为博客增加<time>元素

编辑“blog.html”文件，并在各个文章标题下增加以下日期：

```
<article>
  <h1>Starbuzz meets social media</h1>
  <time datetime="2012-03-12">3/12/2012</time>
  ...
</article>
<article>
  <h1>Starbuzz uses computer science</h1>
  <time datetime="2012-03-10">3/10/2012</time>
  ...
</article>
<article>
  <h1>Most unique patron of the month</h1>
  <time datetime="2012-02-18">2/18/2012</time>
  ...
</article>
```

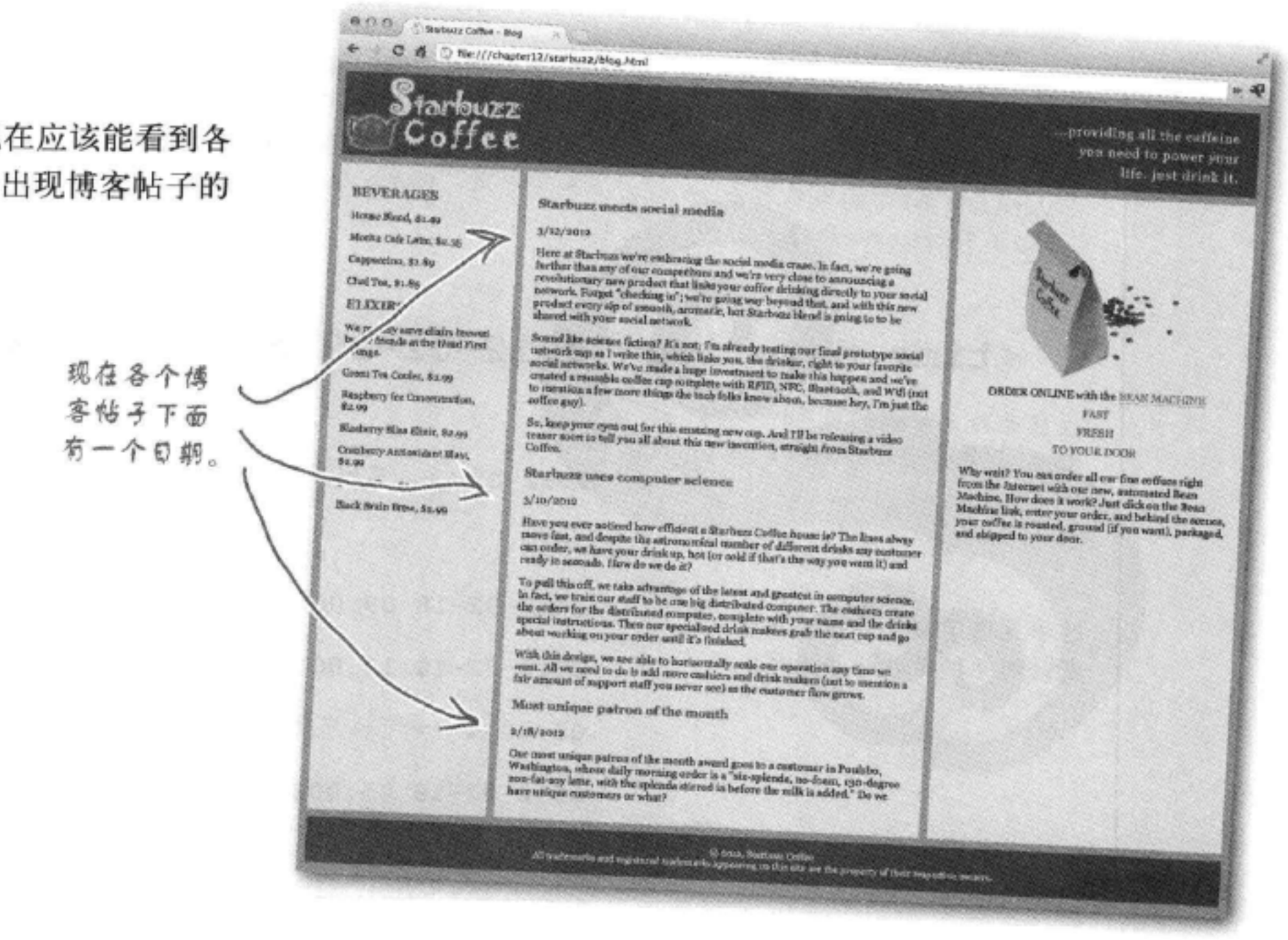
在各个标题下增加了一个<time>元素。

time元素的内容是博客帖子的日期（采用美国写法，月在前）。如果愿意，你也可以写作March 10, 2012。

我们使用了<time>元素的datetime属性，采用官方Internet日期/时间格式指定准确的日期时间。

测试博客

再来测试这个博客，现在应该能看到各个博客帖子标题下面会出现博客帖子的日期。



从语义上讲，看来每个文章都有自己的首部，包含一个标题和日期。我觉得甚至还可以再增加一些类似署名的内容，包含作者名和位置。可以这样使用“article”吗？

当然可以。重申一次，可以把文章看作是一个独立的内容，甚至可以把它取出来，加入到另一个网页的某个位置。如果是这样，你肯定希望增加一些内容，比如署名，指出这个文章是谁写的，什么时间以及在哪里写的。

我们甚至还可以更进一步，因为<header>元素并不只用于主页眉；如果你希望把一些元素组合起来放在一个首部中，都可以使用<header>元素。例如，可以为<article>、<section>，甚至<aside>增加<header>元素。

要看具体怎样做，下面先退一步，首先为Starbuzz的文章增加一些<header>元素。

注意也可以在 section、article和aside元素中使用 footer。在Starbuzz上我们没有这么做，不过很多网站确实会为这些元素创建header和footer。

如何增加更多<header>元素

增加<header>元素很简单。在各个<article>元素中，我们将放置一个<header>包含标题和时间。为此，找到博客区块中的<article>元素，分别增加一个开始和结束<header>标记。

```
  ...
<section id="blog">
<article>
  <header>
     <h1>Starbuzz meets social media</h1>
     <time datetime="2012-03-12">3/12/2012</time>
  </header>
  <p>...</p>
</article>

<article>
  <header>
     <h1>Starbuzz uses computer science</h1>
     <time datetime="2012-03-10">3/10/2012</time>
  </header>
  <p>...</p>
</article>

<article>
  <header>
     <h1>Most unique patron of the month</h1>
     <time datetime="2012-02-18">2/18/2012</time>
  </header>
  <p>...</p>
</article>
</section>
  ...
```

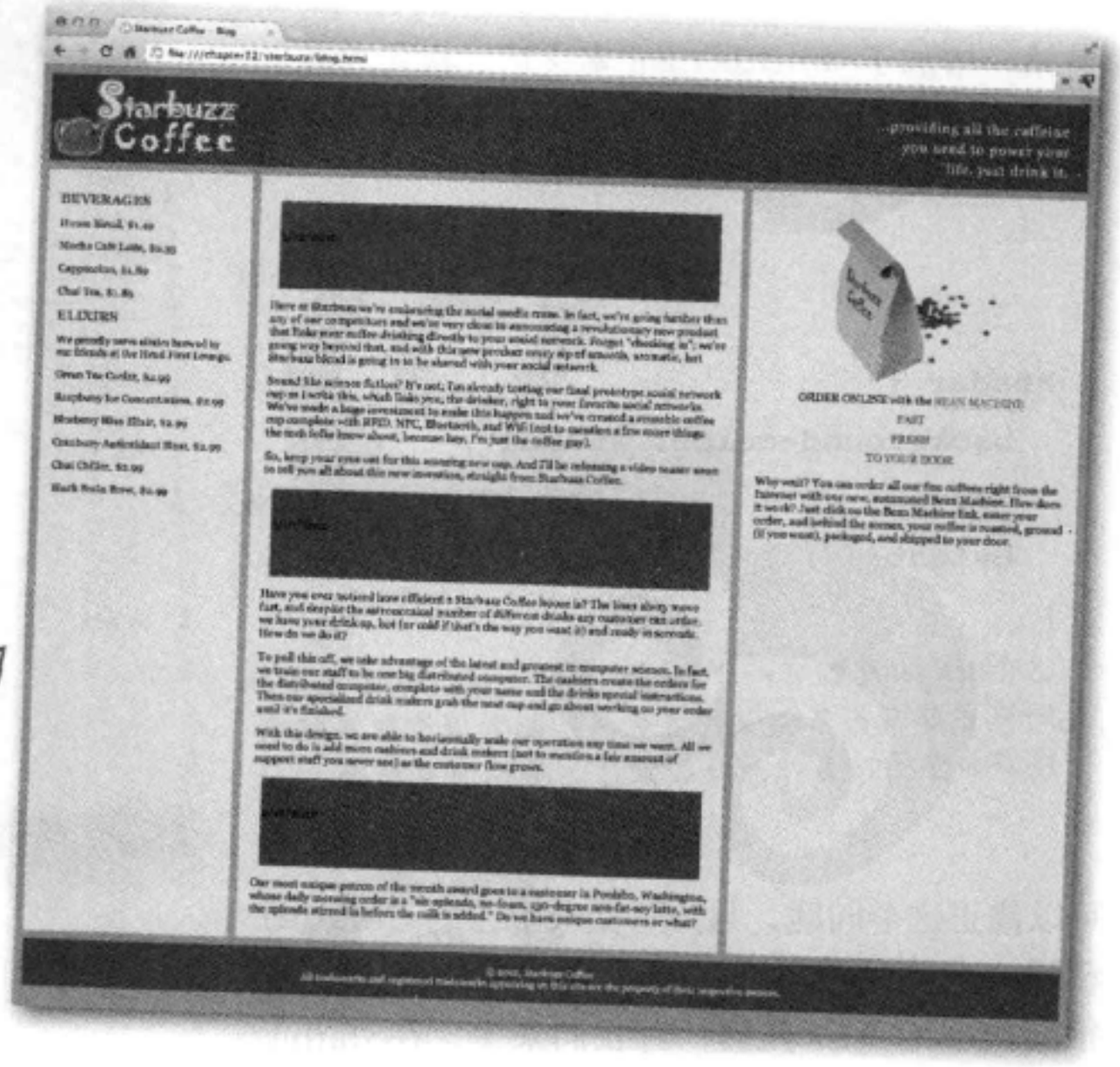
把<header>元素放在这里，包围标题和时间元素。

确保为博客区块中的每个文章都增加一个<header>。

完全可以在首部中增加一个作者署名。嗯，HTML中没有<author>元素。关于标记作者署名你有什么想法?

测试header

为Starbuzz博客增加<header>元素，下面来做个测试。



嗯，注意到了吗？加载页面时，文章的首部看起来不太对劲。格式现在全乱了……

Sharpen your pencil

现在增加了<header>元素，但是间距和格式都乱了，注意到了吗？文章标题下面和日期下面的空间都太大了，另外背景颜色也不对劲。这是为什么？你有什么想法吗？为什么会发生这种现象，请在下面写出你的想法。

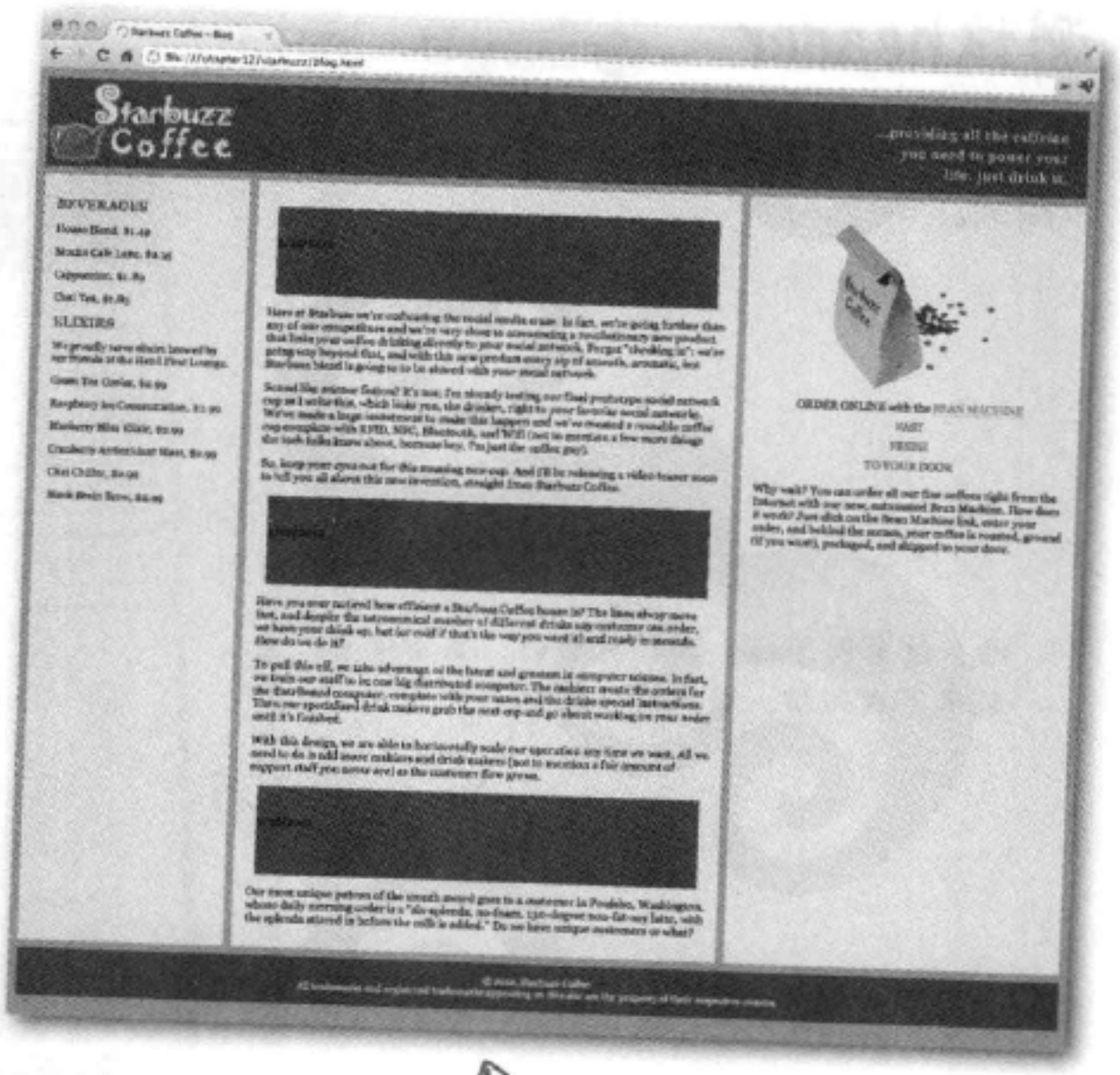
提示：查看你的CSS，看看有没有其他<header>规则可能影响你刚增加的这些新的文章首部。

这些首部到底怎么了？

显然，增加了<header>元素之后，格式有点乱了。为什么？下面再来看“starbuzz.css”文件，检查<header>元素的规则：

```
header {
  background-color: #675c47;
  margin:           10px 10px 0px 10px;
  height:           108px;
}
```

这个header规则有一个height属性，这使得页面中的所有首部（而不仅仅是主页眉）都会设置背景颜色，并增加空间。另外，外边距在这里也没有帮助。



为首部指定样式的规则对于主页眉很合适，但是对于文章中的首部看起来很糟糕。

可以修正这个问题，只为页面最上方的<header>创建一个类。整个网站的区块和文章中可能会有很多<header>元素，对我们来说，在Starbuzz Coffee网站中，页面最上方的<header>（页眉）要与其他首部区别对待，因为它有一个特殊的图形外观。所以，首先找到“blog.html”文件最上面的<header>元素，为这个元素增加一个名为“top”的类：

```
<body>
  <header class="top">
    <img id="headerLogo"
         src="images/headerLogo.gif" alt="Starbuzz Coffee header logo image">
    <img id="headerSlogan"
         src="images/headerSlogan.gif" alt="Starbuzz Coffee header slogan image">
  </header>
...
```

为页面中的第一个<header>元素增加类“top”。

还要为“index.html”文件的第一个<header>增加“top”类。

一旦为“blog.html”和“index.html”文件增加了“top”类，下面要做的就是更新CSS，在header规则的选择器中使用这个类：

```
header.top {
  background-color: #675c47;
  margin:           10px 10px 0px 10px;
  height:           108px;
}

header.top img#headerSlogan {
  float:            right;
}
```

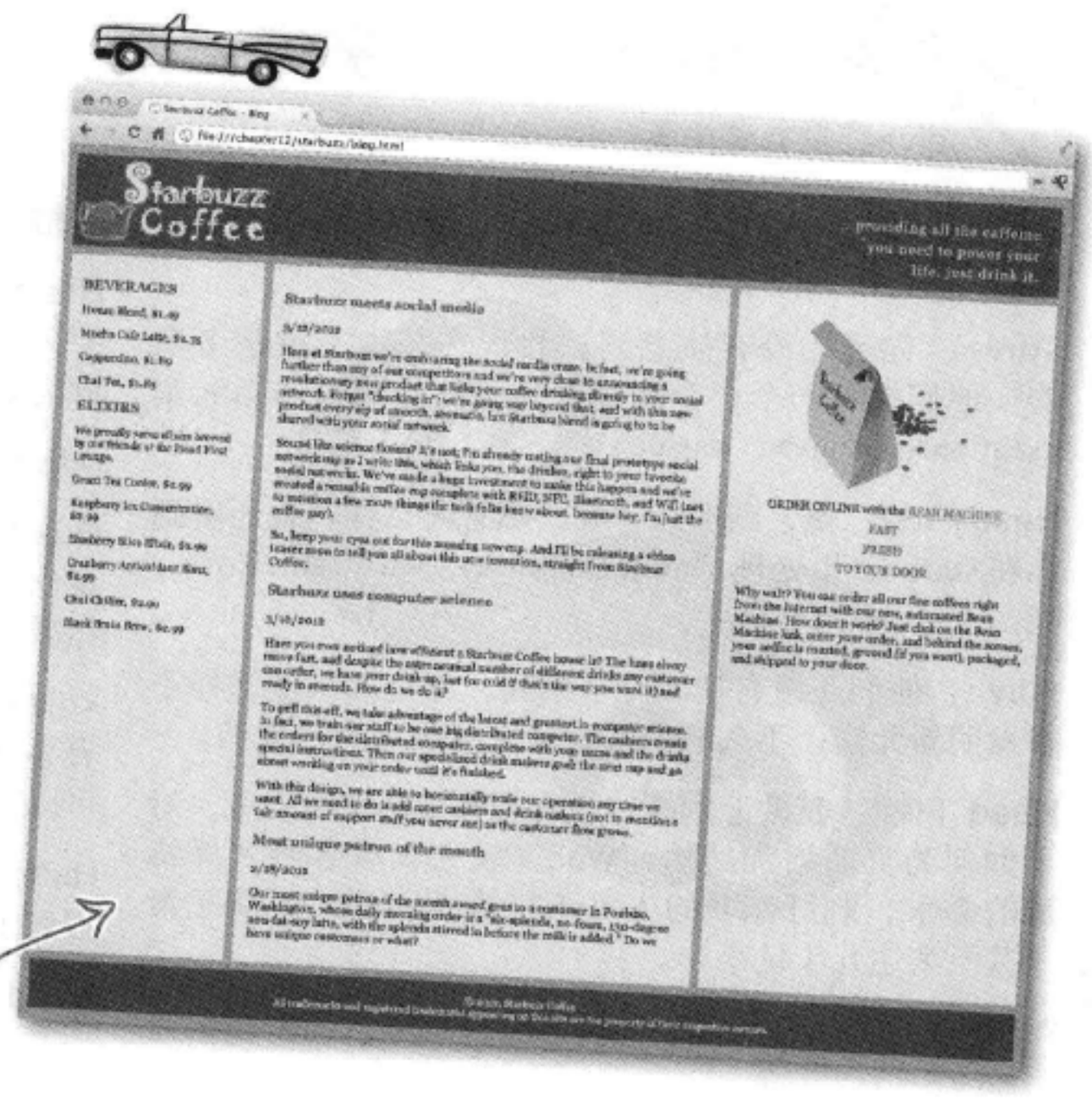
我们在CSS中为header规则增加了.top类选择器。

另外这个规则中也增加了.top。实际上，这个选择器不需要增加.top也能正确地选择元素，不过这样一来，就能在CSS中更清楚地看出我们选择的是哪一个headerSlogan。这算是一个最佳实践。

页眉的最终测试

对“blog.html”、“index.html”和“starbuzz.css”文件完成所有修改之后，重新加载博客页面。

可以注意到，现在<header>规则只应用于页面最上方的<header>，这正是我们想要的。与此同时，文章中的<header>仍采用默认样式，很不错。



现在文章中的首部总算有正确的格式了！

there are no Dumb Questions

问：我们已经做了很多工作，为页面增加了不少元素，但看起来页面和以前还是一模一样的！请再解释一下，这对我来说到底有什么意义？

答：我们替换了一些元素，另外还增加了一些元素，在这个过程中，我们为页面增加了很多含义。对于浏览器、搜索引擎和构建Web页面的应用来说，只要它们愿意，利用这些含义，就能更明智地确定如何处理页面的不同部分。对于你和其他Web开发人员来说，你的页面也更容易理解。尽管看起来是一样的，但实际上页面已经有了更多的含义。

问：<section>和<article>有什么区别，能再说一遍吗？在我看来它们很相像。

答：要确定该使用哪个元素，这确实容易让人糊涂，所以很高兴你能问这个问题。<section>元素比<article>更通用，不过它又不及 通用。例如，如果只是要增加一个元素以便为页面指定样式，那么可以使用 。如果要增加一个元素来标记一些内容，指示这是由相关内容构成的一个结构明确的区块，那就可以使用<section>。如果有些内容可以独立于页面上的其余内容进行重用和分发，那就使用<article>。

问：是不是每个<section>和每个<article>都要有一个<header>？

答：大多数情况下，<section>和<article>都会有一个<header>，或者至少有一个标题（如<h1>）。可以这样来考虑：一个<article>元素中的内容可以在别处重用，所以这个内容很有可能需要一个首部来提供描述或介绍。类似地，<section>元素包含页面中一组相关的内容，所以通常要有某个首部来区分或介绍这部分内容。

问：是不是只有当需要放置多个内容时才应该使用<header>？如果只有一个标题，而没有其他内容，还有必要使用<header>吗？

答：即使只放一个标题也可以使用<header>。<header>元素能提供额外的语义含义，可以将页面、区块或文章的首部与其余的内容区分开。不过，并不要求标题内容非得放在<header>元素中（也就是说，即使没有增加<header>元素，页面也是合法的）。

对 的简短采访

** 感觉有些失落……**

Head First：你好， ，听说你最近心情不太好。怎么了？

：你可能没有注意到，我现在基本上就是多余的！他们把我一个一个地都替换了，换成这些新元素<section>、<nav>、<aside>……

Head First：嘿，别那么悲观。不管怎么样，我还看到你在Starbuzz里处理“tableContainer”和“tableRow”呢。

：他们还没有把我完全剔除，不过如果像这样不断发明新元素，不久的将来我肯定会玩完的，天呐。

Head First：我最近刚查过，你还在HTML规范里。对于如何为页面增加结构，Web开发人员有各种各样特殊的需求，制订标准的人可不打算发明那么多新元素（甚至多达几十亿个）。

：这倒是真的，我还没有看到那种只用来创建通用结构的新元素。

Head First：对呀！所有那些新元素都有特定的用途，要为页面增加某种语义含义，而你更通用。例如，人们需要表格布局时就会回来找你。

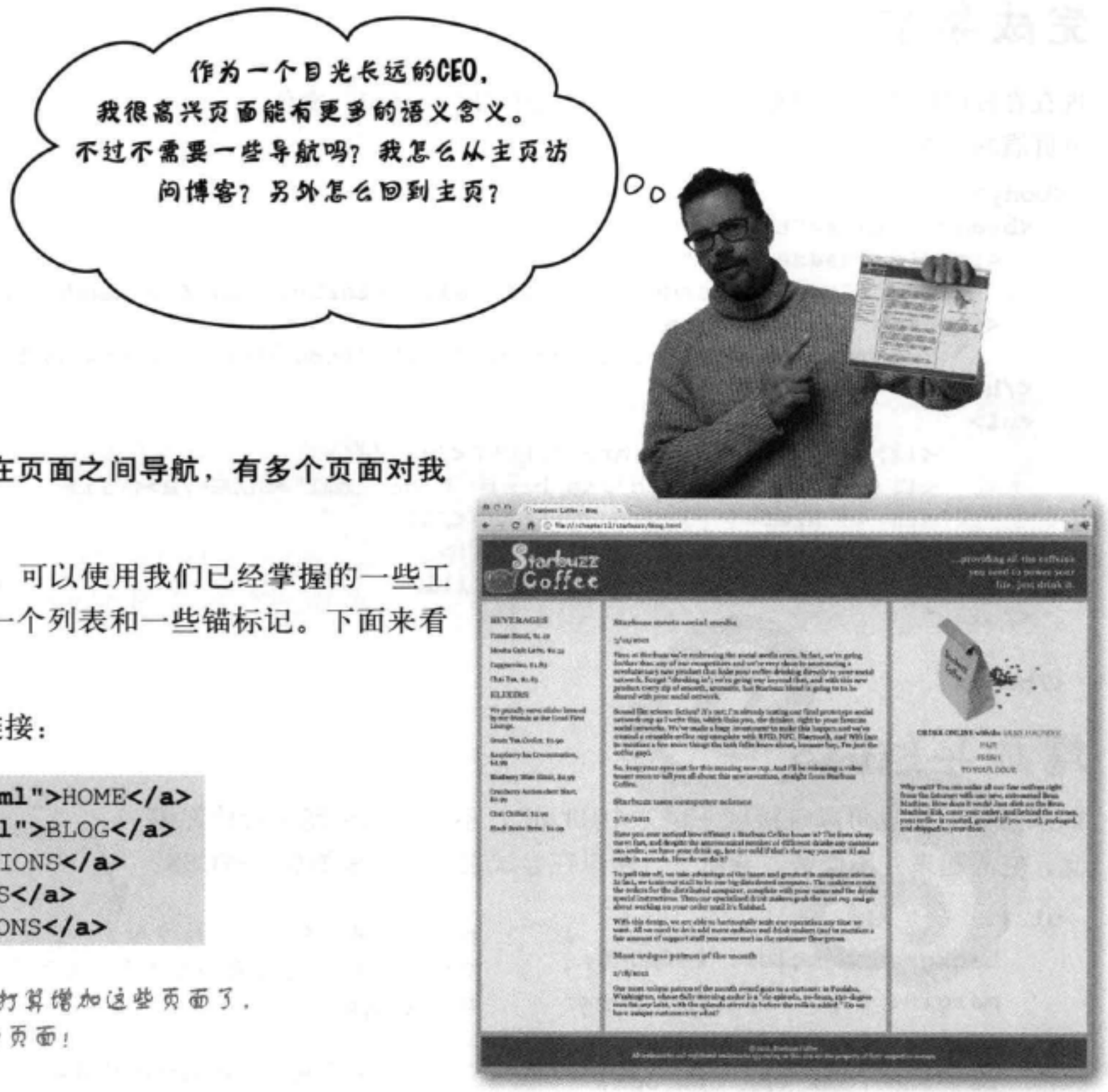
：不会吧！

Head First：如果要问我们的看法，实际上在这些新元素出现之前，你的工作压力实在太重……现在工作负担减轻了，难道你不高兴吗？

：哈，有道理。也许我可以停业一段时间，到世界各地去看看。要知道，我可积攒了不少飞行里程，可以在互联网里到处逛逛。

Head First：先别急，你还不能就这么消失，Web很大程度上还要靠你呢……

Head First：喂？ ？



没错！如果访问者不能在页面之间导航，有多个页面对我们来说就毫无用处。

要为这些页面创建导航，可以使用我们已经掌握的一些工具。具体来说，要用到一个列表和一些锚标记。下面来看如何做到。

首先，为导航创建一组链接：

```
<a href="index.html">HOME</a>
<a href="blog.html">BLOG</a>
<a href="">INVENTIONS</a>
<a href="">RECIPES</a>
<a href="">LOCATIONS</a>
```

这3个链接为空，因为我们不打算增加这些页面了。不过你完全可以自己创建这些页面！

现在，把这些锚包围在一个无序列表中，把它们当作一组列表项。之前我们没有这样做，不过请仔细观察这里的做法，可以看到列表确实非常适合建立导航：

注意，现在每个链接都是一个无序列表中的一个列表项。这看起来可能不太像导航，不过指定一些样式之后这就会有改观。

```
<ul>
    <li><a href="index.html">HOME</a></li>
    <li class="selected"><a href="blog.html">BLOG</a></li>
    <li><a href="">INVENTIONS</a></li>
    <li><a href="">RECIPES</a></li>
    <li><a href="">LOCATIONS</a></li>
</ul>
```

另外要注意，我们使用了一个类来标识被选中的一项。

完成导航

现在在HTML中加入导航。可以把它插入到“blog.html”文件中页眉的下面：

```
<body>
  <header class="top">
    <img id="headerLogo"
         src="images/headerLogo.gif" alt="Starbuzz Coffee header logo image">
    <img id="headerSlogan"
         src="images/headerSlogan.gif" alt="Providing all the caffeine...">
  </header>
  <ul>
      <li><a href="index.html">HOME</a></li>
      <li class="selected"><a href="blog.html">BLOG</a></li>
      <li><a href="">INVENTIONS</a></li>
      <li><a href="">RECIPES</a></li>
      <li><a href="">LOCATIONS</a></li>
  </ul>
  ...
</body>
```

增加导航CSS

如果愿意，你也可以直接试一试这个HTML，不过结果可能不会让你满意，它看起来并不像是“导航”。所以在尝试之前，先来增加一些CSS：

一定要把这个CSS增加到starbuzz.css文件的最后。

```
ul {
    background-color: #efe5d0;
    margin: 10px 10px 0px 10px;
    list-style-type: none;
    padding: 5px 0px 5px 0px;
}
```

增加一个背景颜色，另外增加了一些外边距和内边距。注意下外边距为0，因为表格显示中上方已经有10像素的边框间距(border-spacing)。

还要注意我们删除了列表项的项目符号。

```
ul li {
    display: inline;
    padding: 5px 10px 5px 10px;
}
```

这里将各个列表项的显示从“block”改为“inline”，所以现在列表项的前后不再有回车，它们会像常规的内联元素一样在页面上流入一行。

```
ul li a:link, ul li a:visited {
    color: #954b4b;
    border-bottom: none;
    font-weight: bold;
}
```

我们希望导航列表中的链接看起来与页面中的其余链接有所不同，所以这里们覆盖了<a>的其他规则（也就是CSS中在这个规则之前出现的其他规则），这个规则会为link和visited状态的链接设置属性（所以它们看起来是一样的）。

```
ul li.selected {
    background-color: #c8b99c;
}
```

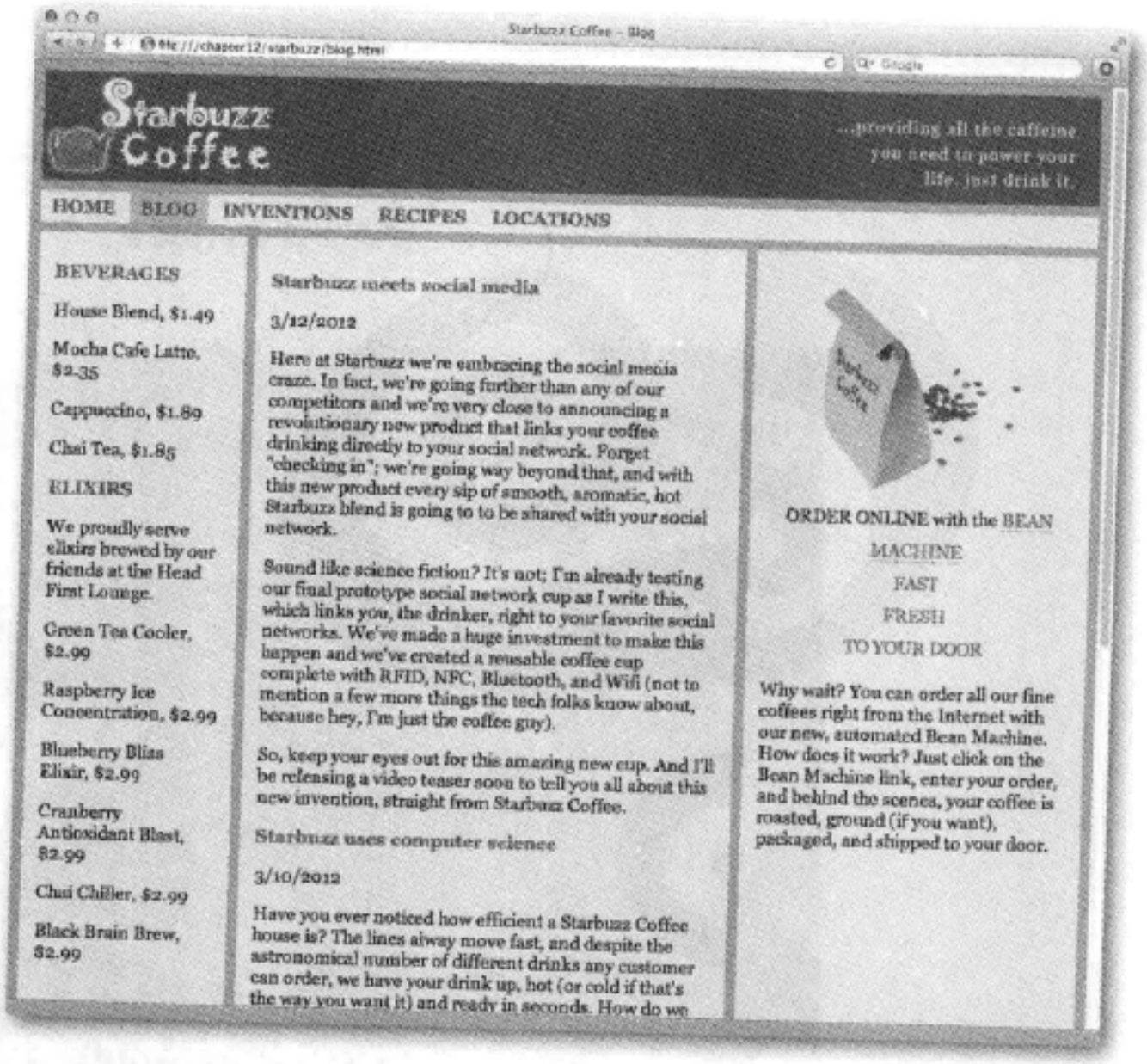
最后，为“selected”类的<li>元素设置背景，这样一来，如果某个导航项正好对应我们所在的页面，它看起来会与其他导航项有所不同。

谁需要GPS？测试导航

下面来试一试。将这些CSS规则输入到CSS文件的最后，然后在浏览器中加载页面。

哈，作为第一次尝试，效果还不错。我们得到了一个漂亮的导航条，甚至能突出显示当前所在的页面（博客页面）。

不过……能不能更进一步呢？毕竟，这一章讨论的是“现代HTML”，我们还没有使用HTML5的新元素实现导航呢！你可能已经猜到了，可以为HTML文件增加一个<nav>元素来加以改进。这样做就会使大家（浏览器、搜索引擎、屏幕阅读器、Web开发人员）了解更多信息，知道对这个列表实际上是什么……



增加<nav>元素……

你已经知道，HTML5中有一个<nav>元素，使用这个元素很简单，只需要把你的导航列表用一个<nav>开始和结束标记包围起来，如下：

这是<nav>开始标记，我们把整个导航列表包围在一个<nav>元素中。

```
<nav>
    <ul>
        <li><a href="index.html">HOME</a></li>
        <li class="selected"><a href="blog.html">BLOG</a></li>
        <li><a href="">INVENTIONS</a></li>
        <li><a href="">RECIPES</a></li>
        <li><a href="">LOCATIONS</a></li>
    </ul>
</nav>
```

我们真的需要好好谈谈最佳实践。要知道，现在你的CSS假设所有无序列表都是导航菜单。所以，如果Starbuzz CEO需要在博客中增加另一个列表，列出他打算新开张的咖啡馆，会怎么样呢？这会带来一场灾难，可能他的博客中间会出现一个导航列表，因为它与我们刚刚在页面中增加的导航列表有相同的样式。

不过，不用担心。要想修正这个潜在的问题，只需要在选择导航列表项时做到更为特定，这并不难，因为我们想选择的唯一的导航列表项都包含在一个<nav>元素中。

继续学习后面的内容之前，仔细考虑应当如何修改CSS来准确地选择导航项，而不是其他无序列表。

让CSS更特定……

OK，我们的HTML中已经有一个<nav>元素，下面就来利用这一点，让选择器更特定。这样一来，我们可以确保将来对HTML的修改不会带来出乎意料的样式　（如以后在页面中的另外某个位置增加一个与导航无关的<ul>元素）。我们的做法如下……不过请注意，我们确实要对<nav>元素的外边距做几处调整，它才能有正确的表现。

```
nav {
    background-color: #efe5d0;
    margin: 10px 10px 0px 10px;
}
nav ul {
    margin: 0px;
    list-style-type: none;
    padding: 5px 0px 5px 0px;
}
nav ul li {
    display: inline;
    padding: 5px 10px 5px 10px;
}
nav ul li a:link, nav ul li a:visited {
    color: #954b4b;
    border-bottom: none;
    font-weight: bold;
}
nav ul li.selected {
    background-color: #c8b99c;
}
```

我们为<nav>元素增加了一个新规则，将设置背景颜色和外边距的属性移到这个规则中，这样<nav>元素中的所有内容都会按这些属性指定样式。

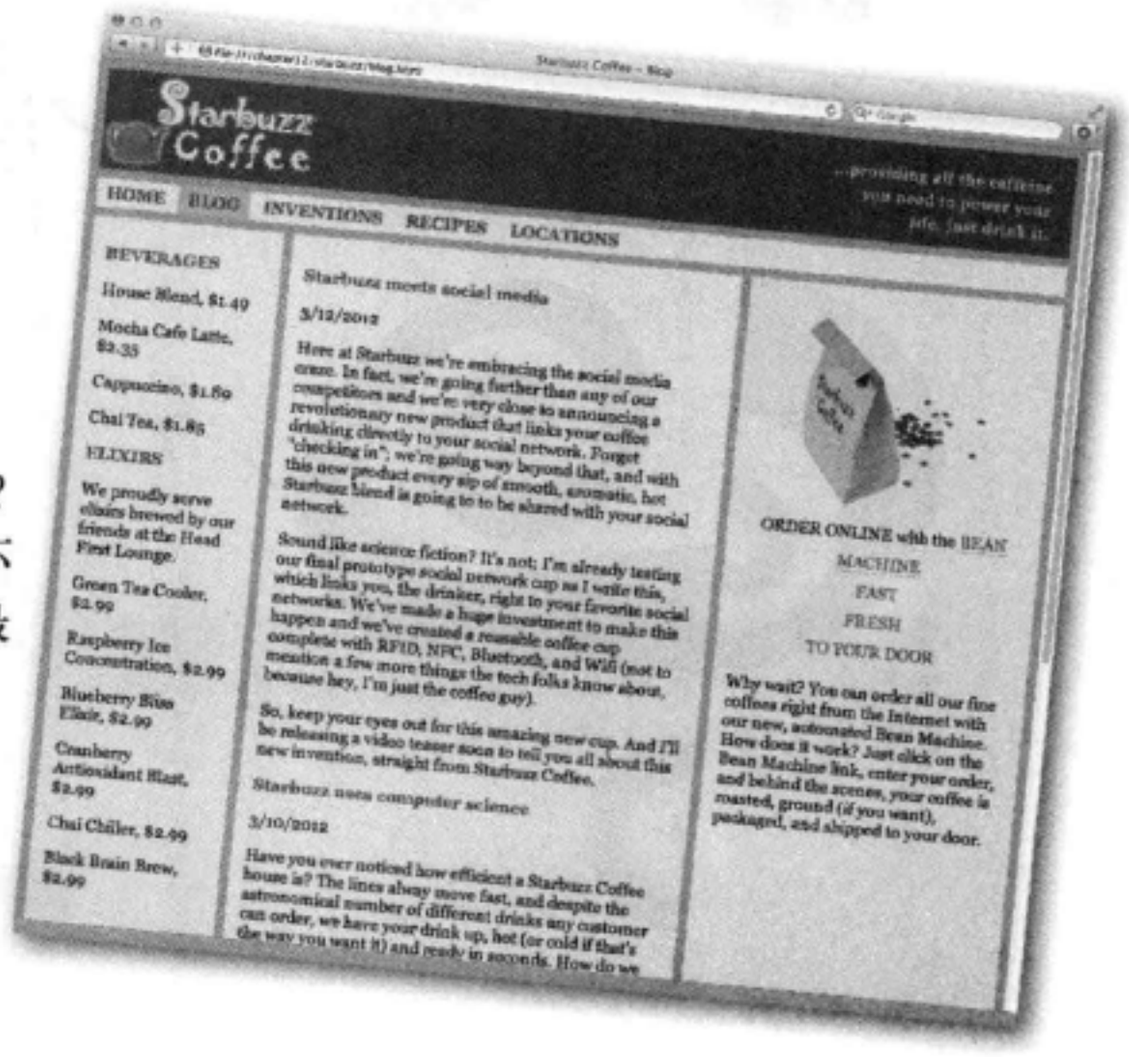
这里增加了一个属性，将<ul>元素的外边距设置为0，使它能紧贴着放在<nav>元素中（否则默认情况下，如果没有明确设置为0，<ul>元素会有一个外边距，这会让<ul>稍稍偏离）。

最后，我们在所有这些规则的前面增加了选择器“nav”，使得这些规则只会影响出现在<nav>元素中的<ul>元素。这样一来，可以确保即使CEO将来要在博客中增加一个<ul>，也不会对它像导航列表那样指定样式！

注意，两个规则都增加了“nav”，这个规则中有两个选择器！

哇！看看这里的导航！

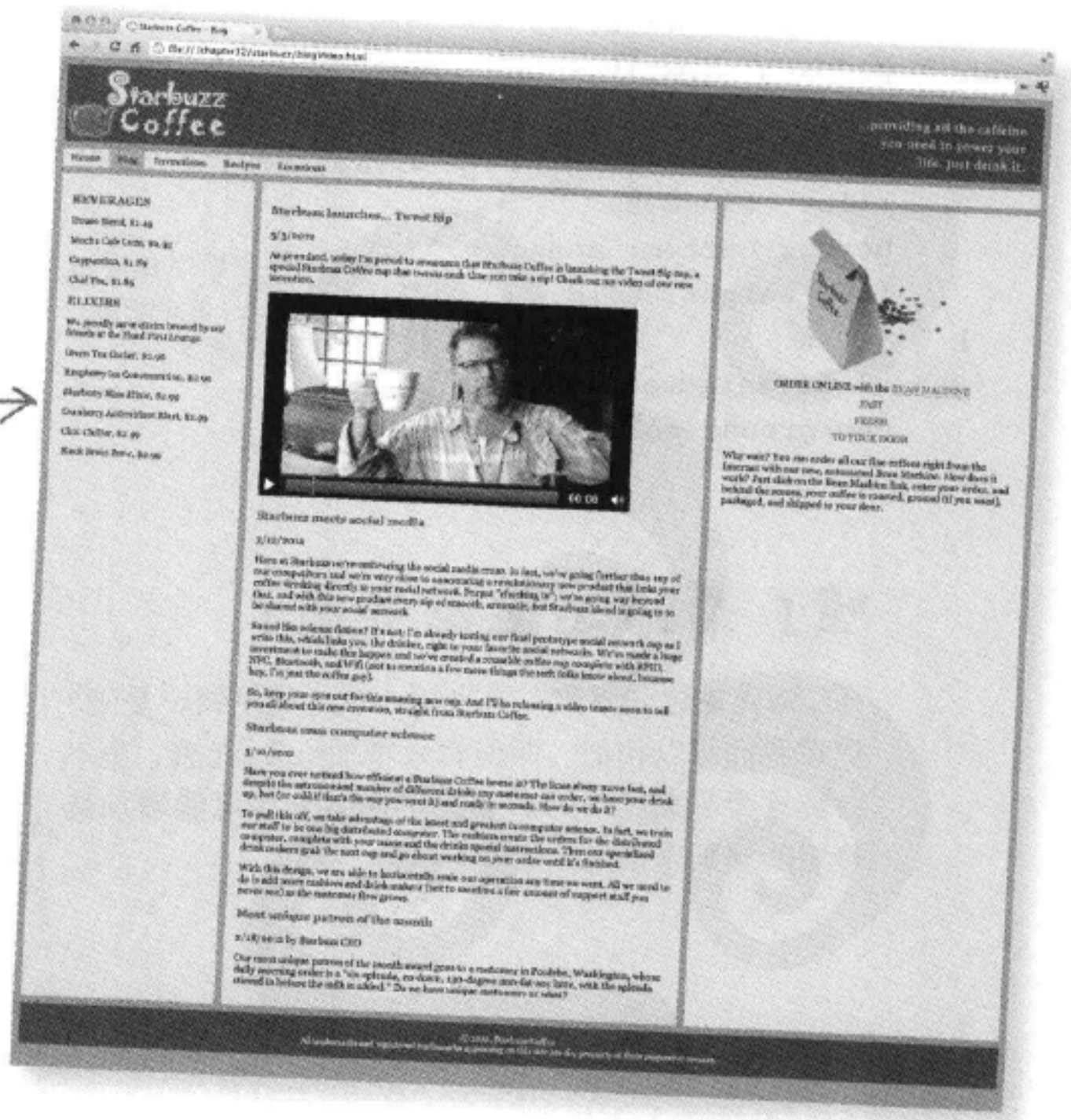
在CSS完成这些修改，再来试一试。还不错，是吧？现在我们可以相信，即使将来增加<ul>元素，也不会受这个导航CSS的影响。记住，要尽可能增加最特定的规则来指定元素的样式。



嘿，能不能先停一下，别再摆弄这些新的HTML5元素了，我有一个好消息：我们刚刚成功地开发了新的Tweet Sip咖啡杯。这是一种创造性的新技术：啜一口咖啡，你在Twitter上的状态就会更新。我刚刚做了一个新视频来演示这个过程！能不能把它放在博客上！

他想把视频放在页面里，就像这样……

这是Starbuzz博客页面，加入了我们所有最新的改进……



嗅，这个Tweet Sip技术实在太有用了，简直令世界震惊，他希望我们能与朋友们分享……我们告诉他你在这方面很擅长。

Jim：嗯，我们以前总是习惯用Flash实现视频，不过有了HTML5，现在我们可以利用一个<video>元素来做到。

Frank：等一下，Flash不是更好些吗？它已经发展了很长一段时间了。

Jim：持这种观点的人确实不少，如果要在桌面上提供视频，短期内可能确实如此，Flash在这方面有它的优势。不过如果你要在某些移动设备上提供视频，而这些设备不支持Flash，该怎么办呢？想想看Starbuzz有多少移动用户，如果我们使用Flash来实现视频，很多顾客就会什么都看不到。

