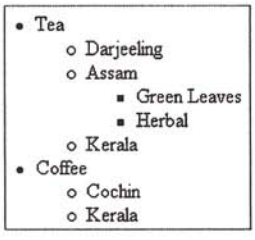


图 7-18 带有列表项的未排序列表

为了向列表项添加加号和减号图片，我们需要应用指定的样式规则。样式表文件 style.css 中会包含如下样式规则：

```
style.css
.plusimageapply{list-style-image:url(plus.jpg);}
```

```
.minusimageapply{list-style-image:url(minus.jpg);}
.noimage{list-style-image:none;}
```

为了在列表项上应用上面的样式规则，并且使它们在选择左侧的加号、减号图片进行展开和折叠，我们需要编写下列的jQuery代码：

```
$(document).ready(function() {
  $('li.drink').addClass('plusimageapply');
  $('li.drink').children().addClass('noimage');
  $('li.drink').children().hide();
  $('li.drink').each(
    function(column) {
      $(this).click(function(event){
        if (this == event.target) {
          if($(this).is('.plusimageapply')) {
            $(this).children().show();
            $(this).removeClass('plusimageapply');
            $(this).addClass('minusimageapply');
          }
          else
          {
            $(this).children().hide();
            $(this).removeClass('minusimageapply');
            $(this).addClass('plusimageapply');
          }
        }
      });
    }
  );
});
```

知其所以然

样式规则plusimageapply会应用到所有其中拥有未排序列表的列表项(处于折叠模式)上，在其中我们将list-style-image属性设置为url(plus.jpg)，从而用加号图标（我们假设存在名为plus.jpg的图片文件）替换传统的圆形符号。

同样，样式规则minusimageapply会应用到所有处于展开模式的列表项上，它将list-style-image属性设置为url(minus.jpg)，从而在列表项的左侧显示减号图标。我们假设图片文件minus.jpg中包含有减号图标。样式规则noimage会应用给所有不含有内嵌的未排序列表的列表项目，它将liststyle-image属性设置为none，从而显示传统的圆形图标。

现在让我们看一下jQuery代码，在所有class为drink的列表项（其中拥有内嵌的未排序列表）上都会应用plusimageapply样式属性，使得在它们的左侧显示加号图标。对于所有其他的列表项（其中不含有内嵌列表项），都会在其上应用noimage样式规则，使得他们以传统的圆形符号显示。初始的时候我们让列表项（其中拥有未排序的列表）的所有内嵌元素可见。即我们让所有其中带有未排序列表项的列表项以折叠模式显示。

为了应用expansion功能，我们为每个列表项（其中拥有未排序列表项）依次添加了click事件。在事件处理器中，我们会检查在上面发生click事件的列表项是否具有样式规则

plusimageapply。如果有的话，那么我们就显示列表项的隐藏内容。

然后我们会移除样式规则 plusimageapply 的样式属性，并应用样式规则 minusimageapply 的样式属性，从而对于展开的列表项，用减号图标替换加号图标。如果被点击的列表项的左侧是减号图片，也就是说，如果在上面没有应用 plusimageapply 样式规则，那么我们就隐藏内嵌的内容。我们还删除了样式规则 minusimageapply 的样式属性，并应用样式规则 plusimageapply 的样式属性，从而将处于折叠状态的列表项的减号图标替换为加号图标。（我们需要把拥有 minusimageapply 样式规则并且被点击的列表项折叠起来。）

初始的时候，列表项会处于折叠模式，左侧是加号图标，如图 7-19 所示。

当你选择加号图标（或者列表项本身）时，它就会展开，并显示内嵌在其中的未排序列表，而加号图标会被替换为减号图标，如图 7-20 所示。由于列表项 Assam 中还包含未排序的列表项，所以它的左侧是加号图标。

图 7-19 折叠模式的两个列表项

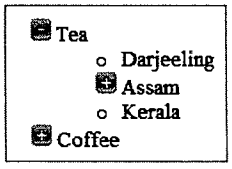


图 7-20 当第一个列表项被点击时显示的内容

当你选择 Assam 列表项左侧的加号图标时，它就会展开并显示其中隐藏的列表项，而加号图标会被替换为减号图标。

当你选择 Coffe 列表项的加号图标（或列表项本身）时，它会展开并显示其中隐藏的列表项，而加号图标会被替换为减号图标。如图 7-22 所示。

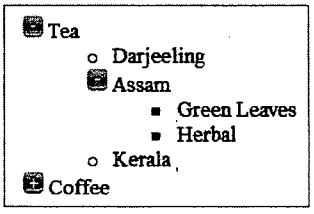


图 7-21 当项目被点击的时候，其中的内嵌列表项内容（折叠的列表项）会显示出来

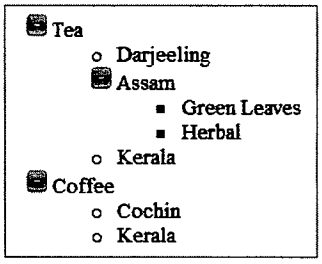


图 7-22 当第二个列表项被点击时，其中的内容会显示出来

7.7 展开和折叠行

问题描述

你拥有一个包含 15 行的表格，每行包含 3 列。表格中显示的是学生的成绩。初始时它会显

示3行。Roll 101-105、Roll 106-110和Roll 111-115。每次你最多想要看到5条记录，也就是说，如果你将鼠标悬停在Roll 101-105行上，它必定会展开，显示编号在101到105之间的学生的成绩。

解决方案

首先我们要创建一个HTML文件，其中的table元素会拥有标题行和包含15行（tr）元素的表格主体，如下所示：

```
<body>
<table border="1">
<thead>
<tr><th>Roll</th><th>Name</th><th>Marks</th></tr>
</thead>
<tbody>
<tr><td colspan=3 class="studgroup" align="center">Roll 101-105</td></tr>
<tr><td>101</td><td>John</td><td>87</td></tr>
<tr><td>102</td><td>Naman</td><td>90</td></tr>
<tr><td>103</td><td>Chirag</td><td>85</td></tr>
<tr><td>104</td><td>David</td><td>92</td></tr>
<tr><td>105</td><td>Kelly</td><td>81</td></tr>
<tr><td colspan=3 class="studgroup"  align="center">Roll 106-110</td></tr>
<tr><td>106</td><td>Charles</td><td>77</td></tr>
<tr><td>107</td><td>Jerry</td><td>91</td></tr>
<tr><td>108</td><td>Beth</td><td>75</td></tr>
<tr><td>109</td><td>Caroline</td><td>82</td></tr>
<tr><td>110</td><td>Hanen</td><td>71</td></tr>
<tr><td colspan=3 class="studgroup"  align="center">Roll 111-115</td></tr>
<tr><td>111</td><td>Douglas</td><td>57</td></tr>
<tr><td>112</td><td>Tim</td><td>86</td></tr>
<tr><td>113</td><td>Michael</td><td>68</td></tr>
<tr><td>114</td><td>Kimbley</td><td>88</td></tr>
<tr><td>115</td><td>Christina</td><td>72</td></tr>
</tbody>
</table>
</body>
```

表格的标题是由内嵌在thead元素中的th元素表示的，而学生的成绩是使用tbody元素中内嵌的tr元素表示的。为了指定5行的数据块，我们定义了3个行（包含跨3栏的td元素）：Roll 101-105、Roll 106-110和Roll 111-115。这些行（跨3栏的td元素）的class名称都是studgroup，可以在jQuery中识别并使用。

为了突出显示这些行，我们在样式表文件中定义了样式规则.hover。样式表文件style.css如下所示：

```
style.css
.hover { background-color: #00f; color: #fff; }
```

当鼠标在某些行上悬停时，相应的隐藏的行会展开，jQuery代码如下所示：

```
$(document).ready(function() {
  $('table tbody tr').hide();
  $('table tbody').find('.studgroup').parent().show();
```

```
$('tbody tr').hover(
  function(){
    var tr=$('table tbody tr');
    var rindex=$(this).parent().children().index(this);
    for(var i=rindex;i<=rindex+5;i++)
    {
      $(tr[i]).show();
    }
    $(this).addClass('hover');
  },
  function(){
    $('table tbody tr').hide();
    $('table tbody').find('.studgroup').parent().show();
    $(this).removeClass('hover');
  }
);
});
```

知其所以然

.hover 样式规则将 background-color 属性设置为#00f，将 color 属性设置为#fff，从而分别将鼠标悬停时的行的背景色和前景色分别设置为蓝色和白色。

jQuery 代码语句的作用如下：

首先我们会隐藏表格的 tbody 元素中内嵌的行，即我们会隐藏除表格标题之外的所有行。

然后我们会显示 3 行，每行都代表 5 行的数据块，它由跨 3 栏的 td 元素组成。(分别是 Roll 101-105、Roll 106-110 和 Roll 111-115。) 之前我们提到过，这 3 行的 class 名称都被赋予 studgroup。

然后我们会向表格的可见行添加 hover 事件。在事件处理器中，当鼠标指针移过行的时候，我们会将其突出显示（样式规则.hover 中定义的属性被应用给该行，分别将它的背景色和前景色变为蓝色和白色）。相反，当鼠标指针从行上移出时，我们会去掉样式规则.hover 的样式属性。在 hover 事件处理器中，我们也会取得表格的所有行，并将其存储在变量 tr 中——现在 tr 是包含表格的所有行的数组——然后我们会找到鼠标悬停的行的索引值，将其存储在变量 rindex 中。然后，借助于 for 循环，我们显示了下面的 5 行，即 class 名为 studgroup 的行标题下面的行会被显示。

当鼠标指针移出行的时候，hover 事件的处理器就会执行，它不仅删除了 hover 样式规则的属性，并且还隐藏了除标题行之外的所有行。

在上面的 jQuery 代码的执行过程中，我们找到了 3 行：Roll 101-105、Roll 106-110 和 Roll 111-115。它们分别指定了图 7-23 所示的记录组。

Roll	Name	Marks
Roll 101-105		
Roll 106-110		
Roll 111-115		

图 7-23 带有指定下面记录组的行的表格

当鼠标悬停在任意一行上时，带有 5 条内嵌记录的组就会被显示。例如，如果我们将鼠标悬停在 Roll 101-105 行上，那么该行就会突出显示。

类似地，当鼠标悬停在 Roll 111-115 上的时候，该行会被突出显示，并且下面编号范围内的学生记录会显示出来，如图 7-25 所示。

Roll	Name	Marks
Roll 101-105		
101	John	87
102	Naman	90
103	Chirag	85
104	David	92
105	Kelly	81
Roll 106-110		
Roll 111-115		

图 7-24　当鼠标在指定它们的组的行上悬停时，学生记录就会显示出来

Roll	Name	Marks
Roll 101-105		
Roll 106-110		
Roll 111-115		
111	Douglas	57
112	Tim	86
113	Michael	68
114	Kimbley	88
115	Christina	72

图 7-25　当鼠标在指定他们的组的行上悬停时，学生记录会再次显示出来

带有加号和减号图标的行

当鼠标在图 7-23 中所示的行上悬停的时候，你希望在左侧显示加号图标。当行被展开时，它应该显示各自范围内的记录，而加号应该替换为减号图标。

为了达到这个目的，jQuery 代码需要做如下改动：

```
$(document).ready(function() {
  $('.studgroup').css(
    {'background-image':"url(plus.jpg)",
     'background-repeat':"no-repeat",
     'background-position':"left"}
  );
  $('table tbody tr').hide();
  $('table tbody').find('.studgroup').parent().show();
  $('tbody tr').hover(
    function(){
      $(this).find('.studgroup').css(
        {'background-image':"url(minus.jpg)",
         'background-repeat':"no-repeat",
         'background-position':"left"}
      );
      var tr=$('table tbody tr');
      var rindex=$(this).parent().children().index(this);
      for(var i=rindex;i<=rindex+5;i++)
      {
        $(tr[i]).show();
      }
      $(this).addClass('hover');
    },
    function(){
      $(this).find('.studgroup').css(
        {'background-image':"url(plus.jpg)",
```

```
            'background-repeat':"no-repeat",
            'background-position':"left"}
        );.
        $('table tbody tr').hide();
        $('table tbody').find('.studgroup').parent().show();
        $(this).removeClass('hover');
      }
    );
});
```

在上面的代码中，我们在显示学号范围的行中（跨 3 列的 td 元素并且 class 被赋值为.studgroup 的行）显示了加号图标，假设该图标存在于图片文件 plus.jpg 中。我们在.css()方法中使用 background-repeat 和 background-position 属性，使得图标只显示一次，并且位于行的左侧。

然后我们会隐藏所有内嵌在表格的 tbody 元素中的行。也就是说，我们会隐藏所有的学生记录（除了表格的标题），因为我们希望只有在鼠标悬停在相关的行上时才显示它们。然后我们会显示带有学号范围内的 3 行（那些 class 被赋值为 studgroup 并且旁边带有加号图标的行）。

当鼠标悬停在任意一行上时，我们会使用.css()方法将它左侧的加号图标会替换为减号图标，并且将 background-repeat 和 background-position 属性分别设置为 no-repeat 和 left，这使得减号图标只在行的左侧显示一次。在 hover 事件处理器中，我们也会取得表格的所有行，并将其存储在变量 tr 中，现在 tr 是包含表格的所有行的数组，然后我们会找到鼠标移过的行的索引值，将其存储在变量 rindex 中。然后，借助于 for 循环，我们会显示下面的 5 行，即 class 名为 studgroup 的行标题下面的行会被显示。最终，对于 hover 事件，我们应用了定义在样式规则 hover 中的属性来突出显示鼠标悬停的行。

当鼠标从悬停的行上移开时，我们会使用.css()方法将它左侧的减号图标替换为加号图标，并且将 background-repeat 和 background-position 属性分别设置为 no-repeat 和 left，这使得加号图标只在行的左侧显示一次。然后我们会隐藏表格的除了显示学号范围的行之外的所有行，并在旁边显示加号图标（这是我们希望鼠标指针从悬停的行上移走时所要显示的）。最终，我们会移除.hover 样式规则中所定义的属性，使得当鼠标指针移走的时候，行的显示和初始时一样。

在上面的 jQuery 代码的执行过程中，我们会找到 3 行——Roll 101-105、Roll 106-110 和 Roll 111-115——它们指定了下面的记录组。我们还会发现每行的左侧都有一个减号图标，表示该行当前是折叠的，如图 7-26 所示。

Roll	Name	Marks
⊞ Roll 101-105		
⊞ Roll 106-110		
⊞ Roll 111-115		

图 7-26 指定下面一组记录的行，左侧带有加号图标

当我们在任意一行上悬停鼠标的时候，该行会被突出显示，并显示相关的学生记录，而鼠标

悬停的行的加号图标会被替换为减号图标，如图 7-27 所示。

类似地，如果我们在行 Roll 106-110 上悬停鼠标，那么它就会突出显示，并且鼠标悬停行的加号图标会被替换为减号图标；同时，之前鼠标悬停的行（现在失去焦点）的左侧会重新显示加号图标（见图 7-28）。

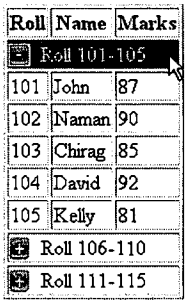


图 7-27　鼠标悬停的行显示记录，加号图标被替换为减号图标

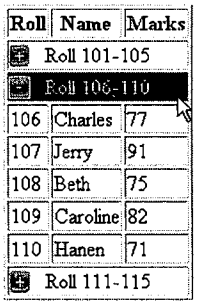


图 7-28　当行恢复折叠模式的时候，重新显示加号图标

7.8　对列表项目排序

问题描述

现有一个包含几个列表项的未排序列表，你想要对其进行排序。

解决方案

让我们创建包含未排序列表的 HTML 文件：

```
<body>
<ul>
  <li>Tea</li>
  <li>Coffee</li>
  <li>Pepsi</li>
  <li>Energy Drink</li>
  <li>Soup</li>
</ul>
</body>
```

当在浏览器中打开这个 HTML 文件的时候，它会显示如图 7-29 所示的列表项。

- Tea
- Coffee
- Pepsi
- Energy Drink
- Soup

图 7-29　未排序的列表项

用来对列表项排序的 jQuery 代码如下所示：

```
$(document).ready(function() {
  var drinks = $('ul').children('li').get();

  drinks.sort(function(a, b) {
    var val1 = $(a).text().toUpperCase();
    var val2 = $(b).text().toUpperCase();
    return (val1 < val2) ? -1 : (val1 > val2) ? 1 : 0;
  });

  $.each(drinks, function(index, row) {
    $('ul').append(row);
  });
});
```

知其所以然

现在来看一下 jQuery 代码，首先我们取得了所有列表项，它们都是未排序列表的子项，并将它们存储在变量 drinks 中。也就是说，drinks 将会成为包含列表项文字的数组。然后我们在 drinks 数组上调用.sort()函数，它会重复，每次取得数组的两个元素，并将它们赋给参数 a 和 b 来进行比较。sort 函数会根据赋给参数 drinks 和 b 的值来返回值。我们需要注意下列函数细节。

- 当函数返回的值小于 0 时，意味着第二个值大于第一个，要将它排在后面。
- 当函数返回值等于 0 时，意味着两个值相等，不需要改变顺序。
- 当函数返回的值大于 0 时，意味着第一个值大于第二个，要将它排在后面。

在调用排序算法之前，我们先会将传递给 sort 函数的两个数组元素变为大写。然后再使用排序算法，返回列表中的值，这些值会按照字母顺序排序。最终，each()函数会操作数组 drinks（包含排序的列表项）；我们提取数组中排好序的元素，并将它附加到未排序的列表元素上。即我们将排好序的列表项附加到未排序的列表元素上，并显示它们。

在上面的 jQuery 代码执行完毕之后，我们得到图 7-30 所示的排好序的列表项。

- Coffee
- Energy Drink
- Pepsi
- Soup
- Tea

图 7-30 排好序的列表项

7.9 对表格排序

问题描述

现有一个表格，其中包含一些行和列。当用户选择表格的任一列时，它的内容就会按照升序

排列。

解决方案

让我们创建一个 HTML 文件，其中定义了一些行和列。

```
<body>
<table border="1">
<thead>
<tr><th>Roll</th><th>Name</th><th>Marks</th></tr>
</thead>
<tbody>
<tr><td>103</td><td>Chirag</td><td>85</td></tr>
<tr><td>102</td><td>Naman</td><td>90</td></tr>
<tr><td>101</td><td>John</td><td>87</td></tr>
</tbody>
</table>
</body>
```

我们定义了边框为 1px 的表格，它有 3 个列标题（使用 th 元素定义）：Roll，Name 和 Marks。另外它还包含 3 行学生的记录。

表格标题嵌入在 thead 元素中，而表格的主体（包含信息的行）嵌入在 tbody 元素中。

我们会通过 3 个步骤来学习这个攻略：

(1) 确定点击的是哪个列标题；

(2) 以升序对列排序；

(3) 以升序和降序对列排序。

确定点击的是哪个列标题

在以升序或者降序对列进行排序之前，你需要知道选择的是表格的哪个列标题。为了突出显示用户选择的列标题，我们需要在样式表文件中定义样式规则 .hover。样式表文件 style.css 可能像下面这样：

```
style.css
.hover{
cursor: default;
color: blue ;
background-color:cyan
}
```

这里是显示选择的是哪列标题的 jQuery 代码：

```
$(document).ready(function() {
  $('th').each(function() {
    $(this).hover(
      function(){
        $(this).addClass('hover');
      },
      function(){
        $(this).removeClass('hover');
      }
    );
```

```
    $(this).click(function(){
      alert($(this).text()+' column is selected');
    });
  });
});
```

知其所以然

在样式表文件中，样式规则.hover 中将 cursor 属性设置为 default，使得鼠标指针以正常的状态显示（箭头的样式）。Color 和 background-color 属性分别被设置为 blue 和 cyan，从而将突出显示的列标题的背景色设为青绿色，前景色设为蓝色。

在上面的 jQuery 代码中我们能够看到，在其中会检查每个表格的标题，看鼠标是否在其上悬停。如果鼠标在任一表格标题上悬停，那么样式规则.hover 中的样式属性就会应用其上，使其突出显示（它的背景色会变为青绿色，而前景色会变为蓝色）。当鼠标指针从列标题上移开的时候，样式规则.hover 的样式属性会被删除，使得列标题像初始时那样显示。我们还会检查标题是否被点击，如果被点击，那么我们就会使用 alert()方法来显示列标题的文字。

当鼠标指针移过任一列标题的时候，标题会突出显示，并且它的名字会通过 alert()方法显示出来，如图 7-31 所示。

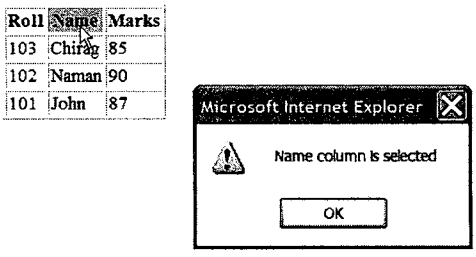


图 7-31 当鼠标在列标题上悬停的时候，会显示列名

使列升序排列

使一列升序排列的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('th').each(function(column) {
    $(this).hover(
      function(){
        $(this).addClass('hover');
      },
      function(){
        $(this).removeClass('hover');
      }
    );

    $(this).click(function(){
      var rec=$('table').find('tbody >tr').get();
```

```
        rec.sort(function(a, b) {
          var val1 = $(a).children('td').eq(column).text().toUpperCase();
          var val2 = $(b).children('td').eq(column).text().toUpperCase();
          return (val1 < val2) ? -1 : (val1 > val2) ? 1 : 0;
        });

        $.each(rec, function(index, row) {
          $('tbody').append(row);
        });
      });
    });
});
```

在继续说明之前，让我们先讨论一下代码中“>”符号的重要性。

“>”符号

“>”符号是一个选择器，它代表选中元素的子元素；例如，在下面的例子中，它会选择元素E1中所有与元素E2匹配的子元素：

语法：

```
E1>E2
```

现在来看一下我们的jQuery代码，它会检查每个列标题，看鼠标是否在上面悬停。如果鼠标在表格的任一标题上悬停，那儿样式规则.hover中定义的样式属性就会应用到标题上，使其突出显示。而当鼠标指针从标题上移开的时候，样式规则.hover的样式属性会被删除，使得标题像初始时那样显示。

然后我们会为表格的每个标题添加click事件。在click事件处理器中，我们会取得表格的所有行（嵌入在tbody元素中）并将其存储在变量rec中。现在rec就是包含表格所有行的数组。通过在rec数组上调用sort()函数，我们会重复地从数组中每次取出两个元素（行），以升序排列。为了达到这个目的，传递给sort()函数的第一个和第二个参数的内容会被提取出来，并在比较之前转换为大写。

然后sort()函数会返回小于0、等于0或者大于0的值，这会帮助我们确定哪个行的内容应该在排列顺序中上移，哪个应该下移。当这个排序函数执行完毕，rec数组中的就是按照升序排列的选定列的所有行。最终，我们会取得rec数组中的排好序的列，并附加到表格的tbody元素中，以显示最终的效果。

Roll	Name	Marks
103	Chirag	85
102	Narnan	90
101	John	87

图7-32　未排序的表格

初始时，表格的显示效果如图7-32所示。

当选择Roll列的时候，它会被突出显示，并且表格的行会按照学号升序排列，如图7-33所示。

类似地，当选择Name列的时候，它会被突出显示，并且表格的行会按照名字的字母顺序升序排列，如图7-34所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85

图 7-33 按照学号升序排列的表格

Roll	Name	Marks
103	Chirag	85
101	John	87
102	Naman	90

图 7-34 按照名字升序排列的表格

以升序和降序对列进行排序

如果某一列标题第一次被点击，那么表格会按照该列升序排列，如果再次点击该列，那么表格就会按照这列降序排列。也就是说，我们想要在每次点击的时候切换排序方式。为了告诉用户当前的排序是应用在哪一列上，我们需要在列标题上显示向上或向下的箭头。向上的箭头表示表格按照该列升序排列，而向下的箭头表示表格按照该列降序排列。

为了在列标题上显示向上和行下的箭头，我们需要在样式表文件中 style.css 中定义两个样式规则，如下所示：

```
style.css
.asc{
background:url('up.png') no-repeat; padding-left:20px;
}

.desc{
background:url('down.png') no-repeat; padding-left:20px;
}
```

现在让我们来修改 jQuery 代码，从而通过选择列来按照升序或者降序对表格排序。

```
$(document).ready(function() {
  $('th').each(function(column) {
    $(this).hover(
      function(){
        $(this).addClass('hover');
      },
      function(){
        $(this).removeClass('hover');
      }
    );

    $(this).click(function(){
      if($(this).is('.asc'))
      {
        $(this).removeClass('asc');
        $(this).addClass('desc');
        sortdir=-1;
      }
      else
      {
        $(this).addClass('asc');
        $(this).removeClass('desc');
        sortdir=1;
      }
```

```
        $(this).siblings().removeClass('asc');
        $(this).siblings().removeClass('desc');

        var rec=$('table').find('tbody >tr').get();
        rec.sort(function(a, b) {
          var val1 = $(a).children('td').eq(column).text().toUpperCase();
          var val2 = $(b).children('td').eq(column).text().toUpperCase();
          return (val1 < val2) ? -sortdir : (val1 > val2) ? sortdir : 0;
        });

        $.each(rec, function(index, row) {
          $('tbody').append(row);
        });
      });
    });
});
```

知其所以然

在样式规则.asc 中我们将 background 属性设置为 url(up.png)，从而在列标题中显示向上的箭头。属性值 no-repeat 会使得箭头在列标题中只显示一次，而 padding-left 属性被设置为 20px，从而在左侧留出一些空间。类似地，样式规则 desc 中包含的 background 属性让我们可以在列标题中显示向下的箭头。

然后，在 jQuery 代码中我们首先会检查每个标题，看鼠标是否在某个标题上面悬停。如果鼠标在表格的任一标题上悬停，那儿样式规则.hover 中定义的样式属性就会应用到这个标题上，使其突出显示。当鼠标指针从列标题上移开的时候，样式规则.hover 的样式属性会被删除，使得列标题像初始时那样显示。

然后我们会为表格的每个标题都添加 click 事件，并检查是否在鼠标悬停的列标题上已经应用了样式规则.asc，即我们会检查表格是否通过选择列标题按照升序排列了。如果是那样的话，那么我们就会移除定义在样式规则 asc 中的样式属性，并应用定义在样式规则 desc 中的样式属性（当列标题再次被点击的时候，表格会按照升序排列）。这样，列标题的左边会显示向下的箭头。另外，变量 sortdir 的值被设为-1，它将会被用来操作 sort 函数的返回值，以按照降序进行排序。

如果在选定的列上已经应用了 desc 样式规则，即表格已经按照选定列的降序排列，那么我们会删除定义在样式规则 desc 中的样式属性，并且应用定义在样式规则 asc 中的属性，这会在列标题的左侧放置向上的箭头。此外，变量 sortdir 的值为 1，这使得 sort 函数对表格进行升序排列。

一旦我们已经对列进行了排序，那么就需要删除样式规则.asc 和.desc 中的属性。从用户未选定的列标题中，我们还需要取得表格所有行（嵌入在 tbody 元素中），并将其存储在变量 rec 中。现在 rec 就是包含表格所有行的数组。

然后我们会在 rec 数组上调用 sort 函数。sort 函数会每次取出数组的两个元素（行），重复进行，并按照变量 sortdir 中的值所确定的排列顺序对其进行排序。在 sort 函数中，首先会

提取传递给 sort 函数的列内容的第一和第二个参数（行），然后在比较之前将它们转换为大写。

这个函数会返回小于 0、等于 0 或者大于 0 的值，这会帮助我们确定哪个行的内容应该在排列顺序中上移，哪个应该下移。当 sort 函数执行完毕，rec 数组会使得所有行按照选定列的升序或者降序排列，这取决于赋给变量 sortdir 的值。最终，我们会取得 rec 数组中的排好序的列，并将其附加到表格的 tbody 元素中，以显示最终的效果。

初始时，表格的显示效果如图 7-35 所示。

Roll	Name	Marks
103	Chirag	85
102	Naman	90
101	John	87

图 7-35 包含一些行列的未排序表格

如果我们选择列标题 Name，那么在它左侧就会出现一个上箭头（代表已经完成升序排序），并且表格已经按照名字的字母顺序升序排列了，如图 7-36 所示。

当我们再次选择列标题 Name 时，在它左侧就会出现一个下箭头（代表已经完成降序排序），并且表格已经按照名字的字母顺序降序排列，如图 7-37 所示。

Roll	▲ Name	Marks
103	Chirag	85
101	John	87
102	Naman	90

图 7-36 按照名字升序排列的表格

Roll	▼ Name	Marks
102	Naman	90
101	John	87
103	Chirag	85

图 7-37 按照名字降序排列的表格

7.10 过滤表格中的行

问题描述

现有一个包含一些行列的表格，在表格前面有一个文本输入框。要基于在文本框中输入的字符来过滤掉表格中的某些行。例如，如果用户输入字符 c，那么表格中所有名字以 c 开头的行会显示，而其他行都会被过滤掉。

解决方案

首先我们会创建一个 HTML 文件，其中包含带有一些行列的表格元素。在表格的前面，我们会显示信息“输入字符”，后面是一个文本输入框。在文本输入框下面，我们会显示一个提交按钮，点击它的时候，会显示表格中过滤后的信息。HTML 文件会像下面这样：