Frank：我明白了。那么怎么利用一个元素来实现视频呢？

Jim：可以把视频看作是<img>元素，我们可以提供一个src属性来引用视频，视频将放在页面中<video>元素所在的位置。

Frank：听上去很容易。简直是小菜一碟。

Jim：嗯，千万别那么快下结论。就像大多数媒体类型一样，视频可能很复杂，特别是处理视频编码方面。

Frank：编码？

Jim：就是对一个视频片段中的视频和音频编码所用的格式。

Frank：这很麻烦吗？

Jim：嗯，这是因为，浏览器制造商对于视频编码还没有达成一致，并没有一个通用的标准。不过后面再考虑这个问题。现在先在页面中增加一个<video>元素，看看可以做些什么。

Frank：听起来不错，开始干吧！

创建新博客条目

先增加一个新的博客条目，按HTML术语来讲，就是增加一个新的<article>元素。把以下HTML增加到<section>元素中，放在其他文章上面：

把它增加到 "blog" <section>的最上面……

```
<article>
  <header>
    <h1>Starbuzz launches……Tweet Sip</h1>
    <time datetime="2012-05-03">5/3/2012</time>
  </header>
  <p>
    As promised, today I'm proud to announce that Starbuzz
    Coffee is launching the Tweet Sip cup, a special Starbuzz
    Coffee cup that tweets each time you take a sip! Check
    out my video of our new invention.
  </p>

</article>
```

要在这里增加视频，放在博客条目中的段落下面。

现在加入<video>元素

乍一看，<video>元素确实很像<img>元素。查看这一章的下载代码包，可以在“video”文件夹中找到一个名为“tweetsip.mp4”的文件。确保“video”文件夹与“blog.html”文件在同一级上。然后在页面中增加这个标记，要把它放在结束</p>标记之后，结束</article>标记之前：

稍后还会回来讨论所有这些属性的细节，不过现在只注意如何设置元素的宽度和高度，以及如何为视频指定一个src URL。

这里是video开始标记，它有很多属性……

```
<video controls autoplay width="512" height="288" src="video/tweetsip.mp4">
</video>
```

稍后还会介绍这里可以放置什么内容……

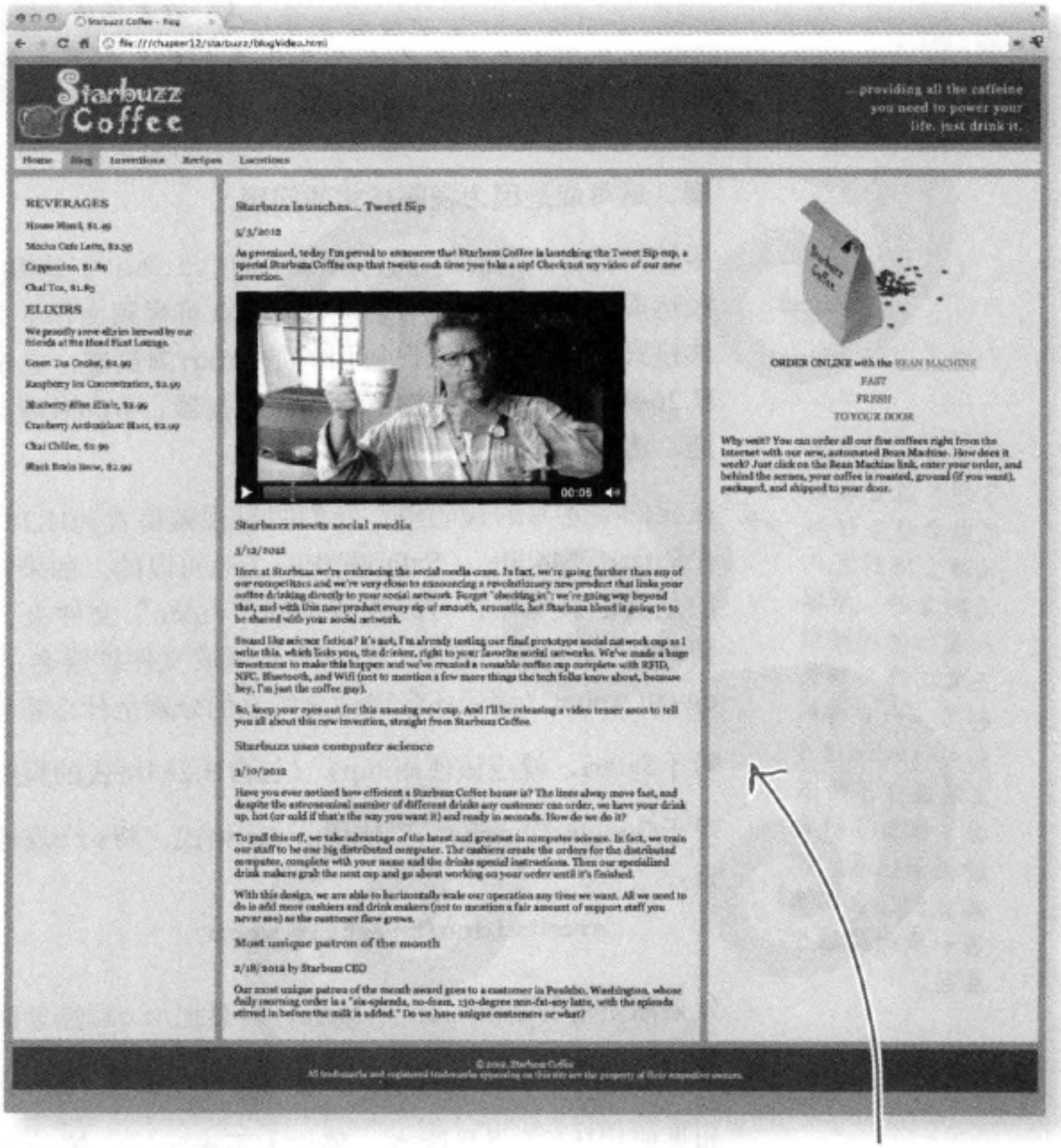
这里是video结束标记。

灯光，摄像，开拍……

加入这个新标记来试一试！希望你能像我们一样看到下面的结果，不过，如果没有看到，也请继续读下去，很快你就会知道如何修正可能出现的问题。

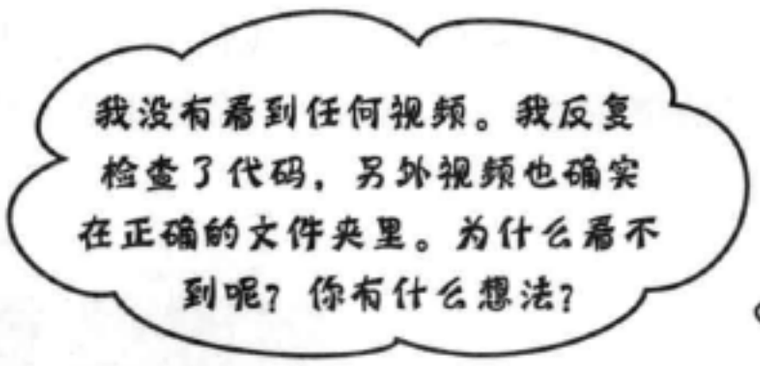
我们的视频现在就在页面中，宽度和高度正是我们刚才指定的。

注意到了吗？这个视频开始自动播放。这是因为，我们提供了一个 "autoplay" 属性。可以把它删掉，这样用户必须点击播放才能看到视频。



另外要注意，这里有一组控件来控制播放、暂停、调节音量等。如果在<video>元素中指定了 "controls" 属性就会提供这组控件。

只用几行标记就得到了这种结果，真不赖，是吧？不过别放松得太早（尤其是如果你还没有看到视频），接下来我们还会了解<video>元素的更多内容。现在就开始吧……



嗯，这可能是因为视频格式的问题。

尽管浏览器制造商对于HTML5中的<video>元素和API已经达成一致，但是并不是所有人都认可视频文件本身的具体格式。例如，如果你使用的是Safari浏览器，它更接受H.264格式；如果你使用Chrome浏览器，WebM则更受欢迎，诸如此类。

等你读到这些文字时，这些格式可能会在各种浏览器上得到更广泛的支持。所以，如果你的视频能正常工作，那太好了。一定要时刻关注Web，这个主题还在不断演进，需要了解它的最新进展。稍后我们还会回来进一步讨论这个主题。

在我们刚才写的代码中，我们假设视频格式为H.264，这在Safari、Mobile Safari和IE9+上是可以的。如果你使用的是其他浏览器，可以查找你的“video”文件夹，会看到3种不同类型的视频，分别有不同的文件扩展名：.mp4，.ogv和.webm（稍后还会详细介绍它们分别是什么意思）。

对于Safari，就应该使用.mp4（包含H.264格式的视频）。

对于Google Chrome，要使用.webm格式，将src属性替换为：

```
src="video/tweetsip.webm"
```

如果你使用的是Firefox或Opera，就要把src属性替换为：

```
src="video/tweetsip.ogv"
```

先来试一试，稍后我们还会回来详细说明。

如果使用IE8或更早版本，那你太不走运了。等一下，这已经是第12章了，你怎么还在使用IE8或更早的版本呢？赶快升级！不过，如果你需要知道如何为IE8用户提供后备内容，稍等片刻，很快我们就会谈到的。

<video>元素如何工作？

现在你已经在页面中加入了一个视频，而且视频可以开始播放，不过在学习后面的内容之前，先退一步，好好研究一下这个<video>元素和它的属性：

注意，controls和autoplay属性与我们之前见过的其他属性有点不同。它们是“布尔属性”，没有值。所以，例如，如果有controls属性，视频控件就会出现。如果没有controls属性，视频控件就不会出现。

如果有controls属性，播放器会提供一些控件，可以控制视频和音频的播放。

如果有autoplay属性，一旦页面加载视频就会开始播放。

```
<video controls
       autoplay
       width="512" height="288"
       src="video/tweetsip.mp4"
       poster="images/poster.png"
       id="video">
</video>
```

页面中视频的宽度和高度。

视频的源位置。

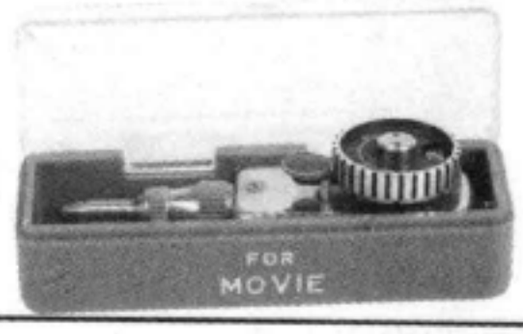
如果愿意，可以提供一个可选的海报图像，视频未播放时就会显示这个图像。

当然，还可以为元素增加一个id，以方便对它应用某些样式。

Web镇的视频礼节：autoplay属性

对于YouTube和Vimeo（或WebvilleTV）之类的网站来说，autoplay确实很好，不过在你的<video>元素中设置这个属性之前一定要三思。通常，用户希望参与决定加载页面时是否播放视频。

仔细检查video属性……



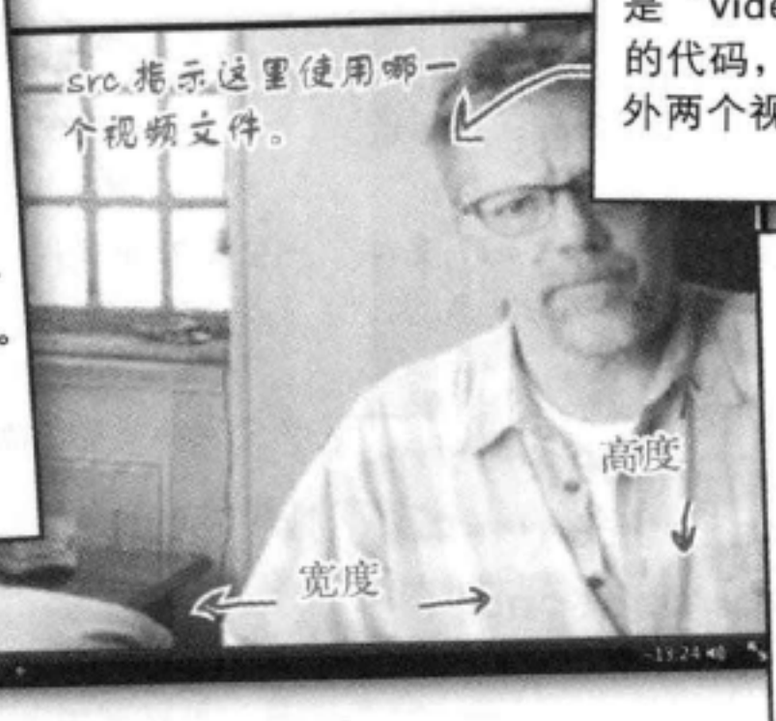
下面再来更仔细地研究一些比较重要的的video属性：

controls

controls属性是一个布尔属性。可以有，也可以没有。如果有这个属性，浏览器就会为视频显示增加内置的控件。不同浏览器提供的控件有所不同，所以要查看各个浏览器，看看会有哪些控件。这是Safari上提供的控件。

src

src属性与<img>元素的src很类似，这是一个URL，告诉video元素在哪里查找源文件。在这里，源文件是“video/tweetsip.mp4”（如果下载了这一章相应的代码，可以在“video”目录中找到这个视频以及另外两个视频）。



preload

属性preload通常用于细粒度地控制视频如何加载，来实现优化。大多数情况下，浏览器会根据是否设置autoplay以及用户的带宽来选择加载多少视频。可以覆盖这种设置，将preload设置为“none”（在用户真正“播放”视频之前不下载视频），也可以设置为“metadata”（下载视频元数据，但不下载视频内容），或者可以设置为“auto”，让浏览器来做决定。

autoplay

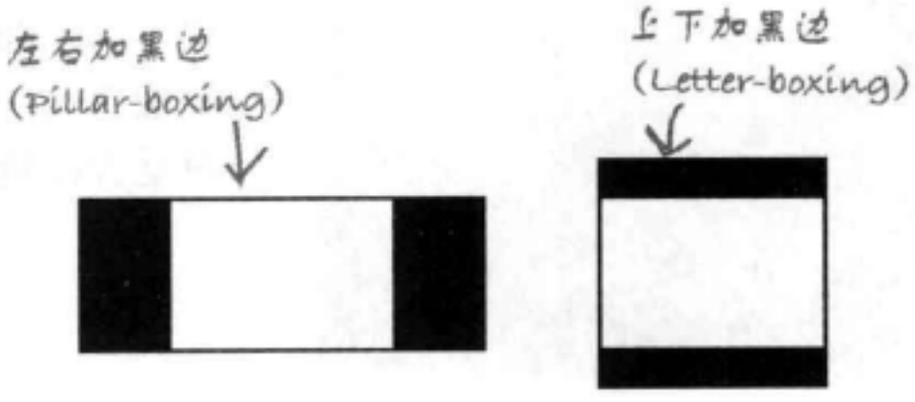
autoplay布尔属性告诉浏览器：一旦有了足够的数据就开始播放视频。你可能会看到，我们的演示视频几乎立即开始播放。

poster

浏览器通常会把视频的一帧显示为“海报”图像，来表示这个视频。如果你删除了autoplay属性，单击播放之前就会看到这个图像。要由浏览器来选择显示哪一帧。通常，浏览器会显示视频的第一帧……这往往是一个黑屏。如果你想显示某个特定的图像，要由你来创建想显示的图像，并使用poster属性来指定。

width, height

width和height属性会设置视频显示区（也称为“视窗”）的宽度和高度。如果指定了一个海报（poster），海报图像会缩放到你指定的宽度和高度。视频也会缩放，不过会保持其宽高比（例如，4:3或16:9），所以，如果两边或者上下边有多余的空间，视频会采用上下加黑边（letter-boxed）或左右加黑边（pillar-boxed）的模式来适应显示区大小。如果你想得到最佳的性能，就应该尽量采用视频原本的尺寸（这样浏览器就不用实时缩放视频了）。



loop

这也是一个布尔属性，如果有loop属性，视频结束播放之后会自动重新开始播放视频。

是的。对于HTML视频，每个浏览器中提供的控件都不同。

控件的外观是那些实现浏览器的人决定的。在不同的浏览器和操作系统上，这些控件看上去确实都不相同。有些情况下，例如，在一个平板电脑上，它们的外观和表现会不同，因为这些设备的工作方式就不同（好在这个问题已经为你考虑到了）。这个道理我们明白，比如说，在各种桌面浏览器上，能有一致的控件是一件好事，但是这并不算HTML5规范中正式的部分，有些情况下，可能某个方法对一种操作系统适用，在面对另一个操作系统的用户界面原则时却会完全失效。所以，要知道控件可能会不同，另外如果你确实觉得有必要，也可以为你的应用实现定制控件。

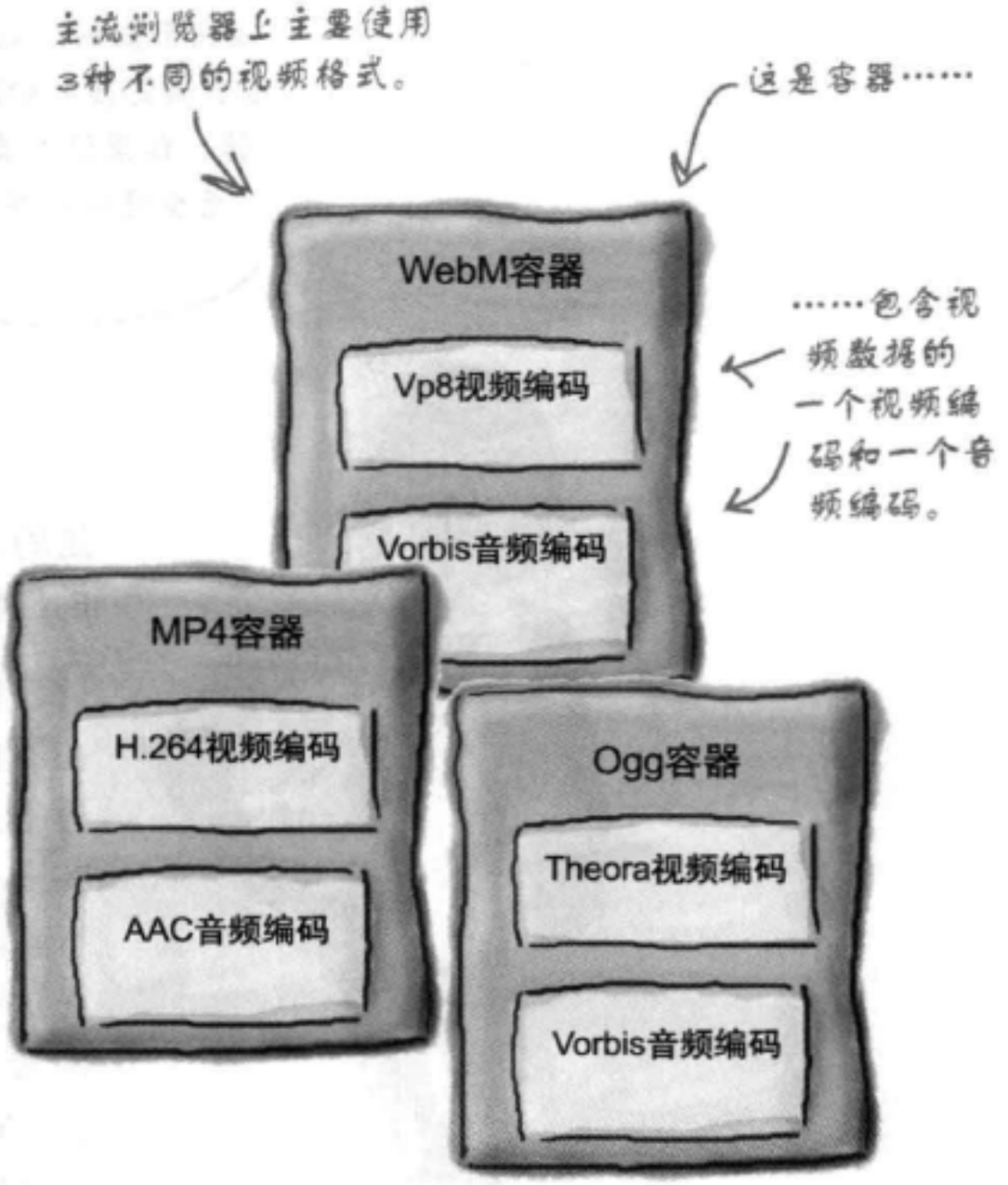
我们在《Head First HTML5 Programming》中就实现了自己的定制控件。加入我们吧，JavaScript真的很有趣！

关于视频格式需要知道什么

我们真希望一切都像video元素和它的属性那样干净整洁，可惜事实上Web上的视频格式相当混乱。视频格式是什么？可以这样来考虑：一个视频文件包含两部分，一个视频部分和一个音频部分，每个部分都使用一种特定的编码类型来编码（这样可以缩小数据大小，并能更高效地播放）。大多数情况下，并没有一种得到大家共识的编码。有些浏览器制造商推崇H.264编码，另外一些却喜欢VP8，还有一些倾向于开源编码Theora。而且，包含视频和音频编码的文件（称为容器）也有自己的格式和格式名，这让情况更为复杂。所以我们面对的真像是一个混合口味的“术语汤”。

不管怎样，如果所有浏览器制造商都能达成一致，在Web上使用同一种格式，这当然是一个让人欢欣鼓舞的大同世界，不过，嗯，出于技术上、政治上甚至哲学方面的一些原因，这是不现实的。但这里并不打算就此展开争论，我们只是希望你对这个主题有足够的了解，对于如何支持你的用户能够做出自己的决定。

下面来看目前流行的一些编码，现在主要有3个竞争对手在努力争霸这个（Web）世界……



等你读到这本书时，你掌握的情况可能会有所不同，因为流行的编码总是在随时间发生变化。

HTML5规范允许采用任何视频格式。具体支持哪些格式由浏览器实现来确定。

视频格式竞争对手

事实上，如果你打算为更大范围的用户提供内容，就不能只提供一种格式。另一方面。如果你只关心（比如说）Apple iPad，那么只支持一种格式可能也行。如今我们有3个主要的竞争对手，下面来一一认识一下：

MP4容器，包含H.264视频和AAC音频

H.264由MPEG-LA公司授权。

目前有多种H.264，每一种分别称为一个“profile”（等级或类）。

MP4/H.264得到了Safari和IE9+的支持。有些版本的Chrome也提供了支持。

WebM容器，包含VP8视频和Vorbis音频

WebM由Google设计，用来处理VP8编码视频。

WebM/VP8得到了Firefox、Chrome和Opera的支持。

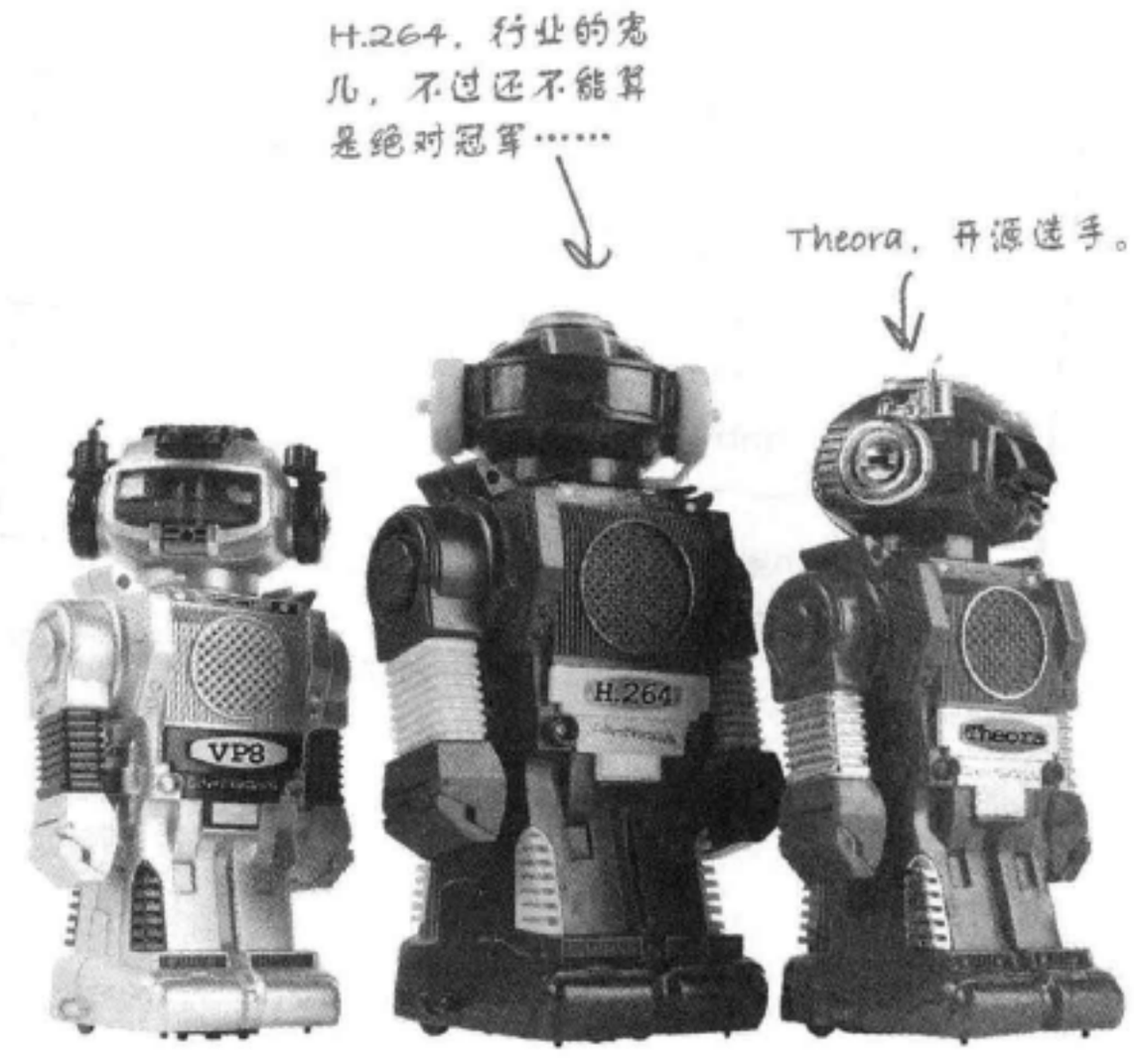
WebM格式的视频扩展名为.webm。

Ogg容器，包含Theora视频和Vorbis音频

Theora是一个开源编解码器。

采用Theora编码的视频通常包含在Ogg文件中，文件扩展名为.ogv。

Ogg/Theora得到了Firefox、Chrome和Opera的支持。



CASE FILE: VIDEO

TOP SECRET

下一项任务：视频侦察

请四处走访一下，确定[redacted]各个浏览器对视频的支持水平（提示，这里给出几个网站，可以从中找到有关的最新信息：**http://en.wikipedia.org/wiki/HTML5_video,** [redacted]**, http://caniuse.com/#search=video**）。这里假设都使用浏览器的最新版本。对于每组浏览器/特性，如果得到支持就画勾，完成任务后，要给出你的任务报告！

iOS和Android以及其他设备。

视频 \ 浏览器	Safari	Chrome	Firefox	Mobile WebKit	Opera	IE9+	IE8	IE7 or <
H.264								
WebM								
Ogg Theora								

如何处理所有这些格式……

现在我们知道了，在视频格式方面，这真是一个混乱的世界，不过该怎么办呢？你可能决定只提供某一种视频格式，也可以提供多种，这取决于你的用户。不论是一种还是多种，都可以利用<video>元素来支持，在<video>元素中可以对应每种格式分别使用一个<source>元素（不要与src属性混淆），这样就能提供一组视频，每个视频分别有自己的格式，可以让浏览器选择它支持的第一种格式。如下：

注意我们从<video>标记删除了src属性……

……并增加了3个source标记，每个标记有自己的src属性，分别包含不同格式的视频版本。

```
<video controls autoplay width="512" height="288"
      src="video/tweetsip.mp4">
    <source src="video/tweetsip.mp4">
    <source src="video/tweetsip.webm">
    <source src="video/tweetsip.ogv">
    <p>Sorry, your browser doesn't support the video element</p>
</video>
```

如果浏览器不支持视频，就会显示这个文本。

浏览器从上向下查找，直到找到它能播放的一种格式。

对于每个source元素，浏览器会加载视频文件的元数据，查看能不能播放这个视频（这个过程可能很耗费时间，不过我们可以在浏览器上更轻松地做到……请看下一页）。

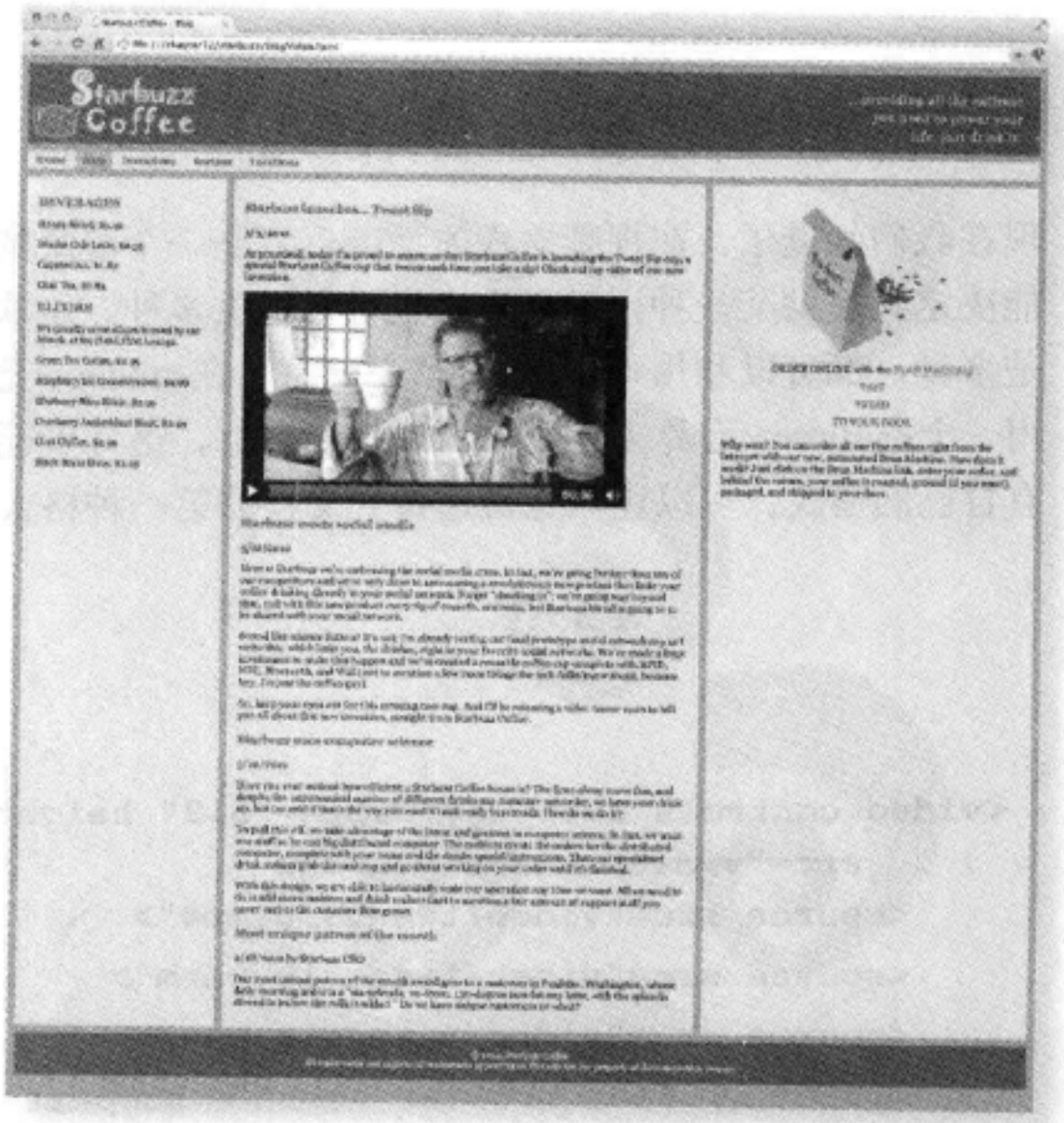
BULLET POINTS

- 容器（container）是用来包装视频、音频和元数据信息的文件格式。常用的容器格式包括：MP4、WebM、0gg和Flash Video。
- 编解码器（codec）是用来对一种特定视频或音频编码完成编码和解码的软件。流行的Web编解码器包括：H.264、VP8、Theora、AAC和Vorbis。
- 要由浏览器决定可以对哪种格式的视频解码，不过并不是所有浏览器制造商都达成了一致，所以如果你想支持所有视频，就需要多种编码。

再来一次：灯光，摄像，开拍……

OK，如果没有看到视频，可以增加上一页的标记，即使你没有遇到麻烦，也请增加这些标记。再来试试这个视频。还要在一些不同的浏览器中尝试。

现在不同浏览器上应该都能看到视频正常播放！



如何更具体地指定视频格式

可以告诉浏览器视频源文件的位置，让它在不同版本中做出选择。不过，浏览器具体确定是否可以播放一个文件之前，还需要做一些侦察工作。你可以为浏览器提供更多帮助，给出有关视频文件的MIME类型和编解码器（可选）的更多信息：

src中使用的文件实际上是具体视频（和音频以及一些元数据）的容器。

codecs参数指定使用哪个编解码器对视频和音频编码，来创建编码的视频文件。

视频编解码器。

音频编解码器。

```
<source src="video/tweetsip.ogv" type='video/ogg; codecs="theora, vorbis"'>
```

type是一个可选属性，这是向浏览器提供的一个提示，帮助它确定能不能播放这种类型的文件。

这是视频文件的MIME类型。指定了容器格式。

注意codecs参数的双引号。这说明type属性两边需要使用单引号。

接下来，可以更新<source>元素，让它包含我们的3种视频的类型信息，

更新和测试

如下更新<source>元素，对页面做个测试：

```
<video controls autoplay width="512" height="288" >
    <source src="video/tweetsip.mp4" type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'>
    <source src="video/tweetsip.webm" type='video/webm; codecs="vp8, vorbis"'>
    <source src="video/tweetsip.ogv" type='video/ogg; codecs="theora, vorbis"'>
    <p>Sorry, your browser doesn't support the video element</p>
</video>
```

如果不知道codecs参数，可以省略，只使用MIME类型。这样效率会低一点，不过大多数情况下都还能接受。

mp4的编解码器比另外两种更为复杂，因为h.264支持多种“等级”，对应不同使用情况（如高带宽和低带宽）会有不同的编码。所以，要想正确使用，需要了解视频编码的更多细节。

大多数情况下，你的视频会像以往一样播放，不过要知道，有了这些额外的类型和编解码器信息，实际上你在后台对浏览器提供了很大帮助。如果你想完成自己的视频编码，就要对source元素中type参数使用的各种选择有更多了解。可以在http://wiki.whatwg.org/wiki/Video_type_parameters得到有关type参数的更多信息。

there are no Dumb Questions

问：以后几年有没有希望协定一种容器格式或者编解码器类型？我们建立标准不就是为了这个目的吗？

答：可能不会很快有这样一种统一的编码。正像我们前面所说的，这个方面混杂着很多问题，各家公司都想在视频领域掌控自己的命运，另外还涉及很多复杂的知识产权方面的问题。HTML5标准委员会意识到了这一点，决定不在HTML5规范中指定视频格式。所以尽管从理论上讲HTML5可以支持（或者至少认识）所有格式，但实际上要由浏览器制造商来决定支持什么和不支持什么。

如果视频对你来说很重要，一定要关注这个方面。在接下来的几年里，如果能看到情况逐渐明朗，这肯定很有意思。而且，与往常一样，一定要记住你的用户需要些什么，另外要竭尽所能地提供支持。

问：如果我想对我自己的视频编码，如何做起呢？

答：目前有很多视频采集和编码程序，选择哪一个取决于你要采集哪种视频，另外还要看你希望如何使用最终结果。有很多书专门介绍视频编码，所以你要做好准备，接下来会进入一个充斥着新编略语和术语的世界。可以先从简单的开始，使用类似iMovie或Adobe Premiere Elements等程序，这些程序都提供了Web视频编码的功能。如果你想用Final Cut Pro或Adobe Premiere等软件程序做一些真正的视频处理工作，这些软件都提供了自己的制作工具。最后，如果你从一个内容分发网络（Content Delivery Network，CDN）分发你的视频，会有很多CDN公司提供编码服务。所以可以根据你的需要在众多方案中做出选择。

问：我能全屏播放我的视频吗？真奇怪API中居然没有这样的一个属性。

答：这个功能还没有标准化，不过如果在Web上搜索，你会找到很多合适的方法，利用这些方法在某些浏览器上就能做到这一点。有些浏览器（例如，平板电脑上的浏览器）提供了一个全屏控件，这就使video元素具备了这一功能。另外要记住，如果想办法实现了全屏，除了基本的播放外，可能会出于安全原因对视频处理有所限制（当前的插件视频方案也存在同样的问题）。

没问题。

如果你喜欢的技术（不论是HTML、Flash或是其他技术）未得到支持，还有一些技术可以作为退路，允许你采用另一种视频播放器。

下面你会看到一个例子，假设浏览器不知道如何播放HTML5视频，这里将展示如何插入你的Flash视频，让它作为HTML5视频的一个退路。显然，这是一个发展非常迅猛的领域，所以一定要关注Web（网上的更新要比书上的更新快得多），要保证你使用的是最新最棒的技术。如果你更喜欢Flash视频，可能还会发现一些方法可以把HTML5（而不是Flash）作为退路。

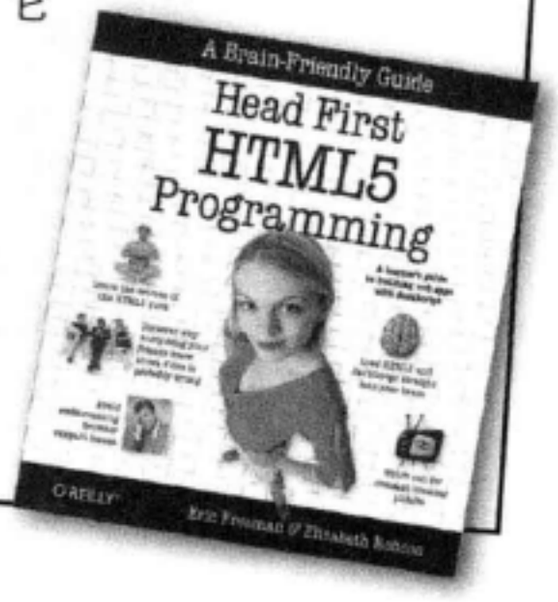
```
<video poster="video.jpg" controls>
        <source src="video.mp4">
        <source src="video.webm">
        <source src="video.ogv">
        <object>...</object>
</video>
```

对于Flash视频，需要一个<object>元素。在<video>元素中插入<object>元素，把它放在<source>标记下面。如果浏览器不认识<video>元素，就会使用<object>，你会看到将播放Flash视频。

我只想说干得真棒！网站已经改头换面，现在只要我们需要就可以加入视频。噢，关于Tweet Sip咖啡杯……嗯，如果你看了视频，我猜你已经知道，我们又在做设计。不过不用担心，我们已经设计出一个新的马克杯，社交网络、游戏化、数字剪贴簿、自动登机和分析统统都包罗其中！我敢打保票，这一个肯定会大获全胜！

你相信吗？我们对视频的讨论才刚刚开始！没错，标记只是第一步。利用HTML5，你还可以使用JavaScript围绕视频创建交互式体验。

不过这远远超出了这本书的范畴（除非你希望这成为一本1400页的大厚书），所以读完这本书之后，请买一本《Head First HTML5 Programming》（当然还是你最喜欢的Head First作者的作品），它会带你更上一个台阶。



元素汤

<progress>

需要显示任务的完成进度吗？比如完成了90%？可以使用这个元素。

<section>

可以使用这个元素定义文档的主要区块。

<header>

有首部的区块和整个文档的页眉可以使用这个元素。

<footer>

这个元素定义一个区块的底部或整个文档的页脚。

这个元素用于表示放在主内容旁边的内容，比如边栏或引用。

<aside>

想在页面中加入一个视频？你肯定需要这个元素。

<video>

<meter>

需要显示某个范围的度量？比如一个从0到212度的温度计，显示现在外面是90度，啊，真热！

这个元素用于突出显示某些文本。就像记号笔一样棒！

<mark>

使用这个元素把网站中用于导航的所有链接组织在一起。

<nav>

可以用它在页面中包含声音内容。

<audio>

用来标记类似新闻报道或博客帖子等独立的内容。

<article>

time元素是一个时间、一个日期或者一个日期时间（如1月21日凌晨2点）。

<time>

这个元素用来在页面中显示用JavaScript绘制的图像和动画。

<canvas>

这个元素用来定义类似照片、图表甚至代码清单等独立的内容。

<figure>

这里有一堆你知道的元素，还有几个你不知道的元素，这些都是HTML5中新增的元素。

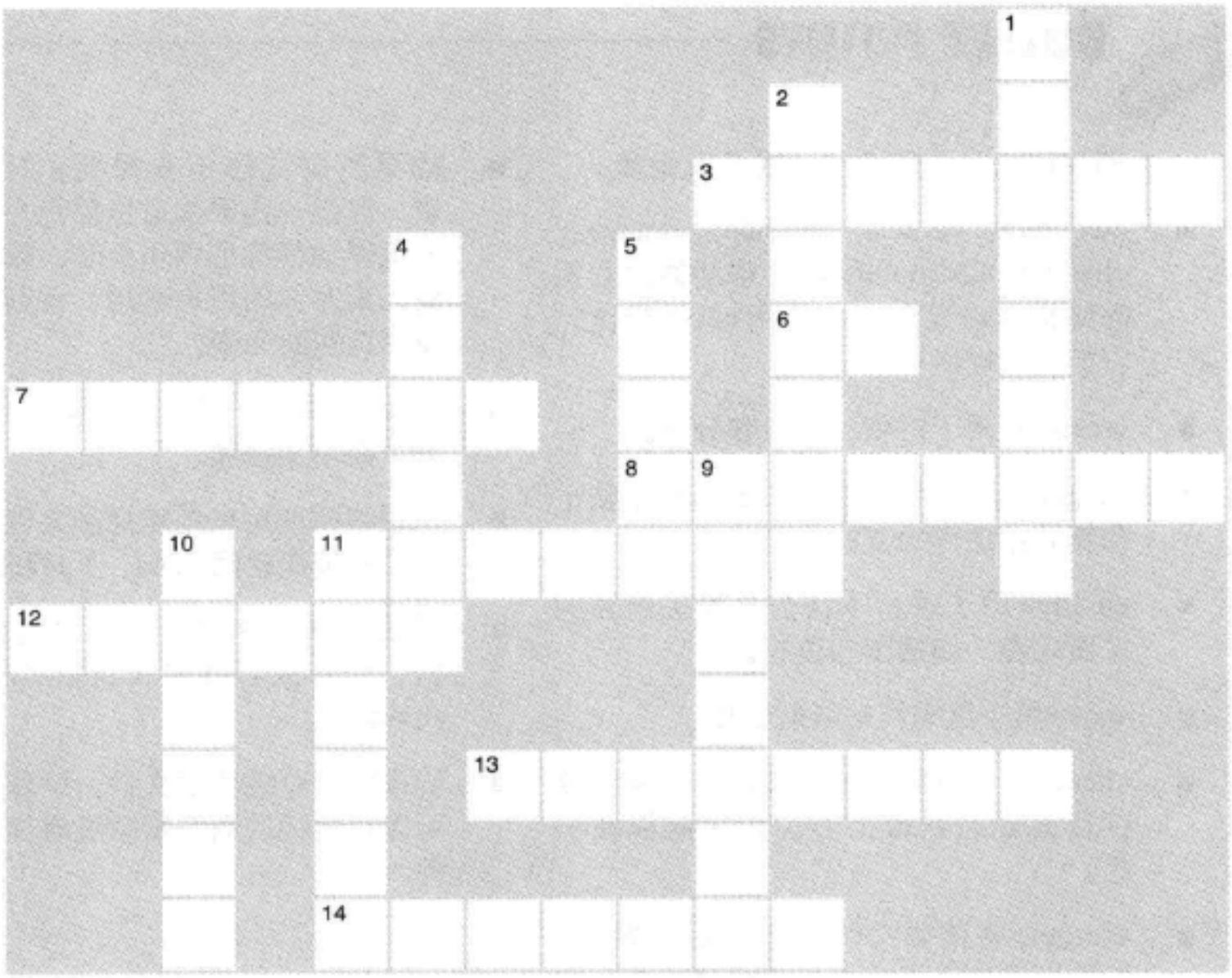
记住，HTML的乐趣一半来自试验！所以你可以自己创建几个文件，来试着用用这些新元素吧。

BULLET POINTS

- HTML5为HTML增加了很多新元素。
- <section>、<article>、<aside>、<nav>、<header>和<footer>都是帮助你建立页面结构的新元素，与使用 相比，它们可以提供更多含义。
- <section>用于对相关的内容分组。
- <article>用于类似博客帖子、论坛帖子和新闻报道等独立的内容。
- <aside>用于表示不作为页面主内容的次要内容，如插图和边栏。
- <nav>用于组织网站导航链接。
- <header>将标题、logo和署名等通常放在页面或区块最上方的内容组织在一起。
- <footer>将诸如文档信息、法律措辞和版权说明等通常放在页面或区块最下方的内容组织在一起。
- <time>也是HTML5中的一个新元素。这个元素用来标记时间和日期。
- 仍然用于建立结构。它通常将元素组织在一起来指定样式，或者有些内容可能不适合放在HTML5中那些与结构相关的新元素中，这些内容就可以使用 创建结构。
- 较早的浏览器不支持新的HTML5元素，所以一定要知道主要用户使用哪些浏览器访问你的Web页面，除非能确保新元素对你的用户适用，否则不要贸然使用这些新元素。
- <video>是一个新的HTML元素，用于为页面增加视频。
- 视频编码是用来创建视频文件的编码。常用的编码包括h.264、Vp8和Theora。
- 视频容器文件包含视频、音频和元数据。流行的容器格式包括MP4、OGG和WebM。
- 要提供多个视频源文件，确保你的用户可以在他们的浏览器中观看你的视频文件。

HTML填字游戏

这一章有很多新想法和新元素。完成下面的填字游戏，把新学的知识牢牢记住。所有答案都可以在这一章中找到。



横向

3. _____属性是没有指定值的属性。
6. TweetSip咖啡杯按 _____度量咖啡。
7. Starbuzz页面的设计有一个主内容 _____。
8. 在<time>元素的 _____属性中指定一个日期。
11. Starbuzz博客中的_____ 样式不正确，直到我们增加了“top”类情况才好转。
12. 浏览器不知道<div id=〃 footer 〃 >表示_____。
13. 在CSS中使用 _____ 选择器，确保不会得到不想要的样式。
14. <section>元素用于将_____ 内容归组。

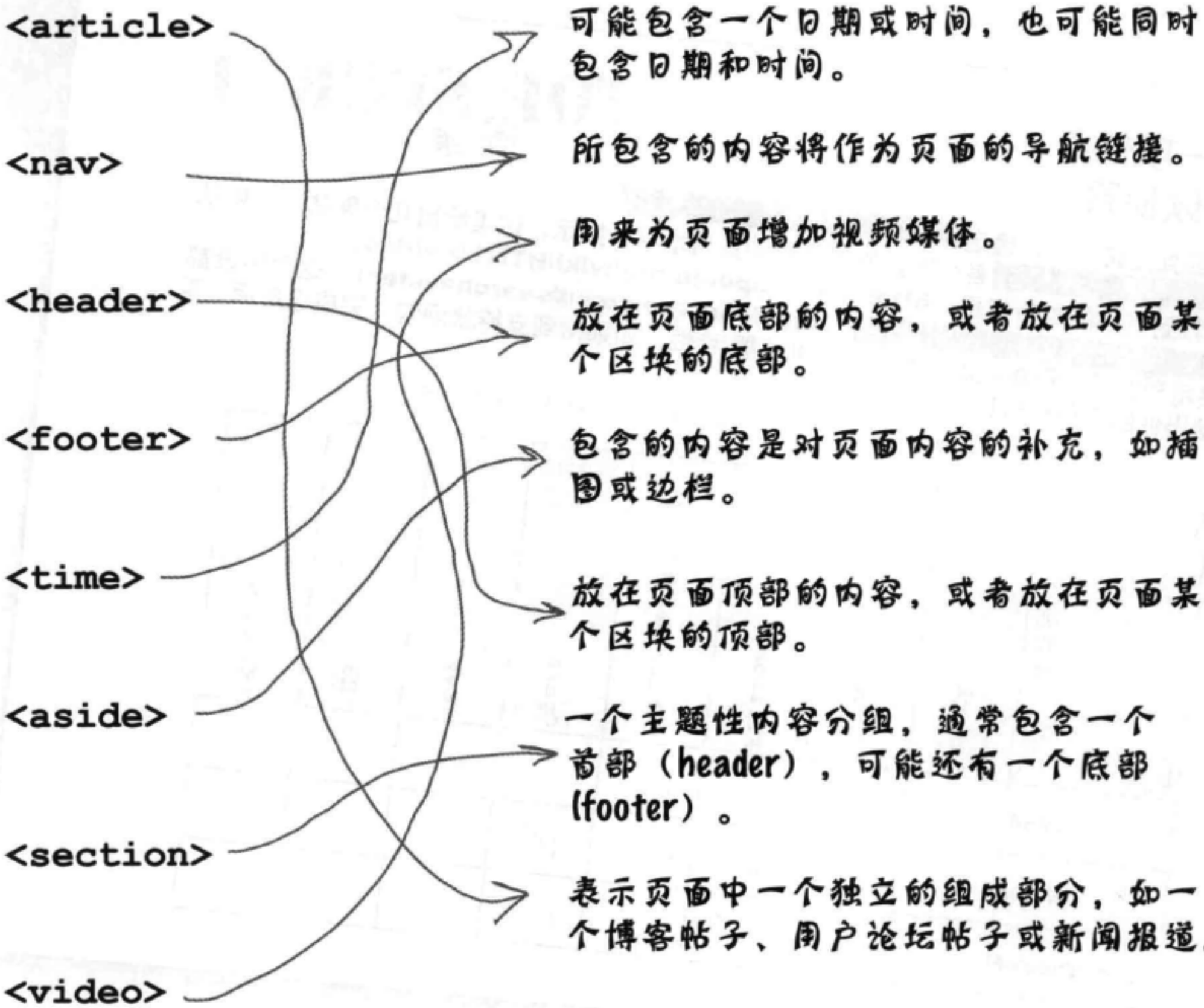
纵向

1. Starbuzz CEO做了一个关于_____ 咖啡杯的视频。
2. 浏览器制造商并没达成一致的视频 _____。
4. 区块可以有一个首部和一个_____。
5. 可以使用这个元素实现边栏。
9. 当地报纸可能使用这种元素来标记新闻报道。
10. _____ 标记用来指定多个视频文件。
11. 可以在页面或文章的最上面使用一个 _____ 。

WHO DOES WHAT?

答案

没错，我们当然可以直接介绍那些新的HTML5元素，不过，由你自己找出来不是更有趣吗？下面在左边给出了新元素（这些肯定不是全部的新元素，不过你会发现比较重要的新元素都在这里），请将各个元素与右边的相应描述连线：



CASE FILE: VIDEO

TOP SECRET

答案

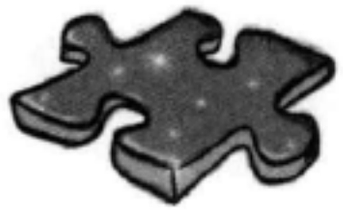
下一项任务：

视频侦察

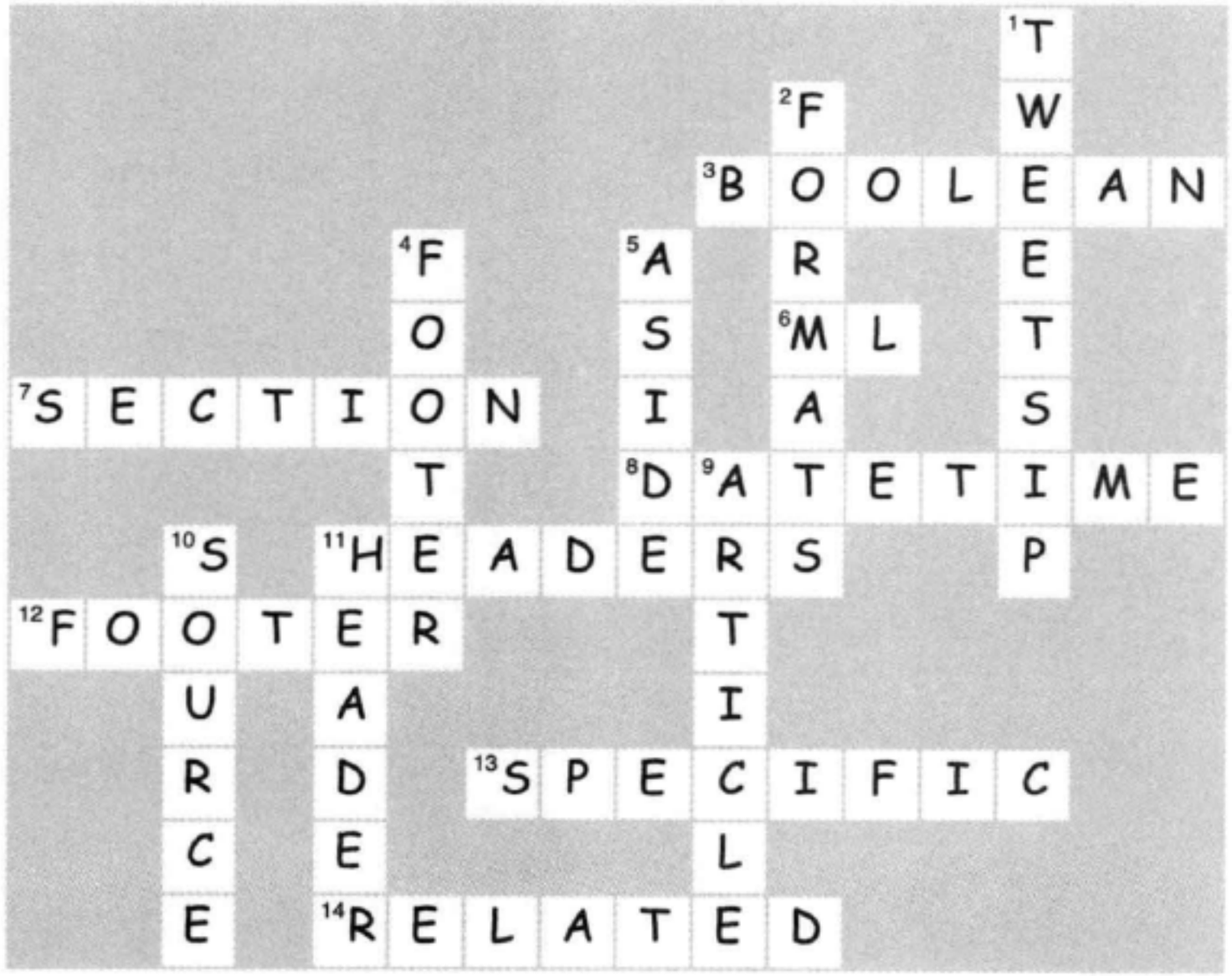
请四处走访一下，确定[redacted]各个浏览器对视频的支持水平（提示，这里给出几个网站，可以从中找到有关的最新信息：**http://en.wikipedia.org/wiki/HTML5_video**，[redacted]，**http://caniuse.com/#search=video**）。这里假设都使用浏览器的最新版本。对于每组浏览器/特性，如果得到支持就画勾，完成任务后，要给出你的任务报告！

iOS和Android以及其他设备。

视频 \ 浏览器	Safari	Chrome	Firefox	Mobile WebKit	Opera	IE9+	IE8	IE7 or <
H.264	✓	部分支持		iOS		✓		
WebM		✓	✓	Android	✓			
Ogg Theora		✓	✓		✓			



HTML填字游戏答案


1T
2F W
3BOOLEAN
4F 5A R E
O S 6ML T
7SECTION I A S
T 8D 9ATETIME
10S 11HEADERS P
12FOOTER T
U A I
R D 13SPECIFIC
C E L
E 14RELATED

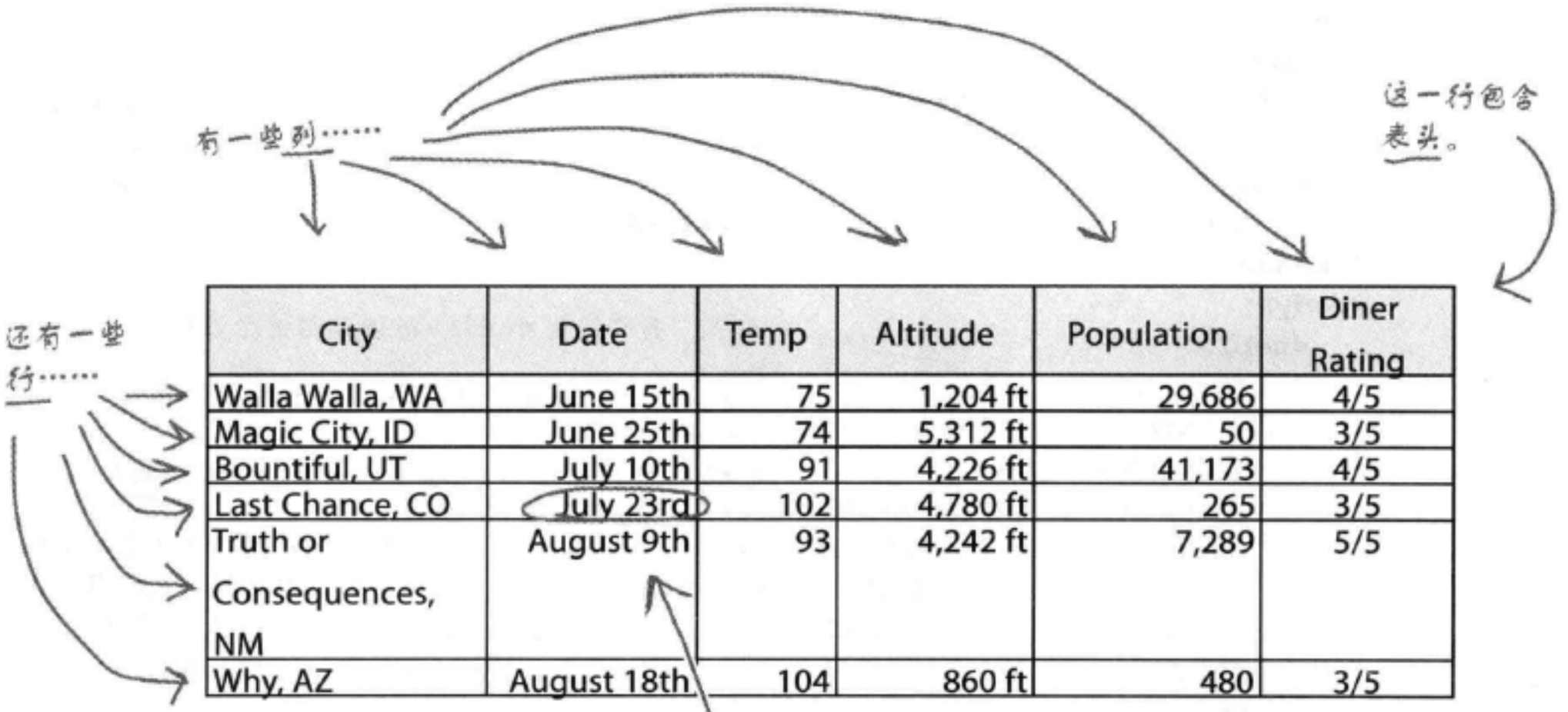
建立表格

如果走路也像表格，说话也像表格……在生活中，总会有一些时候必须处理那些枯燥的表格数据。不论是创建一个页面表示公司去年的库存清单，还是需要为你收藏的玩偶建立一个编目表（放心，我们不会说出去的），你很清楚需要在HTML中建立这些表格，不过怎么做呢？嗯，你拣到便宜了。下单吧，只用一章，我们就能揭开表格的秘密，帮助你把自己的数据正确地放在HTML表格中。还不只这些，我们还会为每张订单提供我们的独家指南，指导你为HTML表格指定样式。另外，如果你现在行动，作为特殊奖励，我们提供指南教你指定HTML列表的样式。别犹豫了，快打电话吧!

City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15	75	1,204 ft	29, 686	4/5
Magic City, ID	June 25	74	5,312 ft	50	3/5
Bountiful, UT	July 10	91	4,226 ft	41, 173	4/5
Last Chance, CO	July 23	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9	93	4,242 ft	7, 289	5/5
Why, AZ	August 18	104	860 ft	480	3/5

如何用HTML建立表格？

Tony说的没错，你确实还没有找到一个好办法使用HTML表示他的表格数据，至少目前还没有。你知道的，可以使用CSS和 来创建一个类似表格的布局（使用CSS表格显示），不过这只是用于建立布局（属于表现范畴），而与内容本身无关。这里我们有一些表格数据，希望用HTML来标记。很幸运，HTML有一个<table>元素专门负责标记表格数据。在深入介绍<table>元素之前，首先来了解表格中有些什么：



City	Date	Temp	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
Why, AZ	August 18th	104	860 ft	480	3/5

如果由你负责设计HTML，你会如何设计（一个或多个）元素来指定表格？表格中可以包含表头、行、列和具体的表格数据。

用HTML创建一个表格

对Tony的网站做出修改之前，先在一个单独的HTML文件中建立表格，让它能像我们期望的那样。我们已经在一个名为“table.html”的HTML文件中开始了这个表格，并输入了表头和表格的前3行，这个文件放在“chapter13/journal/”文件夹中。下面来看这个文件：

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
   <style type="text/css">
      td, th {border: 1px solid black;}
   </style>
   <title>Testing Tony's Travels</title>
</head>
<body>
   <table>
      <tr>
         <th>City</th>
         <th>Date</th>
         <th>Temperature</th>
         <th>Altitude</th>
         <th>Population</th>
         <th>Diner Rating</th>
      </tr>
      <tr>
         <td>Walla Walla, WA</td>
         <td>June 15th</td>
         <td>75</td>
         <td>1,204 ft</td>
         <td>29,686</td>
         <td>4/5</td>
      </tr>
      <tr>
         <td>Magic City, ID</td>
         <td>June 25th</td>
         <td>74</td>
         <td>5,312 ft</td>
         <td>50</td>
         <td>3/5</td>
      </tr>
   </table>
</body>
</html>
```

这是一小段CSS，有了它，我们就能在浏览器中看到表格的结构。现在先不用操心这个CSS。

我们使用<table>标记开始这个表格。

这里是第一行，用一个<tr>开始。

每个<th>元素分别是某一列的表头。

注意，表头是前后排列的。看上去它们好像构成HTML中的一列，但实际上我们在定义整个表头行。再返回去看看Tony的列表，明确他的表头与这里的表头如何对应。

每个<tr>元素构成表格中的一行。

这是第二行的开始，对应城市Walla Walla。

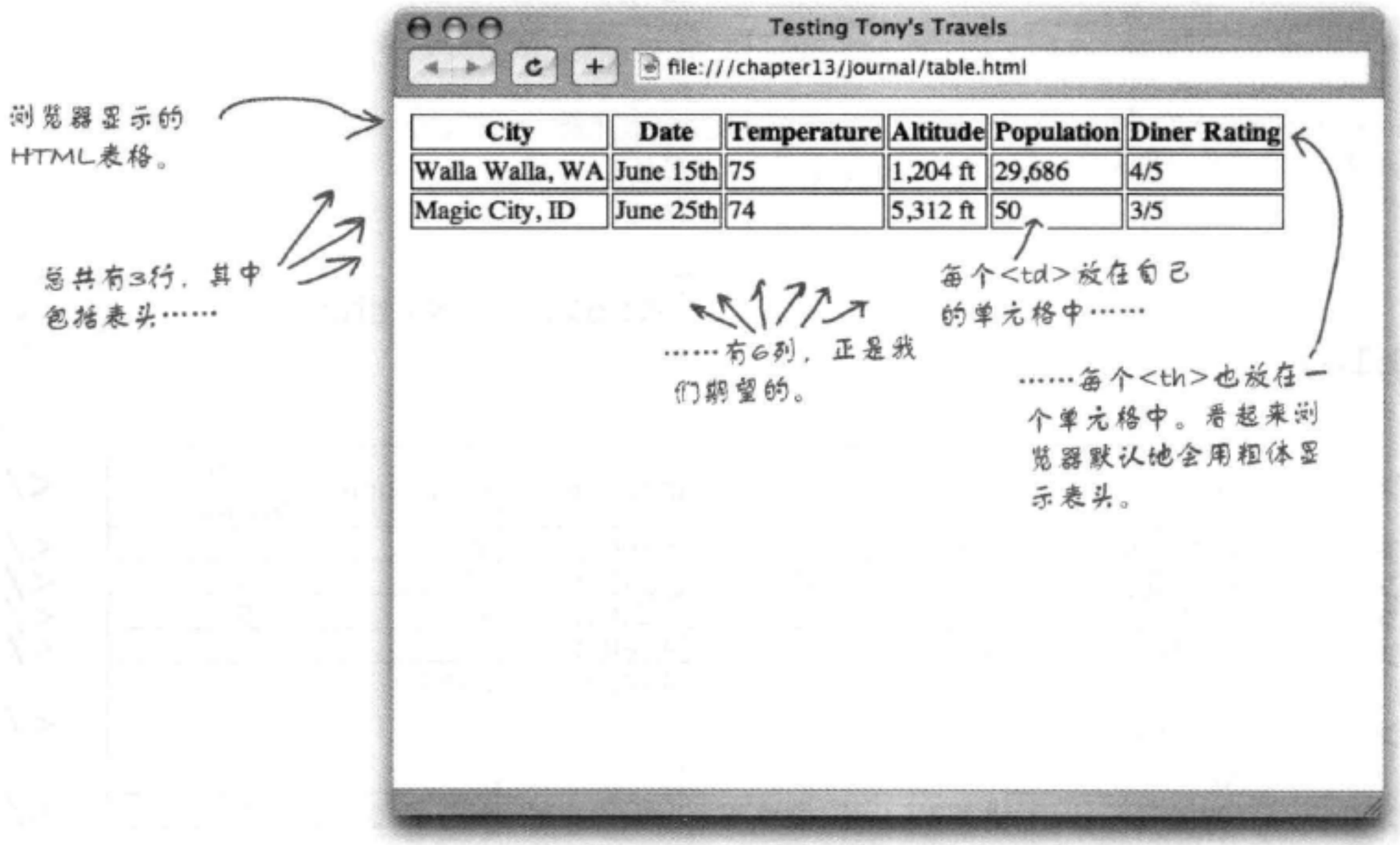
每个<td>元素包含表格中的一个单元格，每个单元格分别构成一个单独的列。

所有这些<td>构成了一行。

这里是第3行。同样的，<td>元素分别包含一个表格数据。

浏览器创建了什么？

下面来看浏览器如何显示这个HTML表格。先提醒一下：这不会很漂亮，不过看上去确实像一个表格。稍后还会考虑它的外观，不过现在先要确保你已经掌握了表格的基础知识。



输入上一页的“Testing Tony's Table” HTML（我们已经在“table.html”中开始了这个表格，不过需要你来完成它）。输入这些代码，尽管很枯燥，不过这会帮助你记住<table>、<tr>、<th>和<td>标记的结构。完成之后，做一个快速测试，然后增加Tony表格中的其余各项。同样要做个测试。

表格剖析

你已经看到了，创建一个表格要使用4个元素：<table>、<tr>、<th>和<td>。下面将逐个地详细介绍这些元素，明确它们在表格中所起的作用。

<table>标记开始整个表格。如果想要一个表格，都要从这个标记开始。

<th>元素包含表格表头中的一个单元格。它必须放在一个表行中。

</tr>标记结束一个表行。

```
<table>
<tr><th>City</th><th>Date</th><th>Temp</th><th>Altitude</th><th>Population</th><th>Diner Rating</th></tr>
<tr><td>Walla Walla, WA</td><td>June 15th</td><td>75</td><td>1,204 ft</td><td>29,686</td><td>4/5</td></tr>
<tr><td>Magic City, ID</td><td>June 25th</td><td>74</td><td>5,312 ft</td><td>50</td><td>3/5</td></tr>
<tr><td>Bountiful, UT</td><td>July 10th</td><td>91</td><td>4,226 ft</td><td>41,173</td><td>4/5</td></tr>
<tr><td>Last Chance, CO</td><td>July 23rd</td><td>102</td><td>4,780 ft</td><td>265</td><td>3/5</td></tr>
<tr><td>Truth or Consequences, NM</td><td>August 9th</td><td>93</td><td>4,242 ft</td><td>7,289</td><td>5/5</td></tr>
<tr><td>Why, AZ</td><td>August 18th</td><td>104</td><td>860 ft</td><td>480</td><td>3/5</td></tr>
</table>
```

每个<tr>元素指定一个表行。所以，放在一行中的所有表格数据都要嵌套在<tr>元素中。

<td>元素包含表格中的一个数据单元格。它必须嵌套在一个表行中。

</table>标记结束表格。

there are no Dumb Questions

问：为什么没有一个表列元素？感觉好像很重要。

答：HTML的设计者决定按行指定表格，而不是按列来指定。不过，要注意，指定每一行的<td>元素时，隐含地就是在指定各列。

问：如果有一行没有足够的元素，会出现什么情况？换句话说，如果实际的元素数小于表格中的列数，会有什么结果？

答：要处理这种情况，最容易的办法就是将数据单元格的内容留空。也就是说，要写为<td></td>。如果省去了这个数据单元格，表格就不能正确地排列对齐，所以要列出所有数据单元格，即使它们的内容为空。

问：如果我希望表格表头放在表格的左侧，而不是放在上方，能行吗？怎么做到呢？

答：嗯，当然可以做到。只需要把表格表头元素放在各行中，而不是都放在第一行中。如果你的<th>元素是各行中的第一项，那么第一列就会包含所有表格表头。

问：我朋友给我展示了一个很酷的技巧，他利用一个表格就能建立所有页面布局，甚至没有使用CSS！

答：还是要用CSS，别走捷径，别贪小利。

在HTML时代，使用表格来建立布局确实很常见，但那是在CSS出现之前，坦率地讲，那时要建立复杂的布局没有更好的办法了。不过，时至今日，利用表格建立布局实在是一种糟糕的方法。如果采用这种方法，不仅很难建立正确的布局，而且还很难维护。实际上，使用CSS表格显示会好得多，不仅可以得到表格布局，而且不必真正创建一个HTML表格（第11章中我们就是利用CSS表格显示为Starbuzz页面建立了样式）。请告诉你的朋友，他的这种技术太老套了，他要赶快学习建立布局的正确方法，也就是要结合使用CSS和HTML。

问：表格不就是关于表现的吗？表现与结构有什么关系？

答：不是这样的。利用表格，可以指定表格数据项之间的关系。另一方面，我们会用CSS来改变表格的外观表现。

问：HTML表格与CSS表格显示有什么关系？

答：HTML表格允许你使用HTML标记指定表格的结构，而CSS表格显示则提供了一种方法，可以用一种类似表格的表现方式显示块级元素。可以这样来考虑，确实需要在页面中创建表格数据时，就使用HTML表格（稍后我们会介绍如何为表格指定样式）。不过，如果只需要对其他类型的内容使用一种类似表格的表现方式，就可以使用CSS表格显示布局。

问：可以使用CSS表格显示为HTML表格指定样式吗？

答：嗯，没必要。为什么？因为你已经用HTML创建了一个表格结构，所以，可以看到，只需要使用简单的CSS就能为表格指定你喜欢的任何样式。

表格提供了一种在HTML中指定表格数据的方法。

表格由行中的数据单元格组成。列隐含地定义在行中。

表格中的列数就是行中数据单元格的个数。

一般来讲，表格不用来提供表现，那是CSS的工作。

扮演浏览器

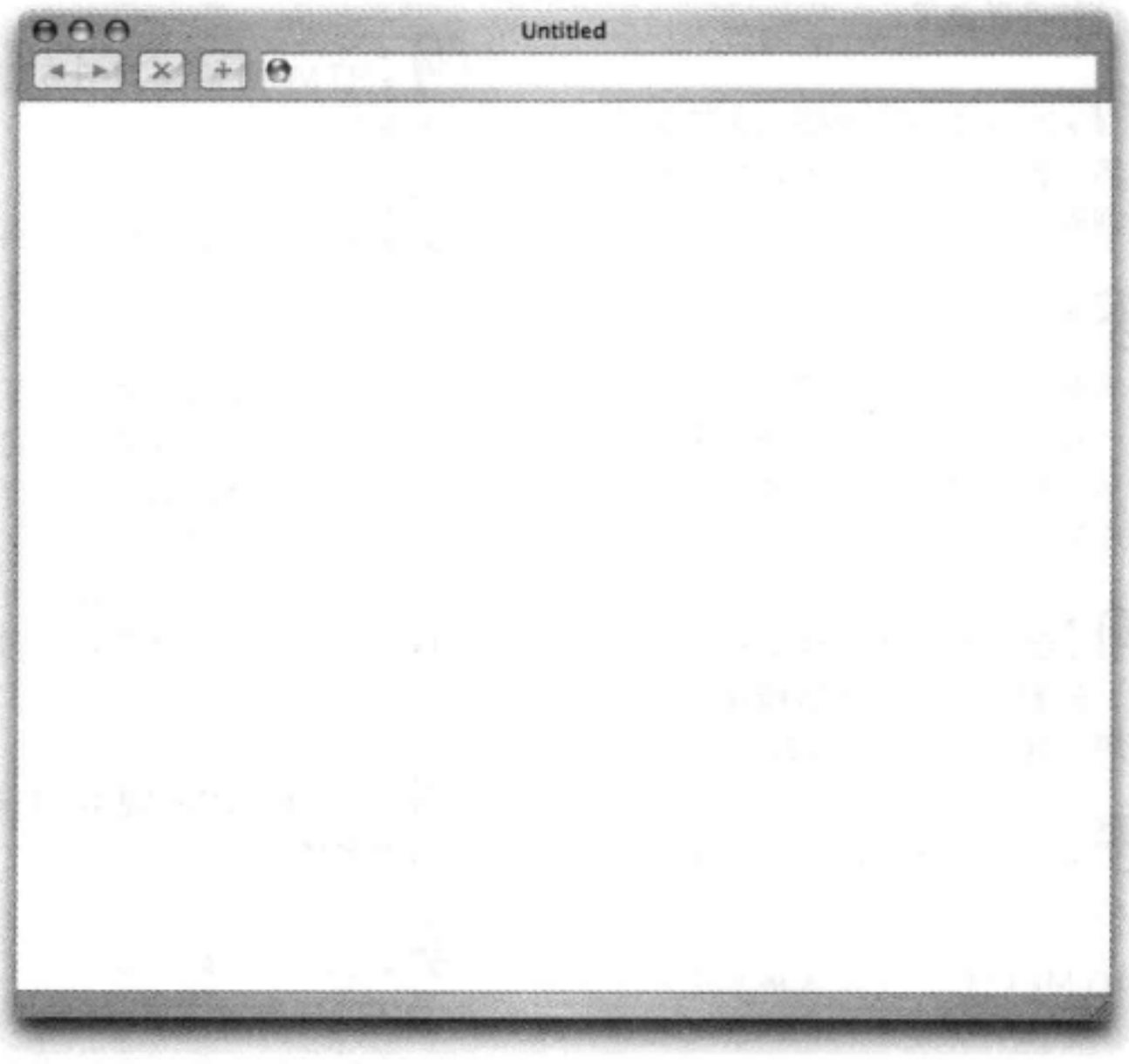
在左边可以看到一个表格的HTML。你的任务是扮演显示这个表格的浏览器。完成这个练习之后，对照这一章最后的答案，检查你做对了没有。

```
<table><tr><th>Artist</th>
<th>Album</th></tr><tr>
<td>Enigma</td><td>Le Roi Est Mort,
Vive Le Roi!</td></tr> <tr><td>LTJ
Bukem</td>
<td>Progression Sessions 6</td>
</tr><tr>
<td>Timo Maas</td>
<td>Pictures</td></tr></table>
```

这是表格的HTML。

哎呀！看来有人得学学如何调整HTML的格式。

在这里画出表格。



增加一个标题

通过增加一个标题，可以让表格立即有所改观。

```
<table>
  <caption>
      The cities I visited on my
      Segway'n USA travels
  </caption>
  <tr>
    <th>City</th>
    <th>Date</th>
    <th>Temperature</th>
    <th>Altitude</th>
    <th>Population</th>
    <th>Diner Rating</th>
</tr>
<tr>
    <td>Walla Walla, WA</td>
    <td>June 15th</td>
    <td>75</td>
    <td>1,204 ft</td>
    <td>29,686</td>
    <td>4/5</td>
</tr>
<tr>
    <td>Magic City, ID</td>
    <td>June 25th</td>
    <td>74</td>
    <td>5,312 ft</td>
    <td>50</td>
    <td>3/5</td>
  </tr>
  .
  .
  .
</table>
```

标题在浏览器中显示。默认地，大多数浏览器会把标题显示在表格上方。

如果你不喜欢标题的默认位置，可以使用CSS重新指定它的位置（稍后我们就会尝试改变它的位置）。要记住，比较老的浏览器还不能完全支持标题位置的调整。

要在HTML中把标题放在表格上方，再使用CSS调整它的位置，比如调整到表格下方（如果你希望这样）。

其余的表格行放在这里。

测试……开始考虑样式

为表格增加标题，保存并重新加载页面。

标题在表格上方。放在下面可能看起来会更好点。

Tony's Table

file:///chapter13/journal/table.html

The cities I visited on my Segway'n USA travels

City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
Why, AZ	August 18th	104	860 ft	480	3/5

我们确实需要为表格数据单元格增加一些内边距，以便于阅读……

……为Tony的网站加一点点橙色，可以把所有内容联系起来。

……另外边框线看起来“很粗”。在表格单元格中可以用“更细”的边框，不过，最好整个表格外面有一个深色边框……

开始指定样式之前，先把表格放在Tony的页面上

开始为Tony的新表格增加样式之前，先要把这个表格放在他的主页上。记住，Tony的主页已经设置了字体系列（font-family）、字体大小（font-size），以及其他一些样式，表格将继承这些样式。所以如果不把表格具体放在他的页面上，就无法准确地知道表格到底是什么样子。

首先打开“chapter13/journal”文件夹中的“journal.html”。找到8月10日的相应条目，完成以下修改。完成之后，在重新加载页面之前先翻到下一页。

```
<h2>August 20, 2012</h2>
<p>
   <img src="images/segway2.jpg" alt="Me and my Segway in New Mexico">
</p>

<p>
Well, I made it 1200 miles already, and I passed through some interesting
places on the way:
</p>

<ol>
   <li>Walla Walla, WA</li>
   <li>Magic City, ID</li>
   <li>Bountiful, UT</li>
   <li>Last Chance, CO</li>
   <li>Truth or Consequences, NM</li>
   <li>Why, AZ</li>
</ol>
```

这是原来的城市列表。将它删除，因为我们要把它换成表格。

```
<table>
   <caption>The cities I visited on my Segway'n USA travels</caption>
   <tr>
      <th>City</th>
      <th>Date</th>
      <th>Temperature</th>
      <th>Altitude</th>
      <th>Population</th>
      <th>Diner Rating</th>
   </tr>
   .
   .
   .
</table>
```

新表格放在这里。可以从上一个文件复制粘贴过来，这是增加这个表格最容易的做法。

现在为表格增加样式

将下面突出显示的新样式增加到“journal.css”样式表的最下面。

```
@font-face {
  font-family: "Emblema One";
  src: url("http://wickedlysmart.com/hfhtmlcss/chapter8/journal/EmblemaOne-Regular.woff"),
       url("http://wickedlysmart.com/hfhtmlcss/chapter8/journal/EmblemaOne-Regular.ttf");
}
body {
  font-family:     Verdana, Geneva, Arial, sans-serif;
  font-size:       small;
}
h1, h2 {
  color:           #cc6600;
  border-bottom:   thin dotted #888888;
}
h1 {
  font-family:     "Emblema One", sans-serif;
  font-size:       220%;
}
h2 {
  font-size:       130%;
  font-weight:     normal;
}
blockquote {
  font-style:      italic;
}

table {
    margin-left: 20px;
    margin-right: 20px;
    border: thin solid black;
    caption-side: bottom;
}

td, th {
    border: thin dotted gray;
    padding: 5px;
}

caption {
    font-style: italic;
    padding-top: 8px;
}
```

最上面就是目前Tony页面上的所有样式。这些样式是第8章中增加的。现在要把表格的新样式增加到这些样式的下面。

首先，为表格指定样式。我们要在左边和右边增加一个外边距，另外还要为整个表格增加一个黑色的细边框。

我们希望把标题移到表格的下方。

还要修改表格数据单元格的边框，设置为一个更细的灰色虚线边框。

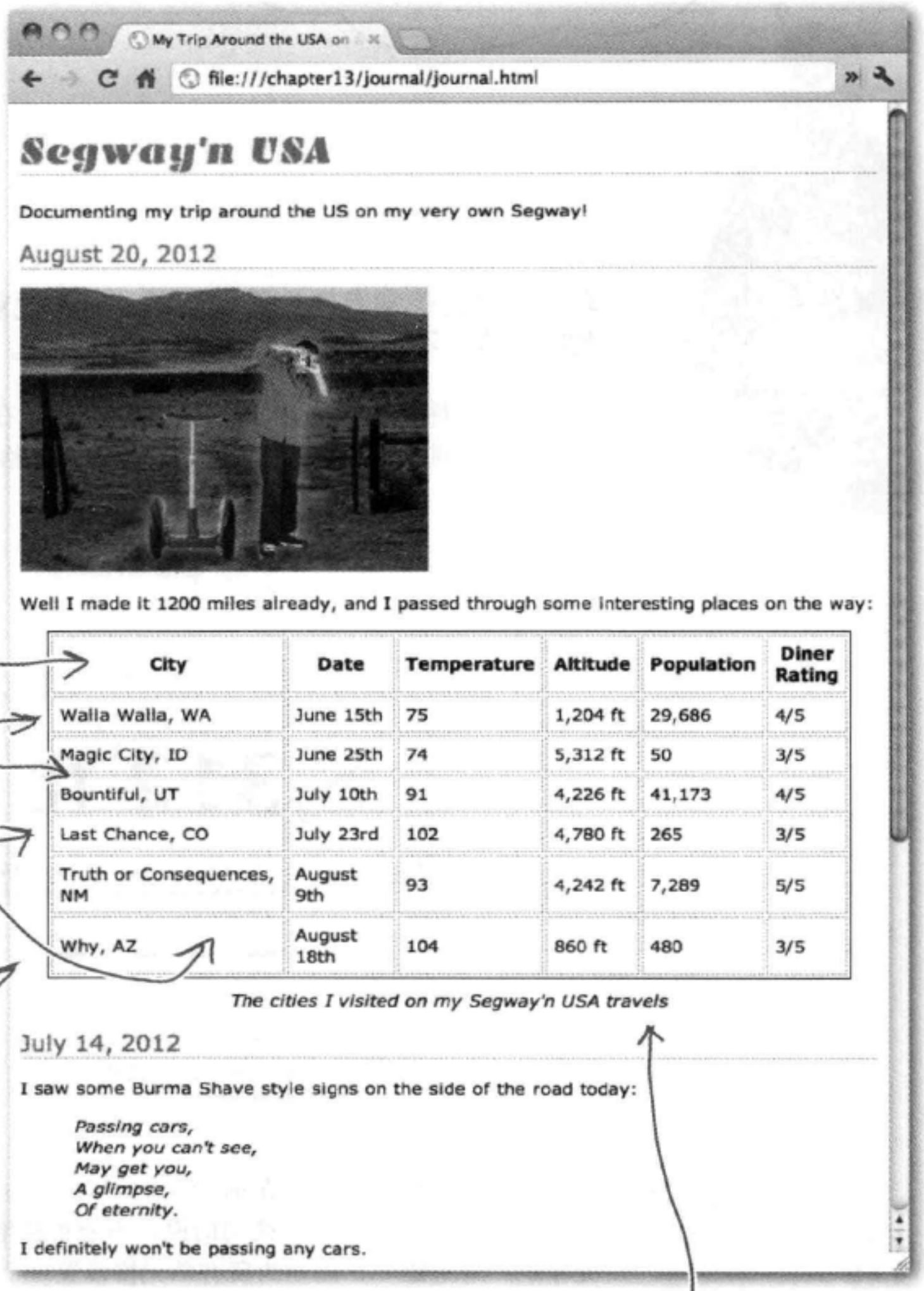
另外为数据单元格增加一些内边距，使数据内容与边框之间有一些空间。

这个规则用于设置表格标题的样式。我们把font-style改为italic，另外增加了一些上内边距。

测试增加样式后的表格

这一次改动可真不少。保存这些修改，而且要完成验证。然后在浏览器中加载“journal.html”。

由于增加了样式，表格现在看起来大不相同。我们还继承了Tony日志中已有的一些样式。



现在所有字体都是sans-serif，而且字体较小。这是从CSS文件中前面已有的样式中选择的。

现在有一个深色虚线边框。

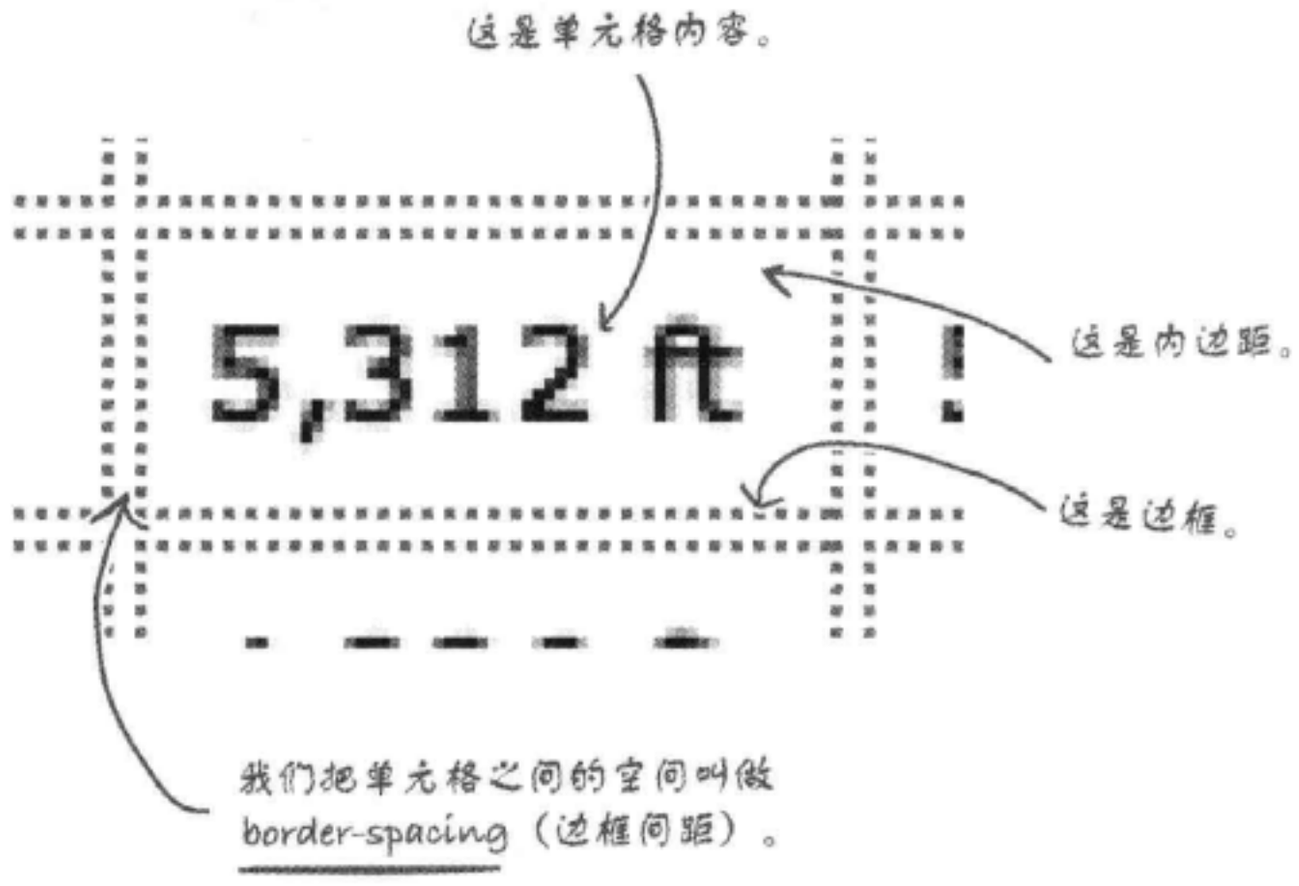
表格上有一些外边距，另外每个表格单元格有一些内边距。

不过，这些虚线看起来很乱，很分散注意力。每对表格单元格之间都重复这些边框没有好处。

记住，在不支持caption-side属性的浏览器中，标题仍然显示在表格上方。

表格单元格确实有内边距和边框，就像之前在盒模型中看到的一样，不过在外边距方面稍有些不同。

盒模型是了解表格单元格的一个好方法，不过在外边距方面它们还是有区别的。下面来看Tony表格中的单元格：

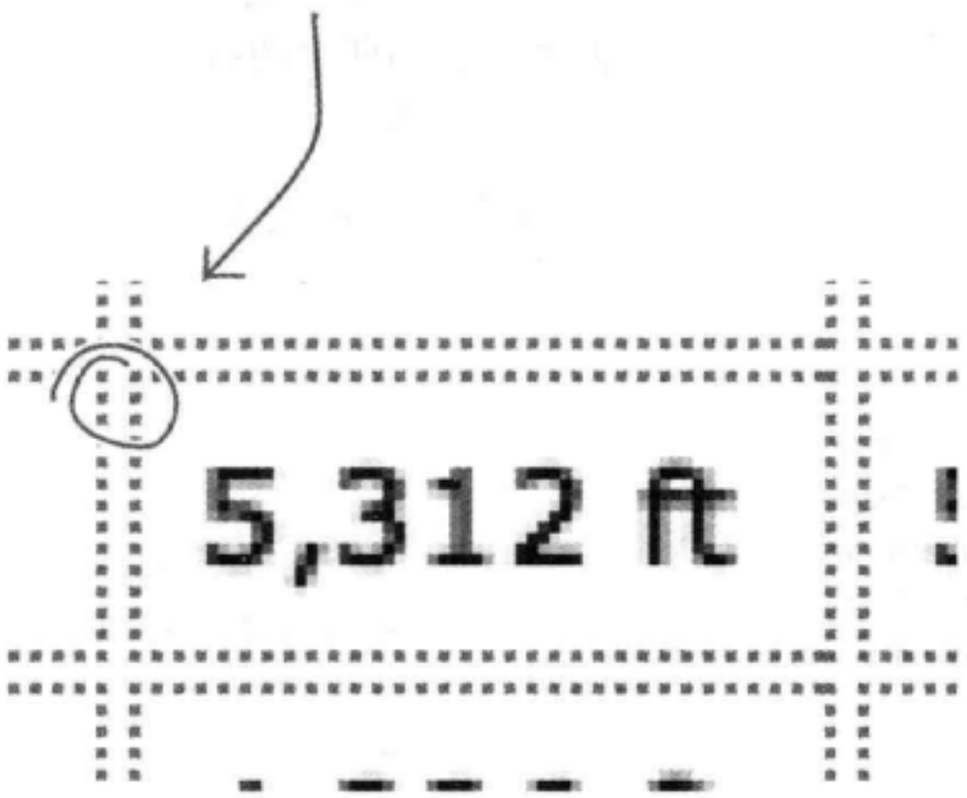


这就像Starbuzz的CSS表格显示布局中使用的border-spacing属性。

所以我们不用外边距，而是有一个border-spacing属性，这是针对整个表格定义的。换句话说，不能单独地设置各个表格单元格的“外边距”，而要为所有单元格设置一个共同的间距。

Sharpen your pencil

双虚线使Tony的表格看上去很乱，很分散注意力。如果每个表格单元格周围只有一个边框，看上去就会好得多。能不能利用你之前学到的知识来指定样式，想办法把双虚线变成单线？试一试，对照这一章最后的答案检查你做的对不对。



there are no Dumb Questions

问：你说过要为整个表格定义边框间距，这么说，我不能为单个表格单元格设置外边距，是吗？

答：对，表格单元格没有外边距，它们只是在边框周围有间距，这个间距是为整个表格设置的。不能单独地控制各个表格单元格的边框间距。

问：嗯，有没有办法在垂直方向和水平方向上设置不同的边框间距？好像这很有用。

答：当然可以。可以像这样指定边框间距：