```
<body>
<div><span class="label">Enter a character </span><input type="text"  class="infobox"
```

```
/></div>
<input class="submit" type="submit" value="Submit"/><br/><br/>
<table border="1">
<thead>
<tr><th>Roll</th><th>Name</th><th>Marks</th></tr>
</thead>
<tbody>
<tr><td>101</td><td>John</td><td>87</td></tr>
<tr><td>102</td><td>Naman</td><td>90</td></tr>
<tr><td>103</td><td>Chirag</td><td>85</td></tr>
<tr><td>104</td><td>David</td><td>92</td></tr>
<tr><td>105</td><td>Kelly</td><td>81</td></tr>
<tr><td>106</td><td>Charles</td><td>77</td></tr>
<tr><td>107</td><td>Jerry</td><td>91</td></tr>
<tr><td>108</td><td>Beth</td><td>75</td></tr>
<tr><td>109</td><td>Caroline</td><td>82</td></tr>
<tr><td>110</td><td>Hanen</td><td>71</td></tr>
<tr><td>111</td><td>Douglas</td><td>57</td></tr>
<tr><td>112</td><td>Tim</td><td>86</td></tr>
<tr><td>113</td><td>Michael</td><td>68</td></tr>
<tr><td>114</td><td>Kimbley</td><td>88</td></tr>
<tr><td>115</td><td>Christina</td><td>72</td></tr>
</tbody>
</table>
</body>
```

可以看到，文本信息嵌入在 class 为 label 的 span 元素中，而文本输入框的 class 名称为 infobox，提交按钮的 class 名称是 submit，从而定义在类选择器.label、.infobox 和.submit（定义在样式表文件中）的样式属性可以自动应用到控件上。样式表文件会像下面这样：

```
style.css
.label {float: left; width: 120px; }
.infobox {width: 200px; }
.submit { margin-left: 125px; margin-top: 10px;}
```

过滤表格的行，以显示以文本输入框中所提供的字符开头的名字的 jQuery 代码如下：

```
$(document).ready(function() {
  var rows;
  var coldata;

  $('.submit').click(function(event){
    $('table').find('tbody tr').hide();
    var data=$('.infobox').val();
    var len=data.length;
    if(len>0)
    {
      $('table tbody tr').each(function(){
        coldata=$(this).children().eq(1);
        if(coldata.text().charAt(0).toUpperCase()==data.charAt(0).toUpperCase())
        {
          $(this).show();
        }
      });
```

```
        }
        event.preventDefault();
    });
});
```

知其所以然

定义在类选择器.label 中的属性将 float 属性设置为 left，这使得标签显示在浏览器窗口的左侧(为显示在它右侧的元素留出空间)。width 属性被设置为 200px，使得标签的宽度在 200px 之内。类选择器.infobox 中将 width 属性设置为 200px，使得文本输入框的宽度为 200px，而类选择器.submit 中将 margin-left 属性设置为 125px，使得提交按钮距离浏览器左边界的距离是 125px（从而显示在文本输入框的下面）。margin-top 属性被设置为 10px，使得提交按钮的顶端离文本输入框的距离为 10px。

在我们的 jQuery 代码中，首先会向提交按钮附加一个 click 事件，其中我们会隐藏表格的所有行（内嵌在 tbody 元素中的 tr 元素），只显示列标题。然后我们会取得文本输入框（class 的名称被赋值为 infobox）中输入的内容，并将其存储在变量 data 中。

接下来我们会找到变量 data 的长度，并扫描表格的每一行（内嵌在 tbody 元素中的 tr 元素）。对于每一行，我们都会取得行元素之下索引为 1（也就是 Name 列的索引）的子元素，并将其存储在变量 coldata 中。

然后我们会将列的内容（在 coldata 中）的第一个字符和文本输入框中输入的字符（在将其转换为大写之后）进行比较，从而开始过滤。如果匹配的话，我们就会显示该行。

最终，我们会调用事件对象的 .preventDefault()方法，从而避免将用户输入的信息提交给服务器。也就是，防止按钮点击的时候执行浏览器的默认行为。

在执行上面的 jQuery 代码时，我们会得到带有提交按钮的文本输入框。如果我们在文本输入框中输入了一个字符，比方说 c，那么表格中所有名字的开头字符为 c 的行就会显示，而其他行都会被过滤掉，如图 7-38 所示。

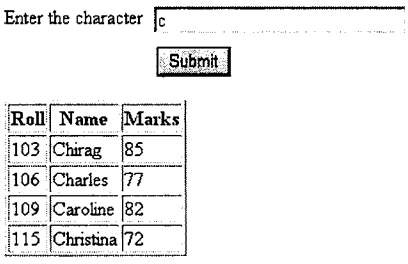


图 7-38 显示包含名字的开头字母为文本输入框输入的字符的行

7.11 小结

本章中，你学到了几个在表格上执行不同功能的攻略，包括突出显示行和列、过滤选定的行、

删除选定的行以及对表格的行进行分页显示。你还学习了在展开和折叠表格的行时所涉及的方法以及表格如何才能按照选定列的升序和降序进行排序。

在下一章中，你会看到如何通过应用 Ajax 技术使得网页更好地响应。其中包括如何使用 Ajax 来显示欢迎信息并执行认证。我们还会探究如何使用 Ajax 来验证用户名和邮件地址，以及如何使用 Ajax 对表格分页显示。

你还将学会使用 Ajax 实现自动完成功能。最终，你会懂得如何使用 Ajax 取得序列化的信息、JSON 数据和 XML 数据。

第8章 Ajax

Ajax是一种让网页的响应更快的技术，因为它允许异步地向服务器发送请求，而不需要重新载入页面。本章中，你会学到一些使用Ajax来提升用户体验的攻略：

- 显示欢迎信息；
- 执行认证；
- 验证用户名；
- 验证邮件地址；
- 使用自动完成；
- 导入HTML；
- 取得JSON数据；
- 取得XML数据；
- 分页显示表格。

8.1 显示欢迎信息

问题描述

表单上有一个标签，一个文本输入框和一个提交按钮，要在用户选择提交按钮的时候向他们显示一条欢迎信息，但前提是他们已经在文本输入框中输入了自己的名字。

解决方案

让我们创建一个HTML文件，其中包含一个`label`元素，一个文本输入框，一个提交按钮以及一个空的`div`元素。HTML文件应该像下面这样：

```
<body>
  <form>
    <label>Enter your Name</label>
    <input type="text" name="uname" class="uname"/>  <br/>
    <input type="submit" id="submit"/>
  </form>
  <div id="message"></div>
</body>
```

你会看到 HTML 文件中在表单内包含有一个标签，一个文本输入框和一个提交按钮。文本输入框的 class 被设置为 uname，目的是为了我们可以通过将要编写的代码来访问它。在表单下面是一个空的 id 为 message 的 div 元素，我们会使用它来向用户显示欢迎信息。

针对我们的解决方案的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('#submit').click(function () {
    var name = $('.uname').val();
    var data = 'uname=' + name;
    $.ajax({
      type:"POST",
      url:"welcome.php",
      data: data,
      success: function (html) {
        $('#message').html(html);
      }
    });
    return false;
  });
});
```

服务器上的脚本文件 welcome.php 如下所示：

```
<?php
$name = $_POST['uname'];
echo "Welcome ".  $name;
?>
```

知其所以然

ajax()方法会使用 HTTP 请求载入远程的页面。它会创建并返回 XMLHttpRequest 对象。该方法有一个键/值对形式对象的参数，它会被用于初始化和处理请求。

.ajax(键/值对的对象)

在这个解决方案中，我们会使用如下几个键/值对。

- Type：我们将会针对请求来使用它，它定义了 HTTP 方法的字符串，或者是 GET，或者是 POST。默认类型是 GET 方法。
- url：这是包含了网页地址的字符串，我们想要向它发送请求。
- Data：这是一个 map 对象或者字符串，我们会将它和请求一起发送给服务器。
- Success：这是一个回调函数，如果发送给服务器的请求成功了，那么它就会被执行。（从服务器）返回的数据会被赋给这个回调函数的参数。

你可以在下面的地址查看所有这些键/值对的选项：http://docs.jquery.com/Ajax/jQuery.ajax#options 。

在 jQuery 代码中，我们会取得用户在文本输入框（class 为 uname）中输入的名字，并将其存储在变量 name 中。我们还定义了变量 data，它会存储一对 uname=name，其中 name 是用户输入的名字。这个 data 变量会被传送给服务器，并且被指定给脚本文件 welcome.php（假设在本解决方案中它已经存在于服务器上了），以得到响应。

我们通过 ajax()方法调用了请求。在 ajax()方法中，我们指定将要使用的请求方法是 POST 类型，并指定了将要在服务器上执行的脚本文件（welcome.php）。要传递给脚本文件的参数包含在字符串 data(uname=name).welcome.php 中，在处理完传入数据之后，我们的解决方案会生成输出，它会由回调函数中的 html 参数指定的 JavaScript 文件来接收。

然后 html 的内容会赋给 id 为 message 的 div 元素，从而向用户显示响应（由服务器上的脚本文件生成）。我们在 click 事件中返回 false，从而抑制浏览器默认的点击行为；即我们想要通过 jQuery 代码采取指定的动作而不是默认的动作。

Welcome.php 脚本文件会从$_POST 数组中取得元素 uname，并将其存储在变量$name 中。然后字符串 Welcome $name 会返回给 jQuery 代码作为服务器做出的响应，其中$name 是用户输入的名字。HTML 文件会生成包含有标签、文本输入框和提交按钮的输出，如图 8-1 所示：

如果我们在文本输入框中输入了名字，并选择提交按钮，那么就会得到由服务器生成的欢迎信息，如图 8-2 所示。

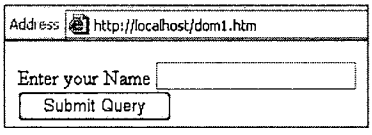


图 8-1 输入用户姓名的屏幕

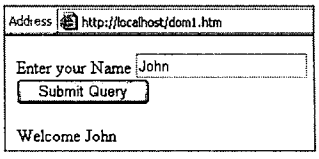


图 8-2 选择提交查询按钮的时候显示的欢迎信息

1. 使用请求的GET HTTP方法

如果你想要使用 GET HTTP 请求载入来自于服务器的数据，那么可以有两种选择：一种是在 ajax()方法中将 type 键的值设为 GET(或者不定义这个键值，因为默认的请求方法就是 GET)，另一种是使用$.get()方法。现在让我们依次试验这两种方法。首先是将 type 键值设置为 GET：

```
$(document).ready(function() {
  $('#submit').click(function () {
    var name = $('.uname').val();
    var data = 'uname=' + name;
    $.ajax({
      type:"GET",
      url:"welcome.php",
      data: data,
      success: function (html) {
        $('#message').html(html);
      }
    });
    return false;
  });
});
```

我们可以看到除了对 type 键值的修改之外，在 JavaScript 文件中没有做任何其他的修改。当你想要使用 GET HTTP 请求从服务器载入数据的时候，这个 type 键值会帮助你。为了让服务器上的脚本从$_GET 数组取得数据，我们需要在 welcome.php 文件中做以下调整。

```
<?php
$name = $_GET['uname'];
echo "Welcome ".  $name;
?>
```

你会看到，数据会从$_GET 数组而不是$_POST 数组中取得的。现在让我们使用$.get()方法来使用 GET HTTP 请求。这个方法向服务器发送 GET 请求来取得数据。它会执行指定 URL 的脚本，并在其中带有传递的参数（如果有的话）：

```
$.get(url, parameters, callback)
```

这些参数如下所示。

- url 是你想要通过 GET 方法执行的服务端脚本。
- parameters 是你想要传递给服务端脚本处理的键/值对。
- callback 是请求完成的时候所要调用的函数。它包含两个参数，第一个是来自于服务端脚本的响应，另一个参数是执行的状态，那么让我们修改上面的 JavaScript 来使用$.get()方法。你的代码可能像下面这样：

```
$(document).ready(function() {
  $('#submit').click(function () {
    var name = $('.uname').val();
    var data = 'uname=' + name;
    $.get(
      "welcome.php",
      data,
      function (html) {
        $('#message').html(html);
      }
    );
    return false;
  });
});
```

你可以看到$.get()方法调用了服务器上的 welcome.php 文件，并将 uname=name 作为参数传递给它（其中 name 中存储的是用户输入的名字），而回调函数通过将它赋给 id 为 message 的 div 元素来显示（来自于服务器的）响应。在上面我们会得到相同的输出效果，如图 8-1 和图 8-2 所示。

2. 创建POST请求

为了向服务器发出 POST 请求，我们使用了$.post()方法，一般你会在想要修改存储在服务器上的信息时用到它。它使用 POST HTTP 请求将数据获取到服务器上。它会执行指定 URL 的脚本，其中带有传递的参数（如果有的话）。这个函数还允许指定单独的回调函数，它会在请求完成的时候执行。下面是方法和参数的形式：

```
$.post(url, parameters, callback)
```

参数的说明如下。

- url 是我们想要向其发送请求（通过 POST 方法）的服务器端脚本。
- parameters 是我们想要传递给服务端脚本处理的键/值对。

- callback 是请求完成的时候所要调用的函数。它包含两个参数，第一个是来自于服务端脚本的响应，第二个参数是执行的状态。

这样，我们可以对上面的 JavaScript 文件进行修改以使用$.post()方法，如下所示：

```
$(document).ready(function() {
  $('#submit').click(function () {
    var name = $('.uname').val();
    var data = 'uname=' + name;
    $.post(
      "welcome.php",
      data,
      function (html) {
        $('#message').html(html);
      }
    );
    return false;
  });
});
```

你可以看到$.post()方法调用了服务器上的 welcome.php 文件，并将 uname=name 作为参数传递给它（其中 name 中存储的是用户输入的名字），而回调函数通过将它赋给 id 为 message 的 div 元素显示了响应（来自于服务器）。在上面我们会得到相同的输出，如图 8-1 和图 8-2 所示。

最后，请你记住，服务器端脚本 welcome.php 必须访问来自于$_POST 数组而不是$_GET 数组的数据，因为这次我们是通过 POST 方法发送的请求。

```
<?php
$name = $_POST['uname'];
echo "Welcome ". $name;
?>
```

8.2 执行认证

问题描述

要求用户在登录表单中输入用户名和密码。如果用户输入的用户名和密码与服务端脚本中的内容相匹配，他就会得到一条欢迎信息，否则会提示他是“未授权的用户”。

解决方案

让我们创建一个 HTML 文件，它显示有两个标签：Enter your Name 和 Enter your Password，还有两个文本输入框，一个提交按钮和一个空的 div 元素。HTML 文件应该像下面这样：

```
<body>
  <form>
    <label>Enter your Name</label>
    <input type="text"  name="uname" class="uname"/>  <br/>
    <label>Enter your Password</label>
```

```
    <input type="password"  name="password" class="passwd"/>  <br/>
    <input type="submit" id="submit"/>
  </form>
  <div id="message"></div>
</body>
```

你会看到HTML文件中包含两个class被赋值为uname和passwd的文本输入框，对class赋值的目的在于要通过jQuery代码访问它们。在表单下面是一个id为message的div元素，我们会使用它来显示服务器端脚本产生的响应。

jQuery代码可能像下面这样：

```
$(document).ready(function() {
  $('#submit').click(function () {
    var name = $('.uname').val();
    var pwd = $('.passwd').val();
    var data='uname='+name+'&password='+pwd;
    $.ajax({
      type:"GET",
      url:"logincheck.php",
      data:data,
      success:function(html) {
        $("#message").html(html);
      }
    });
    return false;
  });
});
```

服务器上的脚本文件logincheck.php会校验用户输入的用户名和密码，代码如下所示：

```
<?php
$name = trim($_GET['uname']);
$pswd = trim($_GET['password']);
if(($name=="guest") && ($pswd=="jquery"))
  echo "Welcome ".  $name;
else
  echo "Sorry you are not authorized";
?>
```

知其所以然

在jQuery代码中，我们首先向id为submit的提交按钮添加了click事件。在这个事件中，我们会取得用户在class分别为uname和passwd的文本输入框中输入的用户名和密码，并分别将其存储在变量name和pwd中。

然后我们会定义变量data，其中会存储uname=name&password=pwd，这里name表示用户输入的用户名，而pwd表示用户输入的密码。这个data变量会被发送给服务器，然后被分配给脚本文件logincheck.php文件，从而进行用户名和密码的验证，并相应地生成响应。

在ajax()方法中，我们指定将要使用的请求方法是GET，而在服务器上将会执行的脚本文件的名称为logincheck.php，并且将要传送给脚本文件的参数包含在字符串data中。

在 logincheck.php 处理完发送给它的数据之后，就会产生输出，这会由回调函数的参数 html 指向的 JavaScript 文件来接收。然后从脚本文件返回的响应会赋给 id 为 message 的 div 元素，从而向用户显示响应信息。

最终，我们在 click 事件中返回 false，从而抑制浏览器默认的点击行为，即我们想要通过 jQuery 代码采取指定的动作而不是默认的动作。

在 logincheck.php 中，我们会看到发送的参数是通过$_GET 数组来访问的，即它会取得存储在$_GET 数组中的 uname 和 password 元素的信息，并分别存储在变量$name 和$pswd 中。然后，我们会检查用户输入的用户名和密码是否分别为 guest 和 jquery，如果是的话就返回 Welcome name 信息，否则就发送 Sorry you are not authorized 信息。

在执行过程中，HTML 文件会显示一个登录表单来输入用户名和密码。如果用户输入的用户名和密码与我们在服务器端脚本文件 logincheck.php 中的不匹配，那么用户会得到消息 Sorry you are not authorized 作为来自于服务器的响应，如图 8-3 所示：

如果输入的用户名和密码分别是 guest 和 jquery，那么会显示欢迎信息给用户，如图 8-4 所示。

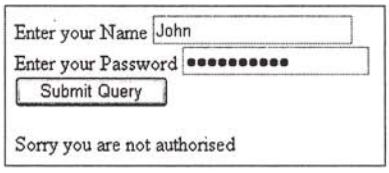


图 8-3 当输入错误的用户名和密码时显示的消息

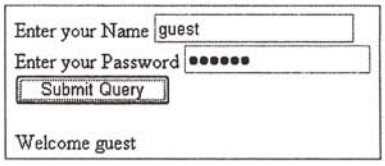


图 8-4 当输入正确的用户名和密码时显示的欢迎信息

8.3 验证用户名

问题描述

要求用户输入用户名，并且要使用服务器端的脚本来确定字段不为空。如果在用户名输入框中没有输入任何字符，那么他就会得到错误信息。

解决方案

让我们首先创建一个 HTML 文件，它显示有一个标签 Enter your Name，一个文本输入框以及一个提交按钮。HTML 文件会像下面这样：

```
<body>
  <form>
    <span class="label">Enter your Name</span>
    <input type="text" name="uname" class="uname"/> <span class="error"> </span><br>
    <input type="submit" id="submit"/>
  </form>
</body>
```

你会看到文本输入框的 class 是 uname，这样做的目的是让我们可以通过 jQuery 代码来访问它。在文本输入框后面是一个空的 span 元素，class 为 error。这个 span 元素中的文字会从生成响应的服务器处得到——前提是用户没有在文本输入框中输入文字。

为了给文本输入框标签指定宽度，为错误信息指定颜色，我们需要在样式表文件中编写下面的样式规则：

```
.label {float: left; width: 120px; }
.uname {width: 200px; }
.error { color: red; padding-left: 10px; }
#submit { margin-left: 125px; margin-top: 10px;}
```

调用服务端脚本的jQuery代码如下，它将用户输入的用户名作为参数传递给 validateuser.php，如果用户没有输入，就会显示错误信息。

```
$(document).ready(function() {
  $('.error').hide();
  $('#submit').click(function () {
    var name = $('.uname').val();
    var data='uname='+name;
    $.ajax({
      type:"POST",
      url:"validateuser.php",
      data:data,
      success:function(html) {
        $('.error').show();
        $('.error').text(html);
      }
    });
    return false;
  });
});
```

脚本文件 validateuser.php 位于服务器上，它的目的是要检查用户是否输入了用户名。代码如下：

```
<?php
if($_POST['uname'] == '')
{
  echo "This field cannot be blank";
}
?>
```

使用正则表达式校验用户名

在上面的方案中，当用户想要输入用户名的时候，他们可以输入任意的文字，包括符号。让我们来改善服务端的脚本，从而它只会接受包含字母和下划线的用户名。为了达到目的，我们可以像下面这样修改脚本文件 validateuser.php：

```
<?php
$name = $_POST['uname'];
if (!eregi("^[a-z0-9_]+$", $name))
{
  echo "Invalid User name";
```

```
}
?>
```

知其所以然

如你在上面所见，在 jQuery 代码中，我们首先隐藏了 class 为 error 的 span 元素，并向 id 为 submit 的提交按钮添加了 click 事件。然后我们会取得用户在文本输入框（class 为 uname）中输入的名字，并将其存储在变量 name 中。

变量 data 中会存储一个字符串 uname=name，其中 name 代表的是用户输入的用户名。这个 data 变量会被传送给服务器，并被指定给脚本文件 validateuser.php（假设在本解决方案中它已经存在于服务器上了），以确定用户名是否为空。

我们通过 ajax()方法来调用请求，其中我们指定将要使用的请求方法是 POST，而在服务器上将会执行的脚本文件的名称为 validateuser.php，我们还指定将要传送给脚本文件的参数包含在字符串 data 中。

在处理完传递过来的数据之后，我们在 validateuser.php 中的代码会产生输出，这会由回调函数的参数 html 所指向的 JavaScript 文件来接收。然后脚本文件（存储在 html 中）返回的响应会赋给 class 为 error 的 span 元素，从而向用户显示由服务器上的脚本文件生成的错误响应。然而，在将响应 html 赋给 class 为 error 的 span 元素之前，我们先要使其可见，因为在代码开始时我们将它隐藏了。最终，我们在 click 事件中返回 false，从而抑制浏览器默认的点击行为；即我们想要通过 jQuery 代码采取指定的动作而不是默认的动作。

你还能够看到上面的 PHP 脚本是如何通过$_POST 数组取得通过参数 data 传递过来的用户名，并且在发现它是空的时，返回错误消息 This field cannot be blank，这个消息会显示在 class 为 error 的 span 元素中。如果用户名不是空的，那么就不会有任何响应。

当文本输入框为空时，如果我们选择提交按钮，就会看到显示出来的错误信息，如图 8-5 所示。

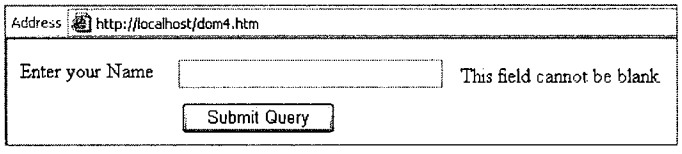


图 8-5 当用户名为空时显示的错误信息

如果输入了合法的用户名，用户名就会被接受，而不会显示任何错误信息，如图 8-6 所示。

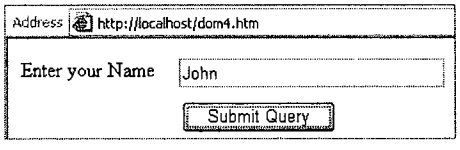


图 8-6 用户名被接受，不显示任何错误信息

当使用正则表达式的时候，你能够看到我们使用 eregi()方法来指定验证用户名的正则表达式。正则表达式中的^和$分别代表的是“开头”和“末尾”。我们在第 4 章表单验证中解释过正则表达式。

“+”符号意味着一次或者多次。这样完整的正则表达式表示的是用户名可能会包含从 a 到 z 的任意字符、从 0 到 9 的任意数字以及下划线。这些字符可以在开头或者末尾显示一次或多次。这样，如果我们在用户名中输入了下划线之外的符号，就会得到“非法的用户名”错误，如图 8-7 所示。

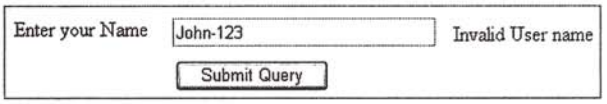


图 8-7　当输入非法的用户名时显示的错误信息

如果用户名只包含字母、数字和下划线，就会被系统所接受，而不会显示任何错误信息，如图 8-8 所示：

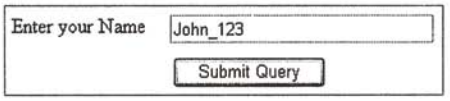


图 8-8　如果用户名只包含字母、数字和下划线，就会被接受而不显示任何错误信息

8.4　验证邮件地址

问题描述

让用户输入他的邮件地址，并使用服务器端的脚本来确定邮件地址是否有效。也就是说，它必须包含字母、数字，没有“-”和“_”之外的符号，并且它必须包含符号“.”和“@“。如果邮件地址无效，就会向用户显示错误信息。

解决方案

让我们创建一个 HTML 文件，它显示有一个标签 Enter your Name，一个文本输入框以及一个提交按钮。HTML 文件会像下面这样：

```
<body>
  <form>
    <span class="label">Enter email address</span>
    <input type="text" name="emailadd" class="emailadd"/>
    <span class="error"></span><br/>
    <input type="submit" id="submit"/>
  </form>
</body>
```

在上面的 HTML 代码中，我们能够看到文本框的 class 名为 emailadd，这样做的目的是

为了通过 jQuery 代码访问它。在文本输入框后面是一个空的 span 元素，class 名为 error。这个 span 元素中的文字会从生成响应的服务器处得到——前提是用户输入了无效的 email 地址。

为了指定标签、文本输入框的宽度以及错误信息的颜色，我们会使用解决方案 8.3 中编写的样式表文件中的样式规则。

调用服务端脚本文件 validatemail.php，用来验证用户输入的 email 地址的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.error').hide();
  $('#submit').click(function () {
    var em = $('.emailadd').val();
    var data='emailadd='+em;
    $.ajax({
      type:"POST",
      url:"validatemail.php",
      data:data,
      success:function(html) {
        $('.error').show();
        $('.error').text(html);
      }
    });
    return false;
  });
});
```

服务器上的脚本文件 validatemail.php 是我们设计用来检查用户的 email 地址是否有效的：

```
<?php
$emid = $_POST['emailadd'];
if (!eregi("^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*{\.[a-z]{2,3})$",
$emid))
{
  echo "Invalid email address";
}
?>
```

一次验证两个字段

现在，让我们将两个秘诀组合在一起，这样通过 Ajax 就能够一次验证多个字段。我们会创建一个 HTML 文件，它显示有两个标签 Enter user id 和 Enter email address，两个用来分别输入 userid 和 email 地址的文本输入框，还有一个提交按钮。HTML 文件会像下面这样：