```
border-spacing: 10px 30px;
```

这会设置10像素的水平边框间距，30像素的垂直边框间距。

问：看起来border-spacing属性在我的浏览器中不起作用。

答：你是不是还在使用一个老版本的Internet Explorer？很遗憾地告诉你，IE 6不支持border-spacing。不过说真的，你是不是该升级浏览器了？

折叠边框

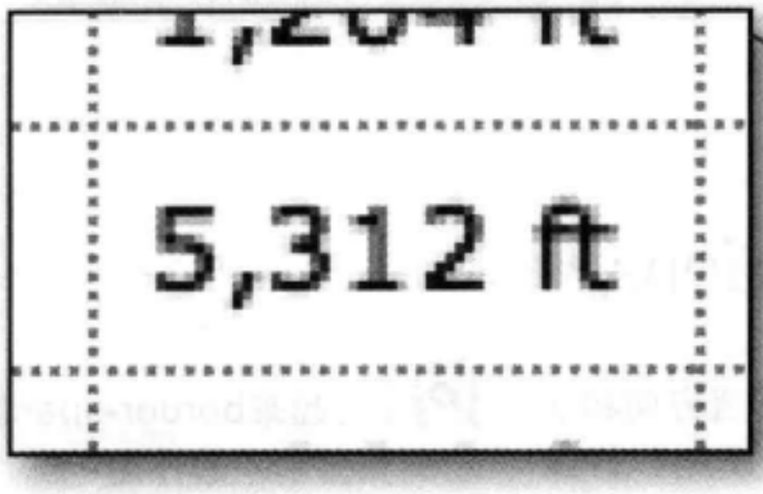
除了border-spacing属性，还有一种办法可以解决这个边框难题。你可以使用一个名为border-collapse的CSS属性来折叠边框，使单元格之间根本没有边框间距。这样一来，浏览器就会忽略表格上设置的所有边框间距。另外还会把紧挨着的两个边框合并成一个边框。这样两个边框就会“折叠”为一个边框。

可以如下设置border-collapse属性。按照下面的设置，对“journal.css”文件完成修改：

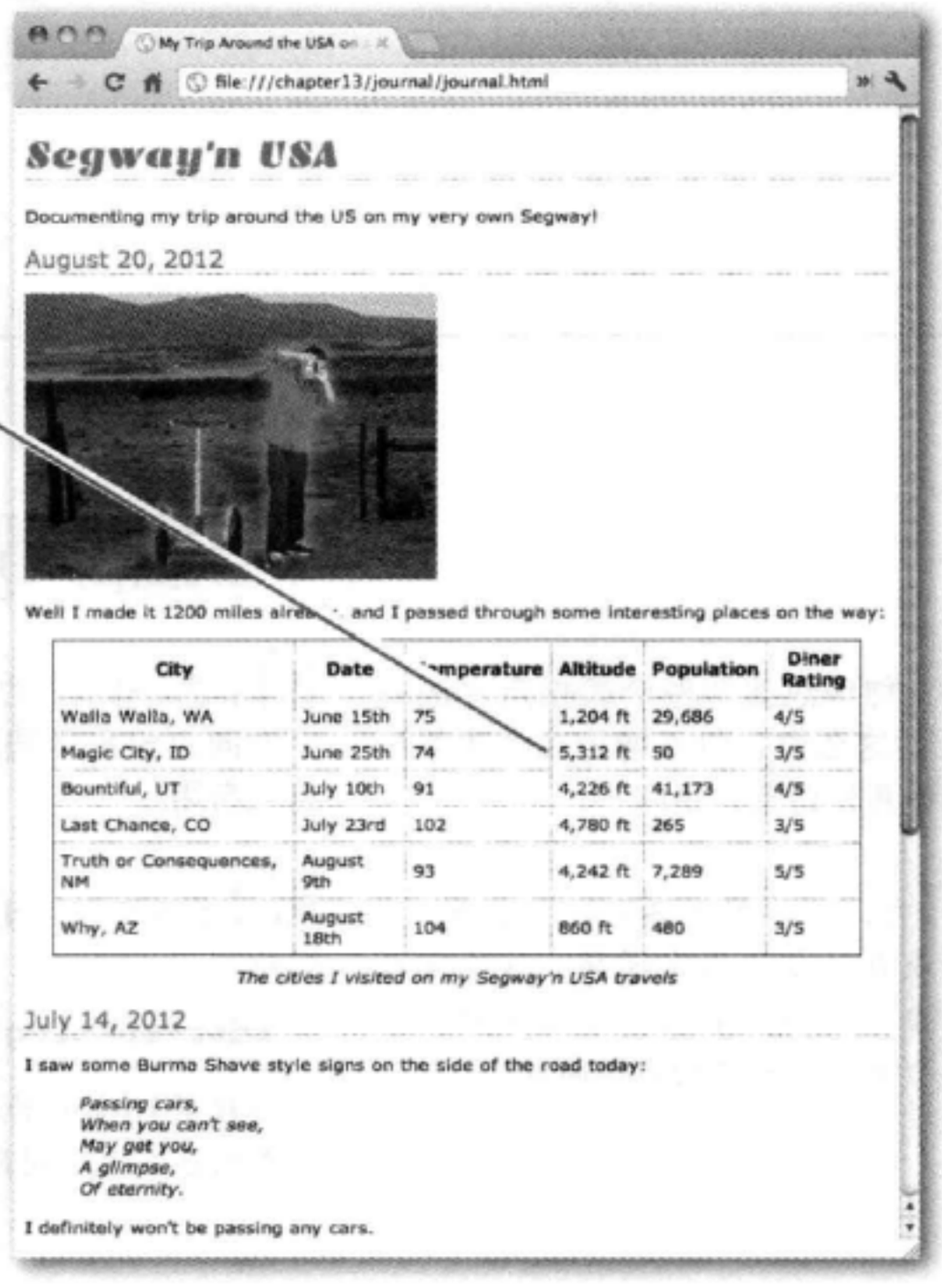
```
table {
    margin-left: 20px;
    margin-right: 20px;
    border: thin solid black;
    caption-side: bottom;
    border-collapse: collapse;
}
```

增加一个border-collapse属性，设置属性值为“collapse”。

保存文件并重新加载页面，然后检查边框的变化。



现在所有表格单元格周围只有一个单线边框。这正是我们想要的，你不觉得吗？现在表格看上去清晰多了！

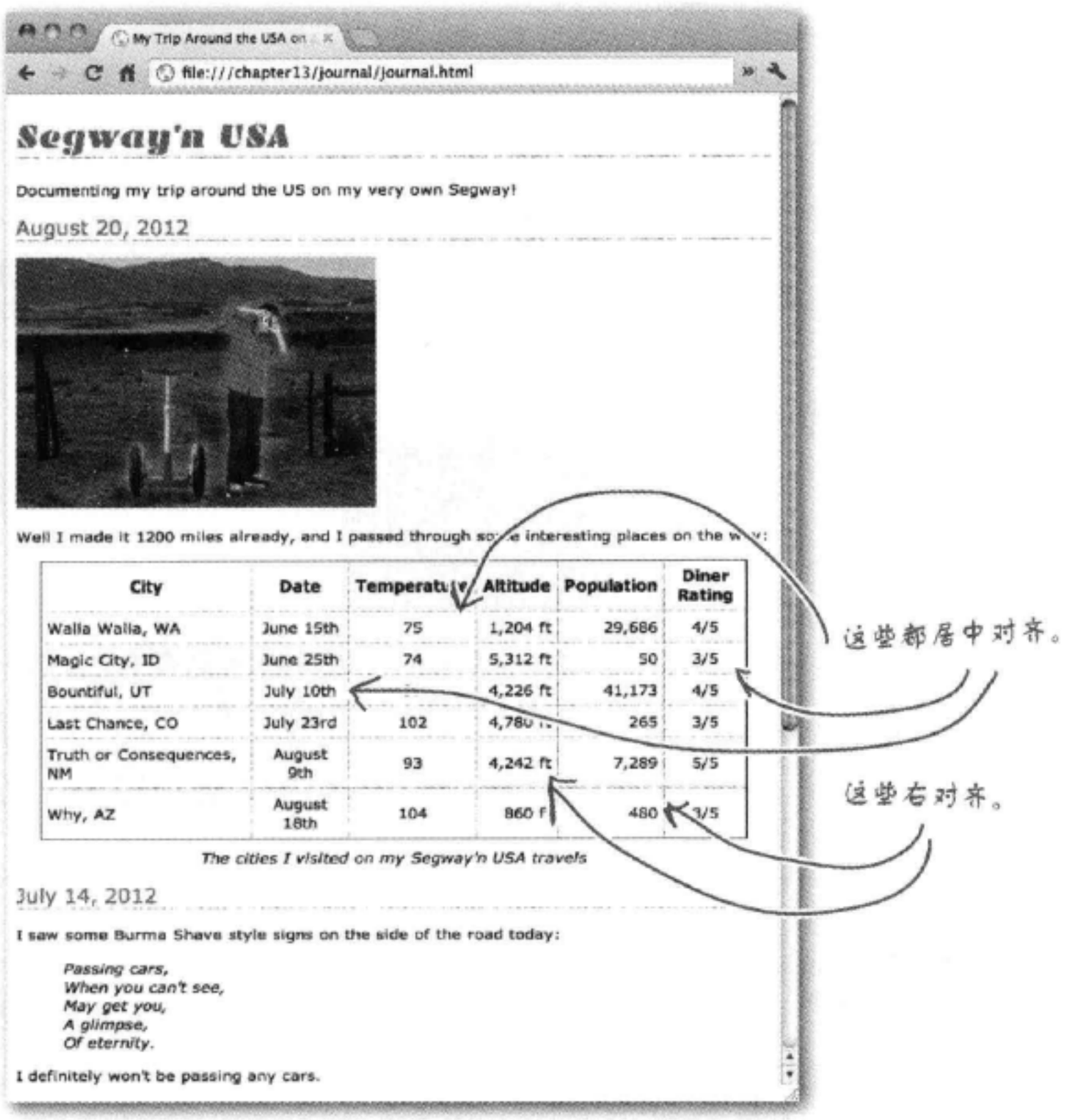


Sharpen your pencil

你对HTML和CSS已经很精通了，所以不妨再通过下面的练习来练练手。来看看这个问题：我们希望再对这个表格做些修饰，先来解决一些文本对齐问题。假设我们希望日期、温度和用餐评分居中对齐，另外让海拔高度和人口右对齐，怎么做到？

给你一个提示：可以创建两个类，一个对应居中对齐，另一个对应右对齐。然后分别在各个类中使用text-align属性。最后，为<td>元素增加适当的类。

这听上去可能有困难，不过我们可以一步一步来完成，你已经了解完成这个任务所需的全部知识。另外，当然了，这一章会在最后给出练习答案，不过在看答案之前，先花点时间看看能不能自己完成。

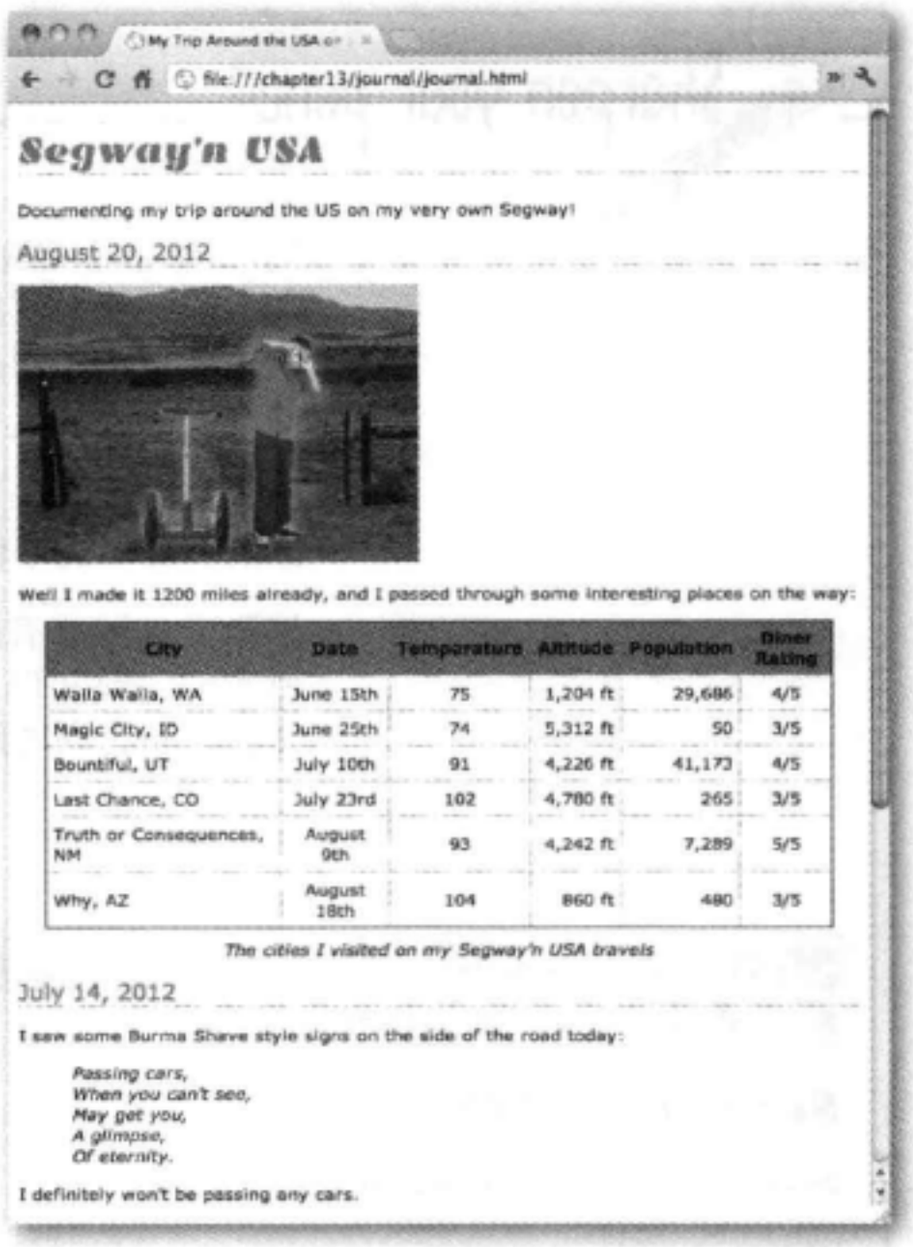


来点颜色怎么样？

你知道的，Tony很喜欢他的个性签名颜色，所以没有理由不在表格中加一点橙色，这样不仅看上去更美观，另外增加一点颜色还会改善表格的可读性。类似于所有其他元素，你要做的就是设置表格单元格的background-color属性来改变它的颜色（注意，我们学到的HTML和CSS知识已经开始融会贯通了）。可以这样做：

```
th {
    background-color: #cc6600;
}
```

把这个新规则增加到“journal.css”文件，然后重新加载页面。你会看到：



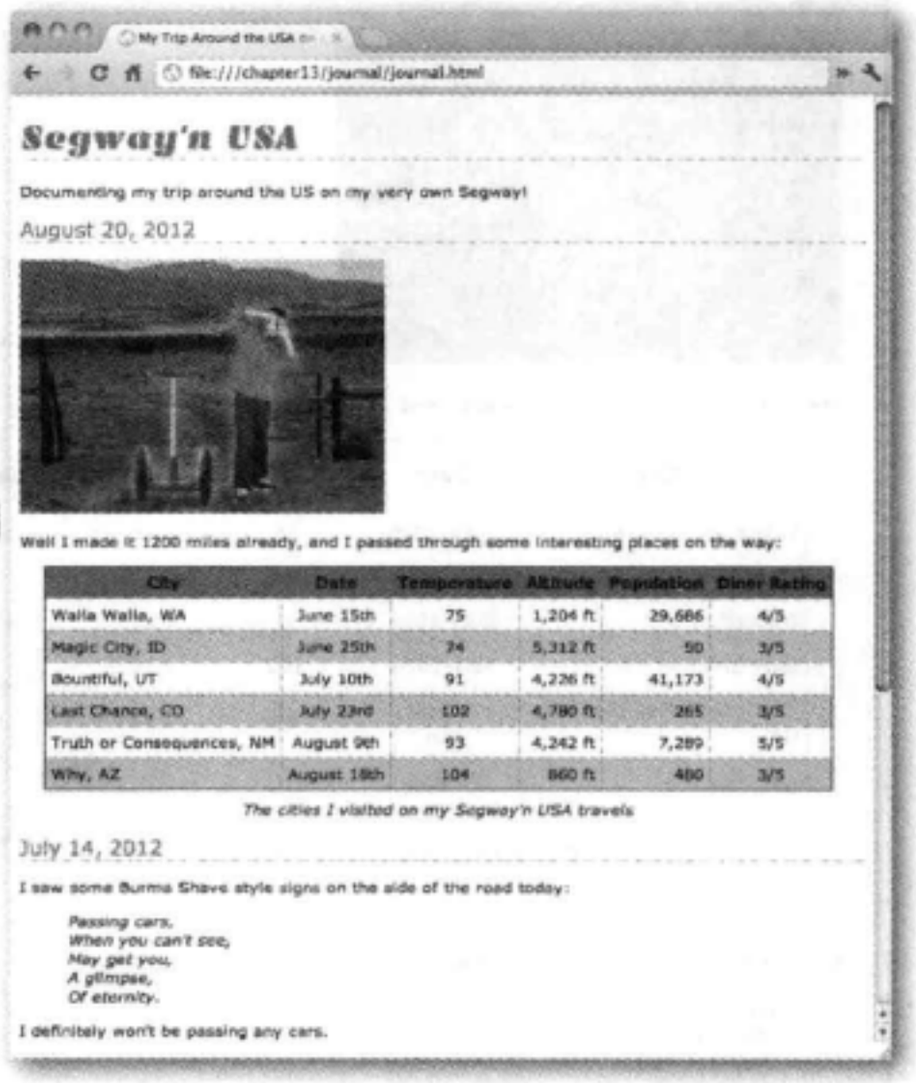
表行里加点颜色怎么样？

到目前为止，颜色看起来很漂亮。下面再更进一步。对表格着色的一种常用方法是为各行指定交替的颜色，这样就能更容易地区分各行，而不会看不清楚哪一列在哪一行中。可以试试看：

用CSS很难做到吧？当然不是。你可以这样做。首先定义一个新类，可以把它叫做“cellcolor”：

```
.cellcolor {
    background-color: #fcba7a;
}
```

然后把这个类属性增加到你希望着色的各行上。所以在这里，你要找到Magic City、Last Chance和Why的<tr>开始标记，分别增加class="cellcolor"。

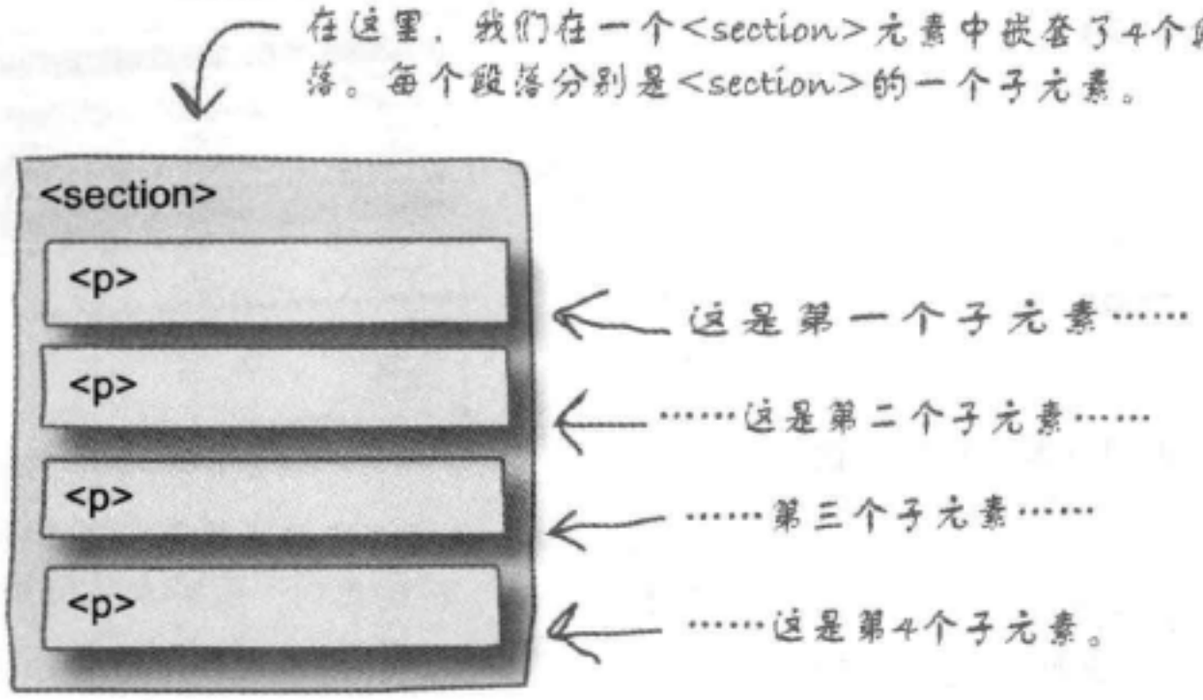


该轮到你了。把class“cellcolor”增加到“journal.css”的CSS中，然后在HTML中，对应要着色的各行，为相应的各个<tr>开始标记增加class="cellcolor"，让这些行有交替的颜色。继续学习后面的内容之前，先检查你的答案。

一些正式CSS

是不是想了解另外一种更高级的方法来为表格隔行增加颜色？这种方法称为nth-child伪类。应该记得，伪类会根据元素的状态来指定元素的样式（如Head First休闲室中使用的a:hover伪类，如果用户把鼠标停在链接上面，a:hover伪类就会指定这个悬停链接的样式）。

对于nth-child伪类，状态则是一个元素相对于它的兄弟元素的数字顺序。下面通过一个例子来看这是什么意思：



假设你想选择偶数段落（也就是第2段和第4段），让它们有一个红色背景，而奇数段落有一个绿色背景，可以这样做：

```
p:nth-child(even) {
  background-color: red;
}
p:nth-child(odd) {
  background-color: green;
}
```

第2段和第4段有红色背景……

……第1段和第3段有绿色背景。

从“nth-child”这个名字可以猜到，这个伪类不只是能选择嵌套在一个元素中的奇数和偶数项，它还可以更加灵活，可以使用数字n指定简单的表达式，从而在选择元素时有更多方式。例如，还可以像这样选择偶数和奇数段落：

选择偶数<p>。

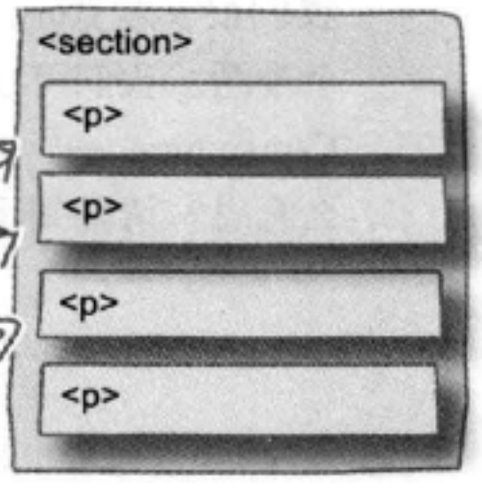
```
p:nth-child(2n) {
  background-color: red;
}
```

选择奇数<p>。

```
p:nth-child(2n+1) {
  background-color: green;
}
```

如果n=0，则2n=0（无段落），2n+1为1，这就是第1个段落。

如果n=1，则2n=2，第2个段落，另外2n+1=3，这是第3个段落。



正式练习

为什么不试着用一用nth-child伪类呢？使用nth-child伪类完成下面的CSS规则，将奇数行设置为淡橙色。

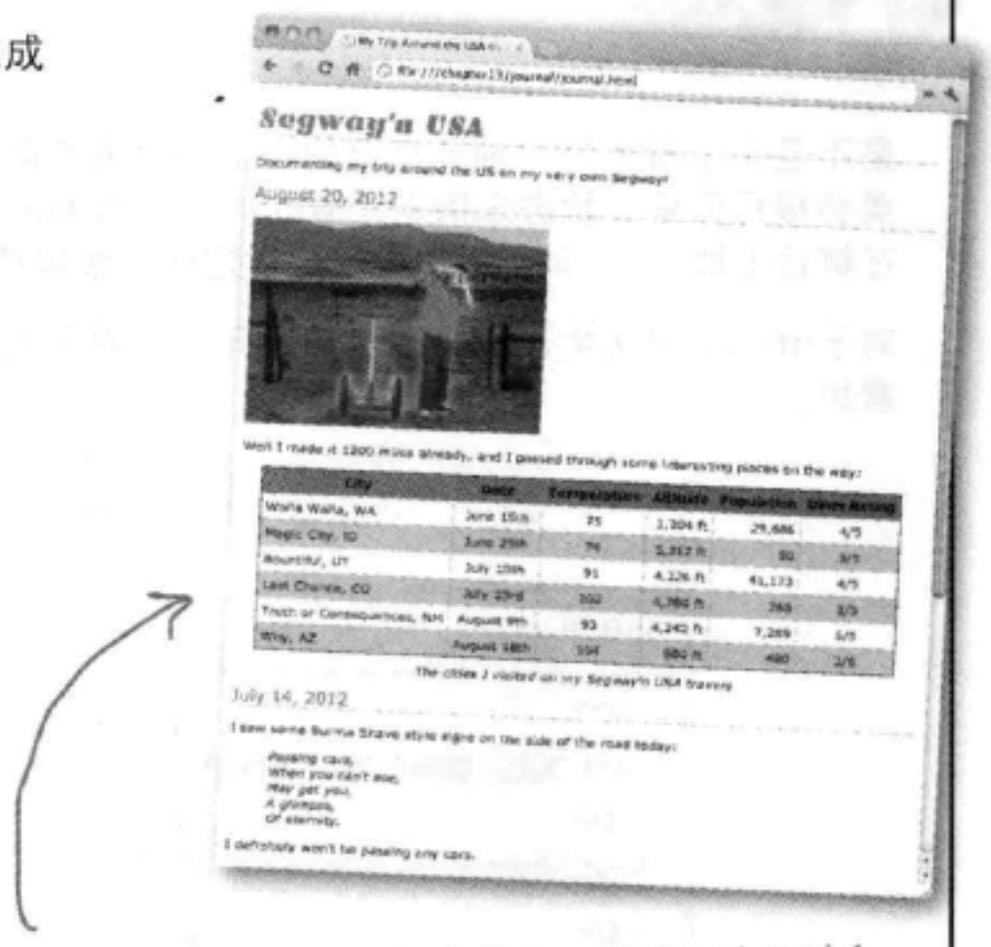
在这里写出你的伪类选择器。

```
tr:_____ {
  background-color: #fcba7a;
}
```

将.cellcolor类注释掉，如下所示：

```
/* .cellcolor {
    background-color: #fcba7a;
} */
```

如果你真想尝试，首先要把你的.cellcolor类注释掉，让它不再起作用。接下来，把新的tr规则放在设置<th>行背景颜色的规则之上（使<th>行仍然为深橙色）。一定要使用一个现代浏览器(IE9+!)，重新加载页面。成功了吗？继续学习下面的内容之前，还要删除这个新规则，去掉.cellcolor规则的注释。

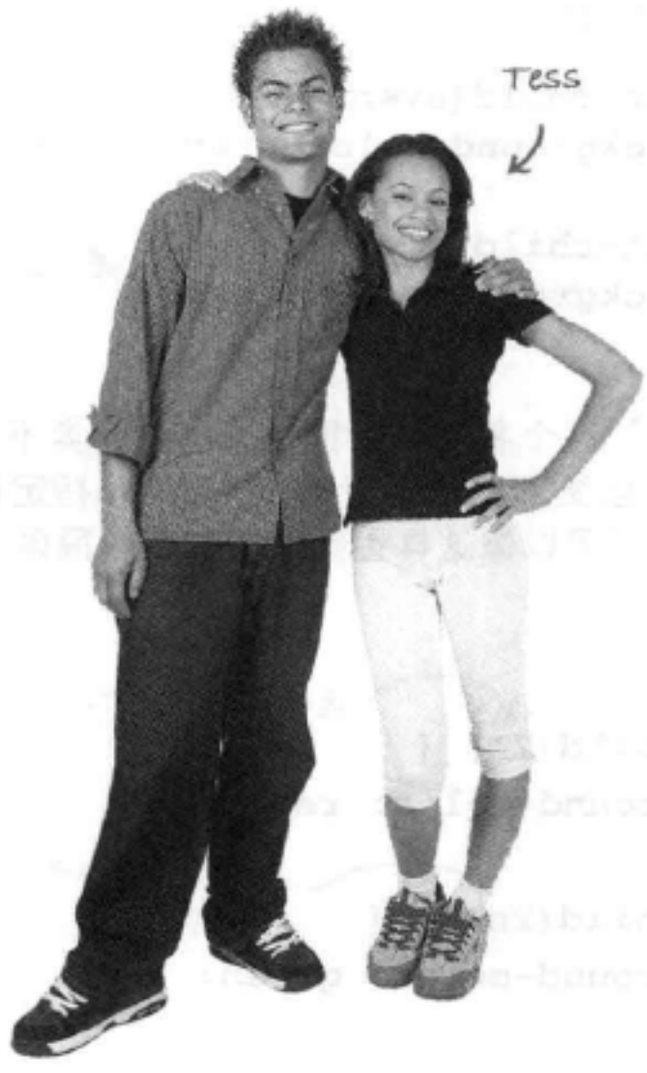


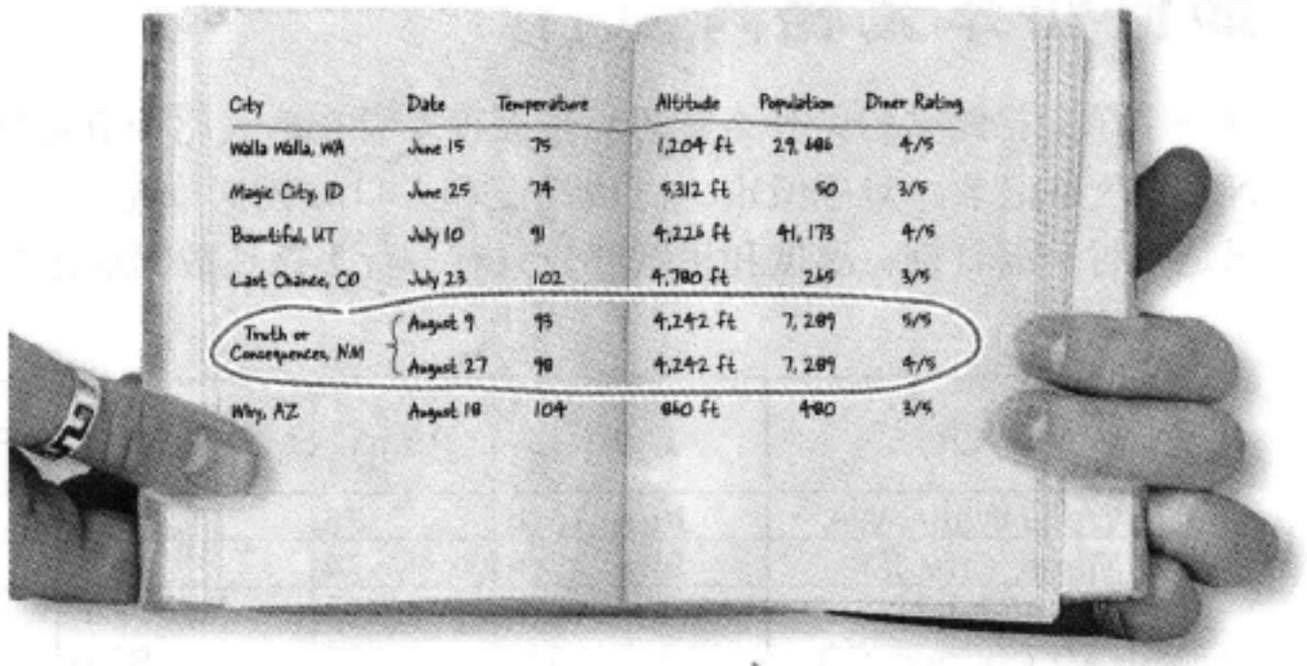
1、3、5和7行都有淡橙色背景。不过th的规则会覆盖“奇数”行的规则，所以表头仍然是深橙色。

我们有没有提到？Tony在新墨西哥的Truth or Consequences有一个有趣的发现？

准确地说，Tony在新墨西哥的Truth or Consequences找到了有趣的东西。实际上，他发现Tess真是很有意思，所以前往亚利桑那州之后又回到了新墨西哥。

我们很喜欢Tony，不过他真是给我们的表格带来一个难题。我们可以直接增加一个新表行对应Truth or Consequences，不过我们还想更高雅一些。这是什么意思？请翻到下一页找出答案。





再来看Tony的表格

由于Tony又返回了新墨西哥，他增加了8月27日的一条记录，就在原来Truth or Consequences那条记录的下面。他还重用了两个信息没有改变的单元格（这是减少表格中信息量的一个很好的技术）。可以看到，增加这个新行时，只需要列出第二次旅行时不同的信息（日期、温度和用餐情况）。

这些都是Tony在Truth or Consequences的旅行记录。

City	Date	Temp	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
	August 27th	98			4/5
Why, AZ	August 18th	104	860 ft	480	3/5

这些表格数据单元格跨两行。

不过，用HTML如何实现呢？看起来必须增加新的一行，只是要重复城市、海拔高度和人口，是吗？嗯，别那么快下结论。我们有办法……通过使用HTML表格，可以实现跨多行（或者跨多列）的单元格。下面来看如何做到……

如何让单元格跨多行

让一个单元格跨多行是什么意思？下面再来看Tony表格中有关Truth or Consequences, NM的两条记录。对应城市、海拔高度和人口的数据单元格都分别跨了两行，而不只是一行，而日期、温度和用餐评分只有一行，这是数据单元格正常的默认行为。

City	Date	Temp	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences NM	August 9th	93	4,242 ft	7,289	5/5
	August 27th	98			4/5
Why, AZ	August 18th	104	860 ft	480	3/5

这些单元格跨两行。

而日期、温度和用餐评分单元格分别只占一行。

那么，在HTML中如何做到呢？这比你想象中要容易，可以使用rowspan属性指定一个表格数据单元格占多少行，然后从这个单元格所跨越的其他行中删除相应的表格数据元素。下面来看一个例子，具体的例子比文字描述更容易理解：

```
<tr>
   <td rowspan="2">Truth or Consequences, NM</td>
   <td class="center">August 9th</td>
   <td class="center">93</td>
   <td rowspan="2" class="right">4,242 ft</td>
   <td rowspan="2" class="right">7,289</td>
   <td class="center">5/5</td>
</tr>
<tr>

   <td class="center">August 27th</td>
   <td class="center">98</td>


   <td class="center">4/5</td>
</tr>
```

这里是包含新墨西哥州数据的两个表行。

对于第二次旅行时没有改变的数据单元格（城市、海拔高度和人口），我们增加了一个rowspan属性，指示这些表格数据跨两行。

由于rowspan设置为2，所以不再需要城市。

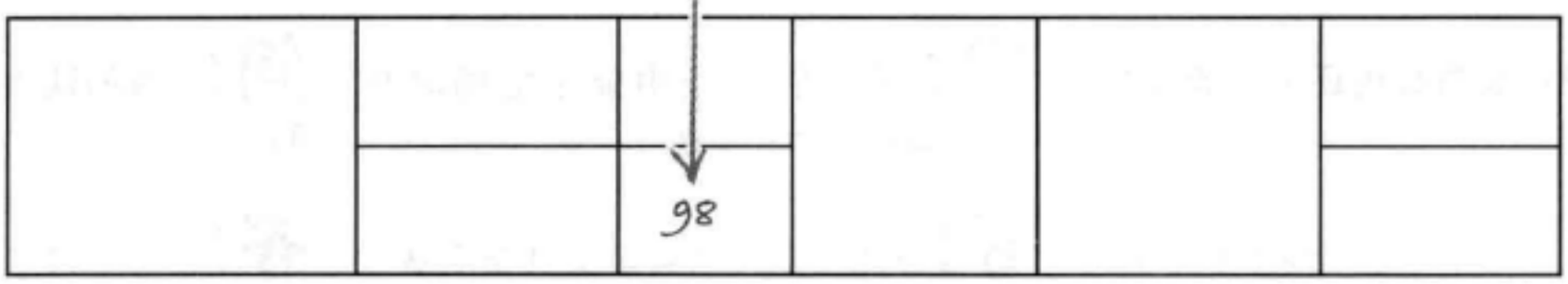
在第二行中，只指定我们需要的列（日期、温度和新的用餐评分）。

海拔高度和人口也一样。

WHO DOES WHAT?

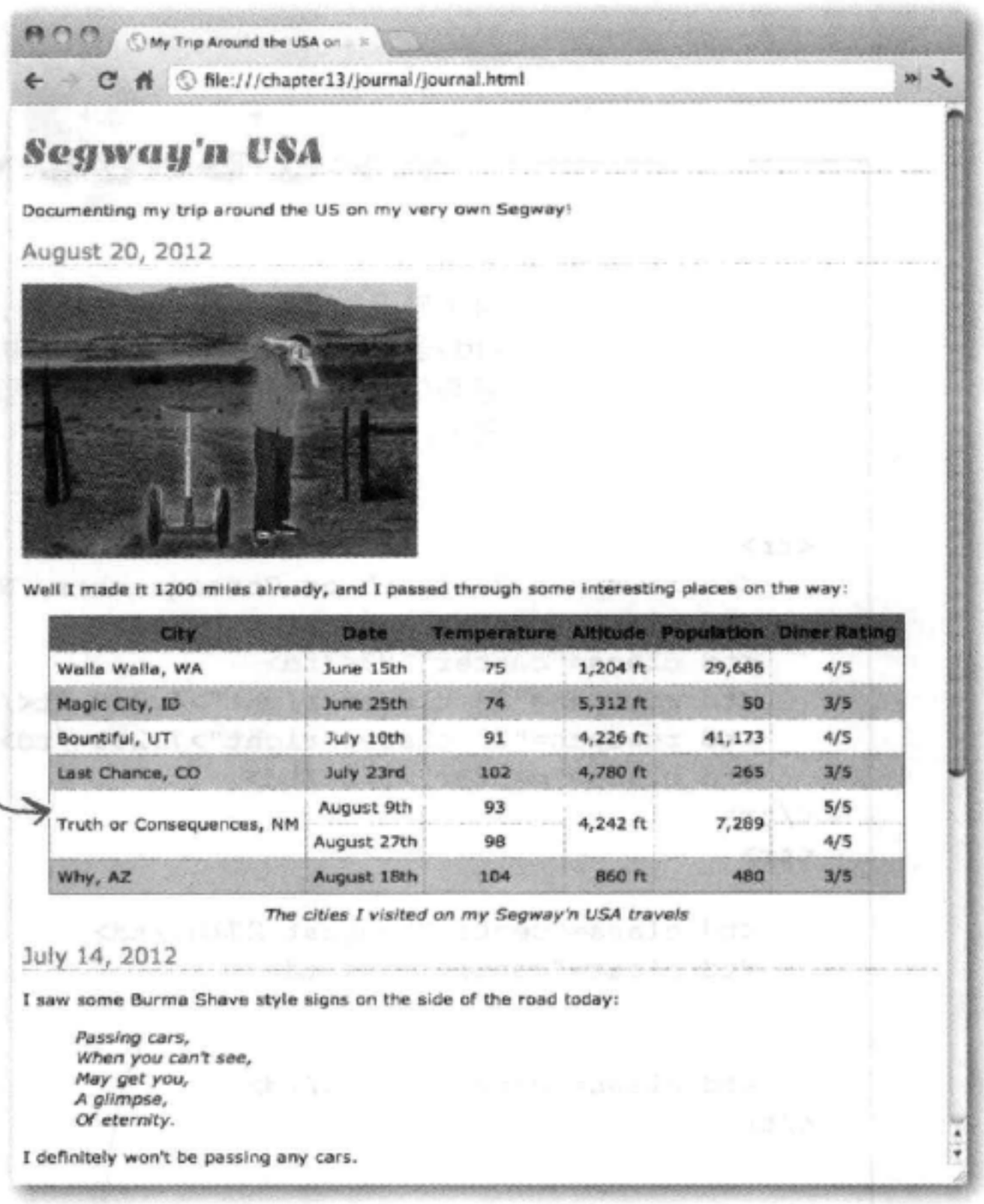
为了确保你真正掌握了这些内容，请在下面找出正确的<td>表格单元格数据，填写表格中的各个单元格。我们已经帮你填了一个空。学习后面的内容之前，请先检查你的答案。

```
<tr>
   <td rowspan="2">Truth or Consequences, NM</td>
   <td class="center">August 9th</td>
   <td class="center">93</td>
   <td rowspan="2" class="right">4,242 ft</td>
   <td rowspan="2" class="right">7,289</td>
   <td class="center">5/5</td>
</tr>
<tr>

   <td class="center">August 27th</td>
   <td class="center">98</td>


   <td class="center">4/5</td>
</tr>
```



测试表格

对“journal.html”中的表格完成修改，然后做个测试。看看这个表格。再想想看你对表格究竟做了哪些处理，要使用HTML来指定某些单元格会占多行，为此，还要删除它们所替换的那些<td>。



现在我们得到了一个漂亮的表格，其中不再有冗余的信息，看起来真的很棒！

there are no Dumb Questions

问：你说过表格数据还可以跨多列，是吗？

答：没错。只需要为<td>元素增加一个colspan属性，然后指定列数就可以了。与rowspan不同，跨多列时，需要删除同一行中的表格数据元素（因为现在所占的是多列，而不是多行）。

问：同一个<td>中能不能同时有colspan和rowspan？

答：当然可以。只是需要调整表格中的其他<td>，考虑到同时有跨行和跨列。换句话说，你需要从同一行和同一列上删除相应数目的<td>。

问：你真的认为这些跨行看起来更好吗？

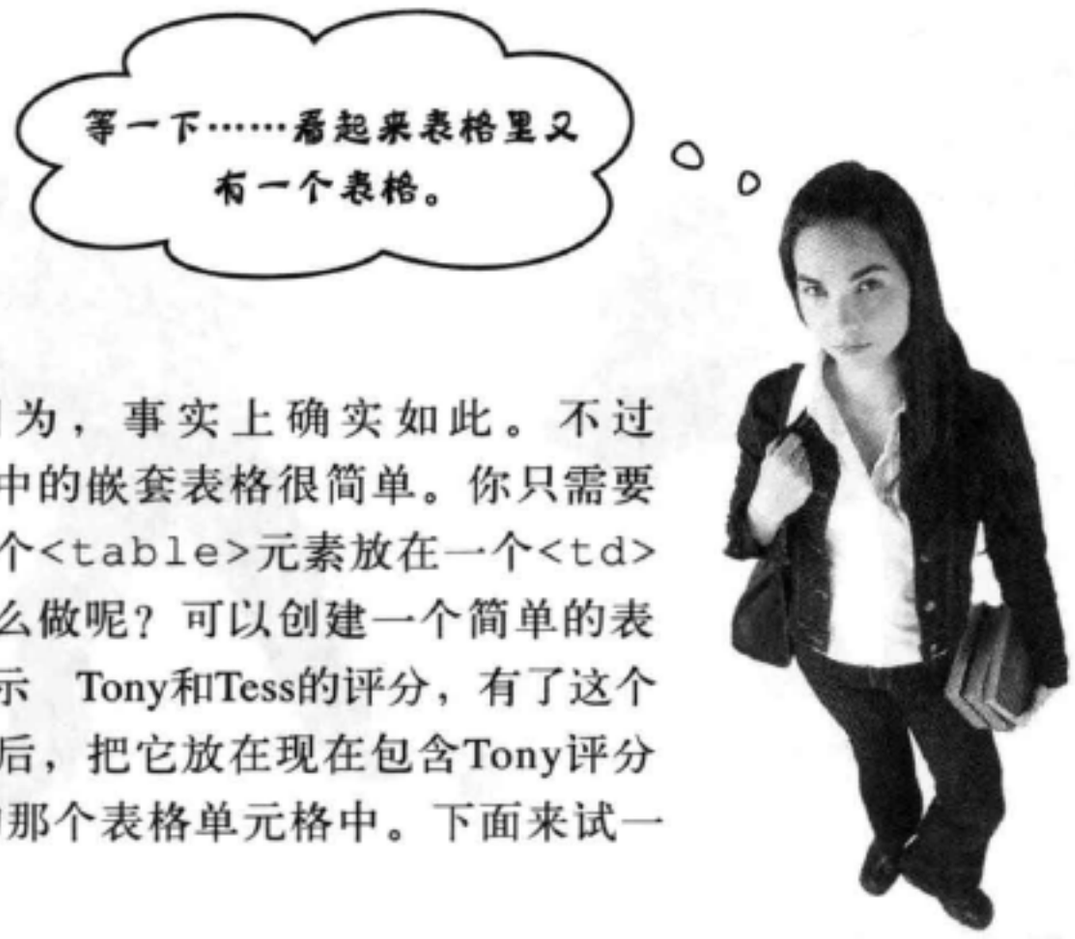
答：嗯，它们可以减少表格中的信息量，这通常是件好事。另外，如果看看现实世界中的表格，你会发现跨行和跨列相当常见，所以能够在HTML中做到跨行和跨列确实很棒。不过，如果你更喜欢以前的表格，当然也可以修改你的HTML，改回为原先的版本。

天堂也有麻烦？

看起来关于8月27日的用餐评分还存在分歧，我们可以让Tony和Tess达成一致，不过为什么要这么做呢？我们有表格，应该可以给出另一个评分。不过怎么做呢？ 我们可不希望只是为了Tess的意见再增加一条记录。嗯……何不像下面这样做？

City	Date	Temp	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
	August 27th	98			Tess 5/5 Tony 4/5
Why, AZ	August 18th	104	860 ft	480	3/5

为什么不把他们俩的评分都放在表格里？这样一来，我们就能得到更准确的信息。



这是因为，事实上确实如此。不过HTML中的嵌套表格很简单。你只需要把另一个<table>元素放在一个<td>中。怎么做呢？可以创建一个简单的表格来表示 Tony和Tess的评分，有了这个表格之后，把它放在现在包含Tony评分为4/5的那个表格单元格中。下面来试一试……

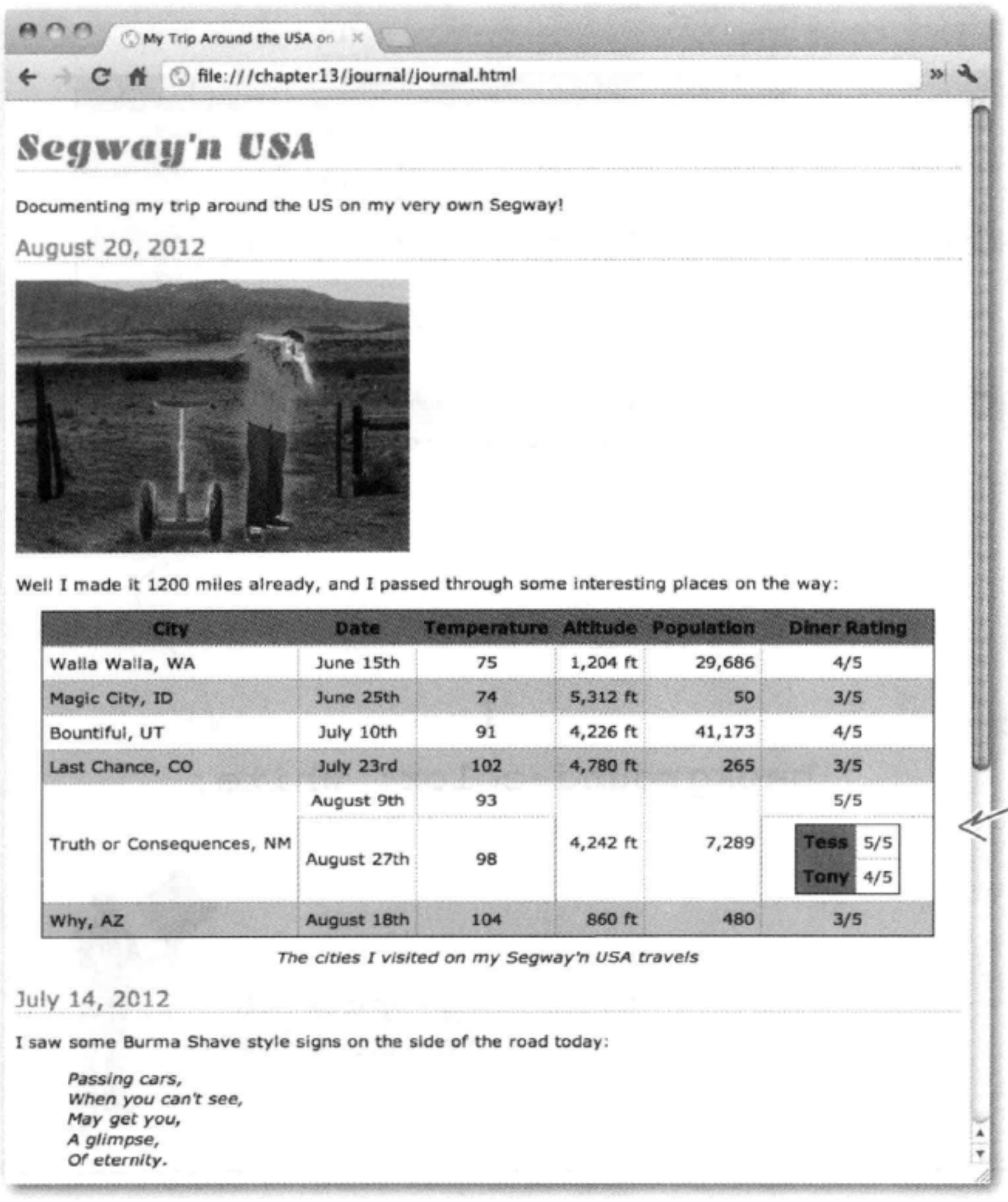
```
<tr>
   <td rowspan="2">Truth or Consequences, NM</td>
   <td class="center">August 9th</td>
   <td class="center">93</td>
   <td rowspan="2" class="right">4,242 ft</td>
   <td rowspan="2" class="right">7,289</td>
   <td class="center">5/5</td>
</tr>
<tr>
   <td class="center">August 27th</td>
   <td class="center">98</td>
   <td>
      ~~4/5~~
      <table>
         <tr>
            <th>Tess</th>
            <td>5/5</td>
         </tr>
         <tr>
            <th>Tony</th>
            <td>4/5</td>
         </tr>
      </table>
   </td>
</tr>
```

首先，删除原来Tony的评分……

……在这个位置上放上一个表格。这个表格包含两个用餐评分：一个是Tess的评分，另一个是Tony的评分。我们使用表格表头表示他们的名字，用数据单元格表示他们的评分。

测试嵌套表格

键入这个新的表格。键入表格时很容易出错，所以一定要进行验证，然后重新加载页面。应该能看到这个新的嵌套表格了。



哇，看起来不错。只是这个背景对于一个嵌套表格来说有点太深了。下面仍然保证名字是粗体，但是去掉它的背景颜色。

BRAIN BARBELL

现在就要利用你之前学到的东西了。你要改变Tony和Tess的表头背景颜色，但是不能改变主表格表头的背景颜色。怎么做呢？需要找到一个选择器，只选择嵌套表格的表头。

City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
	August 27th	98			Tess 5/5 / Tony 4/5
Why, AZ	August 18th	104	860 ft	480	[illegible]

The cities I visited on my Segway'n USA travels

我们想把嵌套表格表头的背景颜色改为白色。

确定选择器，只选择嵌套表格的表头元素。