```
<body>
    <form>
        <span class="label">Enter user id</span>
        <input type="text"  name="userid" class="userid"/>
        <span class="usrerror"> </span><br/>
        <span class="label">Enter email address</span>
        <input type="text"  name="emailadd" class="emailadd"/>
        <span class="emerror"> </span><br/>
        <input type="submit" id="submit"/>
```

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

为了指定标签和文本输入框的宽度，以及错误信息的颜色，我们会使用样式规则。因此让我们在外部样式表文件 style.css 中编写下面的样式规则：

```
.label {float: left; width: 120px; }
.userid {width: 200px; }
.emailadd {width: 200px; }
.usrerror { color: red; padding-left: 10px; }
.emerror { color: red; padding-left: 10px; }
#submit { margin-left: 125px; margin-top: 10px;}
```

调用服务端脚本 validatedata.php，从而验证用户输入的 userid 和邮件地址的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.usrerror').hide();
  $('.emerror').hide();
  $('#submit').click(function () {
    var uid = $('.userid').val();
    var em = $('.emailadd').val();
    var data='userid='+uid+'&emailadd='+em;
    $.ajax({{
      type:"POST",
      url:"validatedata.php",
      data:data,
      success:function(html) {
        var twomsgs = html.split("\n");
        for ( var i in twomsgs )
        {
          var errmsg = twomsgs[i].split("|");
          if(errmsg[0]=='user')
          {
            $('.usrerror').show();
            $('.usrerror').text(errmsg[1]);
          }
          if(errmsg[0]=='email')
          {
            $('.emerror').show();
            $('.emerror').text(errmsg[1]);
          }
        }
      }
    });
    return false;
  });
});
```

我们会使用服务器上的脚本文件 validatedata.php 来验证 userid 和邮件地址，如下所示：

```
<?php
$errors = null;
$name = $_POST['userid'];
if (!eregi("^[a-z0-9_]+$", $name))
{
```

```
  $errors .= "user|Invalid User id\n";
}
else
{
  $errors .= "user|\n";
}
$emid = $_POST['emailadd'];
if (!eregi("^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*(\.[a-z]{2,3})$",
$emid))
{
  $errors .= "email|Invalid email address\n";
}
else
{
  $errors .= "email|\n";
}
echo $errors;
?>
```

知其所以然

首先我们会隐藏 class 为 error 的 span 元素，并向 id 为 submit 的提交按钮添加 click 事件。然后我们会取得用户在文本输入框（class 为 emailadd）中输入的邮件地址，并将其存储在变量 em 中。

变量 data 中存储了字符串 emailadd=em，其中 em 代表的是用户输入的邮件地址。

然后这个 data 变量会被传送给服务器，并将被指定给脚本文件 validatemail.php（假设在本解决方案中它已经存在于服务器上了），以确定 email 地址是否有效。

接下来，我们会通过 ajax()方法来调用请求，其中我们指定将要使用的请求方法是 POST，而在服务器上将会执行的脚本文件的名称为 validatemail.php，我们还指定将要传送给脚本文件的参数包含在字符串 data 中（当然，其中包含字符串 emailadd=em）。

然后，在处理数据并生成输出之后，输出会由 validatemail.php 的回调函数的参数 html 中指定的 JavaScript 文件所接收。然后脚本文件(存储在 html 中)返回的响应会赋给 class 为 error 的 span 元素，从而向用户显示错误响应，而那是由服务器上的脚本文件生成的。然而，在将响应——这里是 html——赋给 class 为 error 的 span 元素之前，我们首先要使其可见，因为在代码开始时我们就将它隐藏了。

我们在 click 事件中返回 false，从而抑制浏览器默认的点击行为；这样做是因为我们想要通过 jQuery 代码采取指定的动作而不是默认的点击动作。

在 PHP 脚本中，我们会看到有 eregi()方法，它被用于为验证 email 地址指定正则表达式。正则表达式中的^和$分别代表的是“开头”和“结尾”。“+”符号意味着一次或多次，而“*”符号意味着 0 次或多次。

让我们来分解一下正则表达式，从而更好地理解它。

- [_a-z0-9-]+ 意味着我们的 email 地址中可以由多个“_”（下划线）、从 a 到 z 的字符、从 0 到 9 的数字以及“-”(中横线)组成。

- (\.[_a-z0-9-]+)* 意味着 email 地址允许在“_”下划线、字母数字、“-”中横线之后没有或者有多个点（.）。
- @[a-z0-9-]+意味着邮件地址必须在一个或多个字母数字、“-”（中横线）之后有一个@符号。
- (\.[_a-z]{2, 3})$ 意味着 email 地址必须以点（.）加上两到三个从 a 到 z 的字符结束，也就是说，在点(.)之后的字符数不能少于两个，也不能大于 3 个。

这样，如果我们输入的 email 地址与上面所提供的正则表达式不匹配，那儿我们就会得到错误信息：无效的 email 地址。如图 8-9 所示。

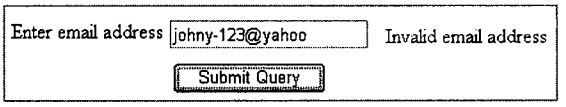


图 8-9　如果输入了无效的 email 地址，就会显示错误信息

现在如果我们输入有效的地址，它遵循服务端的脚本文件 validatemail.php 中所定义的正则表达式，它就会被接受，而不会显示任何错误信息，如图 8-10 所示。

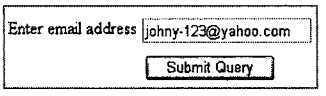


图 8-10　如果 email 地址有效，它就会被接受，而不显示任何错误信息

这样，在我们的 HTML 文件中有两个文本输入域，class 名分别为 userid 和 emailadd，目的是为了让我们能够通过 jQuery 代码访问它们。在每个文本输入框之后，我们都放置了空的 span 元素，它们会被用来显示错误信息。span 元素是用来在输入无效的用户 id 时显示错误信息的；在这种情况下，span 元素的 class 被赋值为 usrerror，并且还有一个显示与无效 email 地址相关的错误信息的 span 元素，它的 class 名称被赋值为 emailadd. 这些 span 元素的文字会从服务端脚本做出的响应赋值。

在样式表文件中，我们将类型选择器 label 的 float 属性设置为 left，从而下一个项目（文本输入框）会显示在它的右侧，width 属性被设置为 120px，从而给显示标签留出足够的空间。类选择器 userid 中将 width 属性设置为 200px，它定义了输入 userid 的文本输入框的大小。类似地，类选择器 emailadd 中将 width 属性设置为 200px，它定义了输入 email 地址的文本输入框的大小。

类选择器 usrerror 和 emerror 中将 color 属性设置为红色以突出显示，将左侧的 padding 设置为 10px，以保持各自与文本输入框之间的距离。最后，类选择器 submit 中将 margin-left 和 margin-top 属性分别设置为 125px 和 10px，这设置了与浏览器的左侧边界和与文本输入框之间的距离，因为我们希望提交按钮显示在文本输入框之下。

在 jQuery 代码中，我们首先隐藏了 class 为 usrerror 和 emerror 的 span 元素。当输入无效数据的时候我们就会显示它们。然后我们为 id 为 submit 的提交按钮添加了 click 事件。

接下来，我们会取得用户在文本输入框（class 分别为 userid 和 emailadd）中输入的用户 id 和邮件地址，并将它们存储在变量 uid 和 em 中。为了将它们发送给服务器，我们定义了变量 data，其中会存储一个字符串 userid=uid&emailadd=em，其中 uid 和 em 分别代表的是用户输入的 userid 和邮件地址。这个 data 变量会被传送给服务器，并将被指定给脚本文件 validatedata.php（假设在本解决方案中它已经存在于服务器上了），以验证我们的用户 id 和邮件地址。

接下来我们通过 ajax()方法来调用请求，其中我们指定将要使用的请求方法是 POST，而在服务器上将会执行的脚本文件的名称为 validatedata.php，我们还指定将要传送给脚本文件的参数包含在字符串 data 中（包含字符串 userid=uid&emailadd=em）。

在处理完传递过来的数据之后，脚本文件（服务器上的 validatedata.php）会产生输出，这会由回调函数的参数 html 指定的 JavaScript 文件来接收。

脚本文件返回的响应（存储在 html 中）会在换行符（\n）处切分（假设脚本文件生成的两条错误信息是用换行符"\n"分隔的），而结果会存储在变量 twomsgs 中。变量 twomsgs 是拥有两个元素的数组，其中包含与用户 id 和邮件地址相关的错误信息。然后我们会依次按照“|“来切分 twomsgs 数组的每个元素（以格式“字段|错误消息”包含数据）并将结果存储在另一个数组 errmsg 中。结果，errmsg 数组的第一个元素会包含"user"或者"email"，而 errmsg[1]中会包含相应的错误信息。

如果 errmsg[0]中包含的是"user"，那么我们就会将 errmsg[0]的内容（包含用户 id 的错误信息）赋给 class 为 usrerror 的 span 元素，从而向用户显示错误信息。在将错误信息赋给 class 为 usrerror 的 span 元素之前，我们首先会使其可见。

如果 errmsg[0]中包含的是"email"，我们就会将 errmsg[1]的内容（包含邮件地址的错误信息）赋给 class 为 emerror 的 span 元素，从而向用户显示错误信息。在将错误信息赋给 class 为 emerror 的 span 元素之前，我们首先会使其可见。我们会在 click 事件中返回 false，从而抑制浏览器默认的点击行为，因为我们希望在 PHP 代码中采用的是我们通过 jQuery 代码指定的动作，而不是默认的点击事件。我们会访问 JavaScript 文件通过参数 data 发送过来的用户 id 和邮件地址，其中会包含来自于$_POST（因为数据是通过 HTTP POST 请求发送的）数组、格式为 userid=uid&emailadd=em 的信息，并将其分别存储在变量$name 和$emid 中。

然后借助于 eregi()方法，我们会为用户 id 和邮件地址指定正则表达式，这在上面我们已经讨论过。如果用户 id 与指定的正则表达式不匹配——意味着用户 id 是无效的——那么我们会显示文本 user|Invalid User id，这是通过将它赋给$errors 变量而达到的。符号“|”后面的内容是被设计为在 span 元素中显示错误信息的，该元素存在于文本输入框之后。如果用户 id 是有效的（即它与我们的正则表达式匹配），那么文本 user|就会赋给$errors 变量。这样，在“|”后面没有提供任何错误信息，从而对于有效的用户 id 不会显示任何错误信息，这正是我们所期望的。对于邮件地址我们会重复同样的工作：如果它是有效的，那么我们就会将变量$errors 赋值为 email|Invalid email address。

这样，如果我们输入无效的用户 id 和邮件地址（与我们在服务端脚本文件中所提供的正则表

达式不匹配），我们就会看到错误信息，如图 8-11 所示。

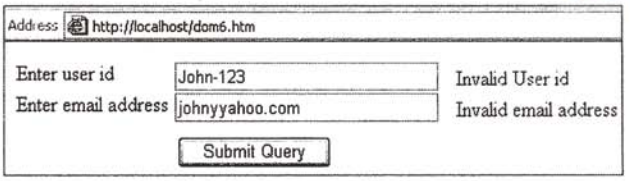


图 8-11　显示用户 id 和邮件地址的错误信息

如果用户 id 是有效的，错误信息就会消失，而只显示与邮件地址相关的错误信息，如图 8-12 所示。

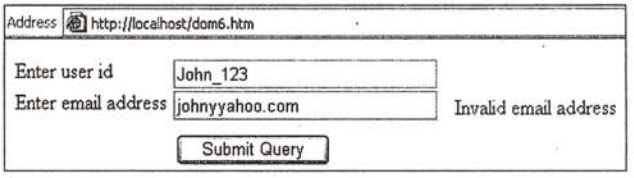


图 8-12　显示针对无效邮件地址的错误信息

最终，如果输入的用户 id 和邮件地址都是有效的，它们就都会被接受，而不会显示任何错误信息，如图 8-13 所示。

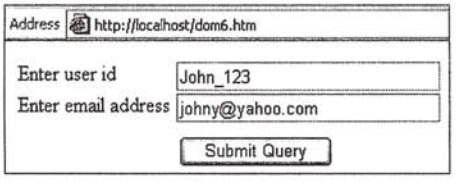


图 8-13　当用户 id 和邮件地址都有效时，不显示任何错误信息

8.5　使用自动完成

问题描述

让用户在文本输入框中输入名字，当用户输入名字的第一个字符的时候，弹出建议框，显示所有以输入的字符开头的名字。让显示在建议框中的名字拥有鼠标悬停的效果——即当鼠标指针在上面悬停的时候，它会突出显示——并且被点击的名字会被插入到文本输入框中。

解决方案

让我们创建一个 HTML 文件，它显示有一个标签 `Enter userid` 和一个文本输入框。在文本输入框下面，我们会创建两个空的 `div` 元素，并将 class 分别赋值为 `listbox` 和 `nameslist`。

HTML 文件会像下面这样：

```
<body>
    <form>
        <span class="label">Enter user id</span>
        <input type="text"  name="userid" class="userid"/>
        <div class="listbox">
            <div class="nameslist">
            </div>
        </div>
    </form>
</body>
```

class 为 listbox 的 div 元素为用户显示列表框，而 class 为 namelist 的 div 元素会被用于显示带有悬停效果的名字。我们在样式表文件 style.css 中定义了将要应用给两个 div 元素的样式规则，如下所示：

```
.listbox {
position: relative;
left: 10px;
margin: 10px;
width: 200px;
background-color: #000;
color: #fff;
border: 2px solid #000;
}

.nameslist {
margin: 0px;
padding: 0px;
list-style:none;
}

.hover {
background-color: cyan;
color: blue;
}
```

下面的 jQuery 代码会在用户在文本输入框中输入第一个字符的时候显示我们的建议框，并且自动将（在建议框中）点击的名字显示在文本输入框中：

```
$(document).ready(function() {
  $('.listbox').hide();
  $('.userid').keyup(function () {
    var uid = $('.userid').val();
    var data='userid='+uid;
    $.ajax({
      type:"POST",
      url:"autocomplete.php",
      data:data,
      success:function(html) {
        $('.listbox').show();
        $('.nameslist').html(html);
        $('li').hover(function(){
```

```
            $(this).addClass('hover');
          },function(){
            $(this).removeClass('hover');
          });
          $('li').click(function(){
            $('.userid').val($(this).text());
            $('.listbox').hide();
          });
        }
      });
      return false;
    });
});
```

服务器上的脚本文件 autocomplete.php 是被设计用来发送名字列表作为响应的，它应该像下面这样：

```
<?php
echo '<li>Jackub</li>';
echo '<li>Jenny</li>';
echo '<li>Jill</li>';
echo '<li>John</li>';
?>
```

取得从数据库得到的名字

上面的服务端脚本的缺陷在于它只能返回开头字母为"j"的名字。即便是用户输入了其他字母，它还是会生成开头字母为"j"的名字。我们希望它能够生成开始字母和用户输入的字母一致的名字。想要达到这个目的，我们需要创建数据库和表，其中包含有成千上万个名字。我已经创建了名为 autofill（使用 mySQL 服务器）的数据库和一个表，其中包含有名字信息。表中包含 name 字段，我们会在其中输入一些开头字符为 a 到 z 的名字……

```
<?php
$name = $_POST['userid'];
$connect =
  mysql_connect("localhost", "root", "mce") or die("Please, check your server
connection.");
mysql_select_db("autofill");
$query = "SELECT name FROM info WHERE name LIKE '$name%'";
$results = mysql_query($query) or die(mysql_error());
if($results)
{
  while ($row = mysql_fetch_array($results)) {
    extract($row);
    echo '<li>' . $name. '</li>';
  }
}
?>
```

知其所以然

在样式表文件中，样式规则.listbox 将 position 属性设置为 relative，从而使它的位置

与其容器（当未定义时通常是浏览器窗口）相关。left 属性被设置为 10px，这使得建议框显示在离浏览器窗口左侧边界 10px 的位置。margin 属性被设置为 10px，使得它与上面的文本输入框之间保持 10px 的距离。width 属性被设置为 200px，使得名字（将要显示在这个框中）占据 200px 的宽度。background-color 和 color 属性分别被设置为#000 和#fff，使得弹出框的背景色是黑色，而名字以白色显示。border 属性被设置为 2px solid #000，使得在建议框中的名字周围有粗细为 2px 的实线框。

查看我们的 jQuery 代码，我们会看到其中在开始时隐藏了 class 为 listbox 的 div 元素，因为我们希望只有在用户在文本输入框中输入了文字的时候才显示它。我们为 class 是 userid 的文本输入框添加了 keyup()事件，从而当用户放开键盘上的键时，事件处理函数就会被触发。

然后我们会取得用户在 class 为 userid 的文本输入框中输入的字符，并将其存储在变量 uid 中；然后定义变量 data，其中会存储字符串 userid=uid，这里的 uid 代表的是用户输入的（名字的）第一个字符。这个 data 变量会被传送给服务器，并将被指定给脚本文件 autocomplete.php（假设在本解决方案中它已经存在于服务器上了），以向用户显示建议框。

我们通过 ajax()方法来调用实际的请求，其中我们会指定将要使用的请求方法是 POST，而在服务器上将会执行的脚本文件的名称为 autocomplete.php，同时我们还指定将要传送给脚本文件的参数包含在字符串 data 中，它包含字符串 userid=uid。在处理完传递过来的数据之后，脚本文件（服务器上的 autocomplete.php）会产生输出，这会由回调函数的参数 html 指定的 JavaScript 文件来接收。然后我们会使 class 为 listbox 的 div 元素可见，并在其中显示名字。我们还会将脚本文件（存储在 html 中）返回的响应赋给 class 为 nameslist 的 div 元素。然后脚本文件会以列表项的形式返回一些名字。

接下来我们会为脚本文件返回的名字（以列表项的形式）附加鼠标悬停事件。在这个事件中，当鼠标指针悬停在名字（列表项的形式）上的时候，我们会对其应用在样式规则 hover 中定义的样式属性，使得它们的背景色为青绿色，文字为蓝色。当鼠标指针移出名字的时候，我们还会从上面移除样式规则 hover 中定义的样式属性。

我们还向以列表项形式存在的名字添加了 click 事件，使得点击的名字显示在 class 为 userid 的文本输入框中，并且隐藏包含名字的建议框。我们在 click 事件中返回 false，从而抑制浏览器默认的点击行为，因为我们想要通过 jQuery 代码采取指定的动作而不是默认的动作。

我们可以看到上面的 PHP 代码只是生成了一些列表项形式的名字，它们将会被传送回 JavaScript 文件。脚本会发送一些开头字母为"j"的名字，假设用户只会输入这个字母（作为第一个字母）。为了让这段脚本生成以用户任意输入的字符开头的名字（从 a 到 z），我们需要带有大量名字的数据库。稍后我们会看一下与这样的数据库的交互过程，但是现在，在执行过程中，我们会看到文本输入框的左边有一个标签 Enter userid，当用户键入字符（任何字符）的时候，服务器发送回来的名字都会以建议框的形式显示出来，如图 8-14 所示。

我们可以让鼠标指针悬停在任意一个名字上。你可能还会发现鼠标悬停的名字会突出显示，如图 8-15 所示。

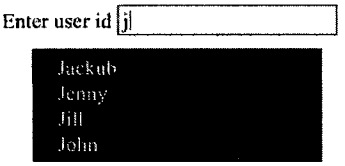


图 8-14　在文本输入框中输入单个字符的时候会显示建议框

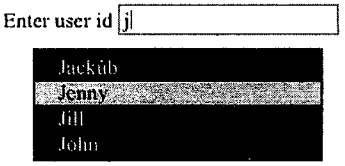


图 8-15　当鼠标悬停时，可选的选项会突出显示

当点击任意一个名字的时候，它会显示在文本输入框中，如图 8-16 所示。

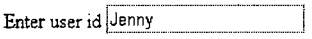


图 8-16　点击的选项显示在文本输入框中

让我们来研究一下带有数据库的解决方案。首先我们会借助于`$_POST` 数组（因为请求是通过 HTTP POST 方法发送的）取得 JavaScript 文件（通过 `data` 参数）发送过来的字符，并将其存储在变量`$name` 中。再次我跳过了 DB 连接的代码，因为那会根据你的设置而有所不同。其中最有趣的部分在于编写 SQL 查询，用来从 `info` 表取得所有以存储在`$name`（它是用户在文本输入框中键入的）中的字母开头的名字。

我们会执行 SQL 查询，而作为查询的输出所返回的记录会存储在`$results` 数组中。然后我们会使用 `while` 循环依次取得每个存储在`$results` 数组中的名字，并将每个名字包含在`<li>` 标签中之后发送给 JavaScript 文件，因为我们想要生成列表项形式的名字。

注意　*我们还可以针对每个用户会话缓存结果，以及在缓存中进行过滤。*

现在，我们就得到了以我们在文本输入框中输入的任意字符开头的名字。这样，如果我们键入字符 `k`，那么就会得到所有以字母 `k` 开头的名字，如图 8-17 所示。

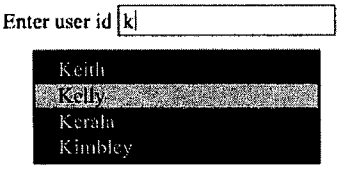


图 8-17　建议框中的选项会根据在文本输入框中输入的第一个字母而改变

8.6　导入 HTML

问题描述

你想要从另一个文件中导入一些 HTML 内容到当前的网页中。

解决方案

让我们先创建一个 HTML 文件，其中包含一个段落元素和一个超链接。我们希望只有在用户选择超链接的时候才导入内容。HTML 文件会像下面这样：

```
<body>
<p>We are going to organize the Conference on IT on 2nd Feb 2010</p>
<a href="abc.com" class="list">Participants</a>
<div id="message"></div>
</body>
```

超链接的 class 被指定为 `list`，从而我们可以通过 jQuery 代码来访问它。在那个超链接下面是名为 `message` 的空白 `div` 元素，我们会使用它来显示导入的 HTML 内容。假设我们想要导入 HTML 内容的文件名称是 namesinfo.htm，并且其中的内容如下所示：

```
<p>The list of the persons taking part in conference</p>
<ul>
<li>Jackub</li>
<li>Jenny</li>
<li>Jill</li>
<li>John</li>
</ul>
<p>We wish them All the Best</p>
```

我们可以看到上面的 HTML 文件中包含两个段落元素和一个列表。导入 HTML 内容的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.list').click(function () {
  $('#message').load('namesinfo.htm');
    return false;
  });
});
```

只导入需要的元素

我们还可以从文件中只导入需要的 HTML 元素。在下面的 jQuery 代码中，我们从 namesinfo.htm 文件中只导入 `list` 元素（不包括段落元素）。

```
$(document).ready(function() {
  $('.list').click(function () {
  $('#message').load('namesinfo.htm li');
    return false;
  });
});
```

知其所以然

在开始理解上面的 jQuery 代码之前，让我们先来看下 `load()` 方法做了些什么，以及在上面的代码中我们是如何使用它的。实质上，这个函数会添加来自于服务器的指定文件，并返回它的 HTML 内容，说明如下。

语法

```
.load(url, parameters, callback function)
```

- url 是定义了服务端脚本文件位置的字符串；
- parameters 包含了我们想要传送给服务端脚本文件以进行某种处理的数据；
- callback 是当请求成功完成时所要执行的函数。

注意　在Internet Explorer中，load()方法会缓存载入的文件。

现在查看一下 jQuery 代码，首先我们为 class 为 list 的超链接添加了 click 事件以及它的事件处理函数，然后会载入文件 namesinfo.htm 的 HTML 内容，并且将其赋给名为 message 的 div 元素。我们在 click 事件中返回 false，从而抑制浏览器默认的点击行为，因为我们想要通过 jQuery 代码采取指定的动作而不是默认的动作。

初始时，我们的网页可能像图 8-18 所示，它包含有一个段落元素和一个带有文本 participants 的超链接：

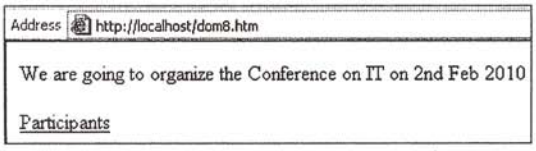


图 8-18　原始的带有超链接的网页

当选择超链接的时候，namesinfo.htm 文件中的内容（包含两个段落元素和列表项）会被导入到当前的网页中，如图 8-19 所示。

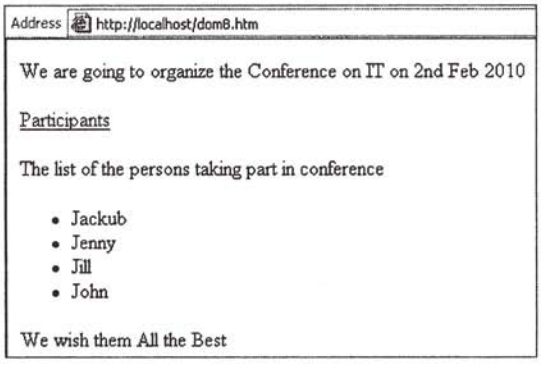


图 8-19　从 html 文件导入内容之后的网页

当只导入需要的元素时，我们会看到在 load()方法中；namesinfo.html 文件后面是 li，表示我们想要只导入文件 namesinfo.htm 中的列表项。结果，当我们选择超链接 Participants 的时候，会发现列表项会被导入到名为 message 的 div 元素中，如图 8-20 所示。

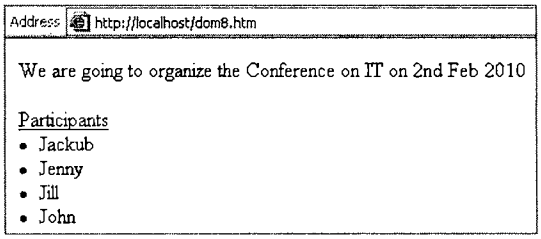


图 8-20 从另一 HTML 文件导入列表项之后的网页

8.7 取得 JSON 数据

问题描述

你有一个存储了一些信息的 JSON 文件。你想要从 JSON 文件中异步地导入信息到当前的网页中。JSON 文件中包含有关于“名称”和“值”的信息。

解决方案

首先我们会创建一个 HTML 文件，其中包含一个段落元素、一个提交按钮和一个空的 div 元素。HTML 文件应该像下面这样：

```
<body>
<p>For information from JSON file click the button given below :<br>
<input type="submit" id="submit"/>
<div id="message"></div>
</body>
```

段落元素只是显示了一条信息；我们想要的是，当用户选择提交按钮的时候，JSON 文件中的信息会导入并显示在名为 message 的 div 元素中。对于这个例子，让我们假设 JSON 文件的名称是 drinkinfo.json，它其中包含的内容如下：

```
[
{"optiontext" : "Tea", "optionvalue" : "Tea"},
{"optiontext" : "Coffee", "optionvalue" : "Coffee"},
{"optiontext" : "Juice", "optionvalue" : "Juice"}
]
```

我们可以看到这里的信息是以两个属性 optiontext 和 optionvalue 的形式存储的。从 JSON 文件中导入信息，并将其以列表的形式显示在当前的网页中的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('#submit').click(function () {
    $.ajax({
      type:"GET",
      url:"drinkinfo.json",
      dataType:"json",
      success: function (data) {
        var drinks="<ul>";
```

```
        $.each(data, function(i,n){
          drinks+="<li>"+n["optiontext"]+"</li>";
        });
        drinks+="</ul>";
        $('#message').append(drinks);
      }
    });
    return false;
  });
});
```

知其所以然

在上面的jQuery代码中，我们向id为submit的提交按钮添加了click事件，并通过ajax()方法调用了请求，其中我们指定想要使用的请求方法的类型是GET，而url是服务器上的JSON文件drinkinfo.json。我们还将dataType键值的值指定为json，表示url包含的是JSON编码形式的数据。

从JSON文件（假设已经存在于服务器上）导入的信息被返回给JavaScript文件，这是以服务器生成的响应的形式返回的，然后由回调函数的参数数据接收。当然，在此data是一个对象数组，其中每个元素都拥有optiontext和optionvalue属性，这与我们要处理的JSON文件的属性是一致的。