```
                                    {
                                      background-color: white;
                                    }
```

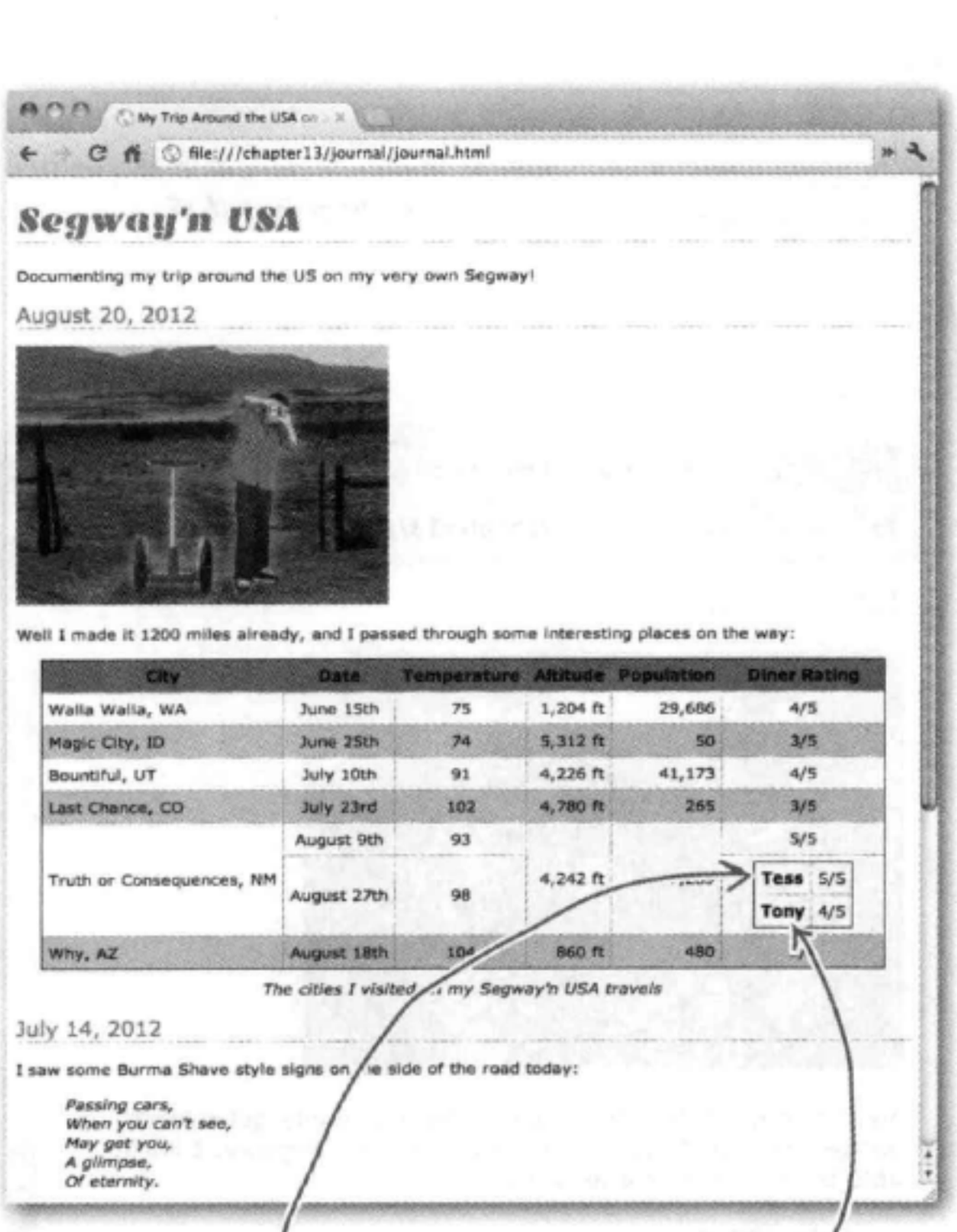
停！完成这个练习之前，先别看下一页！

覆盖嵌套表格表头的CSS

可以使用一个子孙选择器，只选择嵌套表格中的<th>元素。为CSS增加一个新规则，使用“table table th”选择器将嵌套表格表头的背景颜色改为白色：

```
table table th {
    background-color: white;
}
```

现在保存对“journal.css”文件的修改，并重新加载页面。



there are no Dumb Questions

问：我使用了一个类来解决脑力挑战中的那个问题。我创建了一个名为“nestedtable”的类，并把各个表头指定为这个类。然后创建了下面这样一个规则：

```
.nestedtable {
    background-color:  white;
}
```

这个方案也是可行的吧？

答：使用CSS解决问题时会有很多不同的方法，你的方案当然也是一种使用CSS的有效方法，而且是完全合法的。不过需要指出，如果使用子孙选择器，我们就不用对HTML做任何修改。倘若Tony和Tess不断增加用餐评论呢？如果采用你的方案，对于每个评论，就必须为每个<th>增加这个类。而利用我们的方案，样式调整会自动进行。

BRAIN POWER

我们希望Tony和Tess的表行有不同的背景颜色。比如说，分别是蓝色和粉红色。要达到这个目的，你能想出多少种方法？

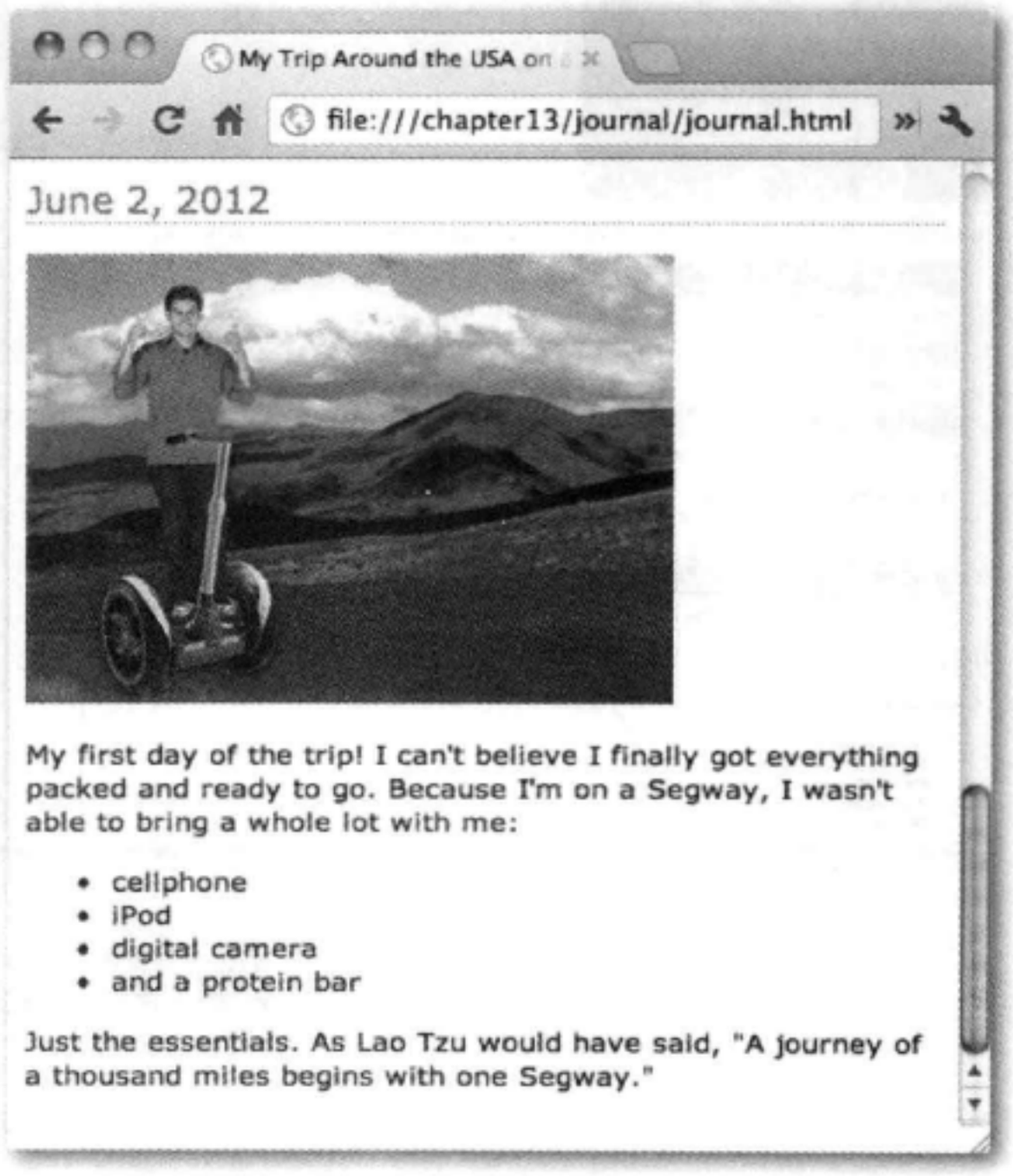
对Tony网站的最后润色

Tony的页面看起来确实很不错，不过还有一个方面可以推敲，我们还没有花时间为他的旅行用品列表指定样式，这个列表中包含了他为旅行准备的各种物品。可以在他6月2日的日志里找到这个列表，下面我们就来看一看：

我们只查看6月2日日志的HTML片段。

```
.
.
.
<h2>June 2, 2012</h2>

<p>
   <img src="images/segway1.jpg"
        alt="The first day of the trip" />
</p>

<p>
   My first day of the trip! I can't
   believe I finally got everything
   packed and ready to go. Because
   I'm on a Segway, I wasn't able
   to bring a whole lot with me:
</p>
<ul>
   <li>cellphone</li>
   <li>iPod</li>
   <li>digital camera</li>
   <li>a protein bar</li>
</ul>
<p>
   Just the essentials. As Lao Tzu
   would have said, <q>A journey of
   a thousand miles begins with
   one Segway.</q>
</p>
</body>
</html>
```

这是Tony的日志（"journal.html"）最下面的部分。还记得他的第一篇日志中给出的物品清单吧？



现在的清单是这样的。

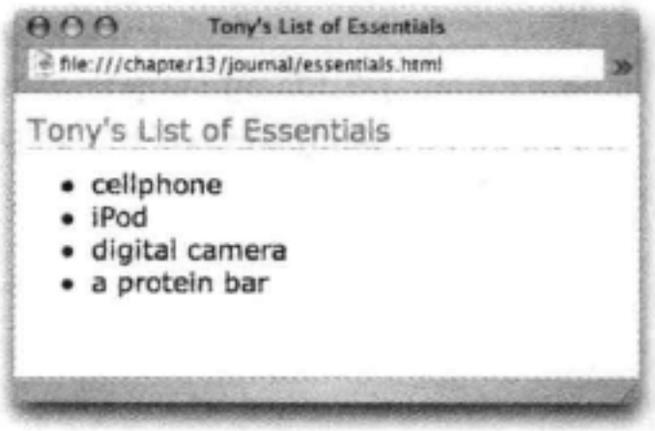
为列表增加一些样式

你已经知道，一旦掌握了基本的CSS字体、文本、颜色和其他属性，就可以对任何元素指定样式，也包括列表。你已经见过一些列表样式（第12章），实际上特定于列表的属性只有几个，所以没有太多的新知识需要学习。列表的主要属性是`list-style-type`，利用这个属性，可以控制列表中使用的项目符号（或者通常也叫做列表标记）。下面给出几种不同的列表标记：

这里要设置<li>元素的样式。也可以对<ul>元素设置这个属性，如果是这样，<ul>列表中的<li>元素就会继承为<ul>元素设置的样式。

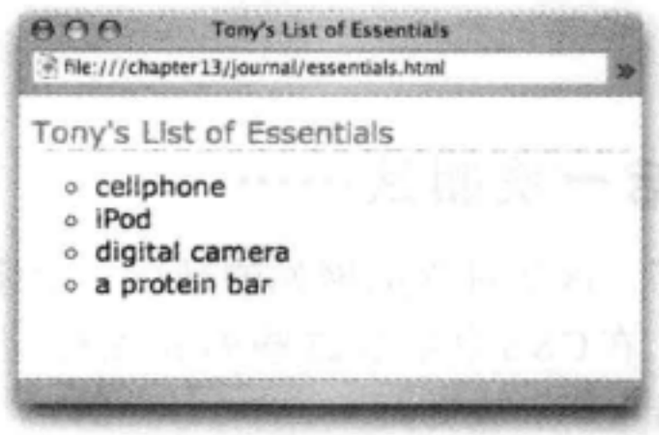
```
li {
    list-style-type: disc;
}
```

disc是默认的列表标记类型。



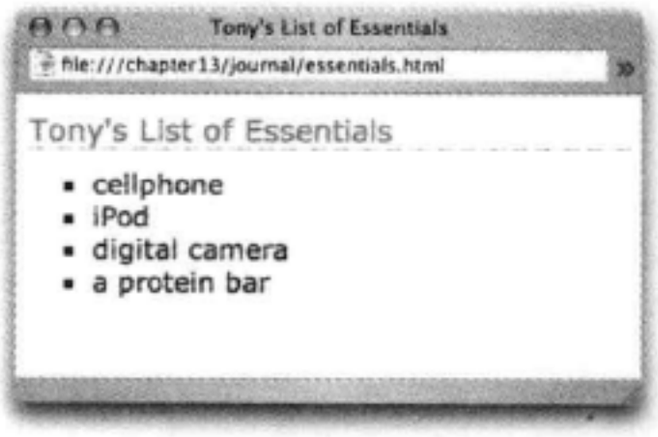

```
li {
    list-style-type: circle;
}
```

circle属性值提供一个简单的圆形标记。



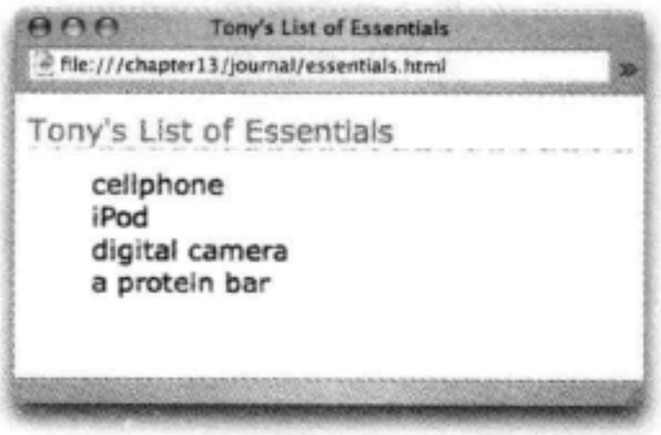

```
li {
    list-style-type: square;
}
```

square属性值提供一个方块标记。




```
li {
    list-style-type: none;
}
```

值为none时，会删除所有列表标记。



如果需要一个定制标记呢？

你是不是觉得Tony肯定想用自己的定制标记？嗯，很幸运，CSS有一个名为`list-style-image`的属性，允许你为列表设置标记图像。下面就在Tony的列表上试一试：

这里是list-style-image属性，我们把它设置为一个URL。

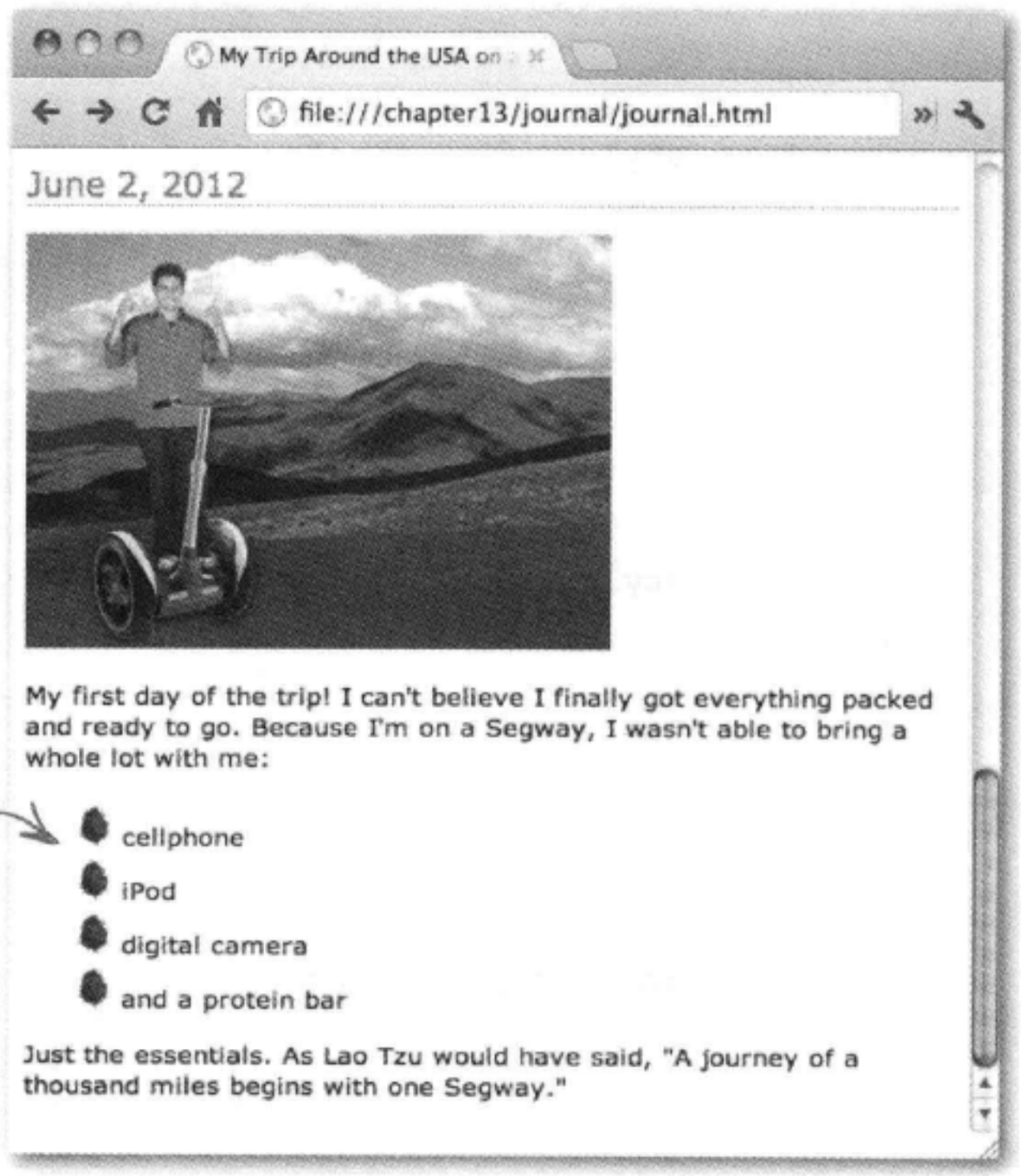
```
li {
    list-style-image: url(images/backpack.gif);
    padding-top: 5px;
    margin-left: 20px;
}
```

图像"backpack.gif"是这个背包图像的一个缩小版，看起来很合适，是不是？而且同样采用了Tony的个性签名颜色。

我们要增加一些外边距，使列表项左边多一些空间，另外还要增加一些上内边距，让每个列表项上面有更多空间。

最后一次测试……

好了，这是对Tony网站的最后一次修改。在CSS中增加这些列表项规则，然后重新加载页面。



来看这里的列表，现在列表标记换成了一个图像，而且增加了一些外边距和内边距。

there are no Dumb Questions

问：有序列表呢？我能改变有序列表的样式吗？

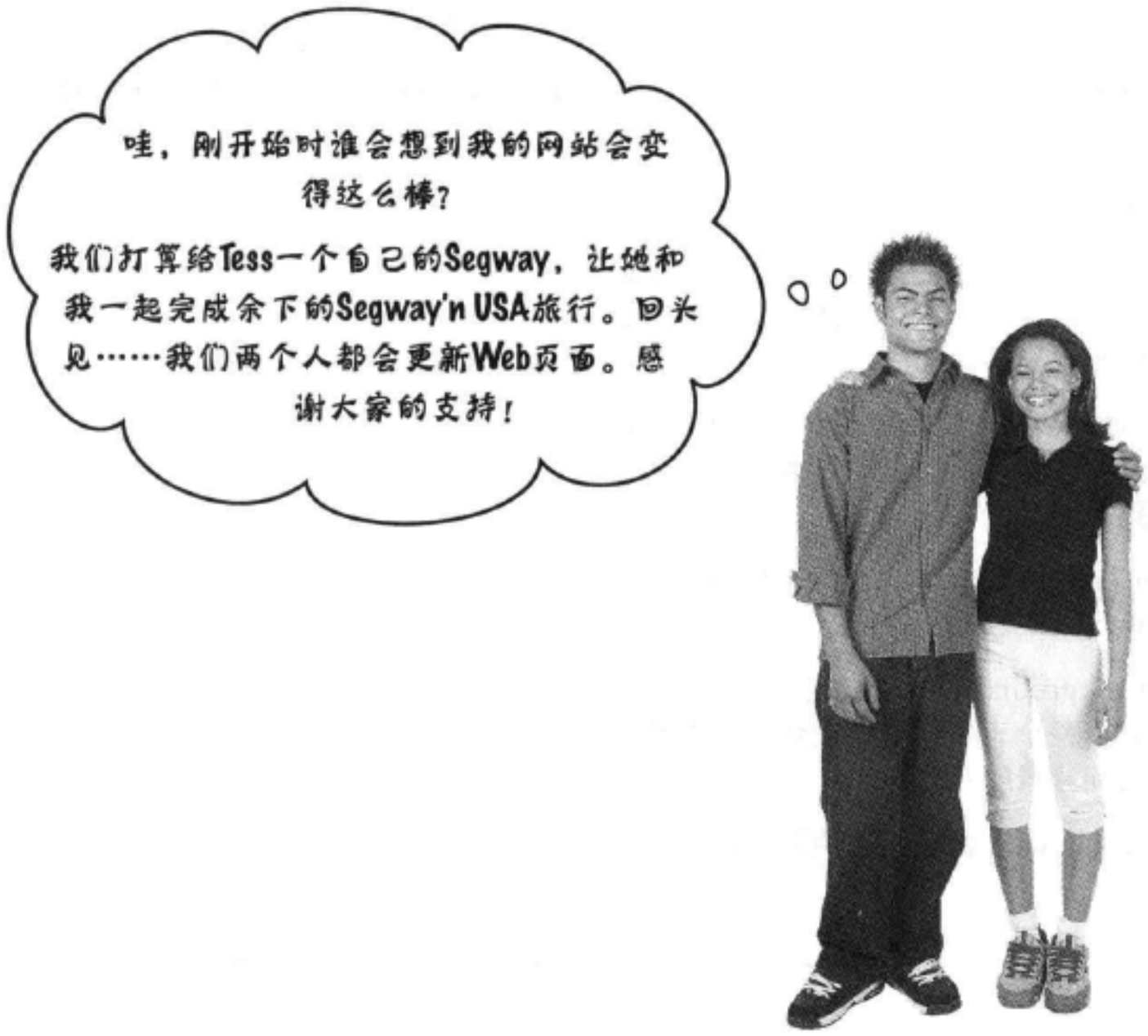
答：不论是有序列表还是无序列表，指定样式的方法都是一样的。当然，有序列表会用一个数字或字母序列作为列表标记，而不是使用项目符号。利用CSS，你可以用list-style-type属性控制一个有序列表的标记是十进制数字、罗马数字还是字母表字符（如a、b、c）。常用的值包括decimal（十进制数）、upper-alpha（大写字母）、lower-alpha（小写字母）、upper-roman（大写罗马数字）和lower-roman（小写罗马数字）。可以找一本CSS参考书来了解更多选择（有很多这样的参考书）。

问：我怎么控制列表上的文本回绕？换句话说，如何控制文本在标记下回绕还是只在文本下回绕？

答：有一个名为list-style-position的属性。如果将这个属性设置为“inside”，文本就会在标记下回绕。如果设置为“outside”，就只在文本下回绕。

问：你确定没错吗？好像反了吧。

答：没错，inside和outside的具体含义是：如果把line-style-position设置为“inside”，标记就在你的列表项里面，所以文本会在它下面回绕。如果这个属性设置为“outside”，说明标记在列表项外面，因此只是文本本身回绕。这里所说的“在列表项里面”是指在列表项盒子的边框里面。

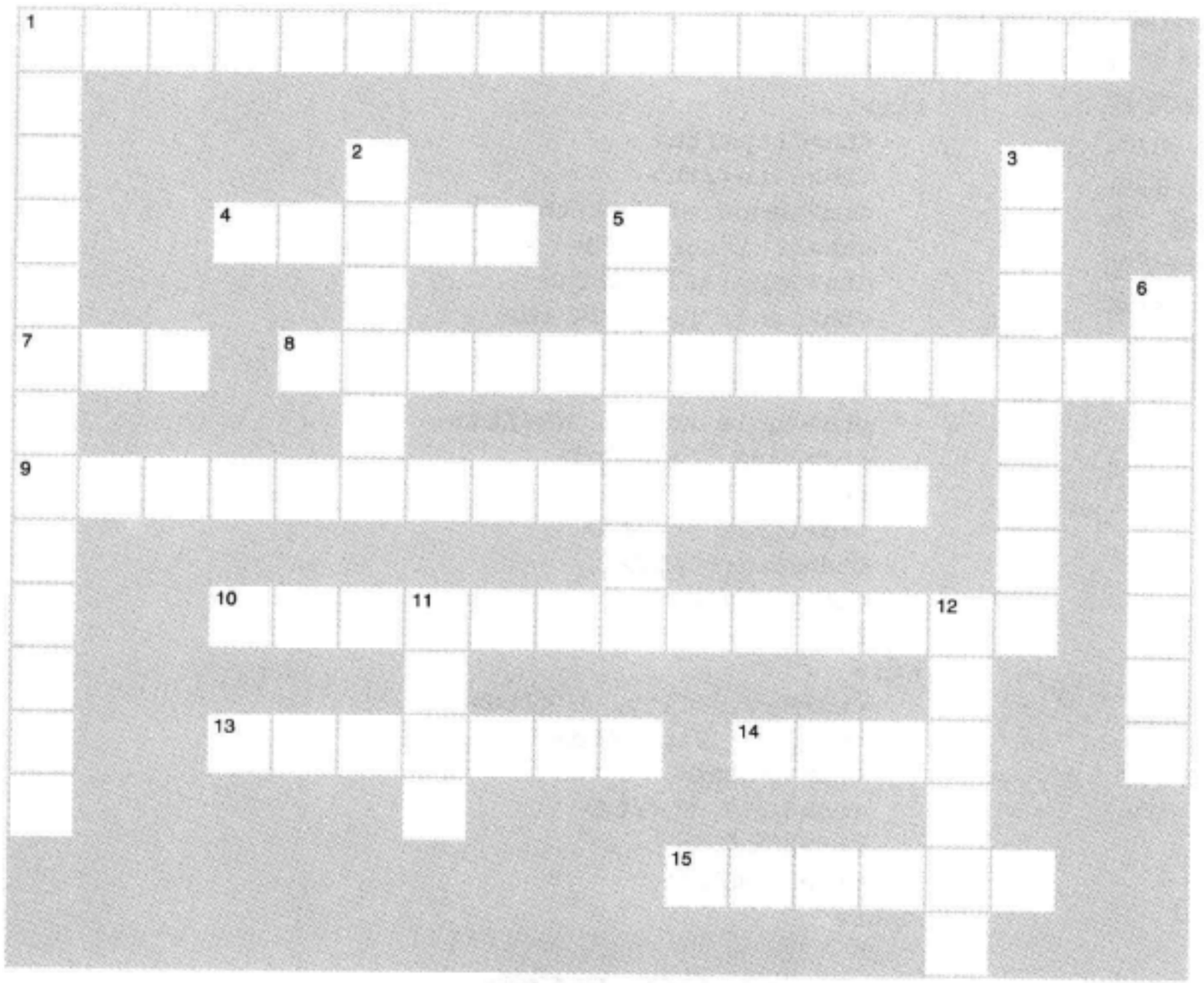


BULLET POINTS

- HTML表格用来建立表格数据结构。
- HTML表格元素<table>、<tr>、<th>和<td>一起用来创建一个表格。
- <table>元素定义并包围整个表格。
- 表格使用<tr>元素按行定义。
- 每行包含一个或多个数据单元格，分别用<td>元素定义。
- 使用<th>元素表示作为行或列表头的数据单元格。
- 表格采用格状布局。每行对应HTML中的一个<tr>……</tr>行，每列对应行中的<td>……</td>内容。
- 可以用<caption>元素提供关于表格的额外信息。
- 表格有边框间距，也就是单元格之间的间距。
- 表格数据单元格还可以有内边距和边框。
- 就像能够控制元素的内边距、边框和外边距一样，可以用CSS控制表格单元格的内边距、边框和边框间距。
- border-collapse是针对表格的一个特殊的CSS属性，允许将单元格边框合并为一个边框，让外观更简洁。
- 可以用text-align和vertical-align CSS属性改变表格单元格中数据的对齐方式。
- 可以用background-color属性为表格增加颜色。可以为整个表格、各行或者单个的数据单元格增加背景颜色。
- 使用CSS nth-child伪类可以为表格隔行增加背景颜色。
- 如果一个数据单元格没有数据，<td>元素中不放置任何内容。不过，需要使用一个<td>……</td>元素维持表格的对齐。
- 如果你的数据单元格需要跨多行或多列，可以使用<td>元素的rowspan或colspan属性。
- 可以在表格中嵌套表格，将<table>元素及其所有内容放在一个数据单元格中。
- 表格应当用于表示表格数据，而不是建立页面布局。另一方面，可以像第11章所介绍的，使用CSS表格显示创建多栏页面布局。
- 与所有其他元素一样，可以用CSS指定列表的样式。有几个特定于列表的CSS属性，如list-style-type和list-style-image。
- list-style-type允许改变列表中使用的列表标记类型。
- list-style-image允许指定列表标记图像。

HTML填字游戏

这个填字游戏看上去有点像表格，是不是？让你的左脑开动起来，完成这个填字游戏。所有答案都可以在这一章中找到。



横向

1. 用来控制标记在列表项边框里面还是外面。
4. 一个数据单元格使用多行或多列时所做的。
7. 标题的默认位置。
8. 用来合并边框。
9. 使用这个属性可以在列表中使用一个图像而不是内置的标记。
10. 边框之间的区域。
13. 增加一个简短描述，与表格一起显示。
14. 按 _____指定HTML表格，而不是按列指定。
15. 我们称项目符号为一种列表 _____。

纵向

1. 使用这个属性可以改变列表标记。
2. 不要为这方面使用表格。
3. list-item-position可以用来控制文本_____行为。
5. 表格单元格有内边距和边框，不过没有_____。
6. <th> 用于这些。
11. <td> 用于这个。
12. 一个表格放在另一个表格中称为 _____。

首先，输入上“Testing Tony’s Table” HTML。输入这些代码，尽管很枯燥，不过这会帮助你记住<table>、<tr>、<th>和<td>标记的结构。完成之后，做一个快速测试，然后增加Tony表格中的其余各项。同样要做个测试。

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
   <style type="text/css">
        td, th {border: 1px solid black;}
   </style>
   <title>Testing Tony's Table</title>
</head>
<body>
   <table>
      <tr>
         <th>City</th>
         <th>Date</th>
         <th>Temperature</th>
         <th>Altitude</th>
         <th>Population</th>
         <th>Diner Rating</th>
      </tr>
      <tr>
         <td>Walla Walla, WA</td>
         <td>June 15th</td>
         <td>75</td>
         <td>1,204 ft</td>
         <td>29,686</td>
         <td>4/5</td>
      </tr>
      <tr>
         <td>Magic City, ID</td>
         <td>June 25th</td>
         <td>74</td>
         <td>5,312 ft</td>
         <td>50</td>
         <td>3/5</td>
      </tr>
      <tr>
         <td>Bountiful, UT</td>
         <td>July 10th</td>
         <td>91</td>
         <td>4,226 ft</td>
         <td>41,173</td>
         <td>4/5</td>
      </tr>
      <tr>
         <td>Last Chance, CO</td>
         <td>July 23rd</td>
         <td>102</td>
         <td>4,780 ft</td>
         <td>265</td>
         <td>3/5</td>
      </tr>
```

继续看下一页。


```
        <tr>
            <td>Truth or Consequences, NM</td>
            <td>August 9th</td>
            <td>93</td>
            <td>4,242 ft</td>
            <td>7,289</td>
            <td>5/5</td>
        </tr>
        <tr>
            <td>Why, AZ</td>
            <td>August 18th</td>
            <td>104</td>
            <td>860 ft</td>
            <td>480</td>
            <td>3/5</td>
        </tr>
    </table>
</body>
</html>
```

Testing Tony's Table

file:///chapter13/journal/table.html

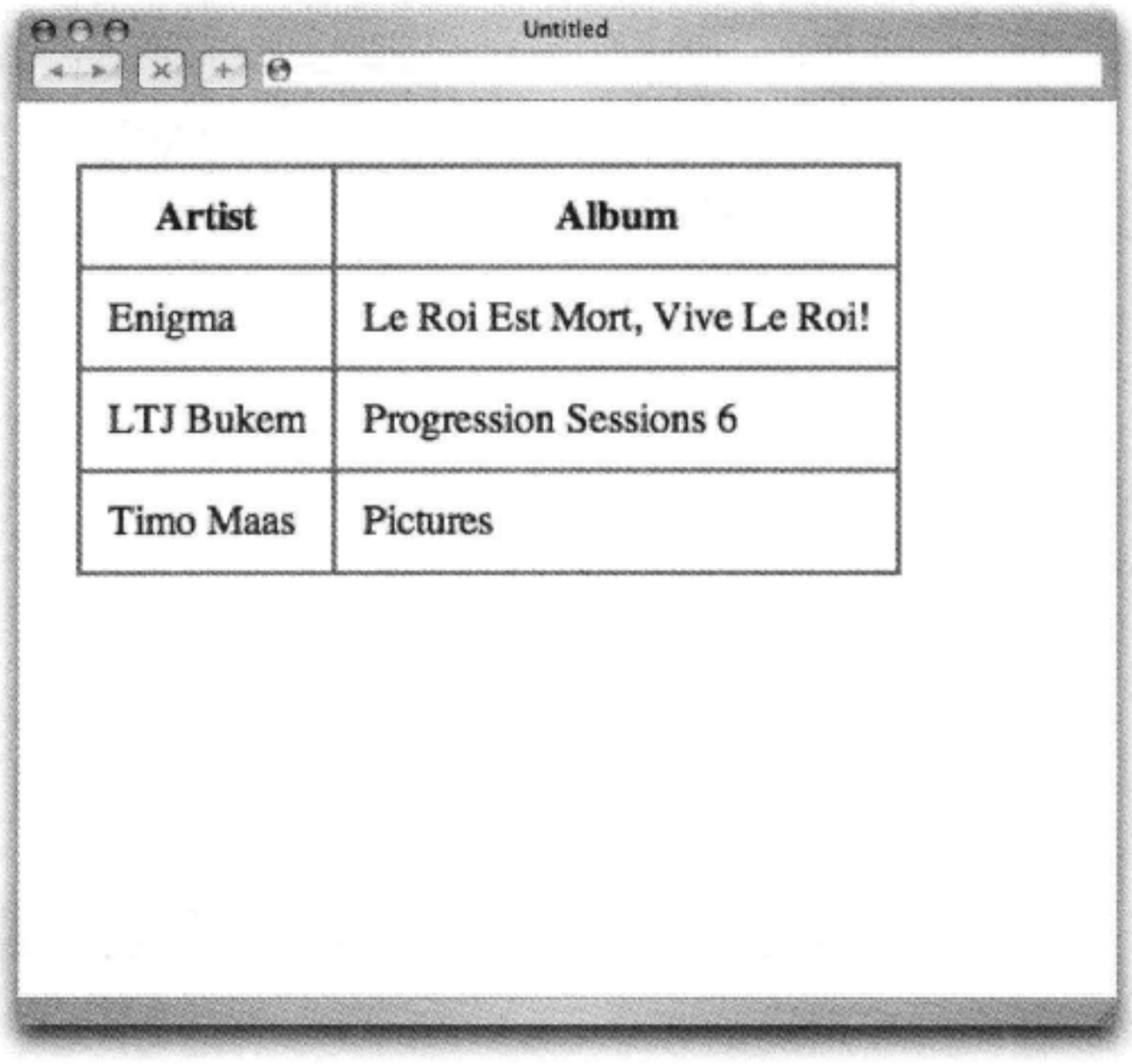
City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
Why, AZ	August 18th	104	860 ft	480	3/5

扮演浏览器答案

在左边可以看到一个表格的HTML。你的任务是扮演显示这个表格的浏览器。这里给出答案。

```
<table>
    <tr>
        <th>Artist</th>
        <th>Album</th>
    </tr>
    <tr>
        <td>Enigma</td>
        <td>Le Roi Est Mort, Vive Le Roi!</td>
    </tr>
    <tr>
        <td>LTJ Bukem</td>
        <td>Progression Sessions 6</td>
    </tr>
    <tr>
        <td>Timo Maas</td>
        <td>Pictures</td>
    </tr>
</table>
```

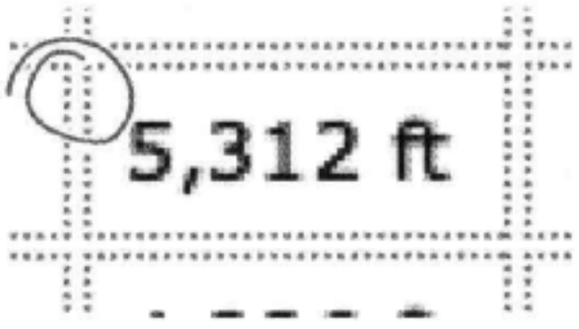
我们调整了这个HTML的格式，这样能让人更容易阅读。



Artist	Album
Enigma	Le Roi Est Mort, Vive Le Roi!
LTJ Bukem	Progression Sessions 6
Timo Maas	Pictures

Sharpen your pencil Solution

双虚线使Tony的表格看上去很乱，很分散注意力。如果每个表格单元格周围只有一个边框，看上去就会好得多。能不能利用你之前学到的知识来指定样式，想办法把双虚线变成单线？可以把border-spacing属性设置为0，去除边框之间的间距。



可以使用border-spacing设置间距为0，这样两条线就会紧挨着。

```
table {
    margin-left: 20px;
    margin-right: 20px;
    border: thin solid black;
    caption-side: bottom;
    border-spacing: 0px;
}
```

好多了，不过还是有两条线，只是它们紧挨在一起，所以我们会得到一个粗的双虚线边框。我们本来希望单元格之间只有一个边框，不是吗？

Sharpen your pencil Solution

假设我们希望日期、温度和用餐评分居中对齐，另外让海拔高度和人口右对齐，怎么做到？可以这样做：

```
.center {
    text-align: center;
}
.right {
    text-align: right;
}
```

这里有两个类，一个对应居中对齐，另一个对应右对齐。

```
<table >
 <caption>The cities I visited on my Segway'n USA travels</caption>
   <tr>
      <th>City</th>
      <th>Date</th>
      <th>Temperature</th>
      <th>Altitude</th>
      <th>Population</th>
      <th>Diner Rating</th>
   </tr>
   <tr>
      <td>Walla Walla, WA</td>
      <td class="center">June 15th</td>
      <td class="center">75</td>
      <td class="right">1,204 ft</td>
      <td class="right">29,686</td>
      <td class="center">4/5</td>
   </tr>
   <tr>
      <td>Magic City, ID</td>
      <td class="center">June 25th</td>
      <td class="center">74</td>
      <td class="right">5,312 ft</td>
      <td class="right">50</td>
      <td class="center">3/5</td>
   </tr>
   .
   .
   .
</table>
```

这里只需要把各个<td>增加到适当的类！

Exercise Solution

要用一个类为Magic City、Last Chance和Why表行创建交替的颜色，可以在表行中为开始<tr>标记增加class=＂cellcolor＂属性，如下所示：

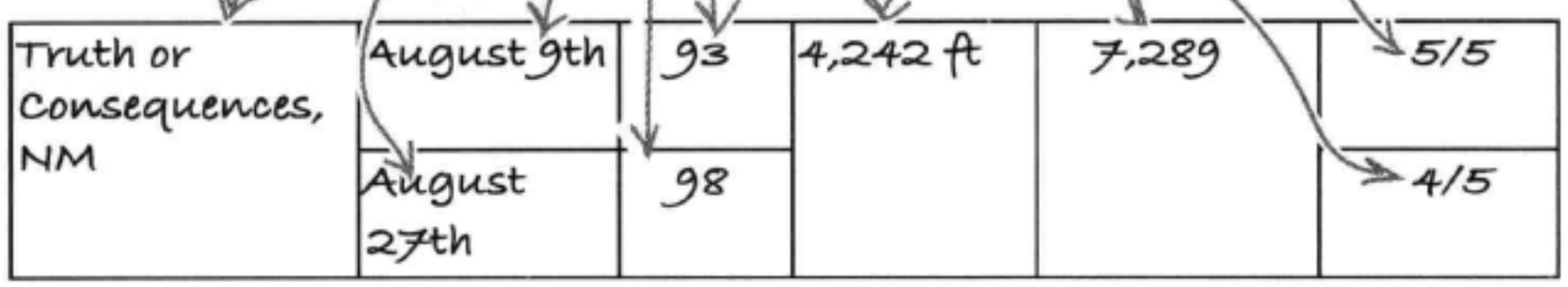
```
<tr class="cellcolor">
  <td>Magic City, ID</td>
  ...
</tr>
```

City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
Why, AZ	August 18th	104	860 ft	480	3/5

WHO DOES WHAT? 答案

为了确保你真正掌握了这些内容，将各个<td>元素与表格中的相应单元格连线。下面给出答案。

```
<tr>
   <td rowspan="2">Truth or Consequences, NM</td>
   <td class="center">August 9th</td>
   <td class="center">93</td>
   <td rowspan="2" class="right">4,242 ft</td>
   <td rowspan="2" class="right">7,289</td>
   <td class="center">5/5</td>
</tr>
<tr>

   <td class="center">August 27th</td>
   <td class="center">98</td>


   <td class="center">4/5</td>
</tr>
```



正式练习答案

要用一个伪类为Magic City、Last Chance和Why表行创建交替的颜色，可以使用nth-child(odd)伪类，选择表格中的奇数<tr>行：

```
tr:nth-child(odd) {
  background-color: #fcba7a;
}
```

City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
Why, AZ	August 18th	104	860 ft	480	3/5

BRAIN BARBELL 答案

现在就要利用你之前学到的东西了。你要改变Tony和Tess的表头背景颜色，但是不能改变主表格表头的背景颜色。怎么做呢？需要找到一个选择器，只选择嵌套表格的表头。

我们使用了一个子孙选择器，只选择嵌套表格表头。可以这样做：

City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
	August 27th	98			Tess 5/5 / Tony 4/5
Why, AZ	August 18th	104	860 ft	480	4/5

(1) 先选择外表……

(2) 再选择内表……

(3) 然后选择表头。

```
(1)    (2)    (3)
table table  th      {
    background-color: white;
}
```

确定选择器只选择嵌套表格的表头元素。

HTML填字游戏答案

[1]L	I	S	T	S	T	Y	L	E	P	O	S	I	T	I	O	N
I																
S					[2]L									[3]W		
T			[4]S	P	A	N	S		[5]M					R		
S					Y				A					A		[6]H
[7]T	O	P		[8]B	O	R	D	E	R	C	O	L	L	A	P	S	E
Y					U				G					P		A
[9]L	I	S	T	S	T	Y	L	E	I	M	A	G	E		I		D
E									N					N		I
T			[10]B	O	R	[11]D	E	R	S	P	A	C	I	[12]N	G		N
Y						A								E			G
P			[13]C	A	P	T	I	O	N		[14]R	O	W	S			S
E						A								T			
										[15]M	A	R	K	E	R		
														D			