接下来我们初始化了变量drinks，并且将其赋予标签<ul>，因为我们想要以未排序列表的形式将响应发送给网页。

在data中接收的信息包括两个属性：optiontext和optionvalue，我们希望只将optiontext属性的内容以列表的形式返回给网页。因此我们使用each()方法来解析存储在data中的每个对象。在each()方法的回调函数中，我们使用了两个参数：i和n，其中i代表对象在data中的索引位置，而n代表包含信息（以optiontext和optionvalue的形式）的对象。我们会依次从n对象中抽取属性optiontext中的信息，并且将其嵌入在
和
标签之间（使得它们以列表项的形式显示），并将它们附加到drinks变量中。

一旦这些工作都已完成，我们就会将内容赋予变量drinks，从而它会包含来自于我们的JSON文件并且存储在optionvalue属性的对象中的信息。现在这是以列表项形式存在的，并且被放置在名为message的div元素中以供显示。最终，我们在click事件中返回false，从而抑制浏览器默认的点击行为；因为我们想要通过jQuery代码采取指定的动作而不是默认的动作。

在执行过程中，我们的网页初始时会显示段落元素和一个提交按钮，如图8-21所示：

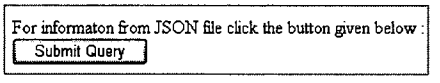


图8-21　带有文本信息和提交按钮的初始页面

当点击提交按钮的时候，存储在optionvalue中的JSON文件的信息会以列表项的形式显示，如图8-22所示。

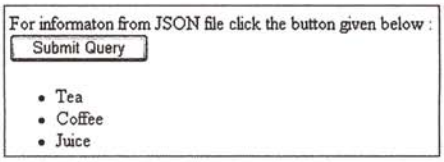


图 8-22 导入的 JSON 数据以列表项的形式显示在网页上

我们还可以使用$.getJSON()方法来简化上述的 jQuery 代码，该方法是特别设计用来从 JSON 文件中取得数据的。这个方法会向指定的 url 地址（位于服务器上）发出 GET 请求，其中带有传递的参数，并且以 JSON 编码的数据格式返回信息。它的语法如下：

```
$.getJSON(url, parameters, callbackfunction)
```

其中的参数如下。

- url 是文件的名字以及它在服务器上的地址。
- parameters 是将要传送给 url 的字符串，其中包含名称/值对形式的信息。
- callback 函数是请求成功完成时，服务器所作出的响应。

使用$.getJSON()方法的 jQuery 代码如下所示，它的工作方式与之前的解决方案基本相似，但是你可以看到，它的代码更加简洁：

```
$(document).ready(function() {
  $('#submit').click(function () {
    $.getJSON('drinkinfo.json', function (data){
      var drinks="<ul>";
      $.each(data, function(i,n){
        drinks+="<li>"+n["optiontext"]+"</li>";
      });
      drinks+="</ul>";
      $('#message').append(drinks);
    });
    return false;
  });
});
```

8.8 取得 XML 数据

问题描述

现有一个 XML 文件，其中包含一些学生的信息。你想要从 XML 文件中异步地导入信息到当前的网页中。XML 文件中包含一些用户自定义的用来组织信息的标签。

解决方案

让我们首先创建一个 HTML 文件，其中包含一个段落元素、一个提交按钮和一个空的 div 元素。HTML 文件应该像下面这样：

```
<body>
```

```
<p>To see the Names of the students extracted from XML file click the button given below :</p>
<input type="submit" id="submit"/>
<div id="message"></div>
</body>
```

段落元素只是向我们的用户以友好的形式显示了一条信息，当然，我们希望用户在准备好的时候点击提交按钮。然后就会从 XML 文件导入信息，并显示在名为 message 的 div 元素中。假设 XML 文件的名称是 student.xml，其中的 XML 内容如下所示：

```
<?xml version="1.0" encoding="utf-8" ?>
<school>
  <student>
    <roll>101</roll>
    <name>
      <first-name>Anil</first-name>
      <last-name>Sharma</last-name>
    </name>
    <address>
      <street>
        22/10 Sri Nagar Road
      </street>
      <city>
        Ajmer
      </city>
      <state>
        Rajasthan
      </state>
    </address>
    <marks>
      85
    </marks>
   </student>
    <student>
    <roll>102</roll>
    <name>
      <first-name>Manoj</first-name>
      <last-name>Arora</last-name>
    </name>
    <address>
      <street>
        H.No 11-B Alwar Gate
      </street>
      <city>
        Ajmer
      </city>
      <state>
        Rajasthan
      </state>
    </address>
    <marks>
      92
    </marks>
  </student>
```

```
</school>
```

下面的 jQuery 代码会导入 XML 标签 first-name 中存储的信息，并以列表项的形式显示在当前的网页中：

```
$(document).ready(function() {
  $('#submit').click(function () {
    $.ajax({
      type:"GET",
      url:"student.xml",
      dataType:"xml",
      success: function (sturec) {
        var stud="<ul>";
        $(sturec).find('student').each(function(){
          var name = $(this).find('first-name').text()
          stud+="<li>"+name+"</li>";
        });
        stud+="</ul>";
        $('#message').append(stud);
      }
    });
    return false;
  });
});
```

显示XML文件中的学号、名字和成绩

在查看代码如何工作之前，有必要提醒大家一下，我们可以很容易地修改上面的 jQuery 代码，从而从其他标签中取得文本——下面是从<roll>、<first-name>、<last-name>和<marks>标签中取得文本的代码：

```
$(document).ready(function() {
  $('#submit').click(function () {
    $.ajax({
      type:"GET",
      url:"student.xml",
      dataType:"xml",
      success: function (sturec) {
        var stud="<table border='1'>";
        $(sturec).find('student').each(function(){
          var roll = $(this).find('roll').text()
          var fname = $(this).find('first-name').text()
          var lname = $(this).find('last-name').text()
          var marks = $(this).find('marks').text()
          stud+="<tr><td>"+roll+"</td><td>"+fname+" "+lname+"</td><td>"+marks+"</td></tr>";
        });
        stud+="</table>";
        $('#message').append(stud);
      }
    });
    return false;
  });
});
```

知其所以然

首先我们向提交按钮添加了 click 事件，然后通过 ajax()方法调用了请求，其中我们指定想要使用的请求方法的类型是 GET，而 url 是服务器上的 XML 文件 student.xml。我们还将 dataType 键值的值指定为 xml，表示 url 包含的是 XML 形式的数据。

从 XML 文件（假设已经存在于服务器上）导入的信息被返回给 JavaScript 文件，这是以服务器生成的响应的形式返回的，然后由回调函数的参数 sturec 接收。这里的 sturec 是对象的数组，其中每个元素都包含下列标签：<school>、<student>、<roll>、<name>、<first-name>、<last-name>、<address>、<street>、<city>、<state>和<marks>。当然这些都与 XML 文件中的标签一致。

我们初始化了变量 stud，并且将其赋给标签
，因为我们想要以未排序列表的形式将响应（学生的名字）发送到网页。在 sturec 中接收的信息包含多个标签，并且我们希望只将<first-name>标签中的内容以列表项的形式返回给网页。因此，我们先是在 sturec 中找到<student>标签，并使用 each()方法来依次解析每个学生的记录。在<student>标签中，我们会找到<first-name>标签，取得其中的文本，并将其存储在变量 name 中。这样做的结果是，每个学生在<first-name>标签中的文本都被提取出来，然后名字会被放在
和
标签之间，使得他们以列表项的形式显示，并且被添加到变量 stud 之中。最终，我们将变量 stud 中的内容赋给名为 message 的 div 元素，从而显示这些数据。

和通常情况一样，我们在 click 事件中返回 false，以抑制浏览器默认的点击行为，因为我们希望那个事件采取通过 jQuery 代码指定的动作，而不是默认的动作。在执行过程中，我们的网页初始时会显示段落元素和一个提交按钮，如图 8-23 所示。

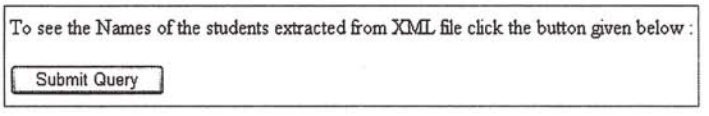


图 8-23　带有文本信息和要导入 XML 内容的提交按钮的初始页面

当选择提交按钮的时候，所有学生（位于 XML 文件中）的名字都会以列表项的形式显示，如图 8-24 所示。

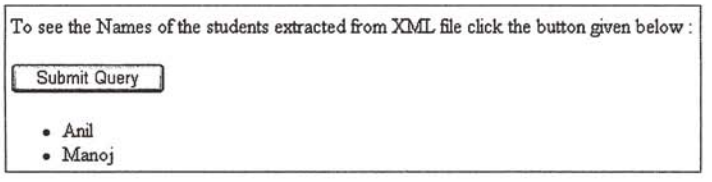


图 8-24　XML 文件中 first-name 标签中的内容以列表项的形式被导入

当我们修改代码的时候，<roll>、<first-name>、<last-name> 和<marks>标签的内容文本也会被取得，并嵌入到<tr>和<td>标签中，从而使得它们以表格的形式显示。在执行过程中，

我们会以表格形式显示所有学生的学号、姓名和成绩，如图 8-25 所示。

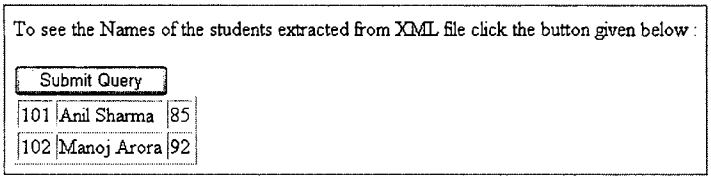


图 8-25 XML 文件中的 学号、名、姓和成绩会以表格的形式导入

8.9 分页显示表格

问题描述

要从 MySQL 服务器的表中取得一些学生信息，并对其进行分页显示，要求每页显示五条记录。

解决方案

首先，我们会创建一个 HTML 文件，其中包含一个名为 message 的空 div 元素，它会被用于分页显示学生记录的表格。HTML 文件会像下面这样：

```
<body>
<div id="message"></div>
</body>
```

下面的 jQuery 代码会为表格中的记录分页，每页 5 行，并且当点击页码的时候，会显示相应的行：

```
$(document).ready(function() {
  $.ajax({
    type:"POST",
    url:"getstudrec.php",
    success:function(html) {
      $('#message').html(html);
      var rows=$('table').find('tbody tr').length;
      var no_rec_per_page=5;
      var no_pages=Math.ceil(rows/no_rec_per_page);
      var $pagenumbers=$('<div id="pages"></div>');
      for(i=0;i<no_pages;i++)
      {
        $('<span class="page">'+(i+1)+'</span>').appendTo($pagenumbers);
      }
      $pagenumbers.insertBefore('table');
      $('.page').hover(function(){
        $(this).addClass('hover');
      }, function(){
        $(this).removeClass('hover');
      });
```

```
        $('table').find('tbody tr').hide();
        var tr=$('table tbody tr');
        for(var i=0;i<=no_rec_per_page-1;i++)
        {
          $(tr[i]).show();
        }

        $('span').click(function(event){
          $('table').find('tbody tr').hide();

for(i=($(this).text()-1)*no_rec_per_page;i<=$(this).text()*no_rec_per_page-1;i++)
          {
            $(tr[i]).show();
          }
        });
      }
    });
});
```

你会看到我们通过 ajax()方法调用了请求，其中我们指定请求的方法类型是 POST，而服务端脚本文件是 getstudrec.php。服务端脚本文件做出的响应——所有的学生记录——会返回给 JavaScript 文件，并且由回调函数中的参数 html 接收。然后我们会将服务器的响应（html）赋给名为 message 的 div 元素以供显示。下面是 getstudrec.php 脚本文件的代码：

```
<?php
$connect =
  mysql_connect("localhost", "root", "mce") or die("Please, check your server
connection.");
mysql_select_db("college");
$query = "SELECT roll, name, marks from student";
$results = mysql_query($query) or die(mysql_error());
if($results)
{
  echo '<table border="1">';
  echo '<thead>';
  echo '<tr><th>Roll</th><th>Name</th><th>Marks</th></tr>';
  echo '</thead>';
  echo '<tbody>';
  while ($row = mysql_fetch_array($results)) {
    extract($row);
    echo '<tr><td>' . $roll . '</td><td>' . $name . '</td><td>' . $marks . '</td></tr>';
  }
  echo '</tbody>';
  echo '</table>';
}
?>
```

为了在页码上应用鼠标悬停的效果，并指定那些页码周围的空间，你需要在样式表文件中定义两个样式规则，如下所示：

```
.hover { background-color: #00f; color: #fff; }
.page{ margin:5px; }
```

知其所以然

在服务端脚本文件中，我们首先指定了与 MySQL 服务器之间的连接，并选择了数据库 college。然后我们编写用于从 student 表中取得所有学生的学号、姓名和成绩的 SQL 查询语句。接下来，我们会执行 SQL 查询，而作为查询的输出所返回的记录会存储在$results 数组中。

样式规则 hover 中将 background-color 和 color 属性分别设置为#00f 和#fff，从而将鼠标悬停的页码的背景色变为蓝色，前景色变为白色。样式规则 page 中将 margin 属性设置为 5px，从而让页码之间保持 5px 的距离。

在 jQuery 代码中，我们通过 ajax()方法调用了请求，其中我们指定将要使用的请求方法的类型是 POST，并且服务端脚本的 url 是 getstudrec.php。服务端脚本文件做出的响应会返回给 JavaScript 文件，并由回调函数的参数 html 接收。注意到此处 html 是 student 记录的数组，因此它是学生记录的表格（包含表示“学号”、“姓名”和“成绩”的学生数据）。

服务端脚本做出的响应被赋给名为 message 的 div 元素，从而将表的记录显示在屏幕上。为了达到这个目的，我们会计算行（嵌入在 tbody 元素中的 tr 元素）的数目，并将其存储在变量 rows 中。我们假设想要每页显示 5 行，并且将变量 no_rec_per_page 的值初始化为 5。

我们用总行数除以每页想要显示的记录的行数得到总页数。页数被赋给变量 no_pages，并且我们还定义了一个名为 pages 的 div 元素，将它赋给变量$pagenumbers。然后借助 for 循环，我们创建一些 span 元素（数量和页数一样），其中包含有一系列的页码 1、2……，span 元素的 class 属性被赋值为 page，从而定义在类选择器 page 中的样式属性会自动应用给所有的页码。最终，所有包含页码的 span 元素都被附加给 ID 为 pages 的 div 元素。

我们会直接将存储在变量$pagenumbers 中名为 pages 的 div 元素插入到 table 元素之前，这样页码会显示在表格上方。

在页码（class 为 pages 的 span 元素）的鼠标悬停事件中，你会看到我们在鼠标悬停的时候会突出显示页码，这是通过操作定义在样式规则 hover 中的属性做到的，它会应用给页码。在那个事件中，我们将它们的背景色和前景色分别变为蓝色和白色，当鼠标不再悬停在上面的时候，我们会移除那些样式属性以及相关的视觉特效。

然后我们会隐藏表格的所有行（即内嵌在 tbody 元素中的 tr 元素），只显示列标题，并将表中的所有记录都取得并存储在变量 tr 中。你要记住，tr 是包含表中所有行的数组，因此我们会使用 for 循环来显示表最前面的 5 行。

我们会向 span 元素附加 click 事件，从而所有的页码和表格的隐藏行都会被显示，当然也包括列标题。最后，我们会使用 tr 数组来显示落在用户点击的页码范围之内的记录。在执行过程中，我们会得到表的最前面 5 行，并在顶部显示页码，如图 8-26 所示。

1 2 3

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85
104	David	92
105	Kelly	81

图 8-26 显示一个表格，顶部带有页码

8.10　小结

本章中，你已经看到我们可以如何使用 AJAX 技术来为网页提供快速响应，以得到更好的用户体验。你看到了如何使用 AJAX 技术来显示欢迎信息，执行动态用户验证、验证用户名和邮件地址等，从而创造出快速响应的用户体验。然后你还学到关于导入 HTML，取得 JSON 数据和 XML 数据，以及如何创建分页显示效果的攻略。

下一章中，我们会使用一些插件，它们会帮助我们创建解决方案，而不需要编写自己的 jQuery 代码，其中包括如何在表格的任一列上过滤，如何为图片添加注解还有拖曳表格的行等。你还会看到如何取得、序列化以及清除表单控件，还有如何使用 Dimension 插件来找到元素的准确位置和范围。你还会使用 3D Image Carousel 插件，用来选择日期的 Datepicker 插件，以及对表格排序的 tablesorter 插件。接下来，我们进入第 9 章，学习插件的相关内容。

第9章 使用插件

jQuery 库非常强大，足以执行很多任务，但使用不同的插件，我们可以为其添加更多附加的功能。在 plugins.jquery.com 有很多 jQuery 插件。在本书中我们不可能试验所有的 jQuery 插件，所以我在下面的攻略中会选择其中几个来说明。本章中，我们将一起试验下列攻略：

- 对表格的任一列进行过滤，并且可以设置每页的行数；
- 为图片添加注解；
- 拖放表格中的行；
- 取得、序列化并清理表单控件；
- 通过 Ajax 提交表单；
- 找到元素的准确位置和大小；
- 以传送带的方式显示图片；
- 使用 `datepicker` 选择日期；
- 对表格排序。

9.1 对表格的任一列进行过滤，并且可以设置每页的行数

问题描述

现有一个文本框，下面是一个表格，在表格的任一列上执行过滤，以找到该列上特定字符的记录。因此，如果在文本框中键入任意一个字符，只有那些包含该字符的行才会显示。类似地，如果在文本框中输入了任意一个数字，则只有那些与文本库中输入的值一致的行才会显示。另外，你还想要使用下拉列表控件来限定表格中显示的行数。也就是说，你想要根据在下拉列表控件中选择的值，将表格的行数限定为 10、25、50 或者 100。

解决方案

对于这个攻略我使用了"DataTables"插件。它的文件名是 jquery.dataTables.js，我们可以从 datatables.net 下载到这个 JavaScript 文件。

对于这个攻略，我们需要创建一个 HTML 文件，其中有一个 `table` 元素，显示了学生的信息。表格会显示 15 个学生的学号、名字和成绩。我们会在 HTML 文件中包含刚刚下载的 JavaScript

文件。HTML 文件中的代码如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title>JQuery Examples</title>
    <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
    <script src="jquery.dataTables.js" type="text/javascript"></script>
    <script src="d1.js" type="text/javascript"></script>

  </head>
</head>
<body>
<table border="1" class="studrec">
<thead>
<tr><th>Roll</th><th>Name</th><th>Marks</th></tr>
</thead>
<tbody>
<tr><td>101</td><td>John</td><td>87</td></tr>
<tr><td>102</td><td>Naman</td><td>90</td></tr>
<tr><td>103</td><td>Chirag</td><td>85</td></tr>
<tr><td>104</td><td>David</td><td>92</td></tr>
<tr><td>105</td><td>Kelly</td><td>81</td></tr>
<tr><td>106</td><td>Charles</td><td>77</td></tr>
<tr><td>107</td><td>Jerry</td><td>91</td></tr>
<tr><td>108</td><td>Beth</td><td>75</td></tr>
<tr><td>109</td><td>Caroline</td><td>82</td></tr>
<tr><td>110</td><td>Hanen</td><td>71</td></tr>
<tr><td>111</td><td>Douglas</td><td>57</td></tr>
<tr><td>112</td><td>Tim</td><td>86</td></tr>
<tr><td>113</td><td>Michael</td><td>68</td></tr>
<tr><td>114</td><td>Kimbley</td><td>88</td></tr>
<tr><td>115</td><td>Christina</td><td>72</td></tr>
</tbody>
</table>
</body>
</html>
```

为了在上面的表格中使用过滤功能，并且限定每页显示的行数，我们会使用如下的 jQuery 代码：

```
$(document).ready(function() {
  $('.studrec').dataTable();
});
```

知其所以然

在代码中可以看到,我们对 class 为 studrec 的 table 元素应用了 dataTable()方法（来自于 JavaScript 文件 jquery.dataTables.js） 表格输出的显示效果如图 9-1 所示。

Show 10 entries
Search:

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85
104	David	92
105	Kelly	81
106	Charles	77
107	Jerry	91
108	Beth	75
109	Caroline	82
110	Hanen	71

Showing 1 to 10 of 15 entries

图 9-1　显示表格的 10 行

我们会看到，在表格上面显示有一个下拉列表控件和一个文本框，下面有一个标签。标签表示的是正在显示的行数。下拉列表控件显示了4种选项：10、25、50和100，用来确定表格中每页所要显示的行数，如图9-2所示。

如果我们在文本框中输入字符“c”，那么所有名字中有字母“C”和“c”（任意位置，在开头或者中间）的学生记录都会显示出来，如图9-3所示。

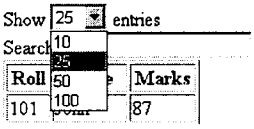


图9-2 下拉列表控件中可用的选项

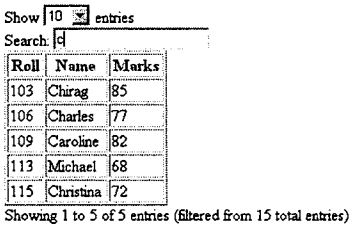


图9-3 显示其中有字母“c”的名字

类似地，我们可以在文本框中输入任意数字，从而看到指定学号或者成绩的记录。

9.2 为图片添加注解

问题描述

首先，你已经拥有一个大图片。你的想法是，当你将鼠标在图片的不同位置滚动时，在图片的重要区域会自动出现注解，告诉用户图片的该区域代表的是什么。

解决方案

对于这个秘诀我使用了jQuery Image Annotation插件。让我们从blog.flipbit.co.uk/2009/03/jquery-image-annotation-plugin.html下载JavaScript文件jquery.annotate.js和CSS样式表文件annotation.css。然后创建一个HTML文件来显示图片shopcart.jpg（这可以是任意的图片）。HTML文件中会包含下载的JavaScript文件，还有指向样式表文件的链接。HTML文件会像下面这样：

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>JQuery Examples</title>
  <link rel="stylesheet" href="annotation.css" type="text/css" media="screen" />
  <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
  <script src="jquery.annotate.js" type="text/javascript"></script>
  <script src="d1.js" type="text/javascript"></script>
  </head>
<body>
<img src="shopcart.jpg" class="shop"/>
</body>
```

让我们来编写jQuery代码，使得注解出现在图片特定的位置上，并且还有代表图片上的那些

位置所代表的意思的文字。jQuery 代码可能会像下面这样：

```
d1.js
$(document).ready(function() {
  $('.shop').annotateImage({
    editable: true,
    useAjax: false,
    notes: [ { "top": 60,
               "left": 300,
               "width": 50,
               "height": 50,
               "text": "Header",
               "id": "h1",
               "editable": false },
             { "top": 200,
               "left": 600,
               "width": 50,
               "height": 50,
               "text": "Picture of Product",
               "id": "s1",
               "editable": false },
             { "top": 200,
               "left": 150,
               "width": 50,
               "height": 50,
               "text": "Menu",
               "id": "m1",
               "editable": false } ]
  });
});
```

知其所以然

在上面的代码中，我们创建了 3 个注解，每个注解的宽度和高度都是 50px，并且都位于在"top"和"left"属性中指定的图片位置。这3个注解的文本分别被设置为"Header","Picture of Product"和"Menu"，从而定义了所代表的图片的位置。另外，这些注解的 id 分别被设置为唯一的"h1"、"s1"和"m1"。editable 属性被设置为 false，因为我们不希望用户编辑这些文本。初始时，shopcart.jpg 文件（没有 jQuery 代码）的样子如图 9-4 所示。

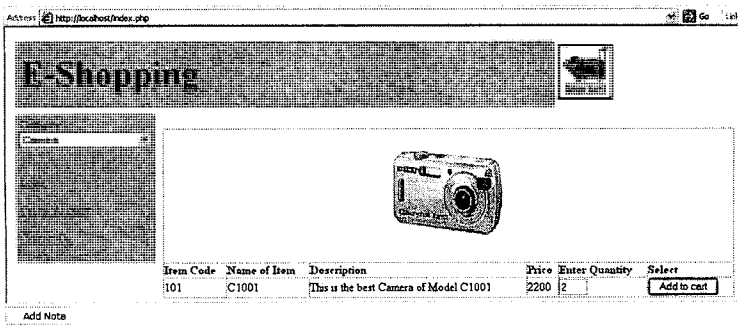


图 9-4　原始的没有注解的图片

应用了 jQuery 代码之后，我们会看到图片上的 3 个用矩形标识的区域，如图 9-5 所示。这些矩形表示图片的这些区域是有注解的。

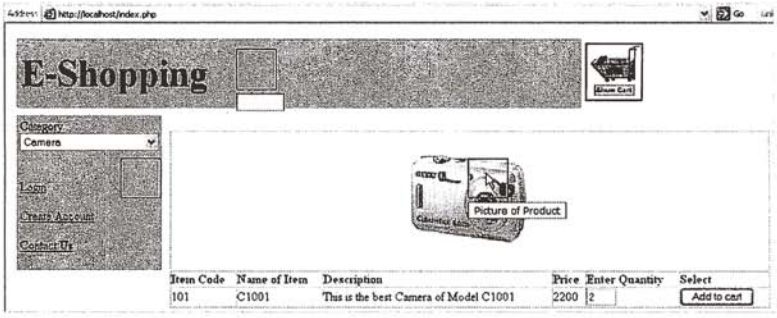


图 9-5 应用了注解的图片

当鼠标指针移到标记的区域上时，我们会看到显示出文本"Picture of Product"。

9.3 拖放表格中的行

问题描述

现有显示了一些信息的表格，你希望用户可以（以任意的顺序）拖放任意行。

解决方案

我会在这个秘诀中使用"Table Drag and Drop JQuery"插件。让我们从 isocra.com/2008/02/table-drag-and-drop-jquery-plugin/下载 JavaScript 文件 jquery.tablednd_0_5.js。然后让我们创建一个 HTML 文件，其中包含有下载的 JavaScript 文件，并且还有显示了 15 个学生的学号、姓名和成绩信息的 table 元素。HTML 文件会像下面这样：

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>JQuery Examples</title>
  <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
  <script src="jquery.tablednd_0_5.js" type="text/javascript"></script>
  <script src="d1.js" type="text/javascript"></script>
  </head>
<body>
<table border="1" class="studrec">
<thead>
<tr><th>Roll</th><th>Name</th><th>Marks</th></tr>
</thead>
<tbody>
<tr><td>101</td><td>John</td><td>87</td></tr>
<tr><td>102</td><td>Naman</td><td>90</td></tr>
<tr><td>103</td><td>Chirag</td><td>85</td></tr>
<tr><td>104</td><td>David</td><td>92</td></tr>
```

```
<tr><td>105</td><td>Kelly</td><td>81</td></tr>
<tr><td>106</td><td>Charles</td><td>77</td></tr>
<tr><td>107</td><td>Jerry</td><td>91</td></tr>
<tr><td>108</td><td>Beth</td><td>75</td></tr>
<tr><td>109</td><td>Caroline</td><td>82</td></tr>
<tr><td>110</td><td>Hanen</td><td>71</td></tr>
<tr><td>111</td><td>Douglas</td><td>57</td></tr>
<tr><td>112</td><td>Tim</td><td>86</td></tr>
<tr><td>113</td><td>Michael</td><td>68</td></tr>
<tr><td>114</td><td>Kimbley</td><td>88</td></tr>
<tr><td>115</td><td>Christina</td><td>72</td></tr>
</tbody>
</table>
</body>
</html>
```

调用表格拖放插件的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.studrec').tableDnD();
});
```

知其所以然

在代码中可以看到我们调用了 JavaScript 文件 jquery.tablednd_0_5.js 中的 `tableDnD()`方法，并将其应用到 class 为 `studrec` 的 `table` 元素上。初始时，我们的表格如图 9-6 所示。

我们可以在任意一行上点击并拖动到其他行上，从而改变它的位置。在图 9-7 中，第 1 行被点击并拖动到第 5 行之后。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85
104	David	92
105	Kelly	81
106	Charles	77
107	Jerry	91
108	Beth	75
109	Caroline	82
110	Hanen	71
111	Douglas	57
112	Tim	86
113	Michael	68
114	Kimbley	88
115	Christina	72