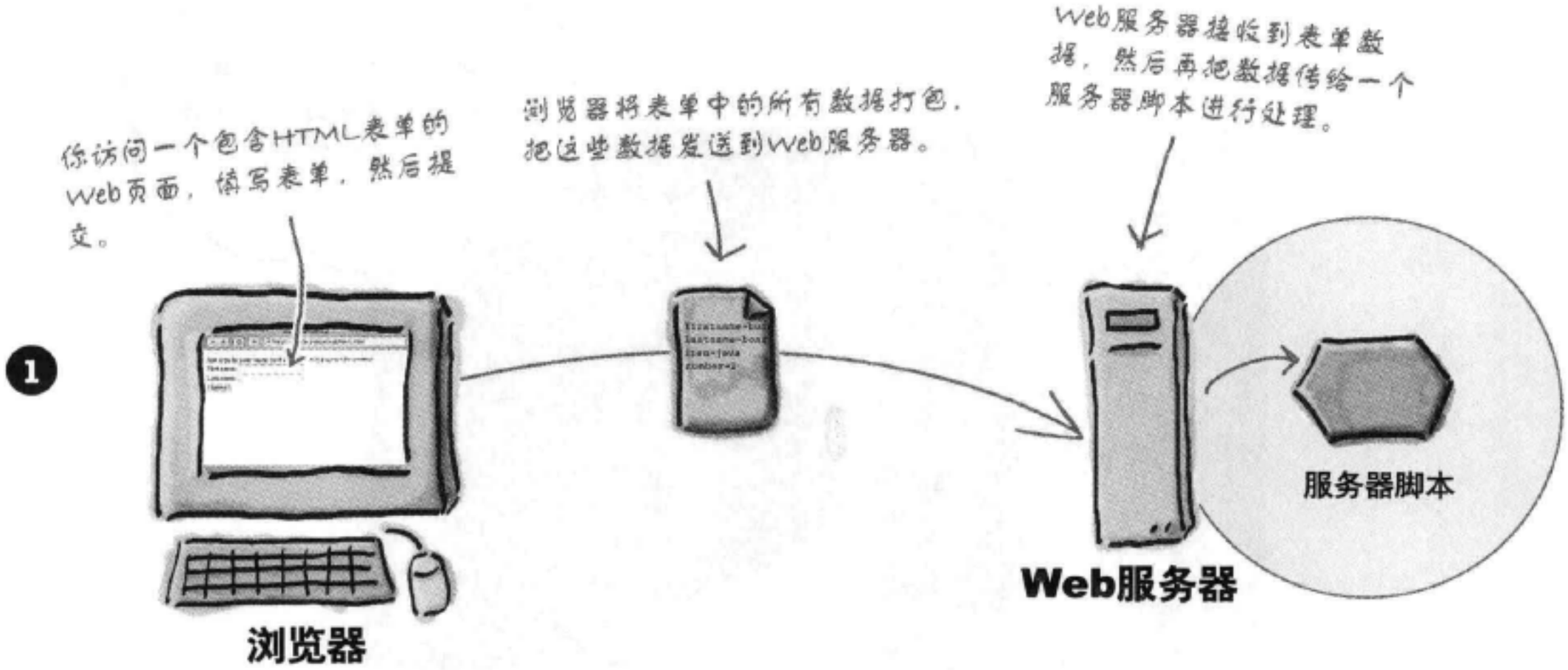
14 HTML表单

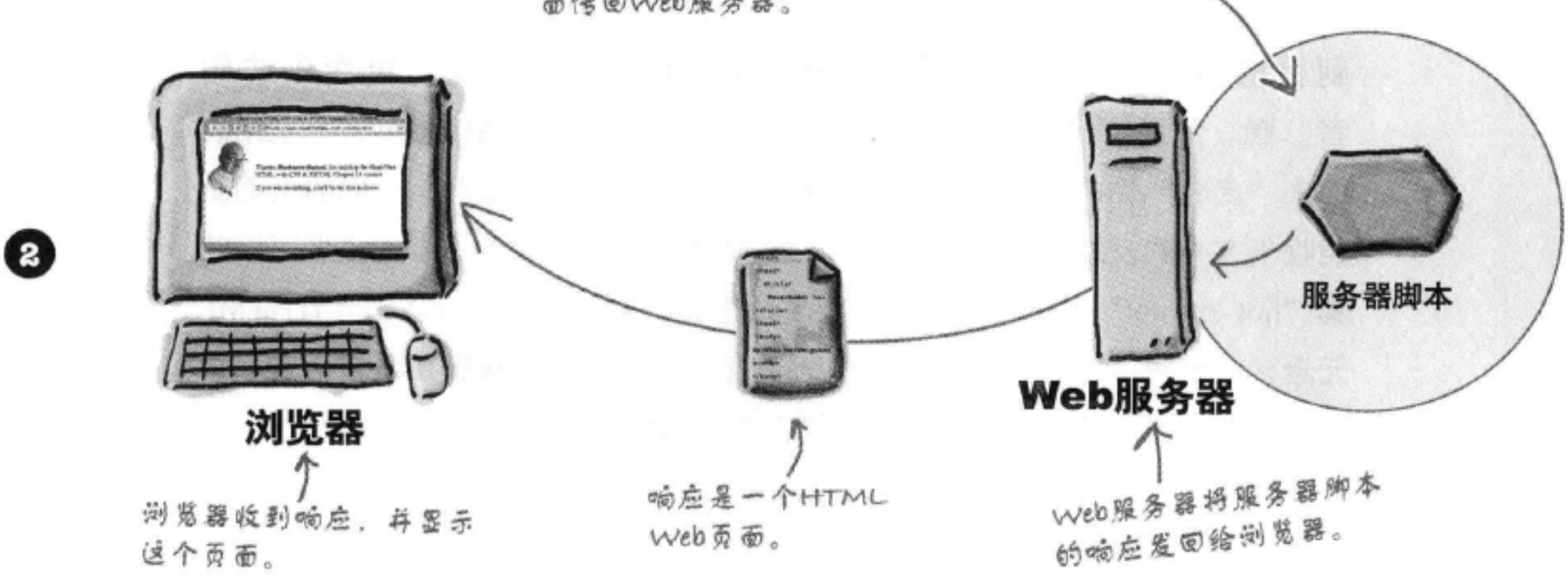
实现交互

到目前为止，你的所有Web通信都是单向的：只是从页面到访问者。嘿，如果访问者能有反馈就好了！这里就要引入HTML表单，一旦用表单来“武装”你的页面（当然还需要Web服务器的一点帮助），你的页面就能收集客户的反馈，得到网上订单，掌握在线游戏中的下一步，或者可以收集“hot or not”（热门与冷门）竞赛的选票。你将在这一章认识大量HTML元素，它们可以一起协作共同创建Web表单。你还会了解服务器在后台如何支持表单。我们甚至还会介绍如何建立表单的样式。

表单如何工作

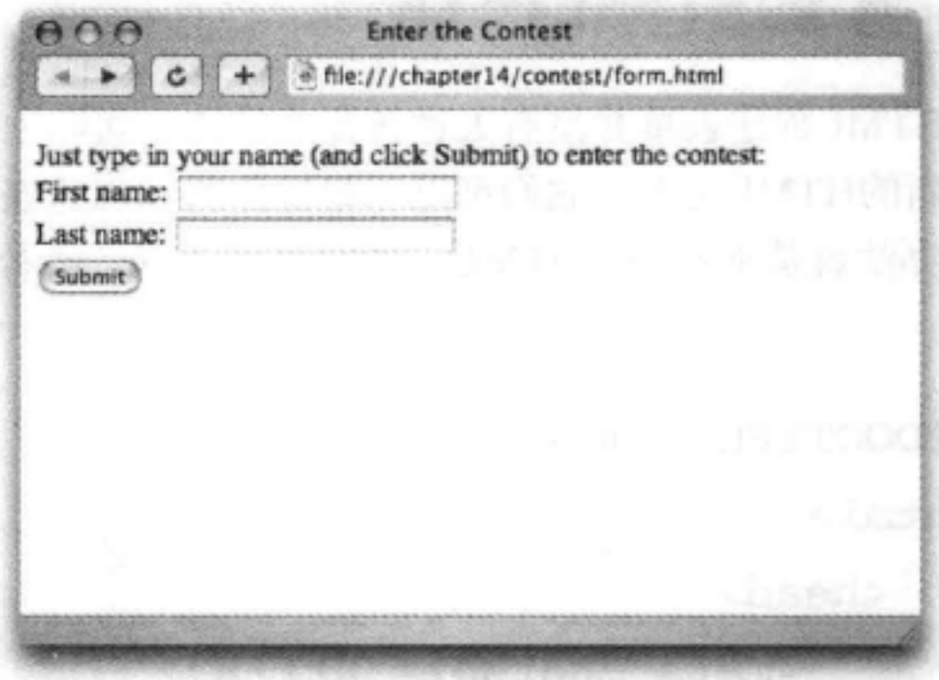
只要你接触过Web，就应该知道表单是什么。不过，你可能没有好好想过表单与HTML到底有什么关系。表单实际上就是一个包含输入域的Web页面，允许你输入信息。提交表单时，这些信息会打包并发送到一个Web服务器，由一个服务器脚本处理。处理完成时，你会得到什么？当然会得到另一个Web页面作为响应。下面来仔细分析这个过程：





表单在浏览器中如何工作

对于浏览器来说，表单只是页面中的一些HTML。你会看到，只需要增加一些新元素，就能很容易地在页面中创建表单。下面从浏览器的角度来看表单如何工作：

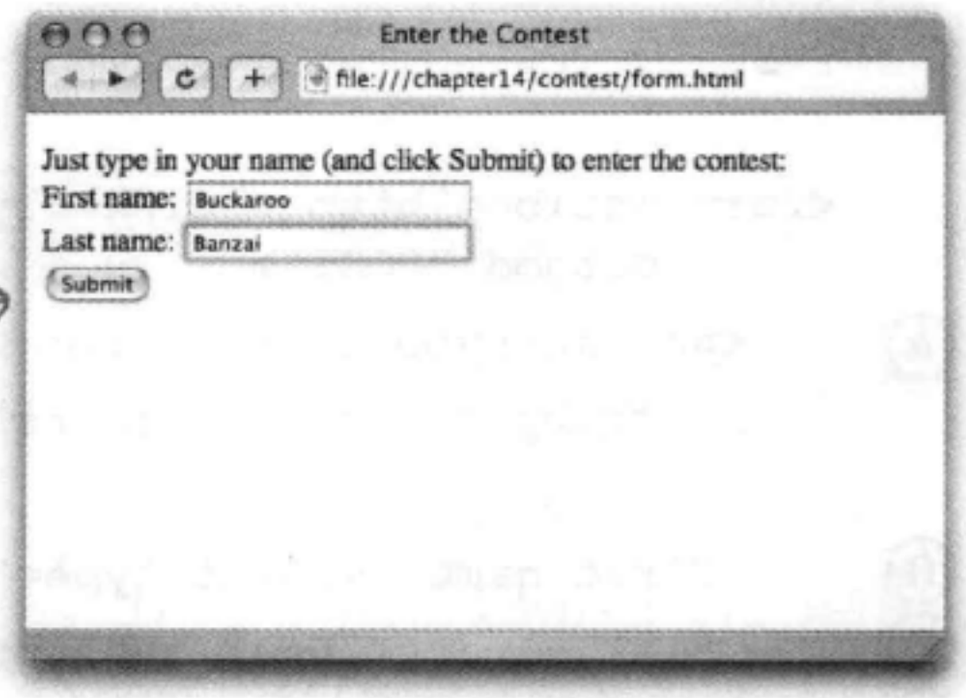


浏览器加载页面

与以往一样，浏览器要加载页面的HTML，遇到表单元素时，它会在页面上创建控件，允许你输入各种各样的数据。控件就是类似按钮、文本输入框或下拉菜单之类的工具，总之，这些控件允许你输入数据。

你输入数据

你使用这些控件输入数据。取决于控件的不同类型，可以有多种不同的输入方式。你可以在文本控件中输入一行文本，或者可以在复选框控件中单击一个选项。稍后我们就会介绍不同类型的控件。

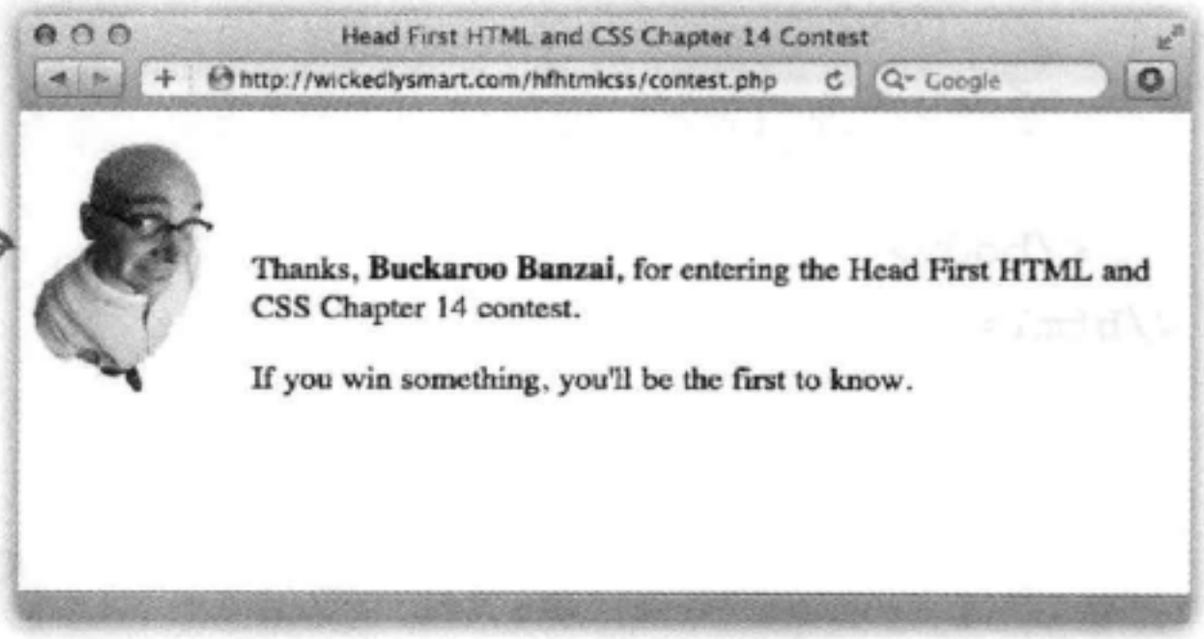


你提交表单

接下来，你单击提交按钮控件，提交这个表单。浏览器看到这个线索时，就会知道它需要打包所有数据，并把这些数据发送到服务器。

服务器响应

一旦服务器得到表单数据，会把这些数据传递给适当的服务器脚本进行处理。这个处理的结果是一个全新的HTML页面，它将返回给浏览器，由于这就是HTML，所以浏览器会为你显示这个页面。



你写的HTML代码

用HTML创建表单并没有太过神秘的地方。实际上，你会在这一章中认识很多新的HTML元素，它们可以一起协作共同创建表单。要想了解表单，最好的办法就是来看一些HTML，然后尝试一下。请看下面这个表单：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Enter the Contest</title>
    </head>
    <body>

        <form action="http://wickedlysmart.com/hfhtmlcss/contest.php"
              method="POST">
Ⓐ         <p>Just type in your name (and click Submit) to
              enter the contest: <br>

Ⓑ          First name: <input type="text" name="firstname" value=""> <br>
Ⓒ          Last name: <input type="text" name="lastname" value=""> <br>
Ⓓ          <input type="submit">
            </p>
        </form>

    </body>
</html>
```

现在这些对你来说已经很熟悉了。

这里是表单。

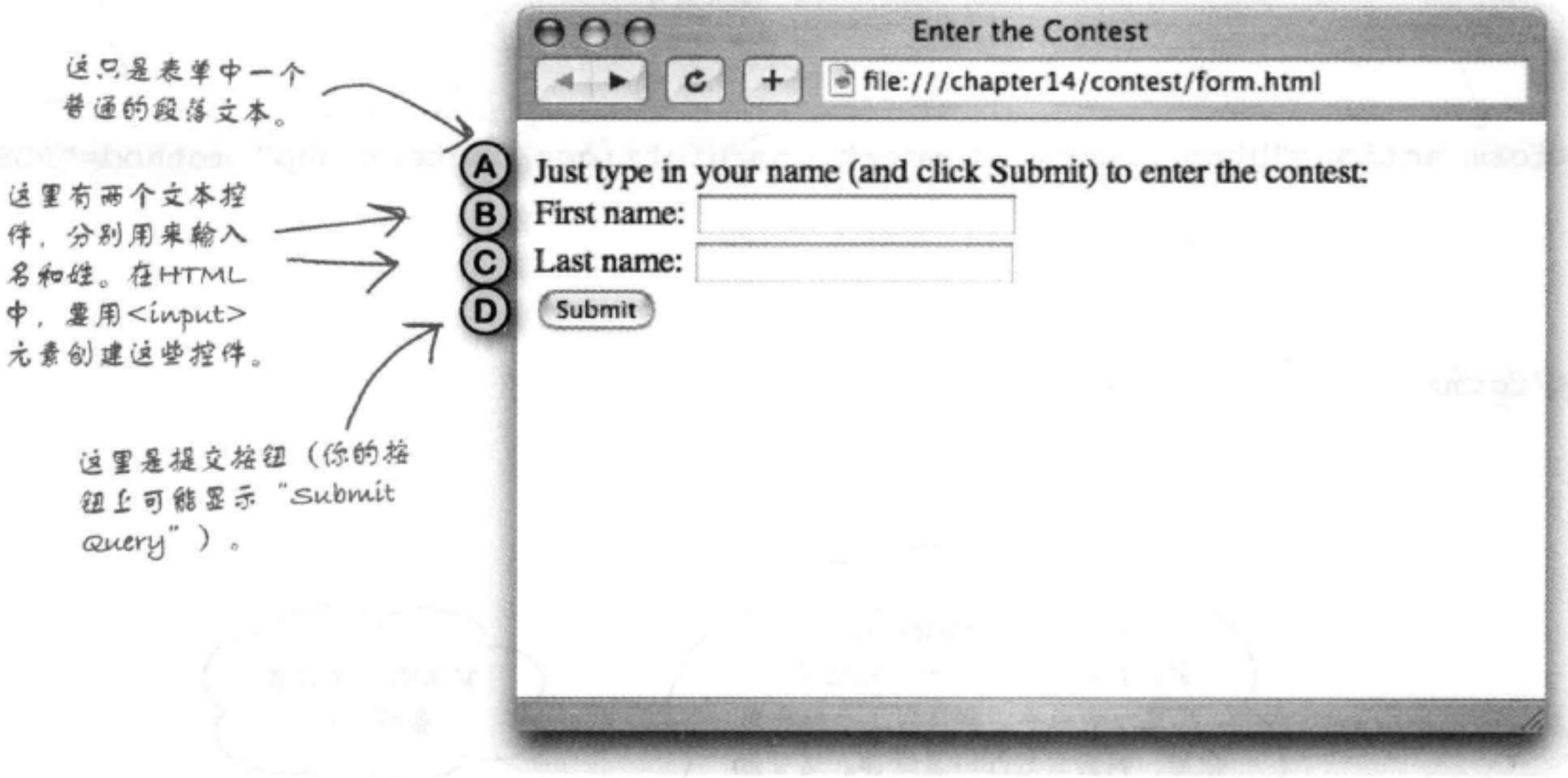
这里是<form>元素本身……

……还有一大堆嵌套在其中的元素。

现在只需要看看这个表单，了解其中有些什么，我们会在这一章一步一步介绍有关的所有细节。

浏览器创建的页面

给你一个惊喜，可以使用<form>元素创建表单。现在，几乎所有块级元素都可以放在<form>元素里，不过你要知道还有一组专门用于表单的新元素。这些表单元素可以提供输入信息的不同方法：文本框、复选框、选项菜单等。我们将介绍所有这些元素，不过先回过头去看看上一页的HTML，了解<form>元素中的元素和内容在页面中如何显示，如下：



在“chapter14/contest”文件夹中可以找到这个“竞赛”表单。打开这个表单，先粗略地看一看，再在浏览器中加载表单来参加竞赛吧。

<form>元素如何工作

下面来仔细分析<form>元素，它不仅包含构成表单的所有元素，还会告诉浏览器当你提交表单时要把表单数据发送到哪里（以及浏览器要用什么方法发送数据）。

这是开始标记。表单中的所有内容都包含在这里。

action属性包含Web服务器的URL……

……脚本所在的文件夹……

……将处理表单数据的服务器脚本的文件名。

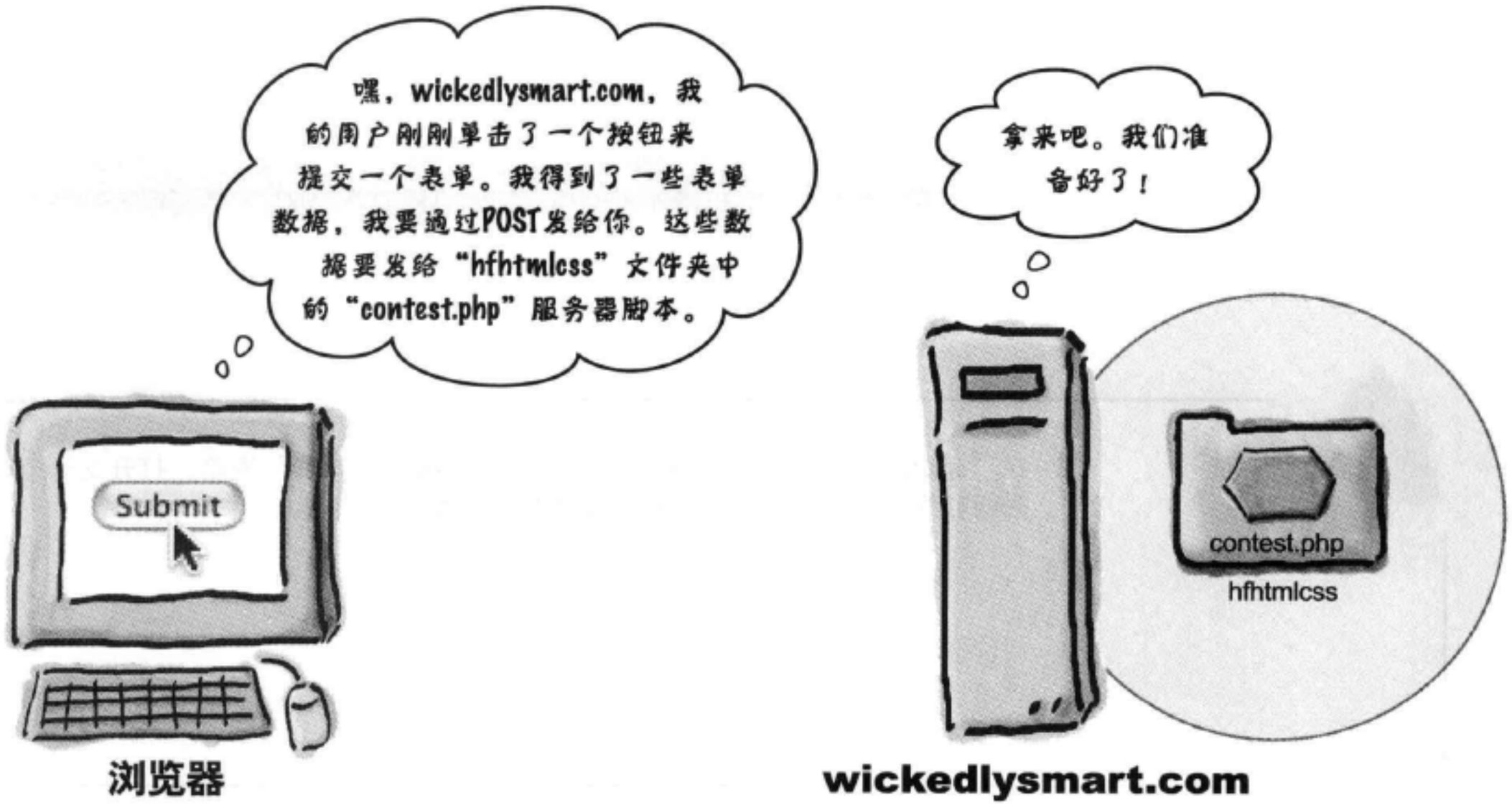
method属性确定表单数据如何发送到服务器。我们将使用最常用的方法：POST。这一章后面还会介绍发送数据的其他方法，另外会解释为什么使用（或不使用）POST。

```
<form action="http://wickedlysmart.com/hfhtmlcss/contest.php" method="POST">
```

表单的所有内容都放在这里……

```
</form>
```

……结束标记结束这个表单。



问得好。

要创建服务器脚本，这本身是一个很大的主题，已经超出了这本书的范畴。嗯，我们本来想加入这些内容的，不过这样一来，这本书可就太重了，可能比你还要重（这可不好）。所以，还是算了吧……

要创建服务器脚本，你要掌握一种脚本或编程语言，而且你的托管公司要支持这种语言。大多数托管公司都支持PHP、Ruby on Rails、Perl、Python、Node.js和Java语言（当然还有很多其他的语言），如果你感兴趣，可以找一本专门介绍创建服务器脚本（也称为服务器端程序）的书。另外可以咨询你的托管公司，他们有时会为客户提供简单的脚本，这样就能减少你的工作，使你不用自己来开发这些脚本。

对于这一章的学习，我们已经开发了你需要的服务器脚本。你要做的只是将这个脚本的URL放在<form>元素的action属性中。

表单里可以有什么？

几乎任何元素都可以放在表单中，不过这不是我们关心的问题，现在我们只对在浏览器中创建控件的表单元素感兴趣。下面简要介绍各种常用的表单元素。先从<input>表单元素开始，它在表单世界里扮演着很多角色。

文本输入

text <input>元素用于输入一行文本。它还有一些可选的属性，允许你为这个控件设置最大字符个数和宽度。

Name: Buckaroo Banzai

type属性为“text”的<input>元素会在浏览器页面中创建一个单行控件。

大多数表单元素都需要一个名字，服务器脚本将使用这个元素名。稍后会看到如何使用。

使用type属性指示你希望得到一个“文本”输入。

```
<input type="text" name="fullname">
```

这个<input>元素是一个void元素，所以后面没有内容。

注意它们都使用相同的HTML元素，只是type属性值不同。

提交输入

submit <input>元素会创建一个按钮，允许你提交表单。点击这个按钮时，浏览器会把表单发送到服务器脚本进行处理。

Submit

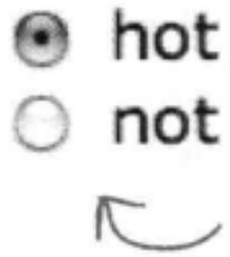
这个按钮的标签默认为“Submit”（提交），或者也可能是“Submit Query”（提交查询），不过你可以改变这个标签（后面会告诉你怎么做）。

```
<input type="submit">
```

提交按钮要指定“submit”作为<input>元素的type属性值。

单选钮输入

radio <input>元素会创建包含多个按钮的控件，但是一次只能选择其中一个按钮。这就像老式的汽车电台按钮，“按下”一个按钮，其余的按钮就会“弹起”。



单选钮只允许从一组选项中选择其一。

每个选项使用一个radio <input>。

与一组给定选项关联的单选钮必须有相同的名字……

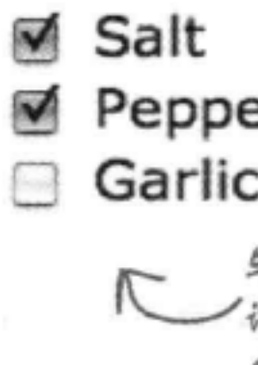
……不过每个选项可以有不同的值。

```
<input type="radio" name="hotornot" value="hot">
<input type="radio" name="hotornot" value="not">
```

这里也是一样，我们仍使用<input>元素，只是有不同的type值。

复选框输入

checkbox <input>元素会创建一个复选框控件，可以选中也可以不选中。多个复选框可以放在一起，如果是这样，你可以根据需要选中多个选项。



与单选钮不同，复选框允许在一组选项中选择0个或多个选项。

类似于单选钮，对各个选项使用相同的checkbox <input>元素。

相关的复选框也共用一个名字。

每个复选框有一个不同的值。

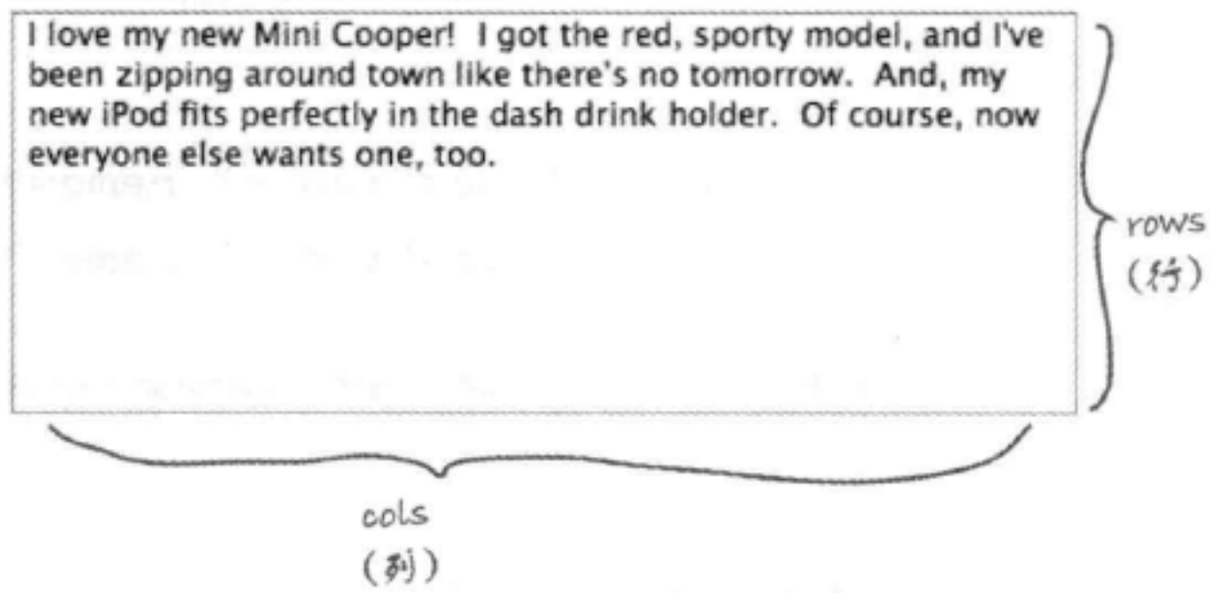
```
<input type="checkbox" name="spice" value="Salt">
<input type="checkbox" name="spice" value="Pepper">
<input type="checkbox" name="spice" value="Garlic">
```

表单里可以有什么（第2部分）

嗯，没错，并不是所有表单元素都是<input>元素。还有很多其他的元素，比如用于菜单的<select>，用于输入多行文本的<textarea>。所以，在继续学习后面的内容之前，何不先来熟悉这些元素？噢，顺便说一句，一旦认识了这些元素，你就差不多掌握了90%的表单元素（甚至是99%的常用表单元素）。

文本区

<textarea>元素会创建一个多行的文本区，可以在其中输入多行文本。如果输入的文本在文本区中放不下，右边还会出现一个滚动条。



<textarea>元素不是一个空元素，所以它有开始和结束标记。

使用name属性为元素指定一个唯一的名字。

cols属性告诉浏览器文本区宽度为多少个字符。

```
<textarea name="comments" rows="10" cols="48"></textarea>
```

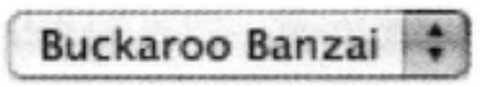
rows属性告诉浏览器文本区高度为多少个字符。

开始和结束标记之间的所有文本会成为浏览器文本区控件中的初始文本。

还可以使用CSS指定文本区的宽度和高度。

select

<select>元素会在Web页面中创建一个菜单控件。菜单提供了一种从一组选项中做出选择的方法。<select>元素与下面的<option>元素结合使用可以创建一个菜单。



select元素会创建一个类似这样的菜单（不过具体的外观可能有变化，这取决于你使用的浏览器）。

<select>元素包围所有菜单选项，把它们组合为一个菜单。

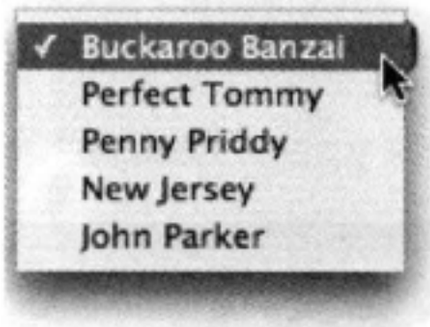
就像其他表单元素一样，要用name属性为select元素指定一个唯一的名字。

```
<select name="characters">
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny">Penny Priddy</option>
    <option value="Jersey">New Jersey</option>
    <option value="John">John Parker</option>
</select>
```

option

<option>元素与<select>元素结合使用可以创建菜单。使用<option>元素来表示各个菜单项。

单击菜单后，会下拉列出菜单项。

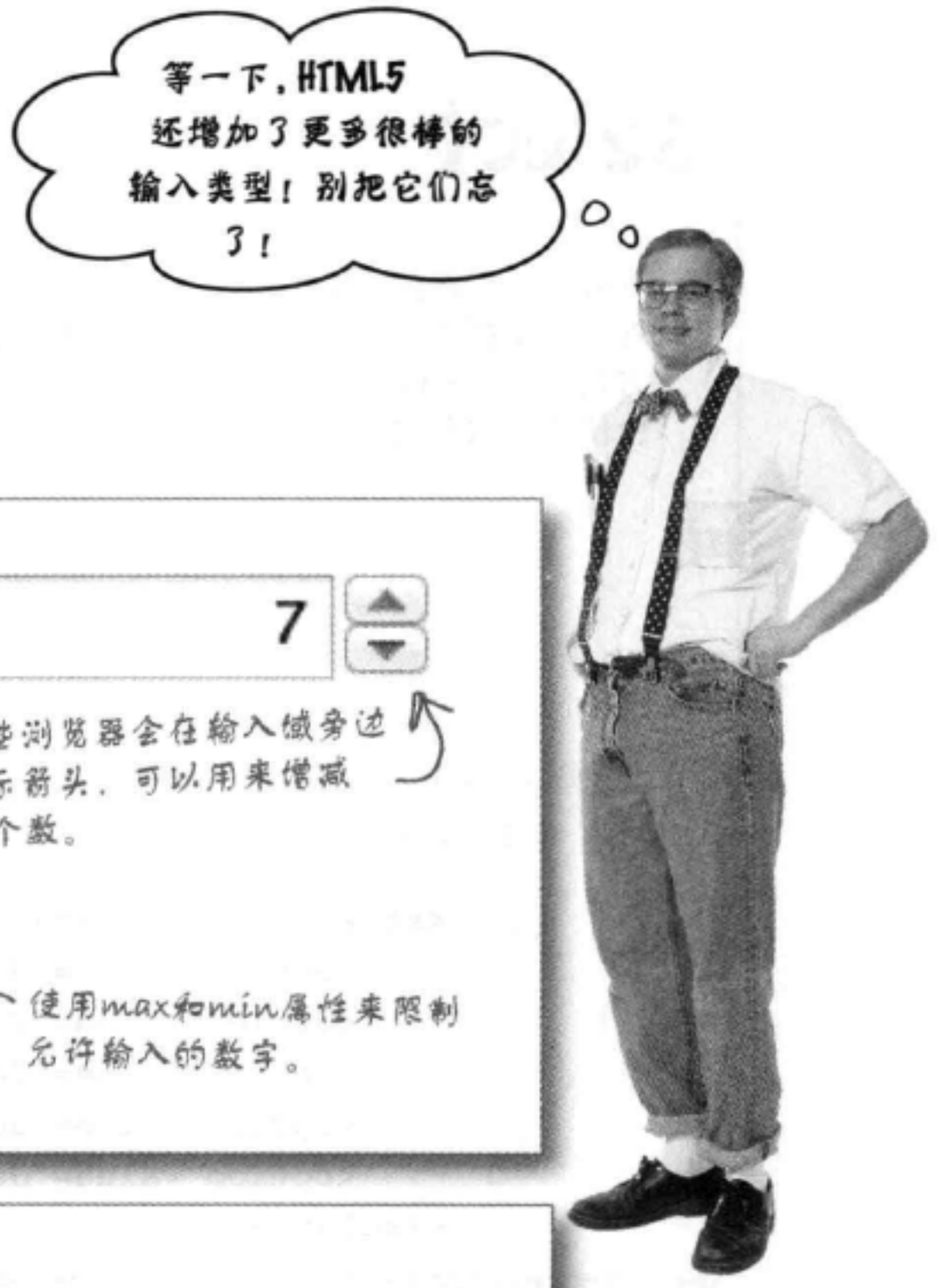



```
<select name="characters">
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny">Penny Priddy</option>
    <option value="Jersey">New Jersey</option>
    <option value="John">John Parker</option>
</select>
```

<option>元素的内容用作为菜单项的描述。每个菜单选项还可以包含一个表示这个菜单项的值。

哇，还有更多元素可以放在表单里！

哈，没错，可不能忘了那些新元素。利用HTML5，我们可以得到更专用的输入表单。下面来看一下：



数字输入

number <input>元素会限制只能输入数字。甚至还可以用可选的属性指定这个元素允许的最小数和最大数。

7

有些浏览器会在输入域旁边显示箭头，可以用来增减这个数。

type为“number”表示你只希望输入数字，而不是文本。

使用max和min属性来限制允许输入的数字。

```
<input type="number" min="0" max="20">
```

范围输入

range <input>元素类似于number，只是它会显示一个滑动条，而不是一个输入框。

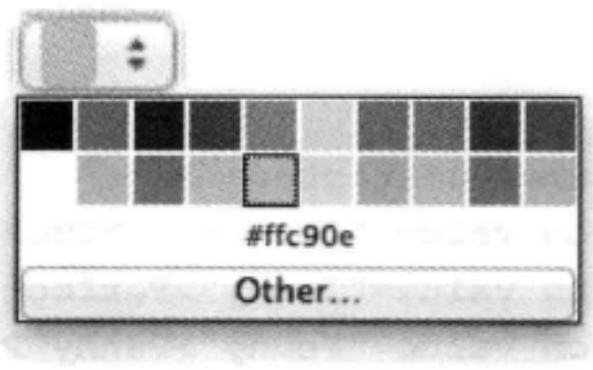
number和range都有一个可选的step属性，可以用来指定值的间隔数（步长）。

```
<input type="range" min="0" max="20" step="5">
```

颜色输入

使用color <input>可以指定一个颜色。单击这个控件时，会弹出一个颜色选择器，允许你选择一个颜色，而不必输入颜色名或值。

如果浏览器不支持颜色输入元素，你会得到一个常规的文本输入控件。




```
<input type="color">
```

日期输入

使用date <input>元素时，可以利用一个日期选择控件指定日期。这个控件会创建一个合法的日期格式串，发送到服务器脚本。


```
<input type="date">
```

与颜色输入元素类似，如果浏览器不支持日期输入元素，你会得到一个常规的文本输入控件。

email输入

email <input>元素就是一个文本输入元素，只不过在一些移动浏览器上，开始输入email时你会得到一个方便输入email的定制键盘。

```
<input type="email">
```

Email: Buckaroo Banzai

tel输入

tel <input>元素也只是一个文本输入元素，不过与email类似，它会在一些移动设备上弹出一个定制键盘。

```
<input type="tel">
```

Phone: 555-1212

url输入

与email和tel类似，url <input>也只是一个文本输入元素，不过会在一些移动设备上弹出一个定制键盘。

```
<input type="url">
```

URL: http://banzai.com

这3种<input>类型都是text <input>的变种。在桌面浏览器上，你看不出任何差别。不过，在移动浏览器上，它们会得到一个定制键盘，可以更容易地输入你需要的字符，如/、@和数字。

即使有这些专用的类型，最后的使用还是取决于你，你要知道服务器脚本期望什么值，并使用适当的<input>类型。

Watch it!

并不是所有浏览器都完全支持这些输入类型。

这两页上的输入类型是HTML5中新增的，可以在所有Web页面中使用这些元素，但是有些浏览器上可能不会像这样显示。

Starbuzz Coffee网站太火爆了。我们现在有一个新概念："Bean Machine"，这是一个在线表单，可以在网上购买我们的咖啡。你能实现这个表单吗？

这是一个咖啡下拉菜单。

选择全豆咖啡还是研磨咖啡（只能选择其一）。

多少包，以及何时到货。

礼品包装，另外是否包括一个商品目录（可以不选择，也可以选择一个或都选）。

送货地址，包括6个文本框。

表单应该是这样的。

Starbuzz Coffee

Choose your beans:

House Blend
Shade Grown Bolivia Supremo
Organic Guatemala
Kenya

Type:

- ○ Whole bean
- ◉ Ground

Number of bags:

Must arrive by date:

Extras:

- ☐ Gift wrap
- ☑ Include catalog with order

Ship to:

Name:

Address:

City:

State:

Zip:

Phone:

Customer Comments:

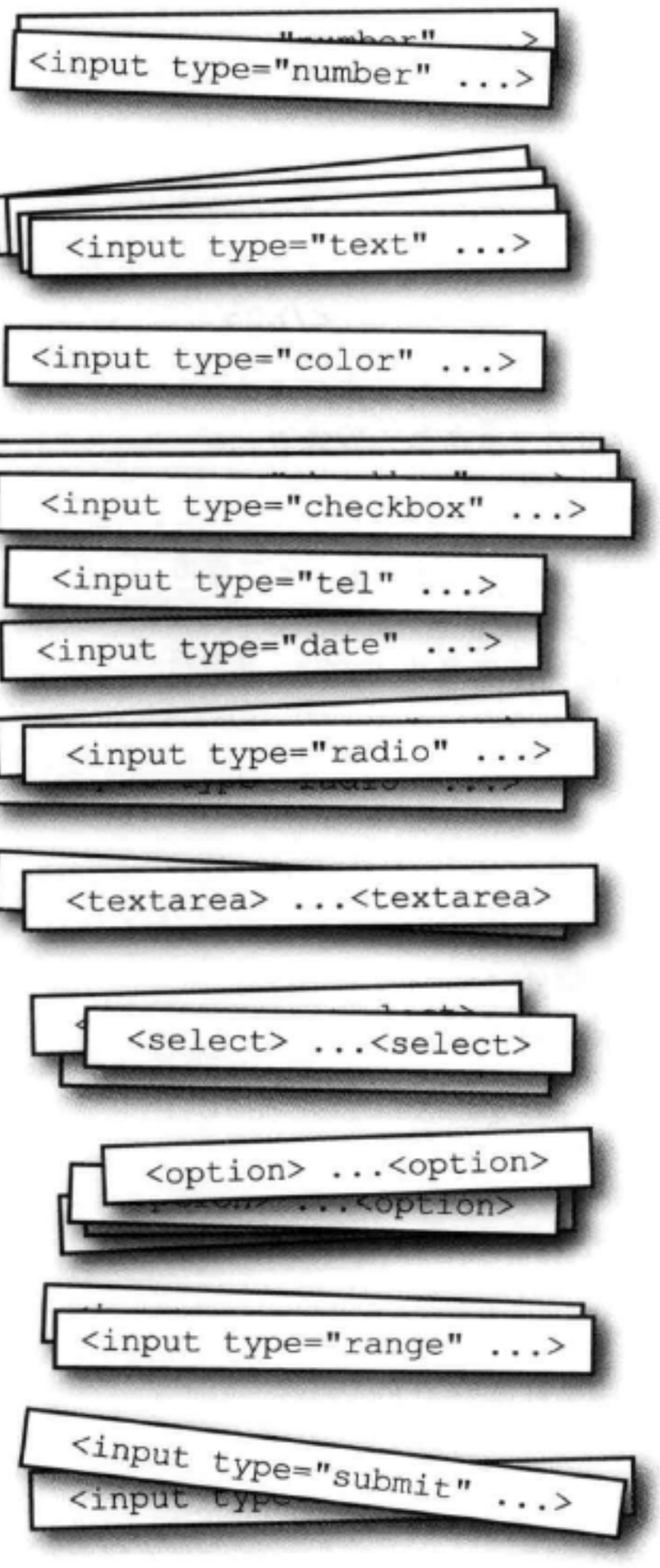
Order Now

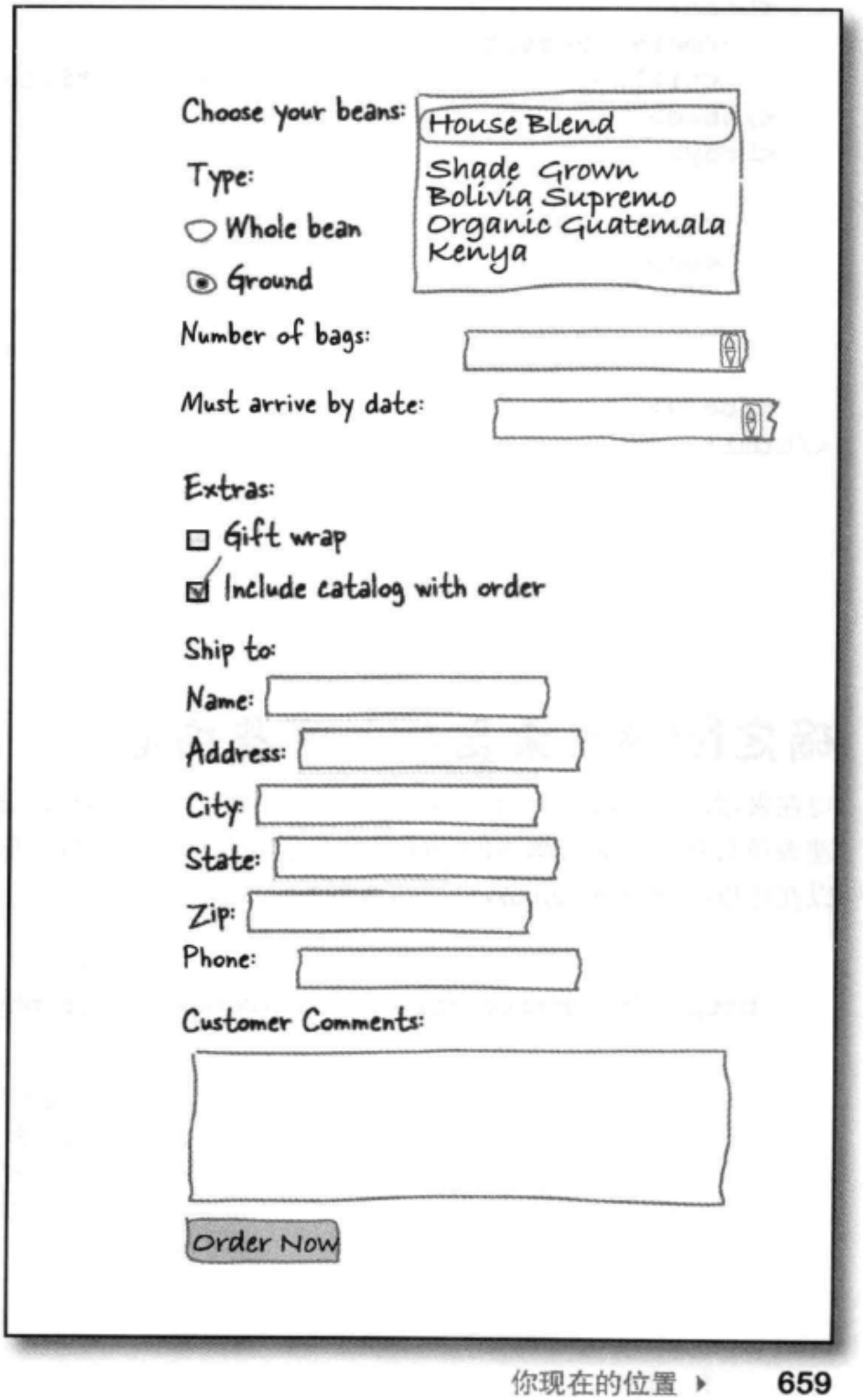
填写客户意见的文本框。

这是一个提交按钮。

标记磁贴

你的任务是把这里的标记元素磁贴放在草图中相应的控件上。完成这个任务时，不一定所有磁贴都会用到；有些可能会剩下。学习后面的内容之前，请对照这一章最后检查你的答案。





准备建立Bean Machine表单

开始构建这个表单之前，先来看一看“chapter14/starbuzz”文件夹中的内容，你会找到文件“form.html”。打开这个文件，简单看一下。这个文件中都是一些基本的HTML标记：