图 9-6　行位于初始位置的表格

Roll	Name	Marks
102	Naman	90
103	Chirag	85
104	David	92
105	Kelly	81
101	John	87
106	Charles	77
107	Jerry	91
108	Beth	75
109	Caroline	82
110	Hanen	71
111	Douglas	57
112	Tim	86
113	Michael	68
114	Kimbley	88
115	Christina	72

图 9-7　将第 1 行拖放到第 5 行之后的表格

9.4 取得、序列化并清理表单控件

问题描述

现有一个带有控件的表单，你想要取得在那些控件中输入的值，将其序列化，并且还想要在点击不同的按钮时分别清理各个控件。

解决方案

对于这个攻略我使用了 Form 插件。让我们从 jquery.com/plugins/project/form 下载 JavaScript 文件 jquery.form.js。然后我们会创建一个 HTML 文件，其中包含带有两个控件的表单，分别可以让我们输入用户 id 和邮件地址。HTML 文件中会包含下载的 JavaScript 文件，代码如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>JQuery Examples</title>
  <link rel="stylesheet" href="style.css" type="text/css" media="screen" />
  <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
  <script src="jquery.form.js" type="text/javascript"></script>
  <script src="d1.js" type="text/javascript"></script>
  </head>
<body>
<form id="myForm">
<div><span class="label">User Id *</span><input type="text"  class="infobox"
name="userid" /><span class="error"> This field cannot be blank</span></div>
<div><span class="label">Email address *</span><input type="text" class="infobox"
name="emailid" /><span class="error"> Invalid Email address </span></div>
<input class="submit" type="submit" value="Submit"/>
<input class="clear" type="submit" value="Clear Form"/>
</form>
</body>
</html>
```

表单的 id 被设置为"myForm"，从而我们可以通过 jQuery 代码来访问它。标签信息 User Id 和 Email address 包含在 span 元素中，它们的 class 被设置为 label。文本输入框的 class 被设置为 infobox，而错误信息“This field cannot be blank”和“Invalid Email address”分别存储在 class 为 error 的 span 元素中。

为所有 3 个项目标签、文本输入框和错误信息都设置 class 的原因是要自动在上面应用定义在类选择器.label、.infobox 和.error 中（定义在样式表文件 style.css 中）的属性。两个按钮的值分别被设置为"Submit"和“清除表单”，它们的 class 被设置为 submit 和 clear，这样就会应用定义在类选择器.submit 和.clear 中的属性。控件都嵌入在 div 元素中，从而会在它们之间保持距离。定义了这些类选择器的样式表可能会像下面这样：

```
.label {float: left; width: 120px; }
.infobox {width: 200px; }
```

```
.error { color: red; padding-left: 10px; }
.submit { margin-left: 50px; margin-top: 10px;}
.clear { margin-left: 125px; margin-top: 10px;}
div{padding: 5px; }
```

类选择器.label 中将 float 属性设置为 left，使得标签显示在它的左边（为显示在右边的文本输入框留出空间），并且将 width 属性设置为 120px，从而定义标签能够使用的宽度。类选择器.infobox 中将 width 属性设置为 200px，从而指定了文本输入框的宽度；类型选择器.error 会将错误信息的颜色设置为红色，并且使得错误信息显示在离左边的元素 10px 的位置。类选择器.submit 和.clear 中将 margin-left 和 margin-top 属性设置为不同的值，从而将两个按钮"Submit"和"Clear Form"放在距离浏览器窗口的左边界和上面的元素适合的位置。类型选择器 div 将 padding 属性设置为 5px，从而在两个 div 元素之间保持一些距离，每个 div 元素中都包含有标签、文本输入框和错误信息。

下面的 jQuery 代码会取得并序列化在表单的两个控件中输入的数据。

```
$(document).ready(function() {
  $('.error').hide();
  $('.submit').click(function(event){
    alert( $('#myForm *').fieldValue());
    alert( $('#myForm *').fieldSerialize());
    event.preventDefault();
  });
  $('.clear').click(function(event){
    $('#myForm').clearForm();
    event.preventDefault();
  });
});
```

在上面的 jQuery 代码中，我们使用了 3 个方法：fieldValue()、fieldSerialize()和 clearForm()。让我们来了解一下这 3 个方法的功能。

1. fieldValue()

fieldValue()方法收集了在所有控件中输入的数据，并将它们以字符串数组的形式返回。只有活动的控件内容（拥有 name 属性并且没有被禁用）才会被这个方法取得。

2. fieldSerialize()

fieldSerialize()方法会取得控件中的内容，并将它们以查询字符串的形式编码，然后再返回。查询字符串会以下面的格式编码：

```
control1=value1&control2=value2…
```

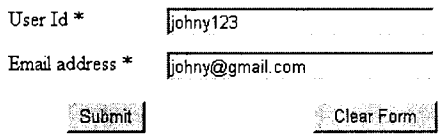
3. clearForm()

clearForm()方法会清除表单所有控件中输入的值。

知其所以然

在上面的 jQuery 代码中，初始时我们隐藏了所有错误信息。当点击提交按钮的时候，我们会取得所有控件的内容，并以字符串数组的形式显示（使用 fieldValue()方法），如图 9-8 所示。

另外，控件中的内容会被序列化，并借助于 fieldSerialize()方法显示。我们会看到在控件中输入的数据会以查询字符串的形式编码并显示，如图 9-9 所示。



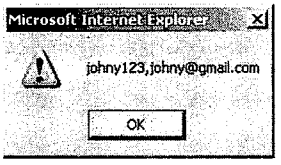


图 9-8 控件中输入的数据以字符串数组的形式显示

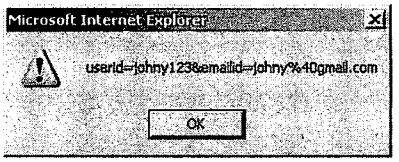


图 9-9 控件中输入的数据以序列化的形式显示

当选择 Clear Form 按钮的时候，控件中输入的数据会被清除。

9.5 通过 Ajax 提交表单

问题描述

现有带控件的表单，要以 AJAX 请求的形式提交表单。

解决方案

在本攻略中，我会再次使用 Form 插件。我们可以从 jquery.com/plugins/project/form 下载 jquery.form.js 文件。然后让我们创建一个 HTML 文件，其中包含带有两个控件的表单，让我们可以分别输入用户 id 和邮件地址。HTML 文件中会包含下载的 JavaScript 文件，代码如下所示：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>JQuery Examples</title>
  <link rel="stylesheet" href="style.css" type="text/css" media="screen" />
  <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
  <script src="jquery.form.js" type="text/javascript"></script>
  <script src="d1.js" type="text/javascript"></script>
  </head>
<body>
<form id="myForm">
<div><span class="label">User Id *</span><input type="text"  class="infobox"
name="userid" /><span class="error"> This field cannot be blank</span></div>
<div><span class="label">Email address *</span><input type="text" class="infobox"
name="emailid" /><span class="error"> Invalid Email address </span></div>
```

```
<input class="submit" type="submit" value="Submit"/>
<input class="clear" type="submit" value="Clear Form"/>
</form>
</body>
</html>
```

带有这些类选择器的样式表可能会像下面这样：

```
.label {float: left; width: 120px; }
.infobox {width: 200px; }
.error { color: red; padding-left: 10px; }
.submit { margin-left: 50px; margin-top: 10px;}
.clear { margin-left: 125px; margin-top: 10px;}
div{padding: 5px; }
```

以 AJAX 请求形式提交表单的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.error').hide();
  $('#myForm').ajaxForm(function() {
    alert("Thank you for your comment!");
  });
});
```

在提交表单（以AJAX请求的形式）之前执行验证

在提交上述表单之前，让我们先对其中的两个控件——用户 id 和邮件地址——进行验证。我们必须编写验证函数，确保用户 id 控件不为空，且邮件地址包含必要的“@”和“.”符号。我们需要对 HTML 文件做出一些修改，如下所示：

```
<body>
<form id="myForm">
<div><span class="label">User Id *</span><input type="text"  class="usrinfo"
name="userid" /><span class="usrerror"> This field cannot be blank</span></div>
<div><span class="label">Email address *</span><input type="text" class="emailinfo"
name="emailid" /><span class="emailerror">Invalid Email address</span></div>
<input class="submit" type="submit" value="Submit"/>
</form>
</body>
```

包含类选择器的样式表（以自动地将其中定义的属性应用到不同的控件上）可能像下面这样：

```
.label {float: left; width: 120px; }
.usrinfo {width: 200px; }
.emailinfo {width: 200px; }
.usrerror { color: red; padding-left: 10px; }
.emailerror { color: red; padding-left: 10px; }
.submit { margin-left: 125px; margin-top: 10px;}
div{padding: 5px; }
```

验证表单控件并且以 AJAX 形式提交表单的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.usrerror').hide();
  $('.emailerror').hide();

  function validate_data()
```

```
{
  var data=$('.usrinfo').val();
  var len=data.length;
  var pattern= new RegExp(/^[\w-]+(\.[\w-]+)*@([\w-]+\.)+[a-zA-Z]+$/);
  var flag=true;
  if(len<1)
  {
    $('.usrerror').show();
    flag=false;
  }
  else
  {
    $('.usrerror').hide();
  }
  var email=$('.emailinfo').val();
  if(pattern.test(email))
  {
    $('.emailerror').hide();
  }
  else
  {
    $('.emailerror').show();
    flag=false;
  }
  if(flag==false)
  {
    return false;
  }
  else
  {
    $('.usrinfo').val("");
    $('.emailinfo').val("");
  }
};

$('#myForm').ajaxForm({
  beforeSubmit: validate_data,
  success: function(){
    alert("Thank you for your comment!");
  }
});
});
```

知其所以然

表单的 id 被设置为 myForm，从而我们可以通过 jQuery 代码来访问它。标签信息 User Id 和 Email address 包含在 span 元素中，它们的 class 被设置为 label。文本输入框的 class 被设置为 infobox，而错误信息“This field cannot be blank”和“Invalid Email address”分别存储在 class 为 error 的 span 元素中。

将类型赋给所有 3 个项目（标签、文本输入框和错误信息）的原因在于要自动地应用在类选择器.lable、.infobox 和.error（定义在样式表文件 style.css）中的属性。两个按钮“Submit”和“Clear Form”的 class 被分别赋值为 submit 和 clear，从而可以应用类选择器.submit

和.clear中定义的属性。控件都嵌入在div元素中，从而在彼此之间会保持一定的距离。

我们可以看到，.ajaxForm()方法被用于在AJAX中提交表单（不需要刷新当前页面）。借助于beforeSubmit选项，我们调用了validate_data函数来验证两个控件（用户id和邮件地址）中的内容。如果表单成功提交，那么方法的success选项会通过alert()方法显示感谢信息。

这个方法以AJAX请求的形式提交表单，也就是提交表单而不刷新页面。

下面是相关的语法：

```
.ajaxForm({
target:,
beforeSubmit:,
success:
});
```

其中target指的是我们想要由服务器响应更新的元素。beforeSubmit选项会在提交表单之前执行任务，而success选项会在表单成功提交的时候执行任务。

在表单成功提交的时候（以AJAX请求的形式），我们会向用户显示提醒信息“Thank you for your comment!”。在两个控件中输入一些数据之后，如果我们选择“Submit”按钮，就会显示提示信息，如图9-10所示。

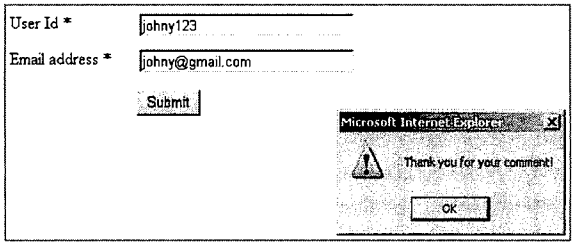


图9-10　成功提交表单时显示的提示信息

注意　我们从上面的HTML表单中移除“清除表单”按钮，因为我们不再需要它。

在第2个解决方案中，表单的id被设置为myForm，因而我们可以通过jQuery代码来访问它。

标签信息User Id和Email address包含在span元素中，它们的class被设置为label。文本输入框的class被赋值为"usrinfo"和"emailinfo"，以便于分别访问他们的内容进行验证。错误信息“This field cannot be blank”和“Invalid Email address”分别被存储在class为usrerror和emailerror的span元素中，从而只有相关控件的错误信息才会显示。控件都嵌入在div元素中，因而会在它们之间保持距离。

validate_data方法会取得用户在class为usrinfo的文本输入框中输入的用户id，并计算它的长度。如果用户id的长度小于1，也就是在文本输入框中没有输入任何文本，那么class为usrerror的错误信息会显示，而变量flag的值会被设置为false，从而表单不会被提交。同样地，在class为emailinfo的文本输入框中输入的邮件地址也会被取得并检查是否符合正则表达式"/^[\w-]+(\.[\w-]+)*@([\w-]+\.)+[a-zA-Z]+$/"。如果邮件地址不符合所提供的正

则表达式，那么与邮件地址相关的 class 为 emailerror 的错误信息就会显示，而变量 flag 会被设置为 false，从而表单不会被提交。

如果用户 id 不为空，且输入了有效的邮件地址，那么 validate_data()方法会返回 true(默认的)，并且表单会被提交。我们会清除输入的内容，以显示数据已经被接受并提交了。当两个字段用户 id 和邮件地址都为空时，我们就会分别得到两个错误信息，如图 9-11 所示。

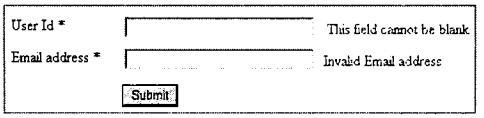


图 9-11 两个文本框都为空时显示的错误信息

如果输入了无效的邮件地址，那么就会显示“Invalid Email address”错误信息，如图 9-12 所示。

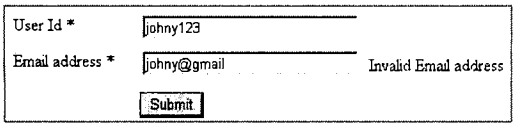


图 9-12 显示“Invalid Email Address”错误信息

如果两个控件中的数据都是有效的，那么表单就会以 AJAX 请求的形式成功提交，并且会显示信息“Thank you for your comment!”，如图 9-13 所示。

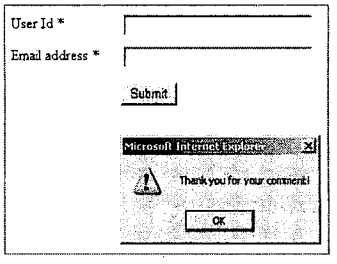


图 9-13 当两个文本框中都输入了有效的数据，就会显示提示信息，并且文本框会被清空以显示数据已经提交

9.6 找到元素的准确位置和大小

问题描述

现有一幅图片，你希望知道它的宽度、高度、内径宽度、位置等。

解决方案

在这里我使用了 Dimension 插件。让我们从 plugins.jquery.com/project/dimensions 下载

JavaScript文件jquery.dimension.js。然后我们会创建一个HTML文件，其中显示了我们想要知道其大小的图片。文件中会包含下载的JavaScript文件，代码如下所示：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>JQuery Examples</title>
  <link rel="stylesheet" href="style.css" type="text/css" media="screen" />
  <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
  <script src="jquery.dimensions.js" type="text/javascript"></script>
  <script src="d1.js" type="text/javascript"></script>
  </head>
<body>
<div class="pic">
<img src="cell.jpg"/>
</div>
</body>
</html>
```

我们会看到，图片嵌入在class为pic的div元素中，从而我们能够通过类型选择器为其应用所需要的样式属性。定义在样式表文件中的类选择器如下所示：

```
.pic {
height: 150px;
width: 100px;
margin: 5px;
border: 5px solid #000000;
overflow: auto;
}
```

我们会看到，类选择器.pic中将height、width和margin属性分别设置为150px、100px和5px（使得图片显示在给定的区域中，并且与边界保持一些距离），还将border属性设置为在图片周围显示粗细为5px的实线黑色边框。overflow属性被设置为auto，使得在图片比指定的宽度和高度大的时候会显示滚动条。

显示图片的宽度、高度和内径宽度的jQuery代码如下：

```
$(document).ready(function() {
  alert("Width: " +$('.pic').width()+" Height :"+$('.pic').height()
    +" Inner Width :"+$('.pic').innerWidth());
});
```

1. 找到偏移量和位置

在找出图片在它的父元素（如果没有的话就是body元素）中的偏移量和位置之前，让我们先简要地说明一下offset()和position()方法。

2. offset()

Offset()方法会帮助我们确定一个元素在页面中的top和left位置，这与position和overflow属性的值无关，也就是说，无论position属性被设置为static、relative或者absolute，还是将overflow属性设置为auto，结果都是一样的。

3. position()

这个方法返回的是元素相对于它的父元素的位置。它有两个属性，top 和 left，分别返回元素的顶部和左侧位置。偏移父元素指的是包围该元素的第 1 个元素（最近的祖先元素）。

在下面的 jQuery 代码中，我们还会滚动图片。滚动元素的两个方法是：scrollTop() and scrollLeft()。

4. scrollTop() and scrollLeft()

这两个方法会使元素向顶部或左侧滚动指定的距离；它们对于可见和隐藏的元素都有效。

显示图片偏移量（即它相对于窗口区域或者 body 元素的位置）和相对于它的父元素的位置的 jQuery 代码如下：

```
$(document).ready(function() {
  alert("Offset top: " +$('.pic').offset().top+", Offset left : "
    +$('.pic').offset().left+", Position top: "+$('.pic').position().top+", Position
left: "
    +$('.pic').position().left);
  $('.pic').scrollTop(50);
  $('.pic').scrollLeft(50);
});
```

知其所以然

让我们来看下 Dimension 插件中的各种方法。

- width()：它会返回窗口、文档对象或者元素的宽度。
- height()：它会返回窗口、文档对象或者元素的高度。
- innerWidth()和 innerHeight()：它会返回包括在元素上应用的填充空间(padding space)在内的宽度和高度。
- outerWidth() & outerHeight()：它会返回包括在元素上应用的填充空间(padding space)以及边框在内的宽度和高度。

借助于提示信息，我们会显示图片的宽度、高度，以及它的内径宽度（除去滚动条和边框所占用的空间），如图 9-14 所示。

当确定偏移量和位置的时候，从图 9-15 所示的输出中，我们可以看到图片位于距窗口区域顶端 20px、左侧 15px 的位置，而相对于偏移父元素的位置为 0px。

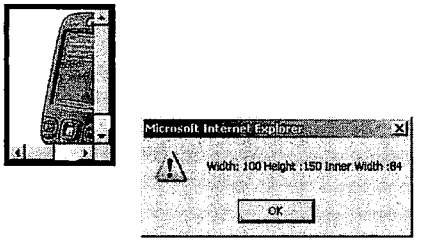


图 9-14 显示赋予图片的宽度、高度和内径宽度

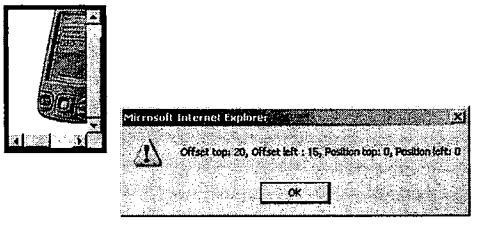


图 9-15 显示图片的偏移量和位置

在选择 OK 按钮的时候，图片还会向顶端和左侧分别滚动 50px（借助于 scrollTop()和 scrollLeft()方法），如图 9-16 所示。

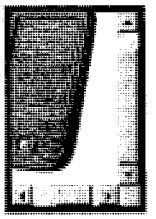


图 9-16　向顶端和左侧滚动后的图片

9.7　以传送带的方式显示图片

问题描述

让一些图片以 3D 环状传送带的形式显示，并且还带有左右箭头。

当鼠标移动到左箭头上时，图片按逆时针方向滚动；当鼠标指针移动到右箭头上时，图片会按顺时针方向滚动。另外，当选择任意一幅图片的时候，它会被放大显示。

解决方案

对于这个秘诀我使用了 3D Image carousel 插件。让我们下载 JavaScript 文件 jquery.carousel3d.js 和 Tween.js，以及它的 CSS 样式表文件（从文档中）。所有文件位于一个 zip 文件中，可以从 plugins.jquery.com/project/carousel3d 下载到（你还需要一个图片的文件夹）。让我们创建一个 HTML 文件，其中显示有一些图片，嵌入在 id 分别为 carousel 和 holder_images 的 div 元素中。HTML 文件中包含了两个下载的 JavaScript 文件，以及指向样式表文件的链接，可能如下所示：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>JQuery Examples</title>
  <link rel="stylesheet" href="style.css" type="text/css" media="screen" />
  <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
  <script type="text/javascript" src="Tween.js"></script>
  <script src="jquery.carousel3d.js" type="text/javascript"></script>
  <script src="d1.js" type="text/javascript"></script>
  </head>
<body>
<div id="carousel"></div>
  <div id="holder_images">
    <img src="image5.jpg"  width="500" height="375"  />
```

```
    <img src="a1.jpg"  width="500" height="375" />
    <img src="a2.jpg"  width="500" height="375" />
    <img  src="a3.jpg"  width="500" height="375" />
    <img src="a4.jpg"  width="500" height="375" />
    <img  src="image1.jpg"  width="500" height="375" />
    <img src="image2.jpg" width="500" height="375" />
    <img src="image3.jpg"  width="500" height="375" />
    <img  src="image4.jpg"  width="500" height="375" />
  </div>
</body>
</html>
```

样式表中包含了 id 选择器，如下所示：

```
#buttonwrapper
{
width: 100px;
height: 50px;
position: relative;
}

#left
{
background: url(left.gif) bottom left no-repeat;
width: 39px;
height: 50px;
float: left;
}

#right
{
background: url(right.gif) bottom left no-repeat;
width: 39px;
height: 50px;
float: right;
}

#left:hover, #right:hover
{
cursor: pointer;
background-position: top left;
}

#holder_images { display: none; }
#carousel img { border: 2px solid #ddd; }
#carousel img.link:hover { border: 4px solid #0e0893; }
```

调用 carousel 的 jQuery 代码如下所示。代码使得图片在鼠标指针悬停在左右箭头按钮上的时候滚动，并且在选择图片的时候将其放大显示。

```
$(document).ready(function() {
  $("#carousel").html($("#holder_images").html()).carousel3d(
    {control: 'buttons', centerX: $('#carousel').offset().left +
$('#carousel').width()/2 }
```

```
    );
});
```

知其所以然

初始的时候，图片以3D环状传送带的形式显示，在顶端带有左右箭头，如图9-17所示。当鼠标分别移到左右箭头按钮上的时候，图片会逆时针或者顺时针方向滚动。

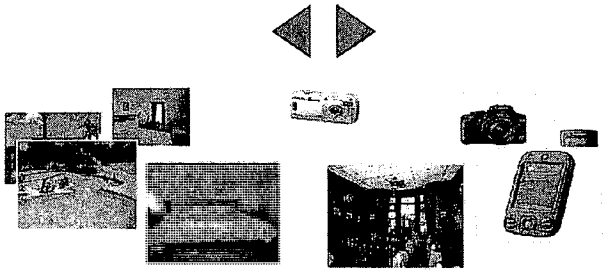


图9-17　图片以3D环状传送带的形式显示，并带有左右箭头

当选择任意一幅图片的时候，它会放大显示，如图9-18所示。

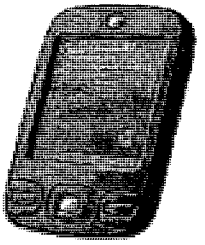


图9-18　选定的图片被放大

9.8　使用 datepicker 选择日期

问题描述

现有一个文本输入框，要在其中输入特定的日期。点击文本输入框，显示当前月份的日历，并且可以方便地向前、向后移动，以查看下个月和上个月的日历。从日历中选择任意一天的时候，将选定的日期插入到文本输入框中。

解决方案

在这里我使用了 Datepicker 插件。让我们从 jqueryui.com/demos/datepicker/下载 JavaScript 文件 ui.datepicker.js 和它的 CSS 样式表文件 ui.datepicker.css。然后我们会创建一个 HTML 文件，其

中显示有一个文本输入框，它嵌入在 class 为 demo 的 div 元素中。文本输入框的 id 被设置为 datapicker。HTML 文件中会包含下载的 JavaScript 文件，以及指向 CSS 样式表文件的链接。代码如下所示：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>JQuery Examples</title>
  <link rel="stylesheet" href="ui.datepicker.css" type="text/css" media="screen" />
  <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
  <script src="ui.datepicker.js" type="text/javascript"></script>
  <script src="d1.js" type="text/javascript"></script>
  </head>
<body>
<div class=demo>
<p>Date: <input id="datepicker"/></p></div>
</body>
</html>
```

调用 Datepicker 插件的 datepicker()方法的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $("#datepicker").datepicker();
});
```

知其所以然

当在文本框中点击的时候，当前月份的日历就会显示，如图 9-19 所示。我们可以选择 Next 或者 Prev 链接，来查看下一月或者上一月的日历。

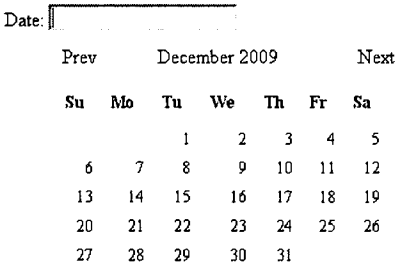


图 9-19 显示当前月份的日历

当选择日期时，它会自动插入到文本输入框当中，如图 9-20 所示。日期的输入格式为 mm/dd/yyyy。

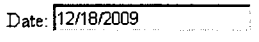


图 9-20 选定的日期显示在文本框中

为了得到特定格式的日期，我们需要使用 dateFormat 选项。以“yy-mm-dd”的格式取得日期的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $("#datepicker").datepicker({dateFormat: 'yy-mm-dd'});
});
```

现在，选定的日期会以特定的格式显示，如图 9-21 所示。

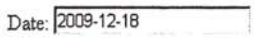


图 9-21　选定格式的日期

9.9　对表格排序

问题描述

现有一个表格，根据选定的列对其进行排序。

解决方案

在这里我使用了 Tablesorter 插件。让我们从 tablesorter.com 下载 JavaScript 文件 jquery.tablesorter.js。然后我们会创建一个 HTML 文件，其中包含有下载的 JavaScript 文件，并且包含显示了 15 个学生的学号、姓名和成绩信息的 table 元素。HTML 文件会像下面这样：

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>JQuery Examples</title>
  <script src="jquery-1[1].3.2.js" type="text/javascript"></script>
  <script src="jquery.tablesorter.js" type="text/javascript"></script>
  <script src="d1.js" type="text/javascript"></script>
</head>
</head>
<body>
<table border="1" class="studrec">
<thead>
<tr><th>Roll</th><th>Name</th><th>Marks</th></tr>
</thead>
<tbody>
<tr><td>101</td><td>John</td><td>87</td></tr>
<tr><td>102</td><td>Naman</td><td>90</td></tr>
<tr><td>103</td><td>Chirag</td><td>85</td></tr>
<tr><td>104</td><td>David</td><td>92</td></tr>
<tr><td>105</td><td>Kelly</td><td>81</td></tr>
<tr><td>106</td><td>Charles</td><td>77</td></tr>
<tr><td>107</td><td>Jerry</td><td>91</td></tr>
<tr><td>108</td><td>Beth</td><td>75</td></tr>
<tr><td>109</td><td>Caroline</td><td>82</td></tr>
<tr><td>110</td><td>Hanen</td><td>71</td></tr>
<tr><td>111</td><td>Douglas</td><td>57</td></tr>
```

```
<tr><td>112</td><td>Tim</td><td>86</td></tr>
<tr><td>113</td><td>Michael</td><td>68</td></tr>
<tr><td>114</td><td>Kimbley</td><td>88</td></tr>
<tr><td>115</td><td>Christina</td><td>72</td></tr>
</tbody>
</table>
</body>
</html>
```

我们会看到上面的 HTML 代码中包含有 class 为 studrec 的 table 元素，它显示了 15 名学生的信息。在 id 为 studrec 的 table 元素上调用 tablesorter()方法的 jQuery 代码如下：

```
$(document).ready(function() {
  $('.studrec').tablesorter();
});
```

知其所以然

在上述 jQuery 代码的执行过程中，初始时表格会以原始的顺序显示各行，如图 9-22 所示。

当选择一个列标题的时候，表格会基于选定的列排序。如果我们选择"Name"列标题，表格会按照名字的字母顺序排序，如图 9-23 所示。

Roll	Name	Marks
101	John	87
102	Naman	90
103	Chirag	85
104	David	92
105	Kelly	81
106	Charles	77
107	Jerry	91
108	Beth	75
109	Caroline	82
110	Hanen	71
111	Douglas	57
112	Tim	86
113	Michael	68
114	Kimbley	88
115	Christina	72

图 9-22 原始的表格

Roll	Name	Marks
108	Beth	75
109	Caroline	82
106	Charles	77
103	Chirag	85
115	Christina	72
104	David	92
111	Douglas	57
110	Hanen	71
107	Jerry	91
101	John	87
105	Kelly	81
114	Kimbley	88
113	Michael	68
102	Naman	90
112	Tim	86

图 9-23 表格按照名字排序

9.10 小结

本章中我们看到了便于执行很多任务的不同插件。使用它们，我们不需要编写任何代码就可以实现很多效果，像在表格的任一列上过滤，给图片添加注解，拖放表格的行，取得、序列化以及清空表单控件，通过 Ajax 提交表单，找到元素的准确位置和大小、以传送带的方式显示图片，使用 datepicker 选择日期以及对表格进行排序等等。在下一章中，我们会看到一些非常依赖 CSS 的攻略。我们会学习识别 HTML 元素、向嵌入在另一元素中的元素应用样式、缩进段落、对段落应用初始大写字母、删除标题和段落之间的间隔、向标题文本应用样式等内容。

第10章

使用 CSS

在最后这一章中，我会提供一系列更依赖于 CSS 的攻略。这些攻略使本书内容更趋完整，因为 CSS 与 JavaScript 开发者的工作总是密切相关的。我在工作中会经常使用一些 CSS 技术，所以我把它们放在这里，你可以在开发 Web 应用程序的时候快速参考相关内容。

本章中，我会讨论以下攻略：

- 区分 HTML 元素；
- 向内嵌在一个元素中的另一个元素应用样式；
- 缩进段落；
- 将段落的首字母设为大写；
- 去除标题和段落之间的间隔；
- 向标题文字应用样式；
- 缩进多个段落的第一行；
- 创建带有悬挂缩进的段落；
- 创建带有边框的提取引用；
- 创建带有图片的提取引用；
- 向列表项应用列表属性；
- 只对选定的列表项应用样式；
- 在列表项之间放置分隔线；
- 向列表应用图片标记；
- 创建水平显示的列表；
- 向超链接和邮件链接应用属性；
- 为 HTML 元素赋予不同的尺寸；
- 放置 HTML 元素；
- 创建多栏的布局；
- 使文字围绕图片；
- 在图片背后放置阴影；
- 当鼠标移过链接的时候改变鼠标样式；
- 在指定的区域中显示长文字；

- 创建圆角的栏；
- 应用文字装饰；
- 缩放图片；
- 设置背景图片；
- 使背景图片在浏览器中央显示；
- 保持背景图片固定。

10.1 区分 HTML 元素

问题描述

在为 HTML 文件中不同的段落或者不同的 h1 元素应用不同的样式时，要为它们赋予不同的 class，从而对其进行区分。另外，我们需要编写能够分别应用给这些 class 的样式规则。

解决方案

首先，我们会编写 HTML 文件，其中包含两个段落和两个 h1 元素。为了区分它们，我们会为其赋予不同的 class。我们将段落元素的 class 分别赋值为 feature1 和 feature2，将 h1 元素的 class 分别赋值为 feature2 和 feature3。

```
<body>
<p class="feature1">Styles make the formatting job much easier and efficient.</p>
<p class="feature2">To give an attractive look to web sites, styles are heavily
used.</p>
<h1 class="feature2">Using jQuery</h1>
<h1 class="feature3">Power of selectors</h1>
</body>
</html>
```

为了向这些 class 不同的 HTML 元素应用样式，可以在样式表中编写下列样式规则：

```
.greencolor{color:green;font-style:italic}
.highlight{background-color:aqua;color:blue;font-family:arial;}
.redandbold{color:red;font-family:arial;font-weight:bold}
```

向段落和 h1 元素应用样式规则的 jQuery 代码如下：

```
$(document).ready(function() {
  $('p.feature1').addClass('greencolor');
  $('.feature2').addClass('highlight');
  $('h1.feature3').addClass('redandbold');
});
```

知其所以然

第一条语句将定义在样式规则 greencolor 中的属性只应用给 class 为 feature1 的段落元素，也就是说，应用给以标签<p class="feature1">开头的元素。第二条语句将样式规则 highlight 中的属性应用给 class 为 feature2 的 HTML 元素。在 HTML 文件中，有一个段落元

素和一个 h1 元素的 class 为 feature2（分别显示为 <p class="feature2">和<h1 class="feature2">标签），因此定义在这个规则中的属性会应用在二者之上。第三条语句将样式规则 feature3 中的属性应用给 class 为 feature3 的 h1 元素。

输出效果如图 10-1 所示。

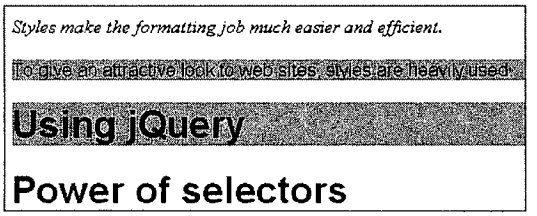


图 10-1　应用在<p>和<h1>标签上的不同类

10.2　向内嵌在一个元素中的另一个元素应用样式

问题描述

有时 span 元素会内嵌在另一个特定 ID 或者 class 的 HTML 元素中，我们需要对这个内嵌的 span 元素应用样式。

解决方案

在下面的 HTML 文件中，我们定义了 class 为 feature 的段落元素。在这个段落元素中我们定义了一个 span 元素：

```
<body>
<p class="feature">Styles make the formatting job much easier and efficient. <span>To
give an attractive look to web sites,</span> styles are heavily used.</p>.
</body>
```

将要应用给 class 为 feature 的段落元素和内嵌在其中的 span 元素的样式规则是在样式表中编写的，如下所示：

```
.greencolor{color:green;font-style:italic}
.highlight{background-color:aqua;color:blue;font-family:arial;}
```

为了向 class 为 feature1 的段落元素和内嵌在段落元素中 class 为 feature 的 span 元素应用样式，我们会使用下面的 jQuery 代码：

```
$(document).ready(function() {
  $('p.feature').addClass('greencolor');
  $('p.feature span'}.addClass('highlight');
});
```

知其所以然

首先，看一下我们是如何定义 CSS 样式的。

下面的代码定义了会应用给所有带有 class="feature"的 HTML 元素的样式：

```
.feature{property:value; property:value;...}
```

下面的代码定义了会应用给内嵌在带有 class="feature"的 HTML 元素中的 span 元素的样式：

```
.feature span {property:value; property:value;...}
```

下面的代码定义了会应用给内嵌在带有 class="feature"的段落元素中的 span 元素的样式：

```
p.feature span {property:value; property:value;...}
```

下面的代码定义了会应用给内嵌在带有 class="feature1"的 HTML 元素中带有 class="feature2"的 span 元素的样式：

```
feature1 span.feature2 {property:value; property:value;...}
```

下面的代码定义了会应用给内嵌在带有 class="feature1"的段落元素中带有 class="feature2"的 span 元素的样式：

```
p.feature1 span.feature2 {property:value; property:value;...}
```

第一条 jQuery 语句会将定义在样式规则 greencolor 中的样式属性应用给带有 class="feature"的段落元素。第二条语句会将定义在样式规则 highlight 中的属性应用给定义在带有 class="feature"的段落元素中的 span 元素。也就是说，样式会应用给和标签之间的文字区域，它们是定义在带有 class="feature"的段落元素中的。应用了样式之后的页面如图 10-2 所示。

Styles make the formatting job much easier and efficient. To give an attractive look to web sites, styles are heavily used.

图 10-2 向内嵌在另一个 HTML 元素中的 span 元素应用样式

10.3 缩进段落

问题描述

现有一个 HTML 文件，其中有 3 个段落，要以 3 种不同的级别对其进行缩进。

解决方案

包含 3 段文字的 HTML 文件如下：

```
<body>
<p class="feature1">Styles make the formatting job much easier and efficient. To give
an attractive look to web sites, styles are heavily used. Styles can be written within
the HTML document or can be attached externally. External styles are considered
better</p>
<p class="feature2">jQuery is a powerful JavaScript library that allows us to add
dynamic elements to our web sites. Not only it is easy to learn, but it's easy to implement
```

```
too.</p>
<p class="feature3"> jQuery Selectors are used for selecting the area of the document
where we want to apply styles. jQuery has the power of handling events also, meaning
we can apply styles when a particular action takes place</p>
</body>
```

我们会看到，3 个段落被赋予不同的 class 名称：feature1、feature2 和 feature3。我们会使用 margin 属性来缩进这些段落。编写在外部样式表中的样式规则如下所示：

```
.indent1{
margin-left:10%;
}
.indent2{
margin-left:20%;
}
.indent3{
margin-left:30%;
}
```

为了向 3 个段落应用样式规则，我们编写的 jQuery 代码如下：

```
$(document).ready(function() {
  $('p.feature1').addClass('indent1');
  $('p.feature2').addClass('indent2');
  $('p.feature3').addClass('indent3');
});
```

知其所以然

第一条语句会从 HTML 文件中选定 class 为 feature1 的段落元素，然后将定义在样式规则 indent1 中的属性应用其上。类似地，第二和第三条语句会选定 class 为 feature2 和 feature3 的段落元素，并将定义在 indent2 和 indent3 中的属性分别应用其上。输出的显示效果如图 10-3 所示。

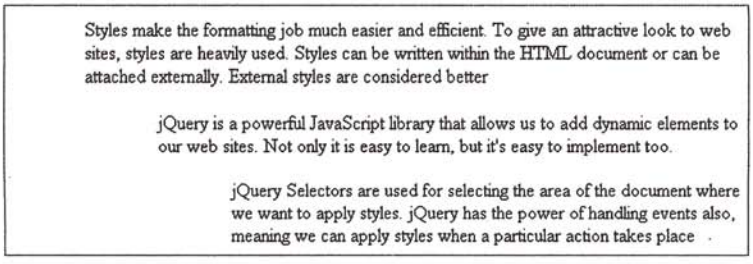


图 10-3　以不同级别缩进的三个段落

10.4　将段落的首字母设为大写

问题描述

要让段落的首字母大写。首字母大写可能以不同的字体或者不同的颜色显示，甚至还可以使

用图片来作为首字母大写。

解决方案

让我们根据下面带有一个段落元素的 HTML 文件来考虑这个问题：

```
<body>
<p><span class="cap">S</span>tyles make the formatting job much easier and efficient.
To give an attractive look to web sites, styles are heavily used. Styles can be written
within the HTML document or can be attached externally. External styles are considered
better
</body>
```

我们将要应用的样式规则是在样式表文件中编写的，如下所示：

```
.initialcap{
font-size: 2em;
}
```

将样式规则应用给 class 为 cap 的 span 元素的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('span.cap').addClass('initialcap');
});
```

知其所以然

我们会看到，在 HTML 文件中，段落的第一个字母与其他文字已经区分开了，它被包含在 span 标签中，并且 class 被赋值为 cap。对于这个 cap 类，我们通过 jQuery 代码为其应用了样式规则。我们会看到首字符的字体大小被设置为（段落其他部分所用）默认字体大小的 2 倍，如图 10-4 所示。

Styles make the formatting job much easier and efficient. To give an attractive look to web sites, styles are heavily used. Styles can be written within the HTML document or can be attached externally. External styles are considered better

图 10-4 段落的第一个字符被设置为首字符大写

我们还改变了首字符的背景色和前景色，样式规则如下所示：

```
.initialcap{
font-size:2em;
background-color:black;
color:white;
}
```

10.5 去除标题和段落之间的间隔

问题描述

当我们想要为任一段应用标题的时候，在标题和段落之间会有一段间隔。要去除这段间隔。

解决方案

带有标题和段落的 HTML 文件如下所示：

```
<body>
<h3>Formatting Makes Attractive</h3>
<p>Styles make the formatting job much easier and efficient. To give an attractive look
to web sites, styles are heavily used. Styles can be written within the HTML document
or can be attached externally. External styles are considered better</p>
</body>
```

移除段落和标题之间的间隔的样式规则如下所示：

```
.heading{
margin:0;
padding:0;
}

p{
margin:0;
padding:0;
}
```

将样式应用到 h3 元素上的 jQuery 代码如下：

```
$(document).ready(function() {
  $('h3').addClass('heading');
});
```

知其所以然

没有应用任何 jQuery 代码的原始 HTML 的输出效果如图 10-5 所示。我们可以看到在标题和段落之间有很大一段间隔。

Formatting Makes Attractive

Styles make the formatting job much easier and efficient. To give an attractive look to web sites, styles are heavily used. Styles can be written within the HTML document or can be attached externally. External styles are considered better

图 10-5　中间带有间隔的段落和标题

在上面的样式表中，我们在 h3 元素上应用了类型选择器 .heading，并对段落元素应用了类型选择器 p{}。

当向段落和标题应用样式的时候，二者之间的间隔就会被移除，如图 10-6 所示。

Formatting Makes Attractive
Styles make the formatting job much easier and efficient. To give an attractive look to web sites, styles are heavily used. Styles can be written within the HTML document or can be attached externally. External styles are considered better

图 10-6　在段落和标题之间通常会存在的间隔被移除

10.6 向标题文字应用样式

问题描述

要在标题上应用样式。

解决方案

我们会使用与攻略 10.5 中相同的 HTML 文件，其中包含有一个段落和一个标题。为了突出显示标题，首先我们需要移除通常在标题和段落之间会存在的间隔。

然后，我们会将其设置为斜体，并为其加上边框。我们会在样式表中编写下面的样式规则：

```
.heading{
margin:0;
padding:0;
font-style: italic;
border-top:5px solid black;
border-bottom:5px solid black;
}

p{
margin:0;
padding:0;
}
```

让我们使用下面的 jQuery 代码为 h3 元素应用样式：

```
$(document).ready(function() {
  $('h3').addClass('heading');
});
```

知其所以然

上述样式规则中的 margin 和 padding 属性会移除标题和段落之间通常会存在的间隔，而 font-style 会让标题以斜体显示。border 属性会在标题上添加顶部和底部的边框。

应用这些样式之后，段落的标题如图 10-7 所示。

Formatting Makes Attractive

Styles make the formatting job much easier and efficient. To give an attractive look to web sites, styles are heavily used. Styles can be written within the HTML document or can be attached externally. External styles are considered better

图 10-7 带有样式处理的标题的段落

10.7 缩进多个段落的第一行

问题描述

要让文档中每个段落的第一行都缩进。

解决方案

让我们先编写一个 HTML 文件，其中带有一些段落，如下所示：

```
<body>
<p>Styles make the formatting job much easier and efficient. To give an attractive look
to web sites, styles are heavily used. Styles can be written within the HTML document
or can be attached externally. External styles are considered better</p>
<p>jQuery is a powerful JavaScript library that allows us to add dynamic elements to
our web sites. Not only it is easy to learn, but it's easy to implement too.</p>
<p> jQuery Selectors are used for selecting the area of the document where we want to
apply styles. jQuery has the power of handling events also, meaning we can apply styles
when a particular action takes place</p>
</body>
```

这个 HTML 文件会以无缩进的方式显示段落。为了在段落的第一行应用缩进，我们需要使用 text-indent 属性。样式表中的样式规则如下所示：

```
.firstindent{
text-indent:10%;
}
```

将样式规则 firstindent 应用到 HTML 文件的所有段落元素上的 jQuery 代码如下：

```
$(document).ready(function() {
  $('p').addClass('firstindent');
});
```

知其所以然

应用了样式之后，HTML 文件中的段落的第 1 行都会缩进，如图 10-8 所示。

Styles make the formatting job much easier and efficient. To give an attractive look to web sites, styles are heavily used. Styles can be written within the HTML document or can be attached externally. External styles are considered better

jQuery is a powerful JavaScript library that allows us to add dynamic elements to our web sites. Not only it is easy to learn, but it's easy to implement too.

jQuery Selectors are used for selecting the area of the document where we want to apply styles. jQuery has the power of handling events also, meaning we can apply styles when a particular action takes place

图 10-8　带有首行缩进的段落

10.8　创建带有悬挂缩进的段落

问题描述

要让文档中每个段落的第一行都悬挂式缩进。

解决方案

在这里，我们会使用与攻略 10.6 中相同的 HTML 文件。其中带有 3 个段落元素。

我们会使用 text-indent 和 margin-left 属性来创建悬挂式缩进。样式规则如下所示：

```
.hangingindent{
text-indent:-10%;
margin-left:10%;
}
```

将 hangingindent 样式应用到段落上的 jQuery 代码如下：

```
$(document).ready(function() {
  $('p').addClass('hangingindent');
});
```

知其所以然

我们将 margin-left 属性设置为 10%，从而将段落设置在离浏览器的左边界 10%浏览器宽度的位置，即整个段落向右移动浏览器宽度 10%的距离。我们还将 text-indent 属性的值设置为 10%，从而使得段落的第一行向左移动浏览器宽度 10%的距离，这使得段落拥有悬挂缩进的效果，如图 10-9 所示。

Styles make the formatting job much easier and efficient. To give an attractive look to web sites, styles are heavily used. Styles can be written within the HTML document or can be attached externally. External styles are considered better

jQuery is a powerful JavaScript library that allows us to add dynamic elements to our web sites. Not only it is easy to learn, but it's easy to implement too.

jQuery Selectors are used for selecting the area of the document where we want to apply styles. jQuery has the power of handling events also, meaning we can apply styles when a particular action takes place

图 10-9 第一行带有悬挂缩进的段落

10.9 创建带有边框的提取引用

问题描述

在一大段文本的中间，要突出显示特定的文字来吸引读者的眼球。在此，要使用边框式引用。

解决方案

让我们编写带有 3 个段落的 HTML，并通过将 class 赋值为 feature 来突出显示我们想要识别的段落。HTML 代码如下所示：

```
<body>
<p>Styles make the formatting job much easier and efficient. To give an attractive look
```

```
to web sites, styles are heavily used. Styles can be written within the HTML document
or can be attached externally. External styles are considered better</a>
<p class="feature">jQuery is a powerful JavaScript library that allows us to add dynamic
elements to our web sites. Not only it is easy to learn, but it's easy to implement
too.</a>
<p> jQuery Selectors are used for selecting the area of the document where we want to
apply styles. jQuery has the power of handling events also meaning we can apply styles
when a particular action takes place</a>
</body>
```

我们会使用 margin、color 和 font-style 属性来突出显示文本。编写在外部样式表中的样式规则如下所示：

```
.quote{
margin:5%;
color:#00a;
font-style: italic;
border:5px solid black;
padding: .5em;
}
```

将 quote 样式规则应用给类名为 feature 的段落元素的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('p.feature').addClass('quote');
});
```

知其所以然

margin 属性使得段落与 4 个边界都保持 5%的距离，color 属性将段落文字的颜色设置为蓝色，而 font-style 属性会使其显示为斜体。为了创建围绕引用部分的边框，我们会在 quote 样式规则中再添加两个属性：border 属性会创建围绕段落的指定宽度的边框；padding 属性会在边框和段落文字之间创建间隔。

应用了样式规则之后，段落就会以边框式的提取引用样式显示，如图 10-10 所示。

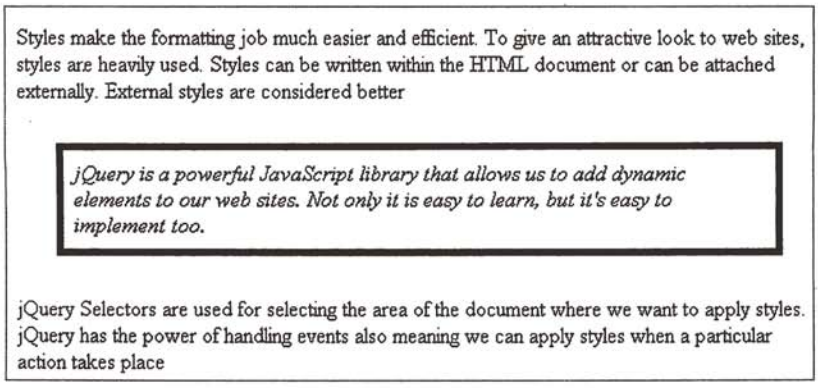


图 10-10　显示为边框式提取引用的段落

10.10 创建带有图片的提取引用

问题描述

为了让文本在一大段文本中吸引访问者的眼球，要创建一个像图片一样的提取引用，从而在文本占很大比重的文档中突出重点，而不需要完整的图片。

解决方案

在这里，我们会使用在攻略 10.7 中用过的 HTML。我们知道，在那个 HTML 中，我们想要从其他文本中识别出来的段落中 class 被赋值为 feature。

为了将图片应用到引用部分的左上角和右下角，我们需要创建两个图片：leftfig.jpg 和 rightfig.jpg。放在提取引用文字左上角的图片 leftfig.jpg，如图 10-11 所示。

放在提取引用文字右下角的图片，如图 10-12 所示。

图 10-11 leftfig.jpg

图 10-12 rightfig.jpg

我们需要对提取引用的文字应用两幅图片。我们只能对一个元素应用一种样式，因此为了向段落元素应用两幅图片，我们会将其放在一个 div 元素中。现在我们可以向段落元素应用一个样式来添加一幅图片，然后向 div 元素应用另一个样式来添加另一幅图片。将 class 为 feature 的段落放在 div 元素中的 HTML 文件如下：

```
<body>
<p>Styles make the formatting job much easier and efficient. To give an attractive look
to web sites, styles are heavily used. Styles can be written within the HTML document
or can be attached externally. External styles are considered better</p>
<div>
<p class="feature">jQuery is a powerful JavaScript library that allows us to add dynamic
elements to our web sites. Not only it is easy to learn, but it's easy to implement
too.</p>
</div>
<p> jQuery Selectors are used for selecting the area of the document where we want to
apply styles. jQuery has the power of handling events also, meaning we can apply styles
when a particular action takes place</p>
</body>
```

向提取引用部分应用两幅图片的样式规则如下：

```
.quote{
background-image:url(leftfig.jpg);
background-repeat: no-repeat;
margin:5%;
color:#00a;
font-style: italic;
```

```
padding:20px 5px 5px 20px;
}

.closing{
background-image:url(rightfig.jpg);
background-repeat: no-repeat;
background-position: bottom right;
}
```

向 class 为 feature 的段落元素添加样式规则 quote，并向 div 元素添加样式规则 closing 的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('p.feature').addClass('quote');
  $('div').addClass('closing');
});
```

知其所以然

样式规则 quote 会在段落的左上角显示 leftfig.jpg。而 background-repeat 的值被设置为 no-repeat，从而这幅图片只会显示一次。margin 属性使得段落的四方都缩进浏览器窗口宽度 5%的距离。font-style 属性使得段落以斜体显示，而 padding 属性设置了段落文本和图片之间的距离。样式规则 closing 会在段落的右下角显示 rightfig.jpg。

应用了样式之后，提取引用的文字的显示如图 10-13 所示。

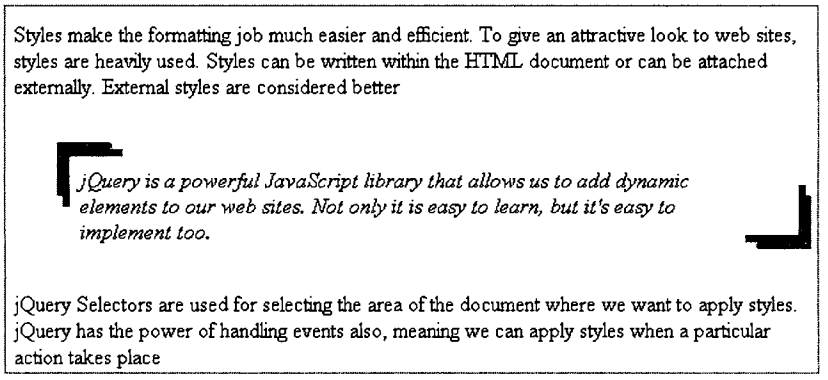


图 10-13　使用图片区分提取引用的段落

10.11　向列表项应用列表属性

问题描述

在显示下拉菜单的时候我们大量使用了列表项，它会显示列表的层次结构。要向列表项应用 list 属性。

解决方案

首先让我们来创建包含有列表项的 HTML 文件。它会像下面这样：

```
<body>
<ul>
  <li>Tea
    <ul>
      <li>Darjeeling</li>
      <li>Assam
        <ul>
          <li>Green Leaves</li>
          <li>Herbal</li>
        </ul>
      </li>
      <li>Kerala</li>
    </ul>
  </li>
  <li>Coffee
    <ul>
      <li>Cochin</li>
      <li>Kerala</li>
    </ul>
  </li>
</ul>
</body>
```

在向其应用 list 属性之前，列表项的显示效果如图 10-14 所示。

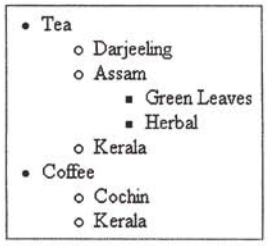


图 10-14 在应用样式之前的未排序列表项

让我们定义一项样式规则，如下：

```
.dispdisc{list-style-type:disc}
```

样式规则 dispdisc 会在列表项前面显示圆盘。向列表项应用样式规则 dispdisc 的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('li').addClass('dispdisc');
});
```

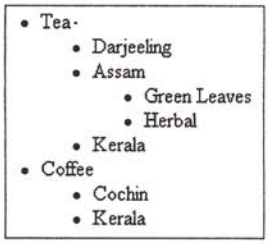


图 10-15 应用了 list 样式之后的未排序列表项

知其所以然

list 的样式类型被设置为 disc，我们在图 10-15 中可以看到，所有列表项前面都显示了圆盘的形状。

10.12 只对选定的列表项应用样式

问题描述

为了让一些列表项突出显示，我们想要对其应用样式。

解决方案

为了只对选定的列表项应用样式，我们需要从其他列表项中识别它们。为了选定一部分列表项，我们会为它赋予class名称或者ID。在本解决方案中，我们对于想要突出显示的列表项，将ID设置为intro：

```
<body>
<ul>
  <li>Tea
    <ul id="intro">
      <li>Darjeeling</li>
      <li>Assam
        <ul>
          <li>Green Leaves</li>
          <li>Herbal</li>
        </ul>
      </li>
      <li>Kerala</li>
    </ul>
  </li>
  <li>Coffee
    <ul>
      <li>Cochin</li>
      <li>Kerala</li>
    </ul>
  </li>
</ul>
</body>
```

让我们在样式表文件中定义一种样式规则，它会被应用给ID为intro的列表项：

```
.dispdisc{color:green;font-style:italic}
```

为了将定义在样式规则中的属性应用给ID为intro的列表项，我们编写的jQuery代码如下：

```
$(document).ready(function() {
  $('#intro').addClass('dispdisc');
});
```

1. 向使用子选择器选择的列表项应用样式

符号“>”是一种子组合符号，它会找到指定ID的元素的子列表项，并且向他们应用给定的样式规则。让我们将未排序的列表的ID设置为drink，如下所示：

```
<body>
<ul>
  <li>Tea
    <ul id="drink">
      <li>Darjeeling</li>
      <li>Assam
        <ul>
          <li>Green Leaves</li>
          <li>Herbal</li>
        </ul>
      </li>
```

```
        <li>Kerala</li>
      </ul>
    </li>
    <li>Coffee
      <ul>
        <li>Cochin</li>
        <li>Kerala</li>
      </ul>
    </li>
  </ul>
  </body>
```