```
<!DOCTYPE html>
<html>
   <head>
      <meta charset="utf-8">
      <title>The Starbuzz Bean Machine</title>
   </head>
   <body>

      <h1>The Starbuzz Bean Machine</h1>
      <h2>Fill out the form below and click "order now" to order</h2>

      <!-- 表单要放在这里。 -->

   </body>
</html>
```

目前为止，我们只有一个标题来标识这个页面，另外还有一些及相关的说明。

对现在来说，我们只构建表单，先不考虑Starbuzz网站中一直使用的样式。这样我们就能集中注意力考虑表单HTML。后面再来增加样式。

确定form元素里加入哪些内容

现在来增加你的第一个<form>元素。创建<form>元素时，首先要知道将处理表单数据的服务器脚本的URL。我们已经帮你写好了这个服务器脚本。可以在这里找到处理Starbuzz订单的服务器脚本：

```
http://starbuzzcoffee.com/processorder.php
```

这个URL指向Starbuzz Coffee网站……

……具体指向服务器上的processorder.php服务器脚本。这个服务器脚本知道如何从我们将要构建的表单得到订单。

增加<form>元素

一旦知道了处理表单的服务器脚本的URL，现在要做的就是把它插入到<form>元素的action属性中，就像这样（照我们说的，在你的HTML输入下面的修改）：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>The Starbuzz Bean Machine</title>
  </head>
  <body>
    <h1>The Starbuzz Bean Machine</h1>
    <h2>Fill out the form below and click "order now" to order</h2>
    <form action="http://starbuzzcoffee.com/processorder.php" method="POST">

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

这里是form元素。

action属性包含服务器脚本的URL。

记住我们要用POST方法向服务器提交表单数据。有关的更多内容将在后面介绍。

还要增加form结束标记。

到目前为止都还好，不过一个空的<form>元素并没有太大意义。再来看前面的表单草图，还有很多内容需要增加，不过我们先从简单的开始，首先完成表单的“Ship to”（送货地址）部分，这部分由一组文本输入和一个数字输入域组成。你对文本输入域已经有所了解，现在再来仔细看一下。Starbuzz表单的文本输入域如下所示：

这里的类型是“text”，因为它将作为一个文本输入控件。

我们使用<input>元素来创建一些不同的控件。type属性确定了这是什么类型的控件。

```
<input type="text" name="name">
<input type="text" name="address">
<input type="text" name="city">
<input type="text" name="state">
<input type="text" name="zip">
<input type="tel" name="phone">
```

对于表单中的每个输入域，分别有一个文本输入控件：Name、Address、City、State、Zip和Phone。

这里类型是“tel”，因为我们希望这里的值是一个电话号码。

name属性相当于用户输入数据的一个标识符。注意每个name属性要设置为一个不同的值。下面来看元素名如何工作……

表单元素名如何工作

关于name属性有一点要知道：它相当于表单和处理表单的服务器脚本之间的一个黏合剂。做法如下：

表单中的每个输入控件都有一个name属性：

在HTML文件中输入表单元素时，会为它们指定唯一的名字。

前面的text和tel输入元素就指定了不同的名字：

```
<input type="text" name="name">
<input type="text" name="address">
<input type="text" name="city">
<input type="text" name="state">
<input type="text" name="zip">
<input type="tel" name="phone">
```

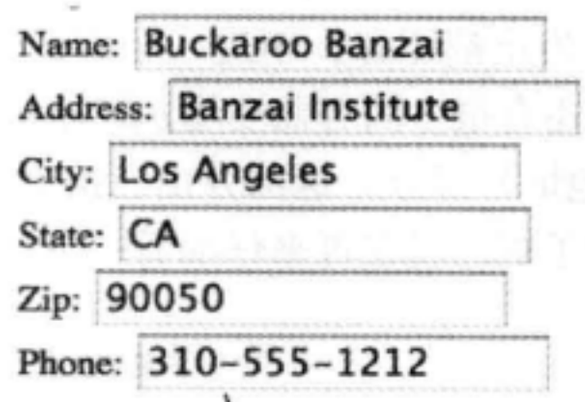
注意，这里有一个元素的名字是"name"（这是完全可以的）。

每个<input>元素都有自己的名字。

提交表单时，浏览器会使用这些唯一的名字打包所有数据：

假设你在表单中输入了姓名（name）、地址（address）、城市（city）、州（state）、邮编（zip）和电话号码（phone），然后单击提交按钮（Submit）。浏览器会得到各部分数据，并用唯一的name属性值作为这些数据的标签。然后浏览器把这些名字和值发送到服务器。如下所示：

表单中输入的值。

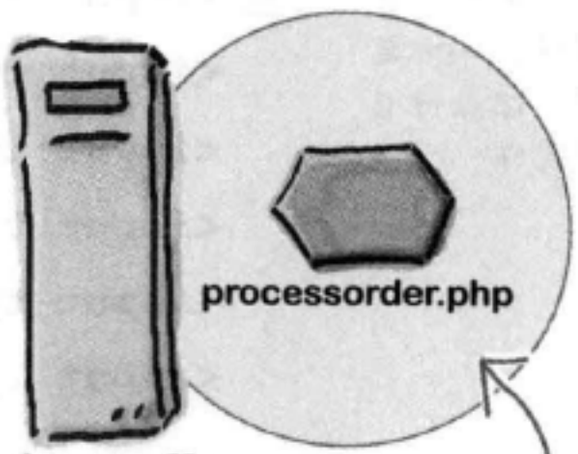


各个表单元素的唯一名字。

```
name = Buckaroo Banzai
address = Banzai Institute
city = Los Angeles
state = CA
zip = 90050
phone = 310-555-1212
```

对于每个唯一的元素名，从你在表单中输入的数据中得到一个相应的值。

浏览器打包这些数据，发送给服务器。



服务器脚本要求这些表单数据有相应的标签，这样它才能分清谁是谁。

there are no Dumb Questions

问：text <input>与<textarea>有什么区别？

答：如果输入只有一行的文本，你可能想使用一个text <input>，如姓名或邮政编码。对于更长的多行文本，就要使用<textarea>。

问：提交按钮上能不能不显示“Submit”，可以显示别的名字吗？

答：可以，只需要为元素指定一个value属性，比如指定为“Order Now”。另外，还可以使用文本输入元素的value属性为这个输入域提供默认文本。

问：在text <input>或<textarea>中可以输入多少文本，有没有限制？

答：对于在text <input>或<textarea>中能够输入多少文本，浏览器确实有一个限制。不过，这个限制范围很大，通常你都不需要输入那么多内容，并不会超出这个限制。如果你希望限制用户在text <input>中输入的文本，可以使用maxlength属性，把它设置为一个特定的字符数。例如，maxlength=“100”，这会限制用户最多能输入100个字符。不过，对于<textarea>，HTML中没有办法限制用户键入多少文本。

问：“tel”、“email”和“url”看起来就像普通的文本输入元素一样。它们有什么区别吗？

答：“tel”、“email”和“url”类型的输入元素都会向服务器脚本发送文本串，所以从这个意义上讲，它们与text类型的输入元素确实是一样的。不过也有区别，例如，由于浏览器知道输入类型是“tel”，所以在为用户提供用户界面时，它会更聪明。在一些移动浏览器上，可能会显示一个数字键盘。

问：我还是不清楚这些名字怎么与表单数据对应起来。

答：OK，你知道每个表单元素都有一个的名字，而且还知道元素有一个相应的值。单击提交按钮时，浏览器会拿到所有名字和相应的值，把它们发送到服务器。例如，如果在一个名为“zip”的text <input>元素中输入了邮政编码“90050”，表单提交时，浏览器会把“zip = 90050”发送到服务器。

问：服务器脚本怎么知道我要在表单中使用哪些名字？换句话说，我怎么为表单元素选择名字？

答：这个问题问得好。实际上应该反过来：你必须知道服务器脚本希望有哪些表单元素名，并相应地编写表单。如果你在使用别人编写的一个服务器脚本，他必须告诉你要使用哪些元素名，或者必须在脚本文档中提供有关信息。也可以寻求托管公司的帮助。

问：为什么<option>元素没有一个name属性？所有其他表单元素都有名字。

答：你真敏锐。所有<option>元素实际上是菜单的一部分，而菜单由<select>元素创建。所以，我们只需要为整个菜单提供一个名字，这已经在<select>元素中指定了。换句话说，<option>元素不需要name属性，因为<select>已经为整个菜单指定了名字。要记住，提交表单时，只会把当前选择的选项连同这个名字发送到服务器。

问：你不是说每个表单元素的名字都必须唯一吗？但是radio <input>元素的名字怎么都是一样的？

答：没错。单选钮是成组出现的。可以这样来考虑：如果按下一个按钮，其余的按钮就会“弹起”。所以，为了让浏览器知道这些单选钮属于同一组，你要使用同一个名字。假设有一组名为“color”的单选钮，值分别为“red”、“green”和“blue”。它们都是颜色，一次只能选择一个颜色，所以为这一组单选钮指定一个名字是有道理的。

问：复选框呢？它们的工作与单选钮类似吗？

答：对，唯一的区别是，复选框允许你选择多个选项。

浏览器将表单数据发送到服务器时，它会把所有复选框值合并为一个值，并将这个值连同复选框名发送到服务器。所以，假设你有一个“spice”复选框，值分别为“salt”、“pepper”和“garlic”，假如你把这些复选框都选中了，浏览器就会向服务器发送“spice = salt&pepper&garlic”。

问：天呐，数据发送到服务器这么复杂，我必须了解所有这些细节吗？

答：你只需要知道服务器脚本期望得到的表单元素的名字和类型。除此以外，对这些细节有所了解有时会有帮助，不过，没错，你不需要知道向服务器发送数据的所有这些后台的细节。

将这些<input>元素放在HTML中

现在要把这些<input>元素放在表单中。查看下面增加的元素，然后在你的“form.html”文件中完成修改。

这只是“form.html”中的表单片段。嘿，节省些篇幅可以少砍些树！

把这个元素直接嵌套在表单中。

先把所有内容放在一个<p>元素中。

```
<form action="http://starbuzzcoffee.com/processorder.php" method="POST">
  <p>Ship to: <br>
    Name: <input type="text" name="name"> <br>
    Address: <input type="text" name="address"> <br>
    City: <input type="text" name="city"> <br>
    State: <input type="text" name="state"> <br>
    Zip: <input type="text" name="zip"> <br>
    Phone: <input type="tel" name="phone"> <br>
  </p>
  <p>
    <input type="submit" value="Order Now">
  </p>
</form>
```

这些都是<input>元素：各元素分别对应表单中“Ship to”部分的各个输入域。

我们为各个输入域增加了一个标签，这样用户就能知道这个输入域中要输入什么内容。

还要知道<input>是一个内联元素，所以，如果你希望<input>元素之间有换行，就必须增加
。正是由于这个原因，才在段落中嵌入了这些
。

最后，别忘了用户需要一个提交按钮来提交这个表单。所以，增加一个提交按钮，在表单最下面插入一个type为“submit”的<input>。另外，增加一个值“Order Now”，这会把提交按钮上的文本由“Submit”改为“Order Now”。

完成这些修改之后，保存你的“form.html”文件，下面来试一试。

别忘了验证你的HTML。表单元素也需要验证！

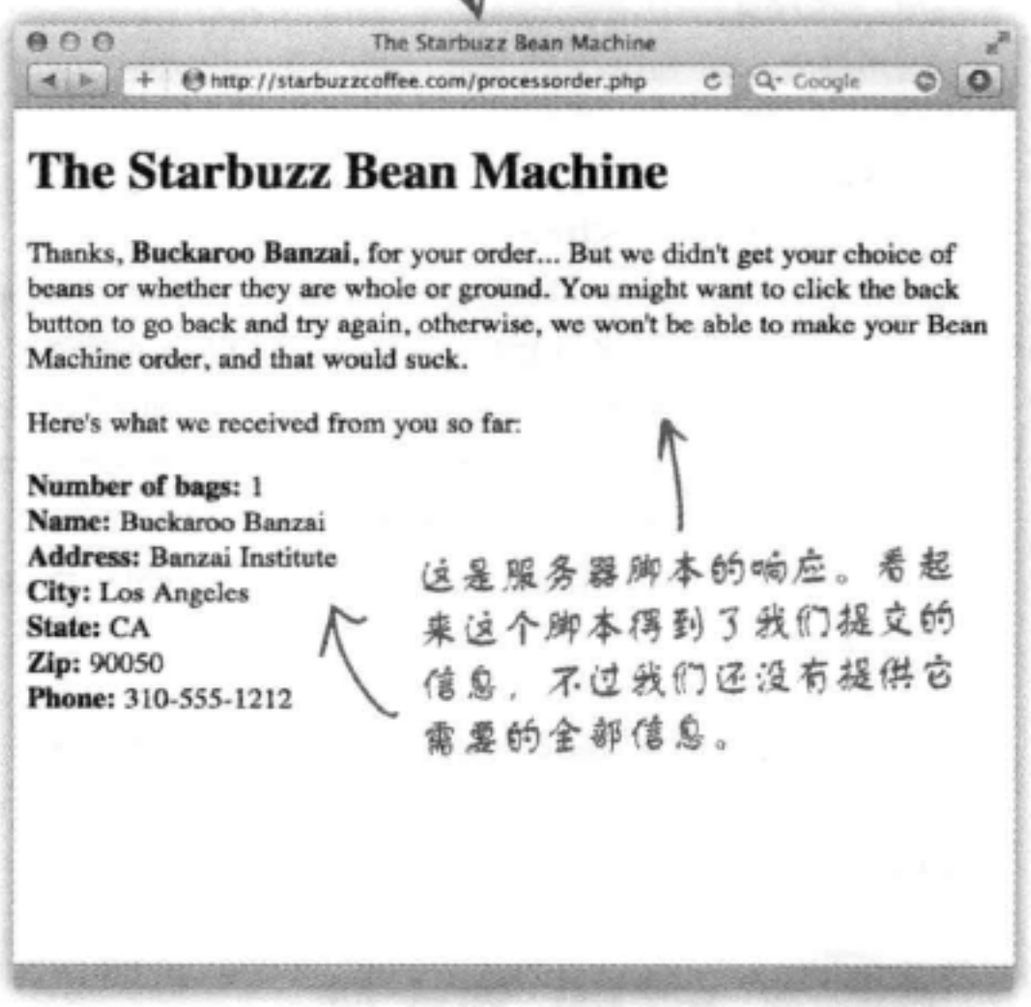
正式的表单测试

重新加载页面，填入文本输入域，然后提交表单。完成之后，浏览器会打包这些数据，把它们发送到action属性指定的URL，即starbuzzcoffee.com。

你以为我们只是给了一个“玩具例子”，它并不能真正工作，是吗？说真的，starbuzzcoffee.com已经准备好了，完全可以接受你提交的表单。试试看吧！

这是我们得到的表单。

注意，提交表单后地址栏中的URL会改变（可以在地址栏中看到表单action属性中指定的URL）。



这是服务器脚本的响应。看起来这个脚本得到了我们提交的信息，不过我们还没有提供它需要的全部信息。

这是提交表单后得到的响应。

为表单增加更多输入元素

看来如果不告诉服务器脚本我们想要什么咖啡，以及想要什么类型的咖啡（研磨咖啡还是全豆咖啡），这个脚本并没有太大意义。下面先增加咖啡选择，在表单中增加一个<select>元素。记住，<select>元素包含一个选项列表，每个列表项将成为下拉菜单中的一个选择。另外，每个选择会与一个值关联，表单提交时，所选菜单项的值会发送到服务器。翻开下一页，来增加这个<select>元素。

增加<select>元素

```
<form action="http://starbuzzcoffee.com/processorder.php" method="post">

  <p>
    Choose your beans:
    <select name="beans">
      <option value="House Blend">House Blend</option>
      <option value="Bolivia">Shade Grown Bolivia Supremo</option>
      <option value="Guatemala">Organic Guatemala</option>
      <option value="Kenya">Kenya</option>
    </select>
  </p>

  <p>
    Ship to: <br>
    Name: <input type="text" name="name" value=""><br>
    Address: <input type="text" name="address" value=""><br>
    City: <input type="text" name="city" value=""><br>
    State: <input type="text" name="state" value=""><br>
    Zip: <input type="text" name="zip" value=""><br>
    Phone: <input type="tel" name="phone" value=""><br>
  </p>
  <p>
    <input type="submit" value="Order Now">
  </p>
</form>
```

这是全新的<select>元素。它也有一个唯一的名字。

在这个元素中放入各个<option>元素，分别对应一种咖啡选择。

HTML放大镜

下面来仔细查看<option>元素。

每个option有一个值。

这个元素的内容将作为下拉菜单中的标签。

```
<option value="Guatemala">Organic Guatemala</option>
```

浏览器将表单元素的名和值打包时，它会使用<select>元素的名和所选选项的值。

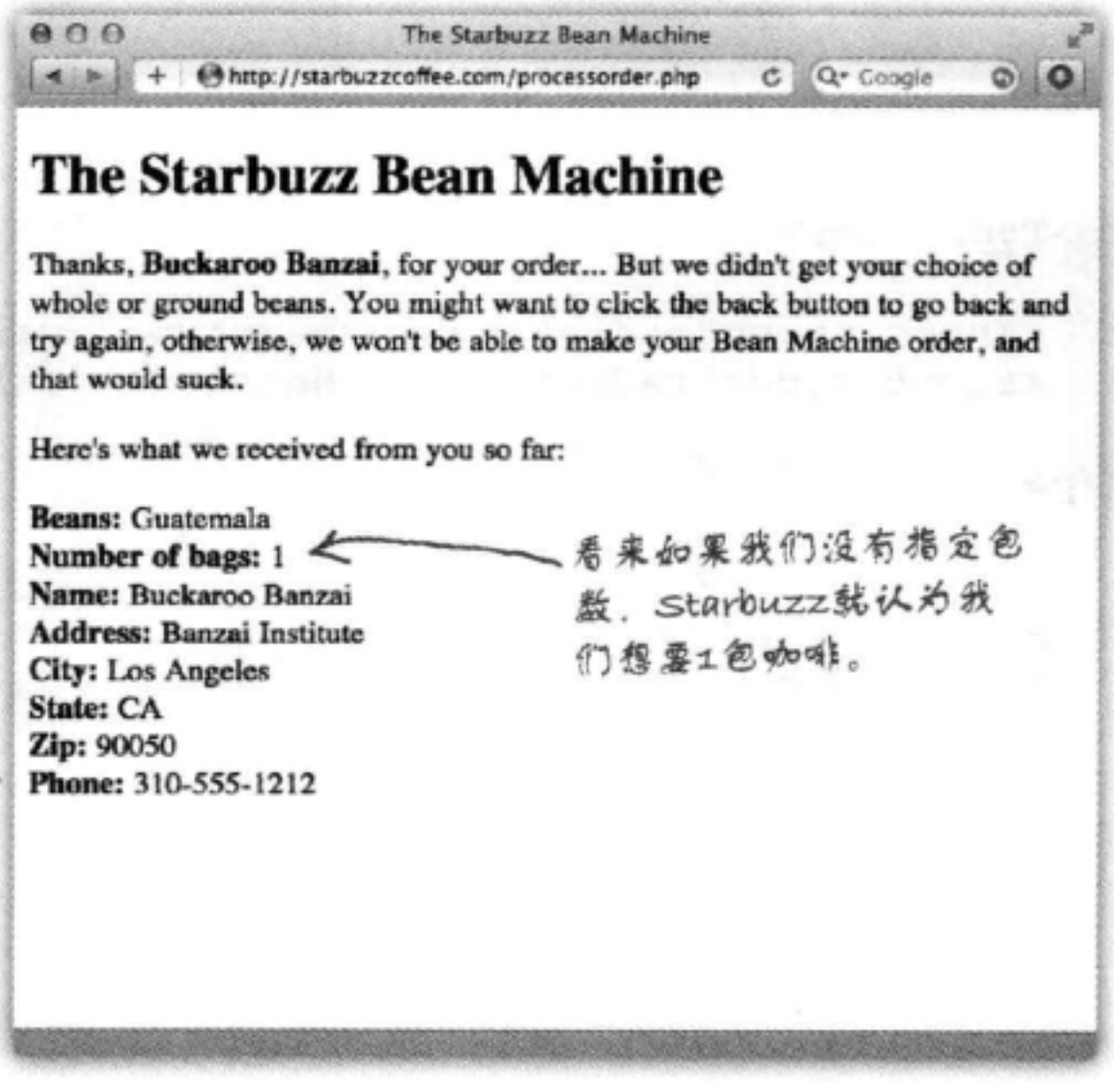
在这个例子中，浏览器会向服务器发送beans = "Guatemala"。

测试<select>元素

下面来测试这个<select>元素。重新加载你的页面，你会看到一个漂亮的新菜单。选择你喜欢的咖啡，填写表单的其余部分，然后提交你的订单。

这是我们得到的表单，现在增加了一个<select>元素。注意这里提供了所有的选项。

我们还是没有提供服务器脚本需要的全部信息，不过起码目前脚本得到了表单里的所有信息。



这是<select>选择的结果。

看来如果我们没有指定包数，Starbuzz就认为我们想要1包咖啡。

这里是所有text输入控件和tel输入控件的结果。

将<select>元素的name属性改为“thembeans”。重新加载表单，再次提交你的订单。这对于从服务器脚本得到的结果有什么影响?

完成这个练习之后，一定要把name属性再改回为“beans”。

允许客户选择全豆咖啡还是研磨咖啡

客户应该能在订单中选择全豆咖啡还是研磨咖啡。为此，我们要使用单选钮。单选钮就像老式汽车电台上的按钮，一次只能按下一个按钮。在HTML中，你要为每个按钮创建一个类型为“radio”的<input>，所以这里我们需要两个按钮：一个对应全豆咖啡，另一个对应研磨咖啡。如下所示：

这里有两个单选钮：一个对应全豆咖啡（whole bean），另一个对应研磨咖啡（ground）。

```
<p>Type: <br>
   <input type="radio" name="beantype" value="whole"> Whole bean <br>
   <input type="radio" name="beantype" value="ground"> Ground
</p>
```

这里要使用<input>元素，type设置为“radio”。

这是唯一的名字。同组的所有单选钮都有相同的名字。

这是将发送到服务器脚本的值。只会发送其中一个值（提交表单时选中的那个单选钮）。

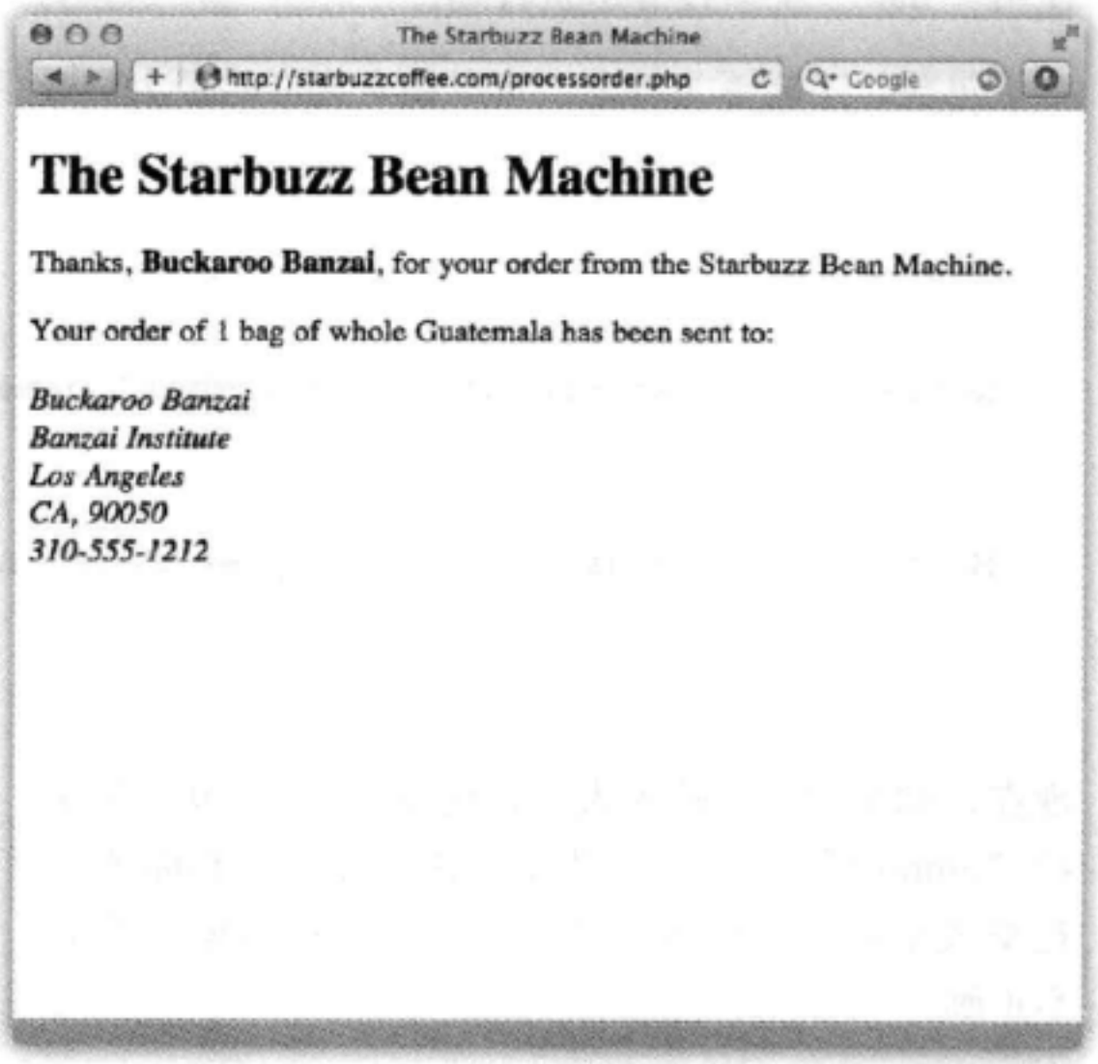
注意我们通常把单选钮的标签放在元素右边。

试试单选按钮

把上一页的单选按钮插入到你的HTML中，放在包含<select>元素的段落下面。一定要重新加载页面，然后再次提交表单。

你可能会注意到，重新加载页面时所有单选钮都未选中（具体情况要看你使用什么浏览器）。

哇！Starbuzz收到我们的订单了，不过我们还没有填好呢。还需要增加包数、送货日期和礼品选项，另外还要提供一个文本区填写客户意见。



如果表单没有包含全部元素，怎么能履行订单呢？嗯，这取决于服务器脚本是如何编写的。按照这里编写的脚本，即使没有随其余的表单数据提交礼品包装、商品目录选项和客户意见，脚本也能处理订单。要想知道一个服务器脚本是否需要某些表单元素，唯一的办法就是与开发脚本的人交流，或者阅读文档。

Exercise

嘿，80%的客户都订购了研磨咖啡。你能不能处理一下，当用户加载页面时，能不能让研磨咖啡类型已经选中？

如果为单选钮输入元素增加一个布尔属性“checked”，浏览器显示表单时就会默认地选中这个元素。为“ground”radio <input>元素增加checked属性，再来测试这个页面。这一章最后给出了答案。

记住，布尔属性不需要值。如果有属性checked，这个输入控件就会选中。

使用更多输入类型

接下来，我们需要得到客户想要购买的咖啡包数，以及送货日期。它们都是<input>元素，不过并不只是使用基本的文本输入，我们可以更特定地指定希望在这些<input>元素中输入什么类型的内容，这里将使用“number”类型提供包数，使用“date”类型设置送货日期。

对于包数，还能更为特定，可以指定允许的最大和最小包数：

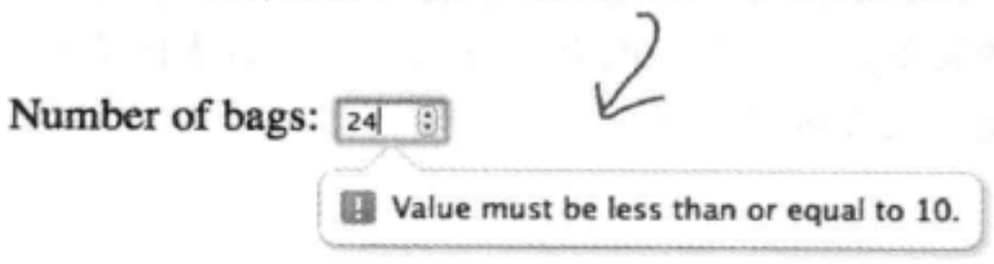
利用“number”类型，并指定最小和最大包数，可以限制输入是一个有效的值（我们不希望客户一次购买10包以上同类咖啡）！

```
Number of bags: <input type="number" name="bags" min="1" max="10">

Must arrive by date: <input type="date" name="date">
```

这里使用了“date”类型，支持这种类型的浏览器会为客户提供帮助，弹出一个日期选择控件。

现在，如果你试图输入大于10包或者少于1包，在支持“number”<input>类型的浏览器中，当你试图提交表单时，会得到一个错误消息，指示输入的值不正确。

如果你输入的值大于允许的最大值（或小于允许的最小值），就会得到一个错误消息。



增加数字和日期输入类型

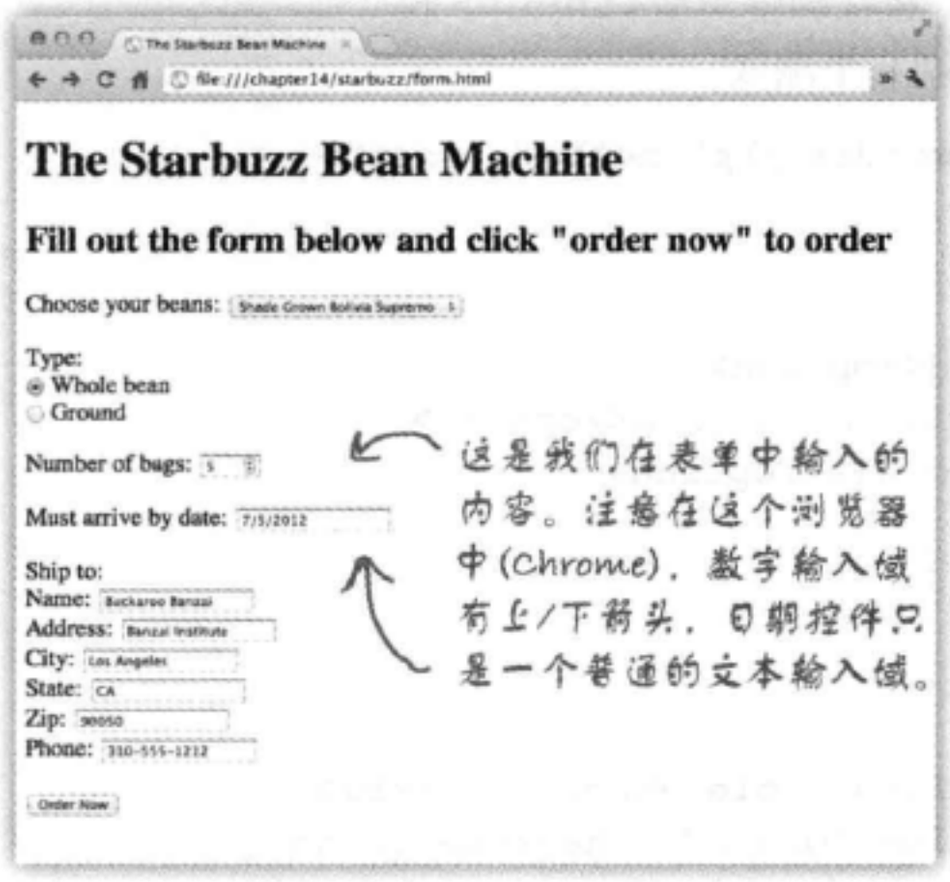
把这两个新的<input>元素增加到你的“form.html”文件，放在咖啡类型<input>下面，而且要在Ship To（送货地址）部分上面，对这些新代码做一个测试。

```
<form action="http://starbuzzcoffee.com/processorder.php" method="post">
    <p>
      Choose your beans:
      <select name="beans">
        <option value="House Blend">House Blend</option>
        <option value="Bolivia">Shade Grown Bolivia Supremo</option>
        <option value="Guatemala">Organic Guatemala</option>
        <option value="Kenya">Kenya</option>
      </select>
    </p>
    <p>
      Type:<br>
        <input type="radio" name="beantype" value="whole">Whole bean<br>
        <input type="radio" name="beantype" value="ground" checked>Ground
    </p>

    <p>
      Number of bags: <input type="number" name="bags" min="1" max="10">
    </p>
    <p>
      Must arrive by date: <input type="date" name="date">
    </p>

    <p>
      Ship to: <br>
      Name: <input type="text" name="name" value=""><br>
      Address: <input type="text" name="address" value=""><br>
      City: <input type="text" name="city" value=""><br>
      State: <input type="text" name="state" value=""><br>
      Zip: <input type="text" name="zip" value=""><br>
      Phone: <input type="tel" name="phone" value=""><br>
    </p>
    <p>
      <input type="submit" value="Order Now">
    </p>
</form>
```

这里增加了新代码。记住，浏览器可能会提供不同的显示，这要看你使用哪一个浏览器。应当在更多的浏览器上试试看！

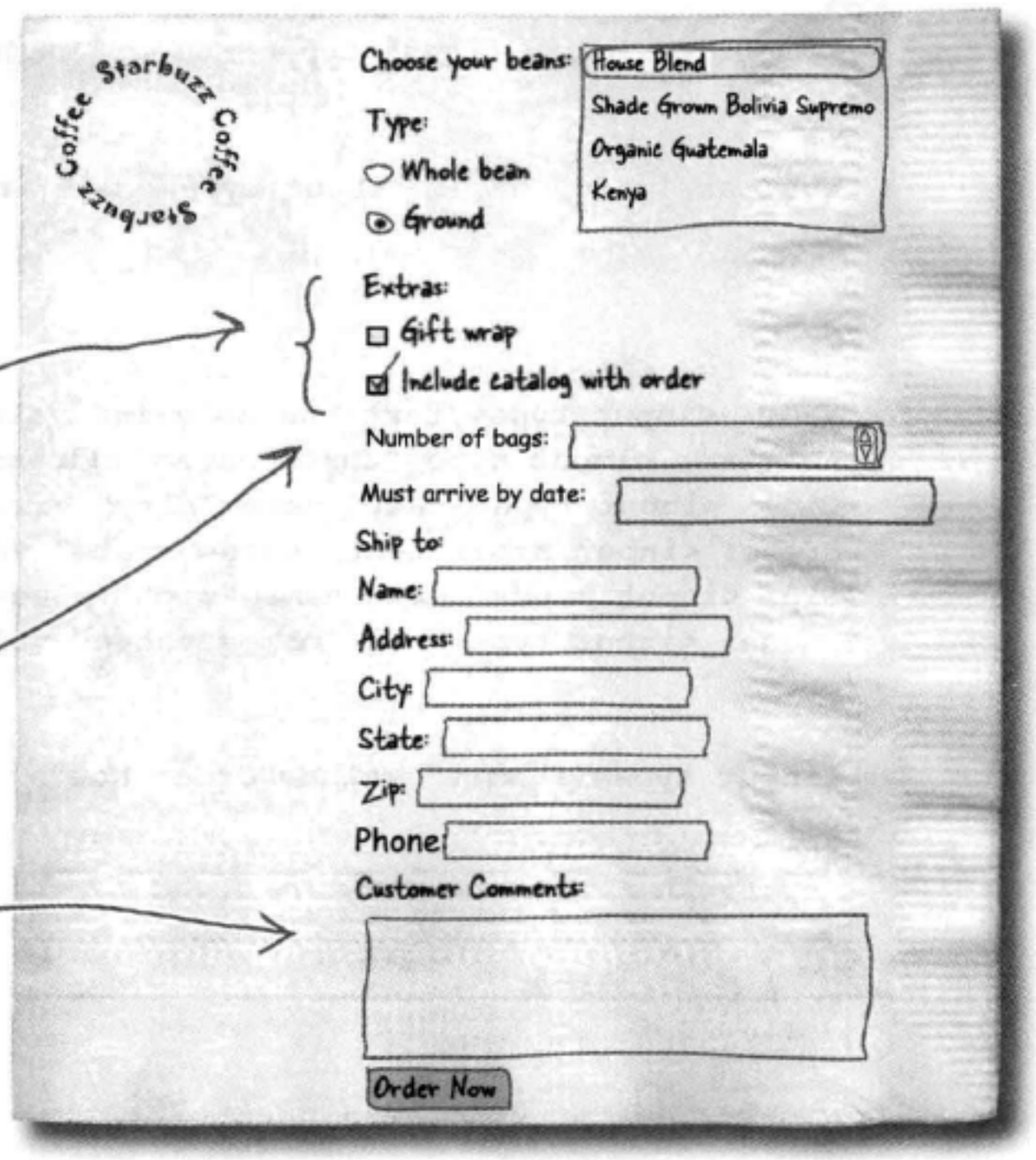
翻开下一页，看看测试的结果……

测试number和date<input>元素



完成表单

就快完工了。还有两个控件需要增加到表单上：首先是包括两个复选框的“Extras”控件，还有一个客户意见控件。你已经对表单很了解了，所以下面同时增加这两个控件。



Extras部分包括两个复选框，一个对应礼品包装，另一个表示是否包括一个商品目录。

看起来“Include catalog”（包含商品目录）选项要默认选中。

Customer Comments（客户意见）部分就是一个<textarea>。

增加复选框和文本区

你应该已经知道怎么做了，查看下面的新HTML，把它增加到你的“form.html”中。

```
<form action="http://starbuzzcoffee.com/processorder.php" method="post">
    <p>
      Choose your beans:
      <select name="beans">
        <option value="House Blend">House Blend</option>
        <option value="Bolivia">Shade Grown Bolivia Supremo</option>
        <option value="Guatemala">Organic Guatemala</option>
        <option value="Kenya">Kenya</option>
      </select>
    </p>
    <p>
      Type:<br>
        <input type="radio" name="beantype" value="whole">Whole bean<br>
        <input type="radio" name="beantype" value="ground" checked>Ground
    </p>
    <p>Number of bags: <input type="number" name="bags" min="1" max="10"></p>
    <p>Must arrive by date: <input type="date" name="date"></p>

    <p>
      Extras:<br>
      <input type="checkbox" name="extras[]" value="giftwrap">Gift wrap<br>
      <input type="checkbox" name="extras[]" value="catalog" checked>Include catalog
        with order
    </p>

    <p>
      Ship to: <br>
      Name: <input type="text" name="name" value=""><br>
      Address: <input type="text" name="address" value=""><br>
      City: <input type="text" name="city" value=""><br>
      State: <input type="text" name="state" value=""><br>
      Zip: <input type="text" name="zip" value=""><br>
      Phone: <input type="tel" name="phone" value=""><br>
    </p>

    <p>Customer Comments:<br>
        <textarea name="comments"></textarea>
    </p>

    <p>
      <input type="submit" value="Order Now">
    </p>
</form>
```

这里为每个选项增加了一个复选框。注意，它们都有相同的名字“extras[]”……

……不过有不同的值。

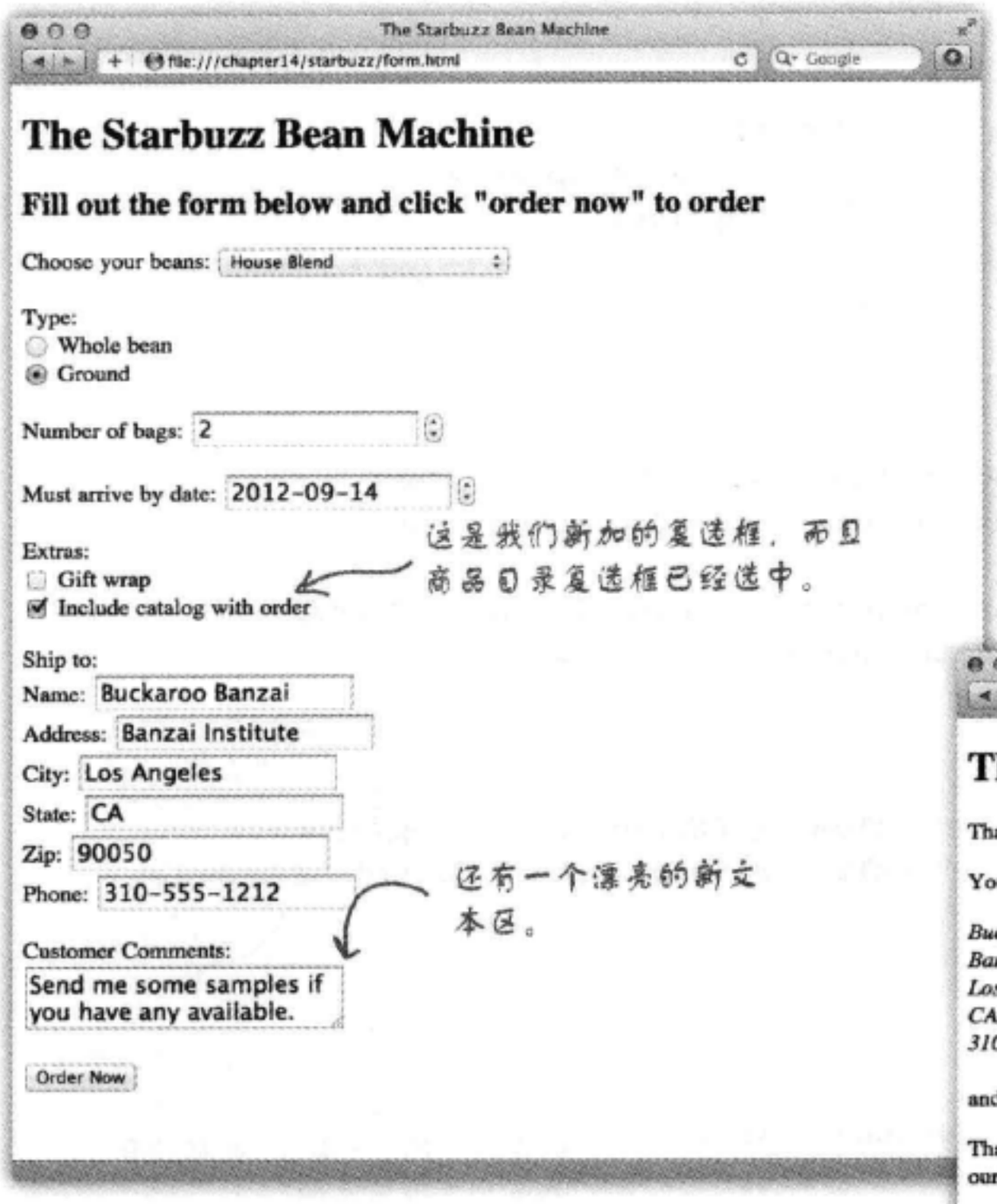
我们使用了checked属性来指定商品目录选项默认是选中的。可以为多个复选框增加checked属性。

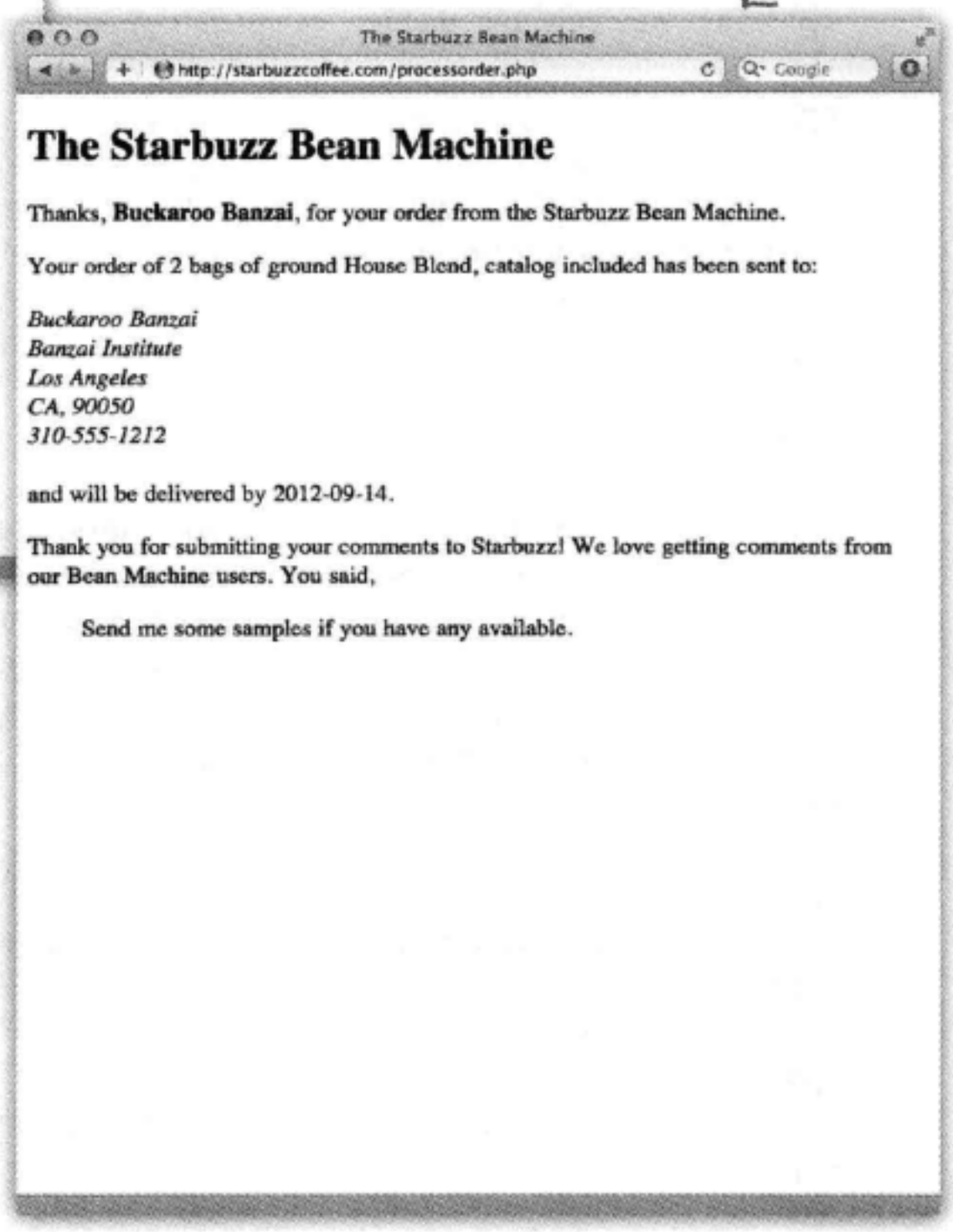
与单选钮一样，我们把这些标签放在复选框的右边。

这里是一个文本区。

最后的表单测试

保存你的修改，重新加载页面，查看这个新表单。 是不是感觉好多了？





尝试按各种可能的组合（有或没有礼品包装，有或没有商品目录，以及不同的咖啡等）发送这个表单，看看是不是都能正确处理。

不管你相不相信，“extras[]”确实是一个完全合法的表单元素名。

不过，尽管这是合法的，但看起来不太对劲，是不是？可以这样来考虑：从HTML的角度看，这只是一个普通的表单元素名，名字里有没有中括号对于浏览器没有任何影响。

那么，为什么要使用这样的元素名呢？这是因为，编写“processorder.php”服务器脚本所用的脚本语言（PHP）希望得到一点提示，想知道一个表单变量可能包含多个值。提供这个提示的做法就是在名字后面增加一个“[]”。

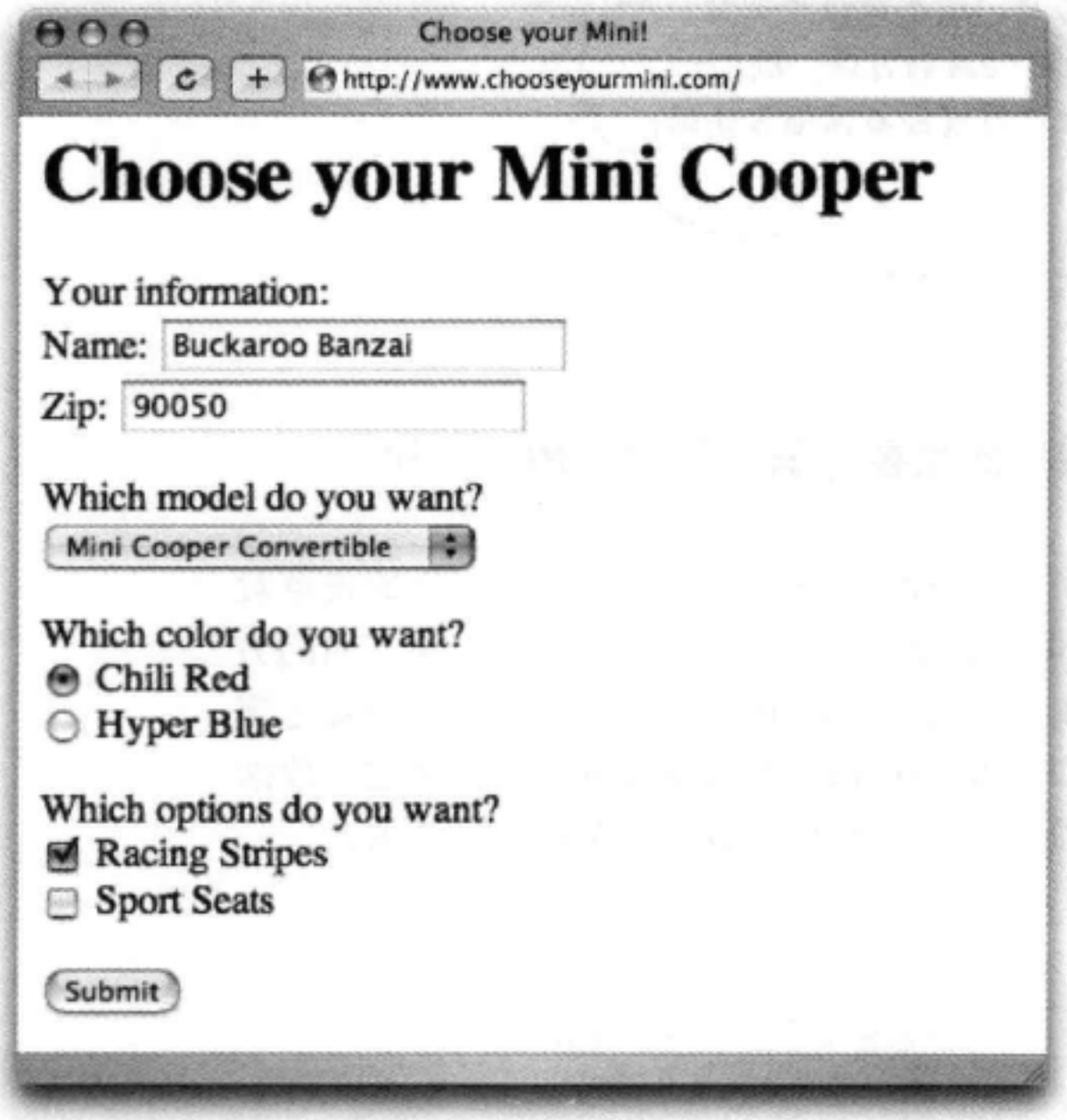
所以，从学习HTML来讲，你完全可以先不考虑它，不过最好还是能记得，万一将来你要写一个使用PHP服务器脚本的表单呢。

扮演浏览器

下面有一个HTML表单，右边是用户输入到这个表单的数据。你的任务是扮演浏览器，把各个表单元素名与用户输入的值配对。完成这个练习之后，检查这一章最后的答案，看看表单元素名和值是否正确匹配。