让我们假设样式表中包含了样式规则 highlight，它会将文本颜色设置为绿色，并使其显示为斜体。

```
.highlight {
  font-style: italic;
  background-color: #0f0;
}
```

向 ID 为 drink 的未排序列表的子元素应用样式规则 highlight 的 jQuery 代码如下：

```
$(document).ready(function() {
  $('#drink >li').addClass('highlight');
});
```

2. 向没有应用CSS类的列表项应用样式

我们还可以向没有指定 CSS 类的元素应用样式。让我们在 JavaScript 文件中编写如下的 jQuery 代码：

```
$(document).ready(function() {
  $('#drink >li').addClass('highlight');
  $('#drink li:not(.highlight)').addClass('redandbold');
});
```

样式表中会包含两条样式规则：highlight 和 redandbold，如下所示：

```
.highlight {
font-style: italic;
background-color: #0f0;
}
.redandbold{
color:red;
font-family:arial;
font-weight:bold
}
```

知其所以然

上述的样式规则向 ID 为 intro 的列表项应用了 color 和 font-style 属性。图 10-16 只显示了突出显示的列表的一部分。

当我们使用子选择器的时候，它会找到 ID 为 drink 的元素的子元素的列表项，并将 highlight 属性应用其上。

图 10-17 显示了如何找到所有未应用 highlight 的列表项，并将定义在 redandbold 中的属性应用其上。

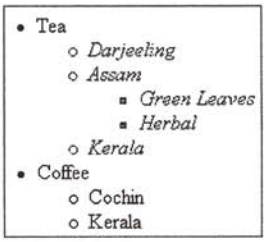


图 10-16　向 ID 为 intro 的列表项应用样式属性

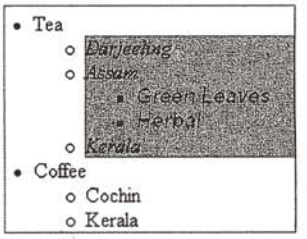


图 10-17　应用两种不同的样式

10.13　在列表项之间放置分隔线

问题描述

要列表项在一列中显示（没有缩进），并且每个列表项之间都由一条线分隔。

解决方案

在这里，我们会使用与攻略 10.10 中相同的 HTML。它显示了特定的列表项，如图 10-14 所示。

样式规则如下所示：

```
.applytopborders
{
border-top: 1px solid black;
}

.applybottomborder
{
border-bottom: 1px solid black;
}

.liststyle {
list-style-type:none;
margin: 0;
}
```

我们为未排序的列表应用了一种样式规则，对除最后一条之外的项目应用了另一种，而对于最后一条项目应用了第 3 种样式规则。

```
$(document).ready(function() {
  $('ul').addClass('liststyle');
  $('li').addClass('applytopborder');
  $('li:last').addClass('applybottomborder');
});
```

知其所以然

我们会在这个攻略中使用 3 种样式规则。

- liststyle——我们会在未排序列表上使用它来移除传统的列表符号，并移除列表项上的层级缩进。
- applytopborder——我们将这个属性应用给不是最后一个的列表项，在每个项目上面显示顶部的边框。
- applybottomborder——我们在最后一个列表项上应用这个属性，从而在它的底部显示边框。

应用上面的样式之后，我们得到的显示效果如图 10-18 所示。

Tea
Darjeeling
Assam
Green Leaves
Herbal
Kerala
Coffee
Cochin
Kerala

图 10-18 在一列上显示列表项，之间带有分隔线

10.14 向列表应用图片标记

问题描述

要在列表上使用图片来替换传统的列表符号。

解决方案

在这里，我们会创建一个 HTML 文件，其中显示有特定的列表项，如下所示：

```
<body>
<ul>
  <li>Tea
    <ul>
      <li>Darjeeling</li>
      <li>Assam
        <ul>
          <li>Green Leaves</li>
          <li>Herbal</li>
        </ul>
      </li>
      <li>Kerala</li>
    </ul>
  </li>
```

```
  <li>Coffee
    <ul>
      <li>Cochin</li>
      <li>Kerala</li>
    </ul>
  </li>
</ul>
</body>
```

在本节中，样式规则会使用两个属性：list-style-type 和 list-style-image。前者会被用于从列表项上移除传统的圆形符号，而后者会使用特定的图片来替换圆形符号。我们想要用来替换圆形符号的图片是 flower.jpg。样式表中的样式规则如下所示：

```
.liststyle {
list-style-type: none;
list-style-image:url(flower.jpg);
}
```

将样式规则 liststyle 应用到未排序列表上的 jQuery 代码如下：

```
$(document).ready(function() {
  $('ul').addClass('liststyle');
});
```

知其所以然

将 list-style-type 属性赋值为 none，使得列表项前面的圆形符号消失，而将 list-style-image 属性赋值为 flower.jpg，将存储在这个文件中的图片应用到列表项上。

当应用了样式规则之后，列表项中传统的圆形符号会被存储在 flower.jpg 中的图片所替换，而我们会得到图 10-19 中显示的效果。

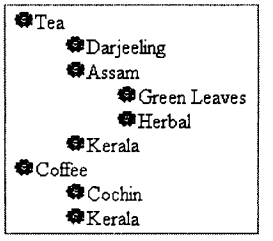


图 10-19　用图片替换了传统圆形符号的列表项

10.15　创建水平显示的列表

问题描述

让列表项显示在水平行上，而没有任何的层次。

解决方案

在本节中，我们会创建一个 HTML 文件，其中显示有特定的列表项，如下所示：

```
<body>
<ul>
  <li>Tea
    <ul>
      <li>Darjeeling</li>
      <li>Assam
        <ul>
```

```
            <li>Green Leaves</li>
            <li>Herbal</li>
          </ul>
        </li>
        <li>Kerala</li>
      </ul>
    </li>
    <li>Coffee
      <ul>
        <li>Cochin</li>
        <li>Kerala</li>
      </ul>
    </li>
  </ul>
  </body>
```

在本节中，样式规则会使用 display、list-style、margin 和 padding 属性，如下样式表中所示：

```
.liststyle {
display: inline;
list-style:none;
margin:0;
padding:0;
}
```

将样式规则 liststyle 应用到未排序列表和列表项上的 jQuery 代码如下：

```
$(document).ready(function() {
  $('ul').addClass('liststyle');
  $('li').addClass('liststyle');
});
```

知其所以然

Display 属性的值被设定为 inline，这使得列表项显示在一行上。将 list-style 的值设置为 none，这会从列表项上移除传统的圆形符号。最终，我们将 margin 和 padding 值设为 0，这会移除列表项上的层级缩进。

应用了样式属性之后，列表项会在一行上显示，而没有任何传统的圆形符号，如图 10-20 所示。

Tea Darjeeling Assam Green Leaves Herbal Kerala Coffee Cochin Kerala

图 10-20 显示在一行上的列表项

10.16 在超链接上应用样式

问题描述

传统情况下，超链接下面会有一条下划线，来和静态的文本相区分。要移除这条下划线，并为这些链接应用不同的样式。

解决方案

为了向超链接应用样式，让我们首先创建带有超链接的 HTML 文件，如下所示：

```
<body>
<div>Styles make the formatting job much easier and efficient. To give an attractive
look to web sites, styles are heavily used. A person must have a good knowledge of HTML
and CSS and a bit of JavaScript.
jQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only it is easy to learn, but it's easy to implement too.
jQuery is an open source project. <a href="abc.com">Click Here</a> for more information
</div>
</body>
```

我们可以看到，页面上的文本“Click Here”是一个超链接，下面带有下划线。当访问者选择这个链接的时候，就会被导航到 www.abc.com。

为了从超链接上移除下划线，并为其应用另一样式属性，我们会在外部样式表文件中编写下面的样式规则：

```
.linkstyle{
font-weight:bold;
background-color: #00f;
color:#fff;
text-decoration:none;
}
```

向超链接应用样式规则的 jQuery 代码如下：

```
$(document).ready(function() {
$('a[@href]').addClass('linkstyle');
});
```

现在让我们来看如何向发送邮件的超链接应用属性。下面是带有发送邮件超链接的 HTML 文件，选择该链接的时候，会打开邮件的客户端：

```
<body>
<div>Styles make the formatting job much easier and efficient. To give an attractive
look to web sites, styles are heavily used. A person must have a good knowledge of HTML
and CSS and a bit of JavaScript.
jQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only it is easy to learn, but it's easy to implement too.
jQuery is an open source project. <a href="mailto:bmharwani@yahoo.com">Contact Us</a>
for more information </div>
</body>
```

包含 jQuery 代码的 JavaScript 文件的内容如下：

```
$(document).ready(function() {
  $('a[@href^="mailto:"]').addClass('linkstyle');
});
```

知其所以然

我们在 CSS 样式规则中使用了 font-weight 属性，使得超链接加粗显示；使用 background-

color 属性将超链接的背景色设置为蓝色；使用 color 属性将文字设置为白色。我们还将 text-decoration 属性设置为 none，以从超链接上移除传统的下划线。

下面的 jQuery 语句：

```
$('a[@href]').addClass('linkstyle');
```

会选择文档中所有带有 href 特性的链接元素，并在其上应用 linkstyle 类。输出效果如图 10-21 所示。

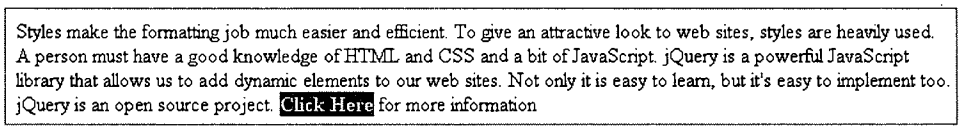


图 10-21 移除超链接上传统的下划线

下面的语句会选择文档中所有开头为 mailto 的带有 href 特性的链接元素，并在其上应用 linkstyle 类。

```
$('a[@href^="mailto:"]').addClass('linkstyle');
```

输出效果如图 10-22 所示。

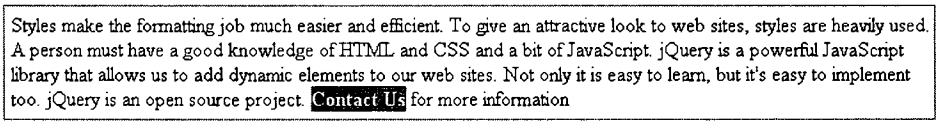


图 10-22 在发送邮件链接上应用 linkstyle 类

10.17 为 HTML 元素赋予不同的尺寸

问题描述

要限制特定段落元素的尺寸。

解决方案

对于这个解决方案，我们会创建带有两个段落元素的 HTML 文件，它们的 class 分别被赋值为 feature1 和 feature2，如下所示：

```
<body>
<p class="feature1">Styles make the formatting job much easier and efficient. To give
an attractive look to web sites, styles are heavily used. A person must have a good
knowledge of HTML and CSS and a bit of JavaScript.  </p>
<p class="feature2">jQuery is a powerful JavaScript library that allows us to add
dynamic elements to our web sites. Not only it is easy to learn, but it's easy to implement
too.  jQuery is an open source project that provides a wide range of features with
cross-platform compatiblity. jQuery has hundreds of plug-ins to extend its features.
jQuery helps in increasing interactions with a web site </p>
```

```
</body>
```

为了向 class 为 feature1 和 feature2 的段落元素应用 width 属性，我们需要编写下面的 jQuery 代码：

```
$(document).ready(function() {
  $('.feature1').css({'width':'50%', 'padding':'10px', 'border':'1px dashed'});
  $('.feature2').css({'padding':'30px', 'border':'2px solid'});
});
```

知其所以然

在上面的解决方案中，我们使用了 css()方法（曾经在第 3 章的攻略 3.7 中讨论过）。在 jQuery 代码中，第 1 条语句将第一段的宽度限定为浏览器窗口宽度的 50%。我们使用 border 属性创建了粗细为 1px 的点线边框，使用 padding 属性在段落文本和边框之间设定了 10px 的空间。第 2 条语句使得段落文本的宽度与浏览器窗口的宽度相同。我们使用 border 属性创建了粗细为 2px 的实线边框，使用 padding 属性在段落文本和边框之间设定了 30px 的空间。输出的显示效果如图 10-23 所示。

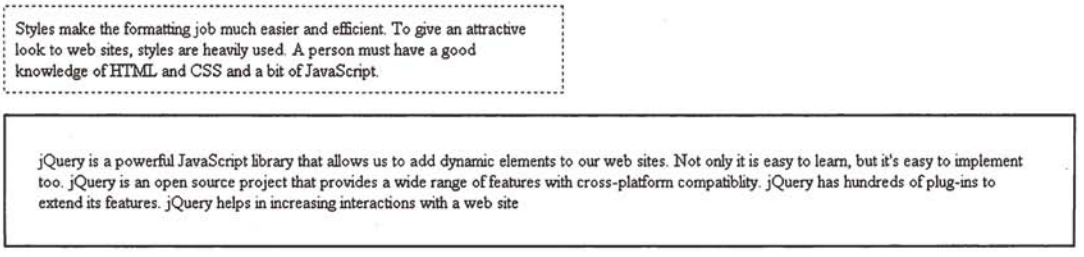


图 10-23　将 width 特性指定为百分比

我们还可以用像素的形式指定宽度，如下 jQuery 代码所示：

```
$('.feature1').css({'width':'300px', 'padding':'10px', 'border':'1px dashed'});
$('.feature2').css({'padding':'30px', 'border':'2px solid'});
```

第一段的宽度会被限定为 300px，如图 10-24 所示。

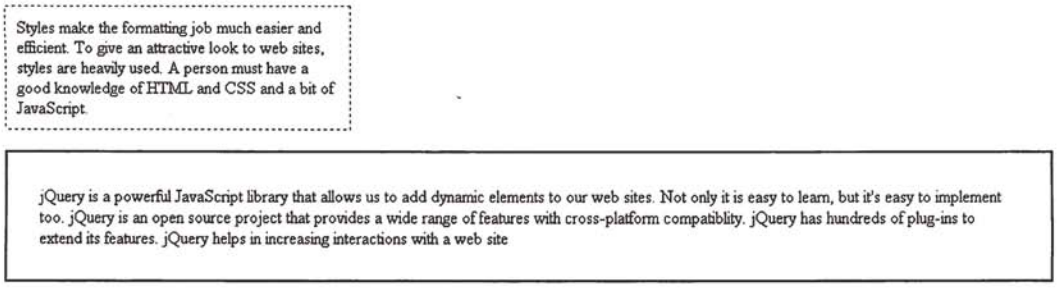


图 10-24　用像素的形式指定 width 特性

10.18 放置 HTML 元素

问题描述

让一个段落元素显示在另一个段落元素的左侧或者右侧。

解决方案

让我们来创建带有两个段落元素的 HTML 文件，它们的 class 分别被赋值为 feature1 和 feature2，如下所示：

```
<body>
<p class="feature1">Styles make the formatting job much easier and efficient. To give
an attractive look to web sites, styles are heavily used. A person must have a good
knowledge of HTML and CSS and a bit of JavaScript. </p>
<p class="feature2">jQuery is a powerful JavaScript library that allows us to add
dynamic elements to our web sites. Not only it is easy to learn, but it's easy to implement
too. jQuery is an open source project that provides a wide range of features with
cross-platform compatiblity. jQuery has hundreds of plug-ins to extend its features.
jQuery helps in increasing interactions with a web site </p>
</body>
```

为了向 class 为 feature1 和 feature2 的段落元素应用 float 属性，我们需要编写下面的 jQuery 代码：

```
$(document).ready(function() {
  $('.feature1').css({'width':'50%', 'border':'1px dashed', 'float':'left'});
  $('.feature2').css({'border':'2px solid'});
});
```

1. 创建两栏的布局

我们还可以让第一段显示在左侧，而第二段显示在右侧。让我们将 jQuery 代码修改如下：

```
$(document).ready(function() {
  $('.feature1').css({'width':'50%', 'border':'1px dashed', 'float':'left'});
  $('.feature2').css({'border':'2px solid', 'float':'right'});
});
```

2. 反转栏

我们还可以交换两栏的位置。即让第一段显示在右侧，第二段显示在左侧。实现这个目标的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.feature1').css({'width':'50%', 'border':'1px dashed', 'float':'right'});
  $('.feature2').css({'border':'2px solid', 'float':'left'});
});
```

知其所以然

在 jQuery 代码中，第一条语句将 float 设置为 left，使得第一段显示在浏览器窗口的左侧，并在右侧留出 50%的空白，这会被第二段所使用，如图 10-25 所示。border 属性会在第一段周

围设置粗细为 1px 的点线框。第二条语句在第二段周围设置粗细为 2px 的实线框。

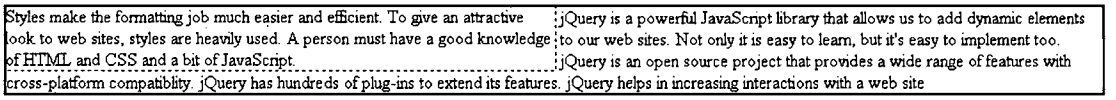


图 10-25　应用 float 属性

在两列的布局中，我们将第一段的 `float` 属性设置为 `left`，这样会在其右侧留出空间（这会被第二段所用）。类似地，当我们将第二段的 `float` 属性设置为 `right` 的时候，它会在浏览器的左侧留出空间，以供第一段所用。图 10-26 显示了这些样式的效果。

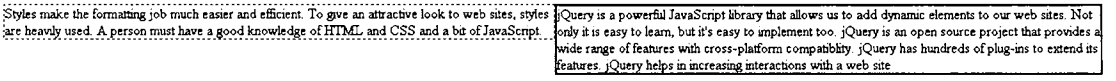


图 10-26　样式的效果

反转布局的效果如图 10-27 所示。

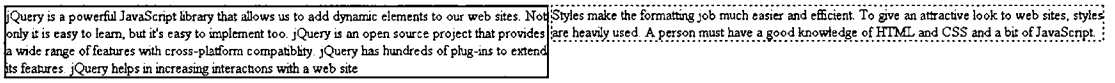


图 10-27　交换两栏

10.19　创建多栏的布局

问题描述

要创建 3 栏的布局；也就是，3 个段落在页面上显示在指定的位置。

解决方案

我们会通过将列放置在页面的不同位置来创建 3 栏的布局。首先，让我们创建带有 3 个段落元素的 HTML 文件，它们的 class 分别被赋值为 `leftalign`、`certeralign` 和 `rightalign`。HTML 文件会像下面这样：

```
<body>
<p class="leftalign">Styles make the formatting job much easier and efficient. To give
an attractive look to web sites, styles are heavily used. A person must have a good
knowledge of HTML and CSS and a bit of JavaScript.   </p>
<p class="centeralign">jQuery is a powerful JavaScript library that allows us to add
dynamic elements to our web sites. Not only it is easy to learn, but it's easy to implement
too. </p>
<p class="rightalign">jQuery is an open source project that provides a wide range of
features with cross-platform compatiblity. jQuery has hundreds of plug-ins to extend
its features. jQuery helps in increasing interactions with a web site. </p>
</body>
```

将 3 个段落元素放置在不同位置的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('.leftalign').css({'position':'absolute', 'left':'50px', 'width':'300px'});
  $('.centeralign').css({'position':'absolute', 'left':'400px', 'width':'300px'});
  $('.rightalign').css({'position':'absolute', 'left':'750px', 'width':'300px'});
});
```

1. 应用float属性

我们可以通过应用 `float` 属性来得到相同的输出效果（3 栏的布局），就像下面的解决方案中所演示的一样。在这个 HTML 文件中，我们只是定义了 3 个段落元素（而没有对 class 赋值）：

```
<body>
<p>Styles make the formatting job much easier and efficient. To give an attractive look
to web sites, styles are heavily used. A person must have a good knowledge of HTML and
CSS and a bit of JavaScript. </p>
<p>jQuery is a powerful JavaScript library that allows us to add dynamic elements to
our web sites. Not only it is easy to learn, but it's easy to implement too. </p>
<p>jQuery is an open source project that provides a wide range of features with
cross-platform compatiblity. jQuery has hundreds of plug-ins to extend its features.
jQuery helps in increasing interactions with a web site. </p>
</body>
```

然后我们编写了下列 jQuery 代码：

```
$(document).ready(function() {
  $('p').css({'float':'left',  'width':'300px','margin':'5px'});
});
```

2. 增加栏之间的空间

栏距指的是栏之间的空间。通过减少列的宽度，并增加 `margin` 的大小，我们可以增加栏之间的空间（栏距）。

让我们将 `width` 属性稍稍调小，而将 `margin` 值稍稍调大，如下 jQuery 代码所示：

```
$('p').css({'float':'left', 'width':'375px','margin':'15px'});
```

知其所以然

在“解决方案”标题下面的第一组 jQuery 代码中，我们看到第一条语句对 class 为 `leftaligh` 的 HTML 元素的 CSS 属性进行了设置。它将该段落元素的 `width` 设置为 300px，并将其放在离包含它的元素（在这种情况下是浏览器）左侧 50px 的位置。同样，第二条语句将 class 为 `centeralign` 的 HTML 元素放在离浏览器窗口左侧 400px 的位置上，而第三条语句将 class 为 `rightalign` 的 HTML 元素放在离浏览器窗口左侧 750px 的位置上。输出效果如图 10-28 所示。

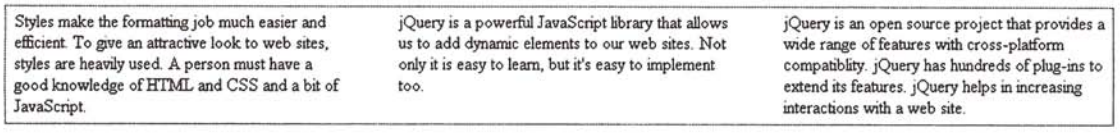


图 10-28 使用 `position` 属性达到的 3 栏布局效果

在使用 float 的示例中，我们将每个段落元素的 width 都设置为 300px，使得他们在浏览器窗口的左侧依次排列。第一段会先显示，宽度为 300px。在 5px 的空白之后显示了第二段；也就是距离浏览器的窗口有 305px 的距离。第二段的宽度也是 300px。最终，第三段会显示在离第二段 5px 的位置上，即显示在浏览器窗口的最右侧，如图 10-29 所示。

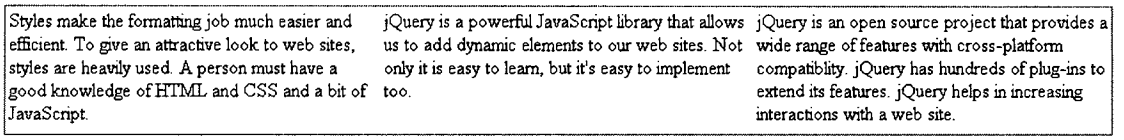


图 10-29　使用 float 属性达到的 3 栏布局效果

最终，图 10-30 显示了增加栏距之后的效果。

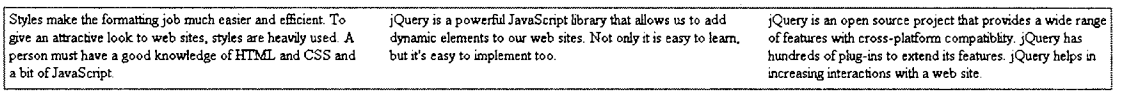


图 10-30　增加了栏距之后的 3 栏效果

10.20　使文字围绕图片显示

问题描述

通常，当我们在页面上显示图片和文字的时候，要么是图片在文字之后，要么是文字在图片之后（这与他们在 HTML 文件中的顺序有关）；二者在默认情况下不会在相邻的位置显示。但有时想让文字围绕图片显示。

解决方案

让我们在 HTML 文件中放置一幅图片，如下所示：

```
<body>
<img src="cell.jpg"/>
</body>
```

现在我们会编写 jQuery 代码来将 img 元素放在 div 元素中，并在 div 元素中增加带有一些文本的段落元素。jQuery 代码如下所示：

```
$(document).ready(function() {
  $('img').wrap('<div></div>');
  $('<p>Styles make the formatting job much easier and efficient. To give an attractive
look to web sites, styles are heavily used. A person must have a good knowledge of HTML
and CSS and a bit of JavaScript. jQuery is a powerful JavaScript library that allows
us to add dynamic elements to our web sites. Not only it is easy to learn, but it's
easy to implement too.  jQuery is an open source project that provides a wide range
of features with cross-platform compatiblity. jQuery has hundreds of plug-ins to extend
its features. jQuery helps in increasing interactions with a web site.
```

```
</p>').appendTo('div');
  $('img').css({'float':'left',  'width':'200px','height':'200px'});
  $('p').css({'clear':'right'});
});
```

知其所以然

应用在图片上的 CSS 属性使用 float 属性让图片显示在浏览器窗口的左侧（让文本显示在右侧）；并使用 width 属性来将图片的大小设置为 200px（任何比浏览器窗口的总宽度小的尺寸），这样就为围绕图片显示的文本留出了空间。我们还是用了 height 属性来将图片的高度设置为特定的大小。

我们将 clear 属性设置为 right，这会使得多出的段落文本移至左侧。也就是说，它会试图在右侧留出空间，而填满左侧的空间。这样，段落中超出图片高度而剩余的文本会移至左侧，从而文本会包围图片。

当我们向图片和段落元素应用上述的 CSS 属性时，会得到图 10-31 所显示的效果。

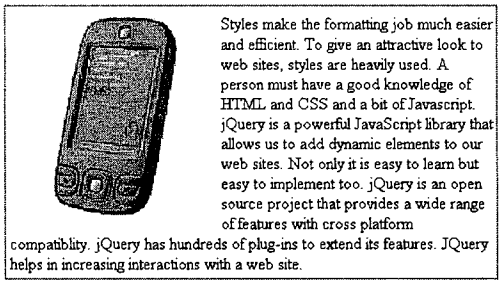


图 10-31 文本围绕图片

10.21 在图片背后放置阴影

问题描述

要在图片之后设置阴影。

解决方案

为了创建阴影，我们首先需要创建两幅图片：一幅作为图片右侧的阴影；另一幅是为了图片底部的阴影效果所创建的。让我们将图片右侧的阴影图片命名为 shadowright.jpg，如图 10-32 所示。

图 10-32 右侧的阴影图片

类似地，图片底部的阴影图片被命名为 shadowbottom.jpg，显示效果如图 10-33 所示。

图 10-33 底部的阴影图片

假设存在名为 image4.jpg 的图片文件，显示这幅图片的 HTML 代码如下所示：

```
<body>
<span class="shadow"><img src="image4.jpg" /></span>
</body>
```

将要应用给 img 元素和 span 元素的样式规则是通过 css()方法赋予的。如以下 jQuery 代码所示：

```
$(document).ready(function() {
  $('span').css({'background':'url(shadowright.jpg)', 'background-repeat': 'no- re-
peat','background-position':'bottom right', 'padding':'0 10px 0 0'});
  $('img').css({'width':'200px','height':'200px','background':'url(shadowbottom.jpg)',
'background-repeat':'no-repeat','background-position':'bottom', 'padding':'0 0 10px
0' });
});
```

知其所以然

在上面的 HTML 文件中，img 元素嵌入在 span 元素中，因为我们需要对 img 元素应用两条样式规则。一条是为了在图片的右侧显示阴影，另一条是为了在图片的底部显示阴影。但是我们对同一元素无法应用一条以上的样式规则。因此，为了向 img 元素应用两条样式规则，我们会将其放在 span 元素中，这样一条样式规则可以应用给 span 元素(它最终还是会应用给 img 元素)，而另一条样式规则可以应用给 img 元素本身。

第一条 css()调用包含 4 个属性。

- 我们设置 background:url 属性，以将存储在文件 shadowright.jpg 中的图片作为背景图片显示。
- 我们将 background-repeat 属性设置为 no-repeat，使阴影图片只显示一次。
- 我们将 background-position 属性设置为 bottom right，以在图片的右下侧显示阴影图片。
- Padding 属性被用于设置阴影图片和实际图片之间的距离。这有助于确定阴影的宽度。

类似地，第二条 css()调用包含 6 个属性。

- Width 和 height 属性被设置为 200px，从而将实际图片的显示宽度和高度设置为 200 像素。
- 我们设置 background:url 属性，以将存储在文件 shadowbottom.jpg 中的图片作为背景图片显示。
- 我们将 background-repeat 属性设置为 no-repeat，使阴影图片只显示一次。
- background-position 属性被设置为 bottom，从而在实际图片的底部显示阴影图片。
- padding 属性被用于设置阴影图片和实际图片之间的距离。

向 HTML 文件中的图片应用了上述属性之后，图片会在右侧以及底部显示阴影效果，如图

10-34 所示。

图 10-34 带有阴影的图片

10.22 当鼠标移过链接的时候改变鼠标样式

问题描述

在鼠标移过链接的时候改变它的样式。

解决方案

对于这个问题，我们创建了一个 HTML 文件，其中在 div 元素中包含了一些信息，还有一个超链接（Click Here），选择它的时候，会将我们导航到 www.abc.com。HTML 文件如下所示：

```
<body>
<div>Styles make the formatting job much easier and efficient. To give an attractive
look to web sites, styles are heavily used. A person must have a good knowledge of HTML
and CSS and a bit of JavaScript.
jQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only it is easy to learn, but it's easy to implement too.
jQuery is an open source project. <a href="abc.com">Click Here</a> for more information
</div>
</body>
```

为了向超链接应用不同的 cursor 属性，我们会编写如下的 jQuery 代码：

```
$(document).ready(function() {
  $(a).hover(
    function(){
      $(this).css({'cursor': 'wait', 'color': 'blue' , 'background-color':'cyan'});
    },
    function(){
      $(this).css({'cursor': 'default', 'color': '#000000' , 'background-color':'#ff-
ffff'});
    });
});
```

知其所以然

hover()方法中包含两项功能：一项会在鼠标指针移过选定的元素时执行，另一项会在鼠标指针移出选定元素时执行。初始的时候，输出的效果如图 10-35 所示。我们可以看到超链接带有下划线，并且鼠标指针是默认的箭头指针。

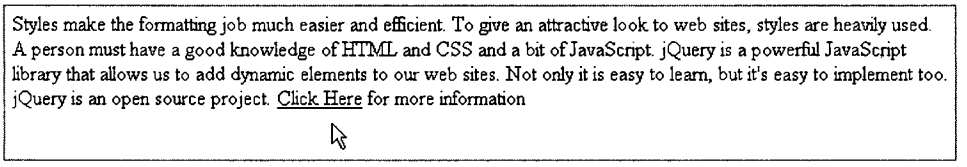


图 10-35　没有在链接上悬停时的默认指针

当鼠标指针移过链接的时候，在 hover()中定义的 CSS 属性会应用其上，从而将鼠标指针变为沙漏样式，并将链接的背景色变为青绿色，前景色变为蓝色，如图 10-36 所示。

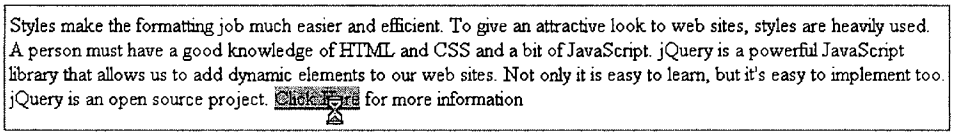


图 10-36　在链接上悬停时，鼠标指针发生了变化

10.23　在指定的区域中显示长文字

问题描述

要在指定的区域中显示长文字。

解决方案

在下面的 HTML 文件中，我们定义了一个段落元素，并且希望将其限制在页面的指定区域之内。

```
<body>
<p>Styles make the formatting job much easier and efficient. To give an attractive look
to web sites, styles are heavily used. A person must have a good knowledge of HTML and
CSS and a bit of JavaScript.  <br/>
jQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only it is easy to learn, but it's easy to implement too.  jQuery is
an open source project that provides a wide range of features with cross-platform
compatiblity. jQuery has hundreds of plug-ins to extend its features. jQuery helps in
increasing interactions with a web site </p>
</body>
```

为了限定文本的位置，并且向段落元素应用 overflow 属性，我们使用了下面的 jQuery 代码：

```
$('p').css({'width':'50%', 'height':'100px','overflow':'scroll'});
```

知其所以然

我们将段落的宽度设置为浏览器窗口宽度的 50%，将高度设置为 100px。我们还将 overflow 的值设置为 scroll，这样当段落元素中的文本在指定高度和宽度的区域内无法完全显示的时候，就会显示滚动条。输出的效果如图 10-37 所示。

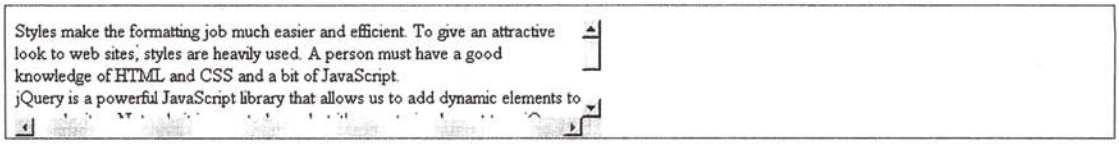


图 10-37 应用带有 scroll 选项的 overflow 属性

让我们看下，将 overflow 的值设置为 hidden 时，会发生什么。这样超出指定区域的段落文本就见不到了，如图 10-38 所示。

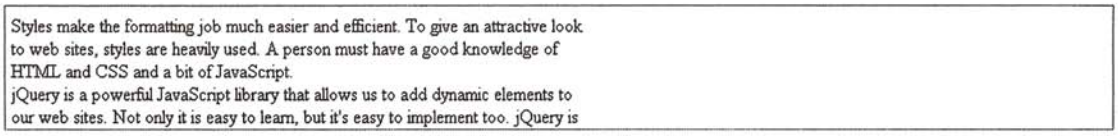


图 10-38 应用带有 hidden 选项的 overflow 属性

让我们将 overflow 属性的值设置为 auto。这样只会显示垂直滚动条，而不会显示水平滚动条（这与将值设为 scroll 时不同）；也就是说，只有在需要的时候才会显示滚动条。应用了样式之后的输出效果如图 10-39 所示。

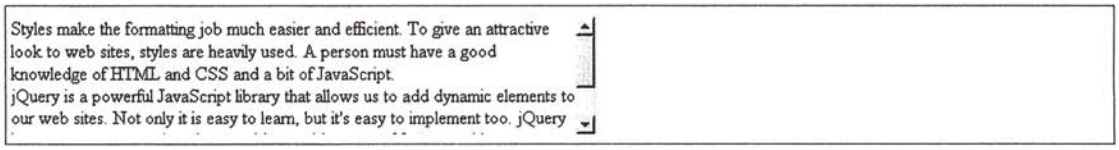


图 10-39 应用带有 auto 选项的 overflow 属性

现在让我们将 overflow 属性的值设置为 visible。段落的文本会全部可见；也就是说，它不会被指定的区域所限制（如图 10-40 所示）。

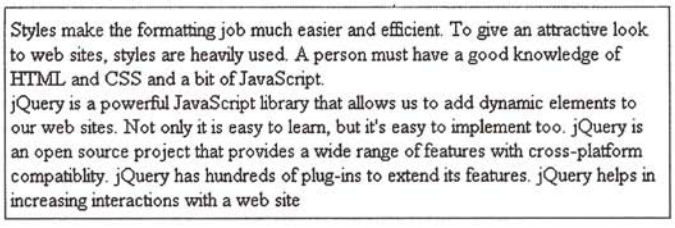


图 10-40 应用带有 visible 选项的 overflow 属性

10.24　创建圆角的栏

问题描述

创建带有圆角的单独的栏。

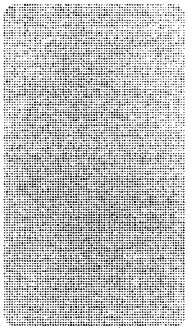


图 10-41　圆角矩形

解决方案

为了将文字栏的形状设置为圆角矩形，我们需要创建带有圆角的矩形的图片，并将其作为文本的背景。让我们创建一个像图 10-41 这样的圆角矩形，并将其命名为 columnfig.jpg。

这个图片会被设置为文本的背景。让我们创建一个 HTML 文件，其中有一个带有一些文本的段落元素，如下所示：

```
<body>
<p>Styles make the formatting job much easier and efficient. To give an attractive look
to web sites, styles are heavily used. A person must have a good knowledge of HTML and
CSS and a bit of JavaScript.
jQuery is powerful JavaScript library used to make dynamic sites.</p>
</body>
```

在样式表中，让我们编写一条样式规则，从而将圆角矩形作为段落元素的背景。样式规则会包含 width、padding、background 和 background-repeat 属性，如下所示：

```
.backfig{
width:150px;
padding:10px;
background:url(columnfig.jpg);
background-repeat:no-repeat;
}
```

让我们来编写 jQuery 代码，从而将 backfig 样式规则应用给段落元素。jQuery 代码如下所示：

```
$(document).ready(function() {
  $('p').addClass('backfig');
});
```

知其所以然

backfig 样式规则会为段落文本设置宽度，这个宽度与圆角矩形的宽度是一致的，这样，段落的文本就会被限定在该矩形的圆角之内。padding 属性被用于在矩形边界和段落文本之间保持一定距离。background 属性是用来将存储在 columnfig.jpg 文件中的圆角矩形图片设置为段落文本的背景的，而 background-repeat 的值被设置为 no-repeat，从而使得圆角矩形只显示一次。

在向背景应用了圆角矩形图片之后，段落文本的显示效果如图 10-42 所示。

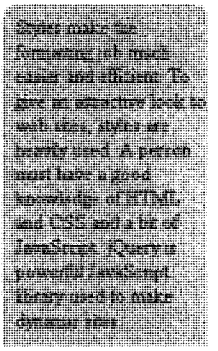


图 10-42　带有圆角的单独栏

10.25 应用文字装饰

问题描述

要对特定的文本应用文字装饰，如上划线和下划线样式，以引起注意。此外，为了与其他文字比较，要应用像删除线这样的效果，例如，想要显示某种商品以前的折扣和当前的折扣。

解决方案

下面是包含 3 个段落元素的 HTML 文件，3 个段落元素的 class 分别被赋予：feature1、feature2 和 feature3。

```
<body>
<p class="feature1">jQuery is powerful</p>
<p class="feature2">Styles make the formatting job much easier and efficient. To give
an attractive look to web sites, styles are heavily used. A person must have a good
knowledge of HTML and CSS and a bit of JavaScript. jQuery is powerful JavaScript library
used to make dynamic sites.</p>
<p class="feature3">10% Discount on all products</p>
<p>20% Discount  on all products</p>
</body>
```

向段落元素应用文本装饰的 jQuery 代码如下所示：

```
$(document).ready(function() {
  $('p.feature1').css({'text-decoration':'underline'});
  $('p.feature2').css({'text-decoration':'overline'});
  $('p.feature3').css({'text-decoration':'line-through'});
});
```

我们还可以对标题（也就是 class 为 feature1 的段落）同时应用上划线和下划线，以使其突出显示。

```
<div>
<p class="feature1">jQuery is powerful</p>
</div>
```

现在样式规则变为下面的样子：

```
$(document).ready(function() {
  $('p.feature1').css({'text-decoration':'underline'});
  $('div').css({'text-decoration':'overline'});
  $('p.feature3').css({'text-decoration':'line-through'});
});
```

知其所以然

第一次对 css()方法的调用将 class 为 feature1 的段落显示为带有下划线的文本。第二次调用将 class 为 feature2 的段落显示为带有上划线的文本。第三次调用将 class 为 feature3 的段落显示为带有删除线的文本。输出的效果如图 10-43 所示。

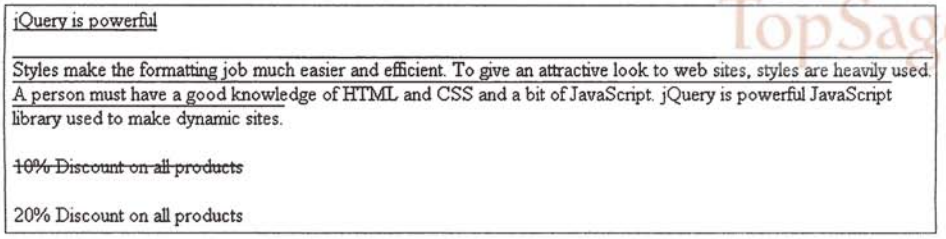


图 10-43　对不同的 class 应用不同的文本装饰方法

使用删除线的优势在于，我们可以显示之前提供的价格是多少，并且能够与当前提供的价格相比较。就像前面的解决方案所显示的，之前的折扣率是 10%，而现在是 20%。

为了突出显示标题，由于我们不能对相同的元素应用两条样式规则，所以我们将 class 为 `feature1` 的段落嵌入在 `div` 元素中，从而我们可以将一条样式规则应用给 `div`，而另一条应用给段落元素。第一条规则使得 class 为 `feature1` 的段落带有下划线显示。而第二条样式规则使得 `div` 元素中的内容——也就是 class 为 `feature2` 的段落——显示有上划线。第三条样式规则，正如你已经看到的，使得 class 为 `feature3` 的段落带有删除线显示。输出效果如图 10-44 所示。

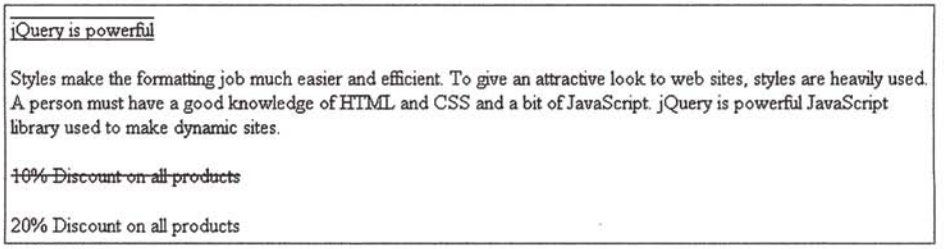


图 10-44　对标题应用上划线和下划线

10.26　缩放图片

问题描述

想让一幅图片可缩放。也就是说，如果容纳图片的区域缩小了，图片也会自动缩小。同样，如果该区域被放大，图片也会自动放大。

解决方案

我们会使用与攻略 10.19 中相同的 HTML（最终的效果请参见图 10-30）。这次，我们会将图片的宽度以包含的 block 元素的百分比的形式定义。由于图片的包含区块是浏览器窗口，所以图片的宽度会根据浏览器窗口大小的改变而缩放。修改后的样式规则如下所示：

```
.moveleft
{
```

```
width:40%;
float:left;
}

.imagewrap {
clear:right;
}
```

向图片和段落元素应用样式规则的 jQuery 代码如下：

```
$(document).ready(function() {
  $('img').addClass('moveleft');
  $('p').addClass('imagewrap');
});
```

知其所以然

如果我们增加攻略 10.19 中浏览器窗口宽度，图片大小会保持不变，如图 10-45 所示。

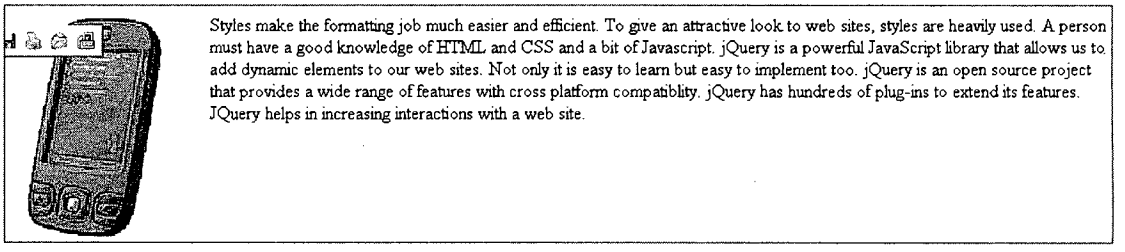


图 10-45 增大浏览器窗口的宽度时，图片没有放大

图片没有被缩放的原因在于应用在上面的原始规则。已经添加的样式如下所示，供你参考：

```
.moveleft
{
width:200px;
height:200px;
float:left;
}
```

图片的宽度被锁定为 200px，因此，不管我们如何改变浏览器窗口的大小，图片的宽度都会保持不变。

新的 moveleft 样式规则包含两个属性。

- width 属性，它被设置为浏览器窗口宽度的 40%（即当浏览器窗口的宽度发生改变时，图片的宽度也会按照 40%的比率改变）。
- float 属性，它被设置为 left，从而使得图片位于浏览器窗口的左侧，为显示在右侧的段落文本留出空间。

样式规则 imagewrap 将会被应用给段落文本，并包含单独的属性 clear，使得多出来的段落文本（超出图片之外的文本）显示在图片下面，使得图片被文本所包围。这个属性的影响只有在调整浏览器窗口的大小时才能够看到，此时段落文本会超过图片的高度。

应用了这些样式之后，图片变得可以根据浏览器窗口的大小来缩放，如图 10-46 所示。

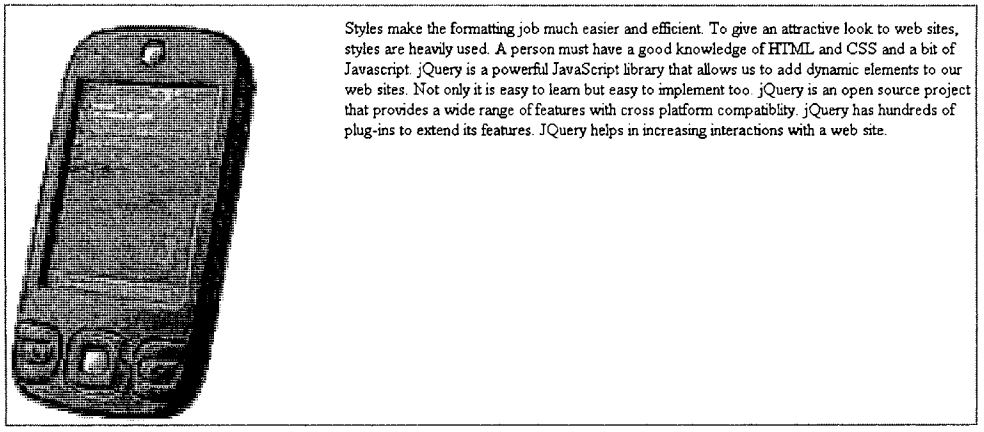


图 10-46　图片会根据浏览器窗口的宽度缩放

10.27　设置背景图片

问题描述

想用一幅图片来做文本的背景。

解决方案

假设我们拥有下面的 HTML 文件，其中包含显示简单文本的段落元素：

```
<body>
<p>Styles make the formatting job much easier and efficient. To give an attractive look
to web sites, styles are heavily used. A person must have a good knowledge of HTML and
CSS and a bit of JavaScript.
jQuery is a powerful JavaScript library that allows us to add dynamic elements to our
web sites. Not only it is easy to learn, but it's easy to implement too.
jQuery is an open source project that provides a wide range of features with
cross-platform compatiblity. jQuery has hundreds of plug-ins to extend its features.
jQuery helps in increasing interactions with a web site. </p>
</body>
```

为了将一幅图片设置为文本的背景，我们需要编写下列样式规则：

```
.placeimage
{
background-image:url(cell.jpg);
background-repeat:no-repeat;
}
```

现在我们需要编写 jQuery 代码来将样式规则 placeimage 应用在 body 标签上。jQuery 代码如下：

```
$(document).ready(function() {
  $('body').addClass('placeimage');
});
```

知其所以然

假设图片文件 cell.jpg 已经存在。在样式规则 placeimage 中，我们使用了两个属性：background-image 和 background-repeat。借助于 background-image，我们让存储在 cell.jpg 中的图片作为文本的背景显示。默认情况下，图片会重复多次，以填满容纳文本的区域。因此，我们将 background-repeat 属性的值设置为 no-repeat，从而图片作为背景只显示一次。

向 HTML 文件的 body 应用了 placeimage 样式规则之后，存储在 cell.jpg 中的图片会作为文本的背景显示，如图 10-47 所示。

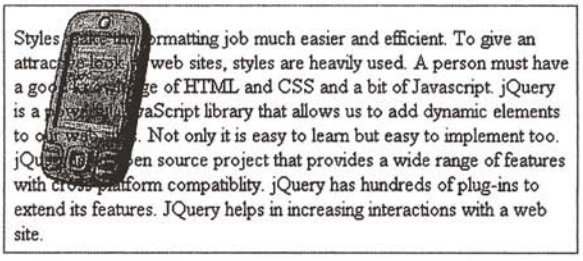


图 10-47 图片被设置为文本的背景

10.28 使背景图片在浏览器中央显示

问题描述

通常，当我们将图片设置为背景的时候，它会显示在浏览器窗口的左侧。让背景图片显示在浏览器窗口的中间。

解决方案

在本节中，我们会使用与攻略 10.26 中相同的 HTML 文件，它会将一幅图片作为背景显示（左对齐）。为了使居左排列的背景图片显示在屏幕的中间，我们会使用 background-position 属性。通过将 background-position 属性的值设置为 center，背景图片会显示在浏览器屏幕的中间。

这样，让我们在样式规则 placeimage（来自于攻略 10.26）中添加 background-position，如下所示：

```
.placeimage
{
background-image:url(cell.jpg);
background-repeat:no-repeat;
background-position:center;
```

```
}
```

将 placeimage 样式规则应用给 HTML 文件的 body 的 jQuery 代码如下：

```
$(document).ready(function() {
  $('body').addClass('placeimage');
});
```

知其所以然

让我们回忆一下，background-image 属性会让 cell.jpg 中的图片作为背景显示。为了让背景图片只显示一次，也就是防止它重复显示以填满整个区域，我们将 background-repeat 属性的值设置为 no-repeat。最终，通过将 background-position 属性的值设置为 center，我们确保背景图片显示在浏览器窗口的中间。

向 HTML 文件的 body 应用了 placeimage 样式规则之后，存储在 cell.jpg 中的图片会作为背景显示在浏览器窗口的中间，如图 10-48 所示。

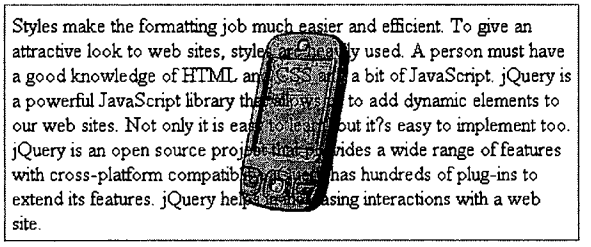


图 10-48　背景图片被放在浏览器窗口的中间

10.29　保持背景图片固定

问题描述

当我们浏览网页的时候，如果向上、向下滚动页面，图片和文本都会随之滚动。要在滚动页面的时候使背景图片能够保持不动。

解决方案

在本节中，我们会使用与攻略 10.26 相同的 HTML 文件。在此，我们会向段落中添加更多的文本，使得它足够大，以便于我们可以滚动页面。为了让背景图片在滚动页面的时候保持固定，我们使用了 background-attachment 属性。

通过将 background-attachment 属性设置为 fixed，我们可以使背景图片保持不动。那么，让我们将 background-attachment 属性添加到用于 10.27 节的样式表中。

```
.placeimage
{
background-image:url(cell.jpg);
```

```
background-repeat:no-repeat;
background-position:center;
background-attachment: fixed;
}
```

将 placeimage 样式规则应用给 HTML 文件的 body 的 jQuery 代码如下：

```
$(document).ready(function() {
  $('body').addClass('placeimage');
});
```

知其所以然

在样式规则 placeimage 中使用的每个属性的功能如下。

- background-image 属性会让 cell.jpg 文件中的图片作为背景显示。
- 通过将 background-repeat 属性的值设置为 no-repeat，背景图片被设置为只显示一次。
- background-position 属性被设置为 center，这使得背景图片显示在浏览器窗口的中间。
- background-attachment 属性被设置为 fixed，这使得背景图片会在我们滚动页面的时候保持不动。

向 HTML 文件的 body 应用了 placeimage 样式规则之后，图片会作为背景显示，如图 10-49 所示。

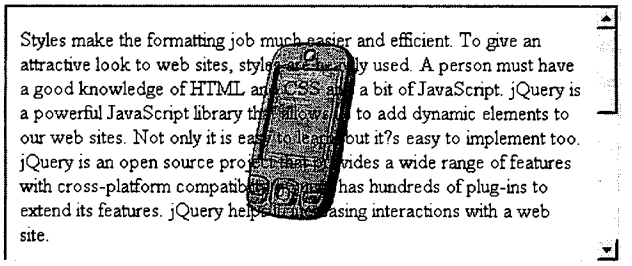


图 10-49 当文本区域比浏览器窗口大的时候，就会显示滚动条

现在，当我们向下滚动页面的时候（通过右侧的滚动条），背景图片会保持在浏览器屏幕中间不动，只有文本会滚动（见图 10-50）。

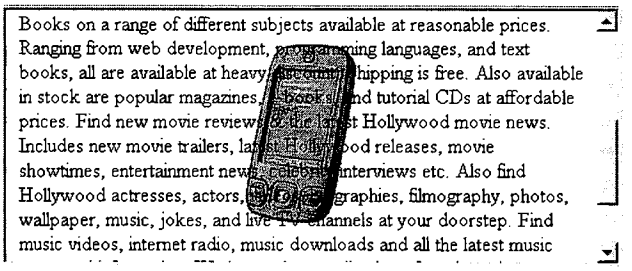


图 10-50 即便在文本滚动的时候，背景也会保持不动

10.30　小结

本章中，我们学习了关于不同CSS技术的攻略。它们经常会应用在网页上，用来识别HTML元素，向内嵌在一个元素中的另一个元素应用样式，缩进段落，对段落使用首字母大写，移除标题和段落之间的间隔，向标题文本应用样式以及缩进多个段落的第一行等等。除此之外，我们还学习了以下过程：创建带有悬挂缩进的段落，创建边框式的提取引用，创建带有图片的提取引用，向列表项应用列表属性，只对选定的列表项应用样式，在列表项之间放置分隔线，向列表应用图片符号并创建横排的列表。最后，我们还学习了如何向超链接和邮件链接应用样式，如何为HTML元素应用不同的尺寸，如何放置HTML元素，如何创建多栏的布局，如何使文本围绕图片显示，如何在图片后面放置阴影，如何在鼠标在链接上悬停的时候改变鼠标指针，如何在特定的区域内显示大段文本，如何创建圆角矩形的栏，如何应用文本装饰，如何缩放图片，如何设置背景图片，如何将背景图片放在浏览器中间，以及如何使背景图片保持固定不动等。

计算机精品学习资料大放送

软考官方指定教材及同步辅导书下载 | 软考历年真是解析与答案

软考视频 | 考试机构 | 考试时间安排

Java 一览无余：Java 视频教程 | Java SE | Java EE

.Net 技术精品资料下载汇总：ASP.NET 篇

.Net 技术精品资料下载汇总：C#语言篇

.Net 技术精品资料下载汇总：VB.NET 篇

撼世出击：C/C++编程语言学习资料尽收眼底 电子书+视频教程

Visual C++(VC/MFC)学习电子书及开发工具下载

Perl/CGI 脚本语言编程学习资源下载地址大全

Python 语言编程学习资料(电子书+视频教程)下载汇总

最新最全 Ruby、Ruby on Rails 精品电子书等学习资料下载

数据库精品学习资源汇总：MySQL 篇 | SQL Server 篇 | Oracle 篇

最强 HTML/xHTML、CSS 精品学习资料下载汇总

最新 JavaScript、Ajax 典藏级学习资料下载分类汇总

网络最强 PHP 开发工具+电子书+视频教程等资料下载汇总

UML 学习电子资下载汇总 软件设计与开发人员必备

经典 LinuxCBT 视频教程系列 Linux 快速学习视频教程一帖通

天罗地网：精品 Linux 学习资料大收集(电子书+视频教程) Linux 参考资源大系

Linux 系统管理员必备参考资料下载汇总

Linux shell、内核及系统编程精品资料下载汇总

UNIX 操作系统精品学习资料<电子书+视频>分类总汇

FreeBSD/OpenBSD/NetBSD 精品学习资源索引 含书籍+视频

Solaris/OpenSolaris 电子书、视频等精华资料下载索引