```
<form action="http://www.chooseyourmini.com/choice.php" method="POST">
   <p>Your information: <br>

      Name: <input type="text" name="name"><br>
      Zip: <input type="text" name="zip"><br>

   </p>
   <p>Which model do you want? <br>
     <select name="model">
         <option value="cooper">Mini Cooper</option>
         <option value="cooperS">Mini Cooper S</option>
         <option value="convertible">Mini Cooper Convertible</option>
     </select>
   </p>
   <p>Which color do you want? <br>
     <input type="radio" name="color" value="chilired"> Chili Red <br>
     <input type="radio" name="color" value="hyperblue"> Hyper Blue
   </p>
   <p>Which options do you want? <br>
     <input type="checkbox" name="caroptions[]" value="stripes"> Racing Stripes
     <br>
     <input type="checkbox" name="caroptions[]" value="sportseats"> Sport Seats
   </p>

   <p>
     <input type="submit" value="Order Now">
   </p>

</form>
```

这里是表单。



这里是已经填写的表单。

将各部分表单数据与相应的表单名匹配，在这里写出你的答案。

```
name = "Buckaroo Banzai"
zip =
model =
color =
caroptions[] =
```

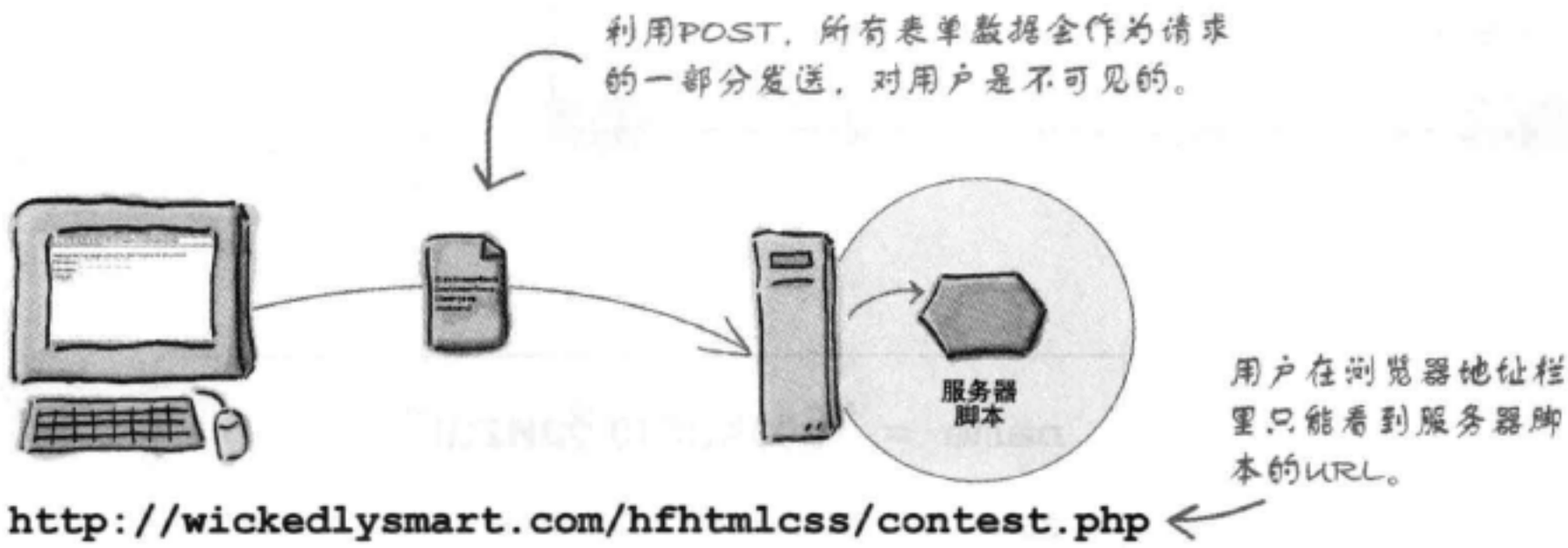
加分题……

既然已经完成了表单，能不能谈谈浏览器向服务器发送数据使用的方法？我们一直都在使用POST，不过你说过还有其他方法的。

浏览器使用的主要方法有两种：POST和GET。

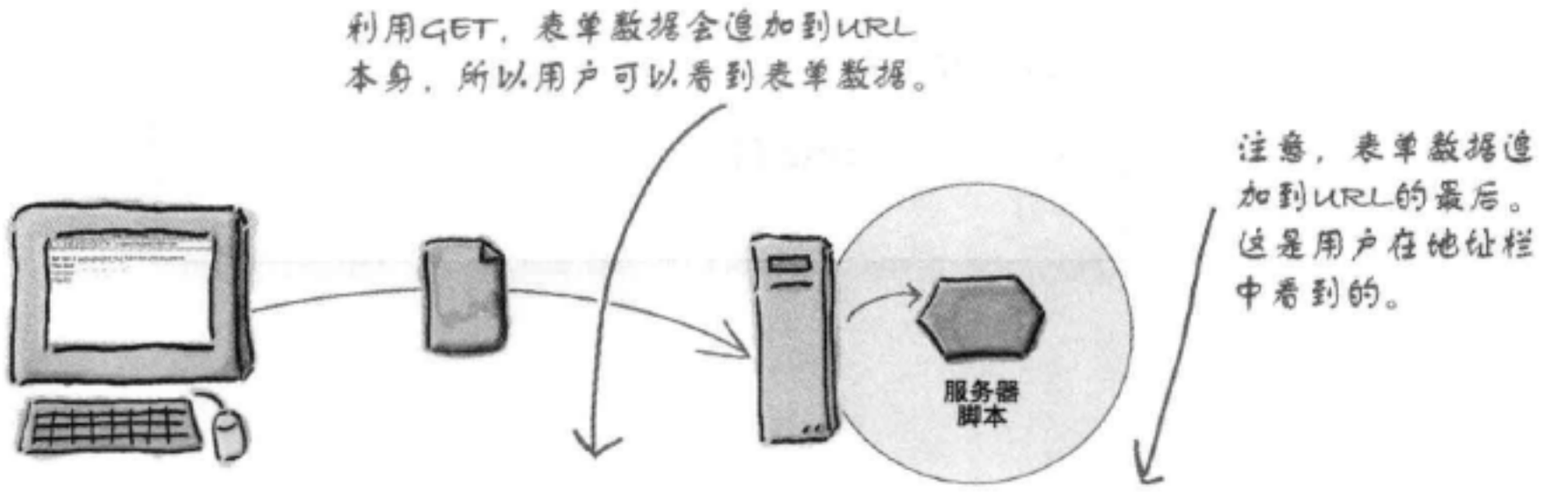
POST和GET完成的任务是一样的，都是将你的表单数据从浏览器发送到服务器，不过采用了两种不同的方式。POST会打包你的表单变量，在后台把它们发送到服务器；GET也会打包你的表单变量，但会把这些数据追加到URL的最后，然后向服务器发送一个请求。

POST



http://wickedlysmart.com/hfhtmlcss/contest.php

GET



http://wickedlysmart.com/hfhtmlcss/contest.php?firstname=buckaroo&lastname=banzai

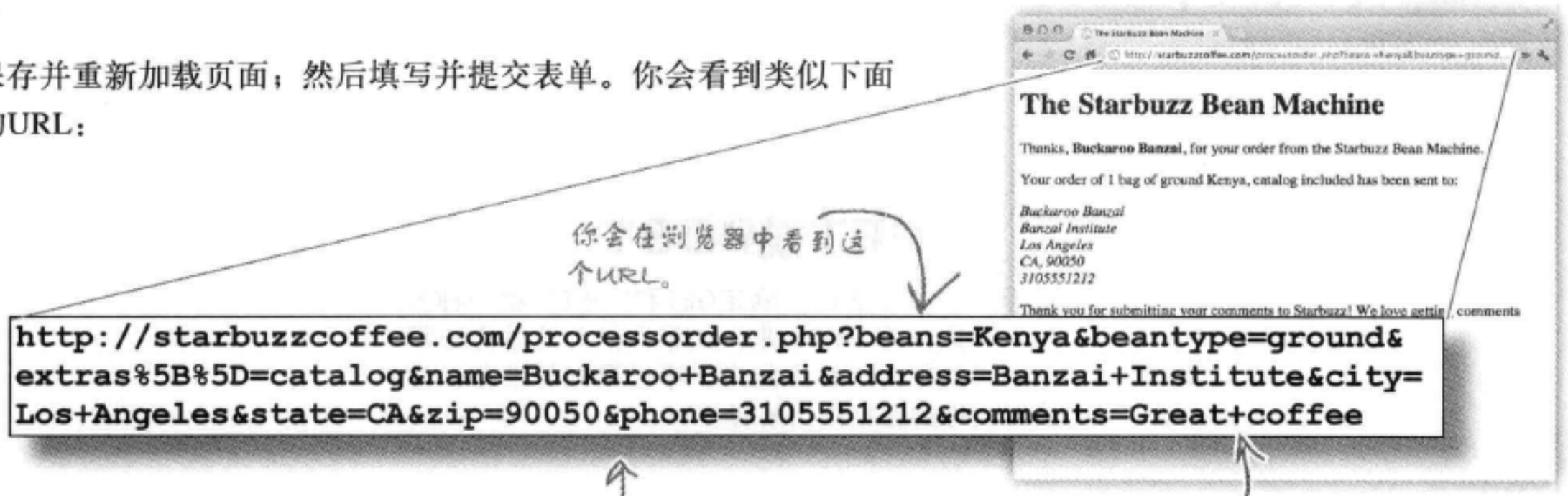
GET的实际使用

要了解GET，最好的办法就是看它的实际使用。打开“form.html”文件，做下面这个小小的修改：

只需要把方法从“POST”改为“GET”。

```
<form action="http://starbuzzcoffee.com/processorder.php" method="GET">
```

保存并重新加载页面；然后填写并提交表单。你会看到类似下面的URL：



你会在浏览器中看到这个URL。

```
http://starbuzzcoffee.com/processorder.php?beans=Kenya&beantype=ground&
extras%5B%5D=catalog&name=Buckaroo+Banzai&address=Banzai+Institute&city=
Los+Angeles&state=CA&zip=90050&phone=3105551212&comments=Great+coffee
```

在这个URL中可以看到每个表单元素名和相应的值。

注意浏览器会对一些字符编码，如空格。服务器脚本接收到这些字符时会自动对它们解码。

there are no Dumb Questions

问：既然我们在向服务器发送数据，为什么这种方法叫GET？

答：问得好。浏览器的主要任务是什么？就是从服务器得到Web页面。使用GET时，浏览器像以往一样用正常的方式得到Web页面，只不过，由于有一个表单，它会在URL的最后追加另外一些数据。除此以外，浏览器会把它当作一个普通请求来处理。

另一方面，利用POST，浏览器实际上会创建一个小数据包，并把它发送到服务器。

问：那么为什么我要使用POST而不是GET，或者反之，为什么有时要使用GET而不是POST？

答：它们确实有几个很大的差别。如果你希望用户能够对提交表单后的结果页面加书签，就必须使用GET，因为如果使用POST，返回的结果页面就无法加书签了。什么时候想要加书签呢？假设你有一个服务器脚本，它会返回一个搜索结果列表，你可能希望用户能够对这些结果加书签，这样他们就能直接查看这些结果，而不用再填写表单。

另一方面，如果你有一个处理订单的服务器脚本，可能不希望用户对这个页面加书签（否则，每次他们返回到这个书签时，都会重新提交这个订单）。

还有一种情况你肯定不想使用GET，比如你的表单中的数据是私有的，如信用卡或一个口令。因为URL是明文可以看到，别人只要查看你的浏览器历史，就能看到这些信息。或者如果将GET结果设置了书签，别人也会很容易看到这些私有信息。

最后，如果你使用了一个<textarea>，就应该使用POST，因为可能会发送大量数据。GET和POST请求对于发送的数据量都有一个限制，不过对POST请求的限制通常要宽松得多。

Sharpen your pencil

GET或POST

对于以下各个描述，确定GET和POST中哪一种方法更合适，把它圈出来。如果你认为两种方法都可行，就把两个都圈起来。不过要做好准备，你要能对你的答案做出解释……

GET **POST** 输入用户名和口令的表单

GET **POST** 订购CD的表单。

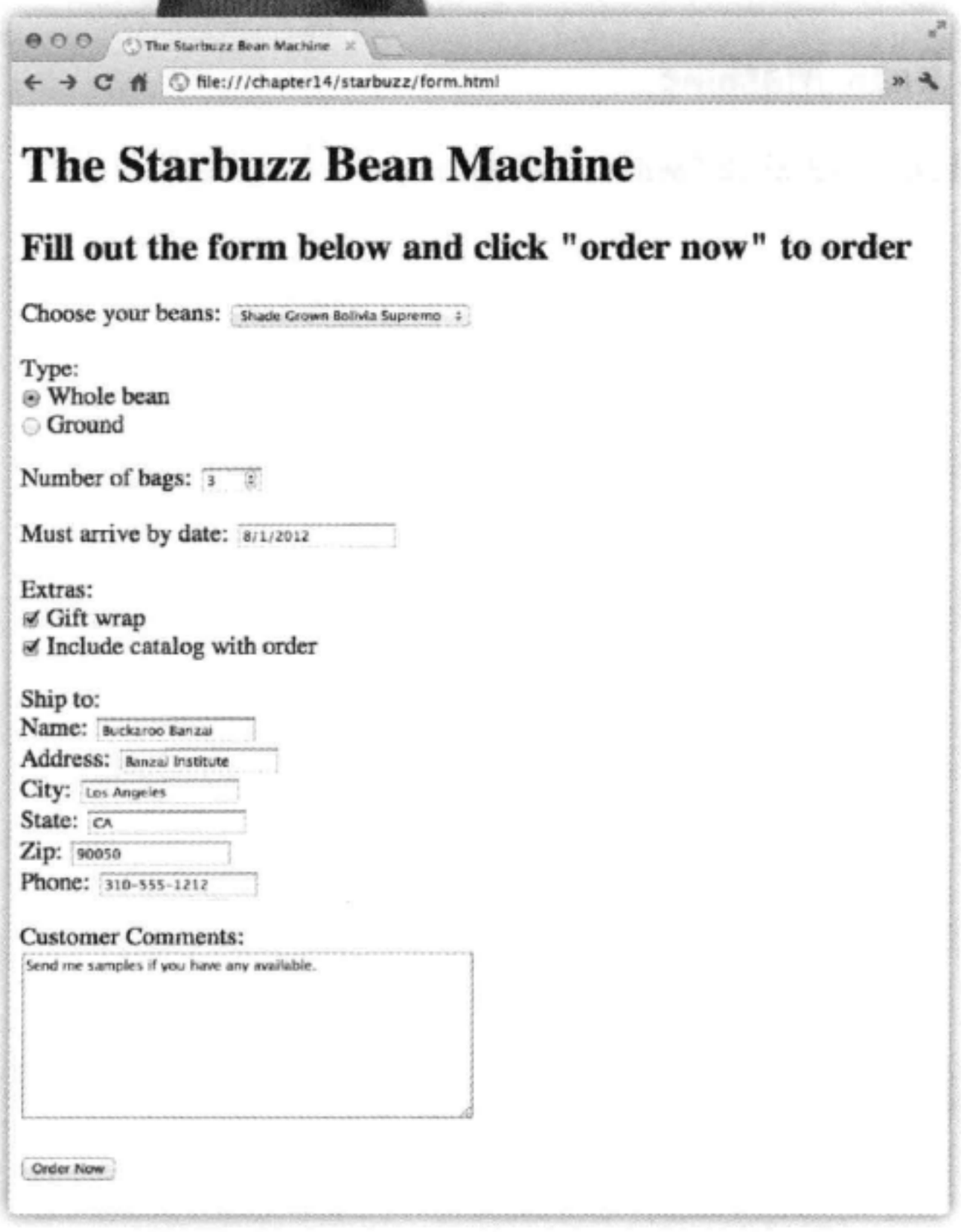
GET **POST** 查看当前时事的表单。

GET **POST** 提交书评的表单。

GET **POST** 按身份证号查看福利的表单。

GET **POST** 发送客户反馈的表单。

我不得不承认，你的Bean Machine页面做得真不错！这肯定会大大促进我们的咖啡销售。现在还需要给这个表单增加一点样式，我们已经准备好迎接客户了。



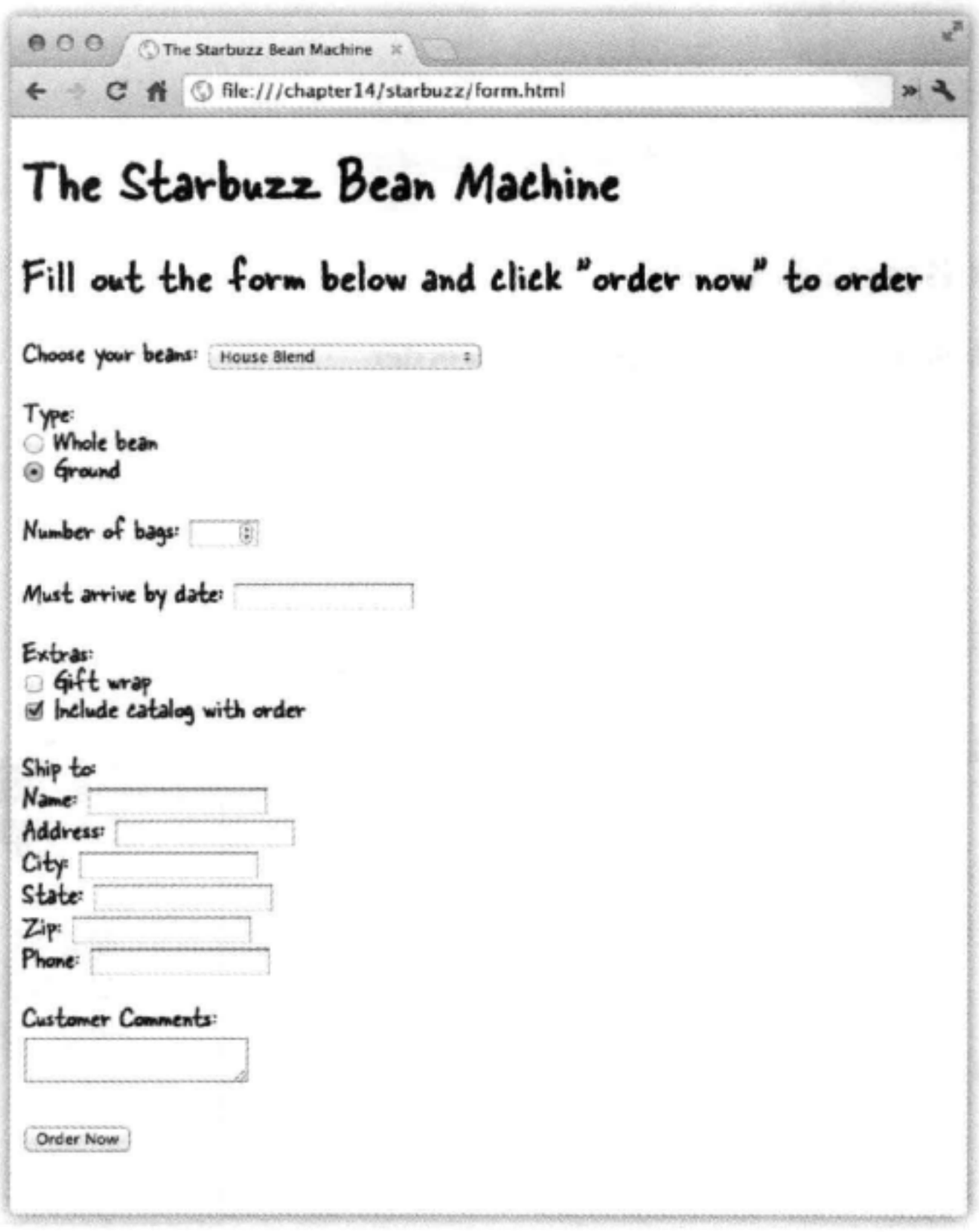
根据你了解的HTML和CSS的全部知识，如何为这个表单增加样式？

Sharpen your pencil

表单通常采用表格布局，所以你可能发现，CSS表格显示布局对于设计这个表单的外观很合适……我们就采用这种方式来建立Bean Machine表单的布局。通过使用这种表格显示布局，页面才会像一个真正的表单，而不是一组输入元素随便地堆在一起，而且这样更容易阅读。

在具体设计布局之前，先来明确这个表单固有的表格结构。先从下面的草图开始，把这些元素放在一个表格中（提示：我们发现刚好能放在2列14行中），所以每行用一个块元素表示，每个单元格也用一个块元素表示。注意，可能需要为HTML增加一些结构，它才能正常工作。

完成这个练习之前，别偷看下一页。真的！可以把下一页先盖住。

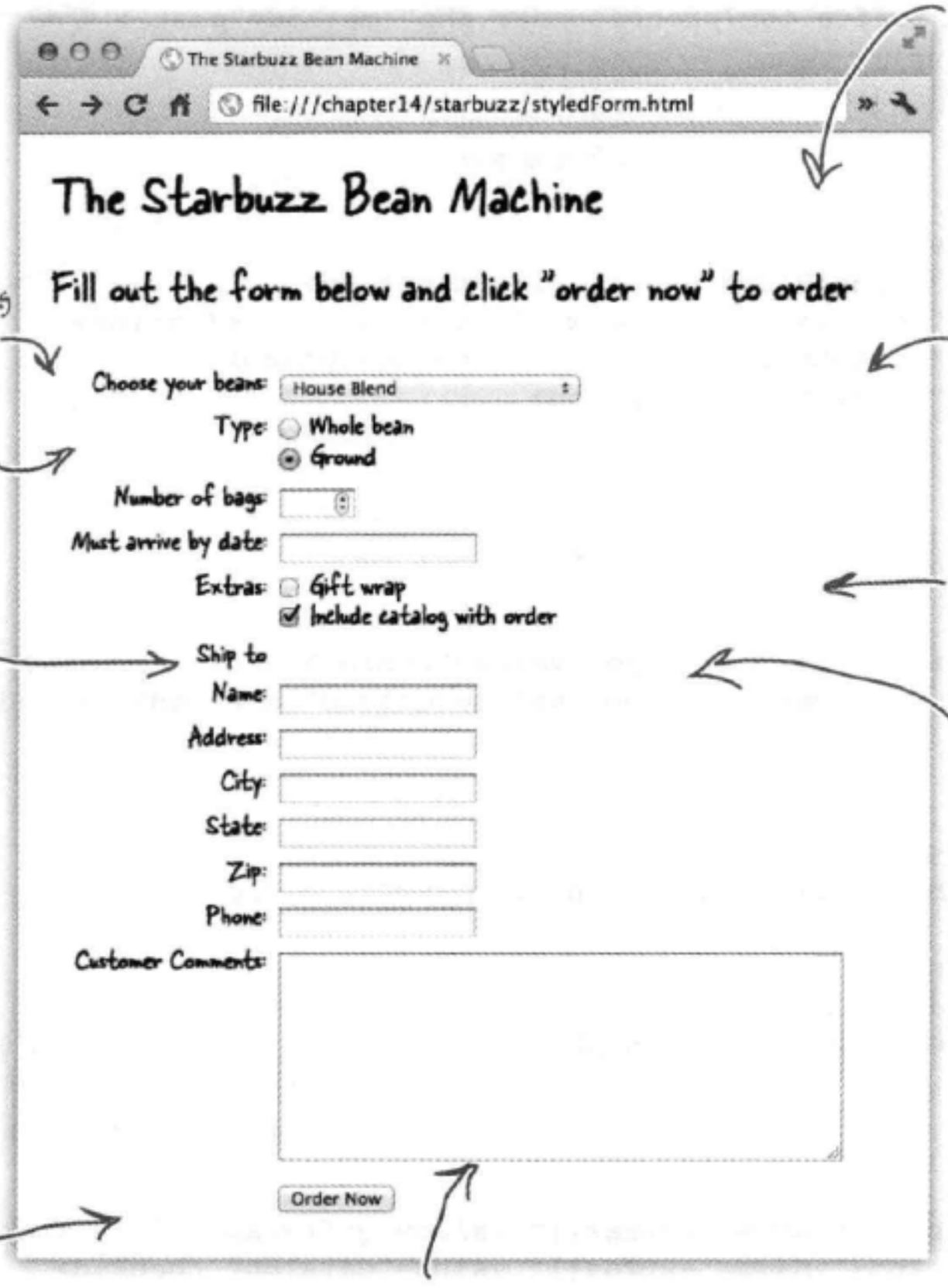


Sharpen your pencil Solution

表单通常采用表格布局，所以你可能发现，CSS表格显示布局对于设计这个表单的外观很合适……我们就采用这种方式来建立Bean Machine表单的布局。通过使用这种表格显示布局，页面才会像一个真正的表单，而不是一组输入元素随便地堆在一起，而且这样更容易阅读。

在具体设计布局之前，先来明确这个表单固有的表格结构。先从下面的草图开始，把这些元素放在一个表格中（提示：我们发现刚好能放在2列14行中），所以每行用一个块元素表示，每个单元格也用一个块元素表示。注意，可能需要为HTML增加一些结构，它才能正常工作。

这是我们的答案……在学习后面的内容之前，请对照检查你的答案！



这里是表格的草图。这是一个简单的表格显示布局，共2列、14行，各行分别对应表单的各个主要部分。

每个表单元素的标签放在左列。

所有输入元素都放在右列中。

单元格值在垂直方向上都向上对齐。

注意我们把各组复选框和单选钮都归组到一个单元格中。

"Ship to"右边的单元格为空，这里没有控件。

记住，每个单元格对应一个块元素，所以我们会增加更多<p>元素，确保对于每个单元格都有一个单独的块元素。

各行还需要另外一些块元素。就像前面一样（第11章），我们将使用 元素。

最后，需要一个元素包含所有内容，作为表单本身。为此可以使用form元素！

提交按钮左边的单元格为空。这里没有放置标签。

文本区也更大了！

将表单元素放入HTML结构实现表格显示布局

既然你已经知道如何把表单元素组织在一个表格显示布局中，下面需要测试一下你编写HTML的能力。开始输入你的HTML吧！

开玩笑的。我们不会让你输入所有这些代码……毕竟，这一章的重点是表单，而不是表格显示布局。我们已经为你输入了这些HTML，就在“chapter14/starbuzz”文件夹下的“styledform.html”文件中。尽管看起来很复杂，实际上情况没那么糟糕。我们已经在下面增加了一些标注，指出了主要的部分。

这里是<form>元素，我们将使用这个元素实现“表格”显示部分。

我们将使用一个class为“tableRow”的 表示表格中的各行。

每个单元格中的内容嵌套在一个<p>元素中。

对于咖啡选择菜单、“beantype”单选钮和“extras”复选框，我们把对应各个菜单的所有表单元素都放在一个数据单元格中。

```
<form action="http://starbuzzcoffee.com/processorder.php" method="post">
<div class="tableRow">
    <p>
      Choose your beans:
    </p>
    <p>
        <select name="beans">
          <option value="House Blend">House Blend</option>
          <option value="Bolivia">Shade Grown Bolivia Supremo</option>
          <option value="Guatemala">Organic Guatemala</option>
          <option value="Kenya">Kenya</option>
        </select>
    </p>
</div>
<div class="tableRow">
    <p> Type: </p>
    <p>
        <input type="radio" name="beantype" value="whole"> Whole bean<br>
        <input type="radio" name="beantype" value="ground" checked> Ground
    </p>
</div>
<div class="tableRow">
    <p> Number of bags: </p>
    <p> <input type="number" name="bags" min="1" max="10"> </p>
</div>
<div class="tableRow label">
    <p> Must arrive by date: </p>
    <p> <input type="date" name="date"> </p>
</div>
<div class="tableRow">
    <p> Extras: </p>
    <p>
        <input type="checkbox" name="extras[]" value="giftwrap"> Gift wrap<br>
        <input type="checkbox" name="extras[]" value="catalog" checked>
        Include catalog with order
    </p>
</div>
```

代码未完，见下一页。

对于只包含标签“Ship to”的行，为<p>增加了一个类“heading”，以便对这个文本加粗。

注意，右列中还有一个空单元格，所以在这里直接放一个空的<p>元素。

所有行都很简单：一个“tableRow” 对应表行，另外每个单元格放在一个<p>中。

对于最后一行，左列有一个空单元格，所以同样的，可以在那里放置一个空的<p>元素。

```
<div class="tableRow">
    <p class="heading"> Ship to </p>
    <p></p>
</div>
<div class="tableRow">
    <p> Name: </p>
    <p> <input type="text" name="name" value=""> </p>
</div>
<div class="tableRow">
    <p> Address: </p>
    <p> <input type="text" name="address" value=""> </p>
</div>
<div class="tableRow">
    <p> City: </p>
    <p> <input type="text" name="city" value=""> </p>
</div>
<div class="tableRow">
    <p> State: </p>
    <p> <input type="text" name="state" value=""> </p>
</div>
<div class="tableRow">
    <p> Zip: </p>
    <p> <input type="text" name="zip" value=""> </p>
</div>
<div class="tableRow">
    <p> Phone: </p>
    <p> <input type="tel" name="phone" value=""> </p>
</div>
<div class="tableRow">
    <p> Customer Comments: </p>
    <p>
        <textarea name="comments" rows="10" cols="48"></textarea>
    </p>
</div>
<div class="tableRow">
    <p></p>
    <p> <input type="submit" value="Order Now"> </p>
</div>
</form>
```

用CSS建立表单样式

我们已经有了需要的所有结构，现在只需要增加一些样式规则，就大功告成了。因为这个表单是Starbuzz网站的一部分，所以我们会重用“starbuzz.css”样式表中的一些样式，另外还会创建一个新的样式表“styledform.css”，为Bean Machine表单增加新的样式规则。现在所有这些CSS对你来说都应该很熟悉了。我们没有使用任何表单特定的规则，这里用到的还是前几章使用的内容。

这个CSS可以在文件夹“chapter14/starbuzz”下的“styledform.css”文件中找到。

我们要依赖Starbuzz CSS建立一些样式，不过还会为body增加Starbuzz背景图像和一个外边距。

```
body {
    background: #efe5d0 url(images/background.gif) top left;
    margin: 20px;
}
```

这里使用form利用表格显示布局来表示表格……

```
form {
    display: table;
    padding: 10px;
    border: thin dotted #7e7e7e;
    background-color: #e1ceb8;
}
```

……在表单周围增加一个边框，另外在表单内容和边框之间增加一些内边距，还增加了一个背景颜色，使它与背景区分开。

```
form textarea {
    width: 500px;
    height: 200px;
}
```

让表单中的textarea控件更大一些，通过设置它的宽度和高度，可以有更大的空间来输入客户意见。

```
div.tableRow {
    display: table-row;
}
```

每个“tableRow” 相当于表格显示布局中的一行。

```
div.tableRow p {
    display: table-cell;
    vertical-align: top;
    padding: 3px;
}
```

嵌套在一个“tableRow” 中的各个<p>元素分别是一个表格单元格。我们将各个<p>中的内容垂直对齐，使得各行的内容都与单元格顶部对齐。另外还在这里增加了一点内边距，来增加行之间的间距。

```
div.tableRow p:first-child {
    text-align: right;
}
```

这个选择器对应嵌套在“tableRow” 中的<p>元素，这个规则在这个选择器上使用了first-child伪元素。这表示各行的第一个<p>元素要右对齐，所以它们都与这一列的右边对齐。

```
p.heading {
    font-weight: bold;
}
```

对于类为“heading”的<p>元素，我们将文本设置为粗体，使它们看起来像标题。“Ship to”单元格中会使用这个样式。

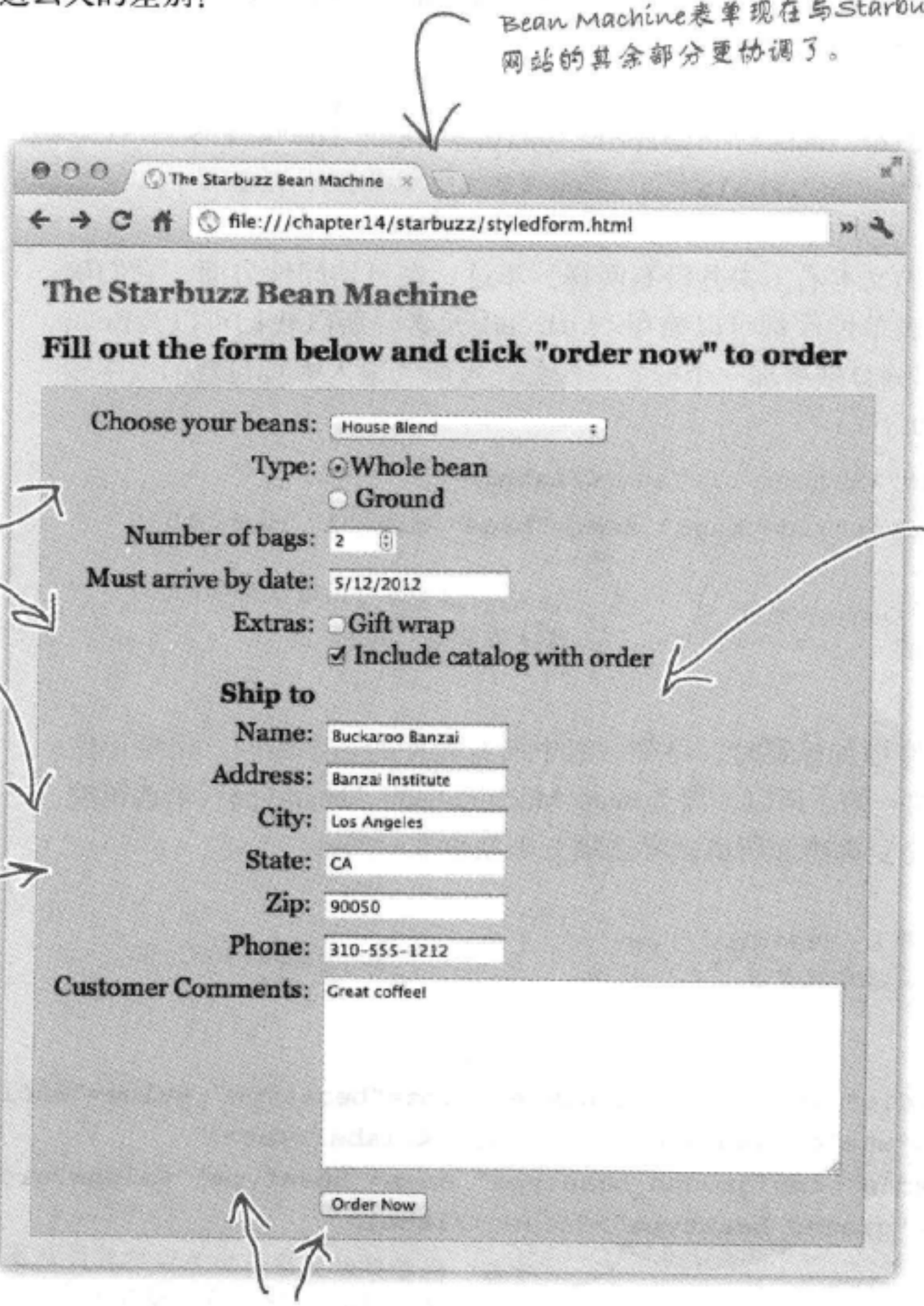
测试增加样式后的表单

在“styledform.html”的HTML中，为<head>增加两个<link>元素，分别链入第12章中的Starbuzz样式表“starbuzz.css”和你新建的样式表“styledform.css”。一定要保证正确的顺序：先链接“starbuzz.css”文件，然后链接“styledform.css”。链接了这两个样式表之后，保存并重新加载你的页面。你会看到浏览器中会显示增加样式后漂亮的Starbuzz Bean Machine页面。

如果你想提高自己的HTML和CSS技能，看看能不能为Bean Machine页面增加Starbuzz页眉和页脚，让那些元素在这个Bean Machine页面看上去很协调。

哇，一点点样式竟然带来这么大的差别！

Bean Machine表单现在与Starbuzz网站的其余部分更协调了。



这些标签与表单元素顶部对齐，另外它们同时还右对齐。通过这种对齐方式，可以更容易地看出哪个标签属于哪个控件。

正如我们期望的，“Ship to”标题是粗体。

行之间的间距让表单大为改观，另外也更容易阅读。

这里有两列，行中的所有内容都能很好地对齐！

关于可访问性

我们使用label创建了Bean Machine页面的一个完整版本，另外相应地更新了CSS。请查看下载代码中的accessform.html和accessform.css。

到目前为止，我们只使用简单的文本为表单元素增加标签，不过实际上应该用<label>元素来标记这些标签。<label>元素可以提供页面结构的更多信息，使你能更容易地使用CSS对标签设置样式，另外对于有视力障碍的人，也有助于他们使用的屏幕阅读器更准确地标识表单元素。

要使用<label>元素，首先为表单元素增加一个id属性。

```
<input type="radio" name="hotornot" value="hot" id="hot">
<label for="hot">hot</label>
<input type="radio" name="hotornot" value="not" id="not">
<label for="not">not</label>
```

然后增加一个<label>，设置其"for"属性为相应的id。

现在这些单选钮旁边的文本就是一个真正的标签（label）。

默认地，标签与普通的文本看上去并没有两样。不过，在可访问性方面，它们确实有很大不同。任何表单控件都可以使用<label>元素，所以我们可以为Bean Machine表单的各个部分分别增加一个标签。例如，可以为输入咖啡包数的数字输入域增加一个标签，如下：

```
<label for="bags">Number of bags:</label>
<input type="number" id="bags" name="bags" min="1" max="10">
```

我们已经为这个<input>元素增加了id "bags"。

name和id属性可以使用相同的值，在这里就是"bags"。

为单选钮或复选框控件增加标签时，尽管一组中所有控件的名字相同，但要记住每个控件的id必须是唯一的。所以，要为Bean Machine中的"beantype"单选钮控件增加标签，就要为全豆咖啡和研磨咖啡选项分别创建唯一的id：

这两个控件的名字都是"beantype"，所以向服务器脚本提交这个表单时，它们会归组在一起。

不过，每个id必须是唯一的。

```
<input type="radio" id="whole_beantype" name="beantype" value="whole">
    <label for="whole_beantype">Whole bean</label><br>
<input type="radio" id="ground_beantype" name="beantype" value="ground" checked>
    <label for="ground_beantype">Ground</label>
```

注意，标签可以放在与它关联的控件前面或后面。只要for属性的值与id匹配，标签放在哪里并不重要。

表单中还可以有哪些元素？

我们已经介绍了表单中常用的所有元素，不过还有一些元素你可能也想增加到你的表单中。这里将简要介绍这些元素，将来如果你想深入研究表单，这可能会有帮助。

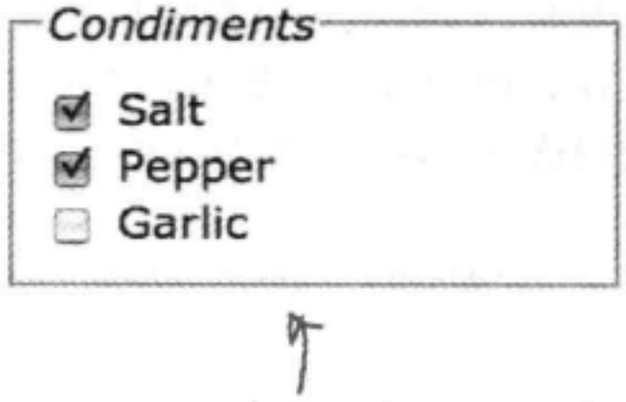
fieldset和legend

表单变得越来越大时，在视觉上对元素分组会很有帮助。尽管可以用 和CSS也可以做到，不过HTML还提供了一个<fieldset>元素，可以用来将公共元素组织在一起。<fieldset>又使用了另一个元素，名为<legend>。下面来看如何结合使用这两个元素：

<fieldset>元素包围一组输入（input）元素。

<legend>为这一组提供一个标签。

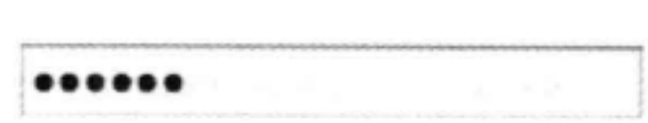
```
<fieldset>
   <legend>Condiments</legend>
      <input type="checkbox" name="spice" value="salt">
         Salt <br>
      <input type="checkbox" name="spice" value="pepper">
         Pepper <br>
      <input type="checkbox" name="spice" value="garlic">
         Garlic
</fieldset>
```



这是某个浏览器上显示的fieldset和legend。你会发现，在不同的浏览器上，这些元素的显示也可能不同。

passwords

password <input>元素的工作与text <input>元素很类似，只是你输入的文本会加掩码。如果表单中需要输入口令、密码或者其他敏感信息（你不希望其他人在你输入时看到），这个元素就很有用。不过，要记住，表单数据并不会采用一种安全的方式从浏览器发送到服务器脚本（除非你采取了安全措施）。要想提高安全性，请联系你的托管公司。



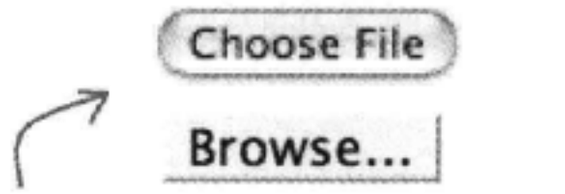
```
<input type="password" name="secret">
```

password <input>元素的工作与text <input>元素很类似，只是你输入的文本会加掩码。

可以放在表单中的其他元素

文件输入

还有一个新的输入元素我们没有谈到。如果你需要向服务器脚本发送整个文件，同样可以使用<input>元素，不过这一次要把它的类型设置为“file”。这样一来，这个<input>元素会创建一个文件输入控件，允许你选择一个文件，表单提交时，文件的内容会随其余的表单数据一同发送给服务器。记住，你的服务器脚本希望上传一个文件，另外需要说明，使用这个元素的前提是必须使用POST方法。



这是两个不同浏览器中看到的文件输入元素。

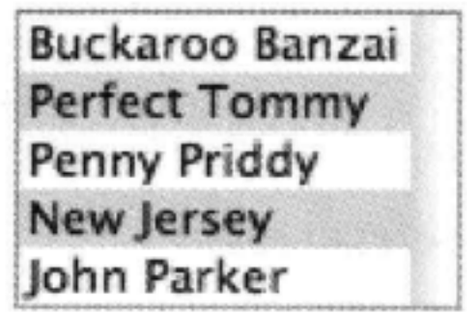
```
<input type="file" name="doc">
```

要创建一个文件输入元素，只需要将<input>元素的type属性设置为“file”。

多选菜单

这不是一个元素，更应算是使用已知元素的一种新方法。如果为<select>元素增加布尔属性multiple，就会把你的单选菜单变成一个多选菜单。不再显示一个下拉式菜单，你会得到一个多选菜单，在屏幕上显示所有选项（如果选项太多，还会有一个滚动条）。选择时通过同时按下Ctrl (Windows)或Command (Mac)键，可以选择多个选项。

利用多选菜单，你可以一次选择多个选项。

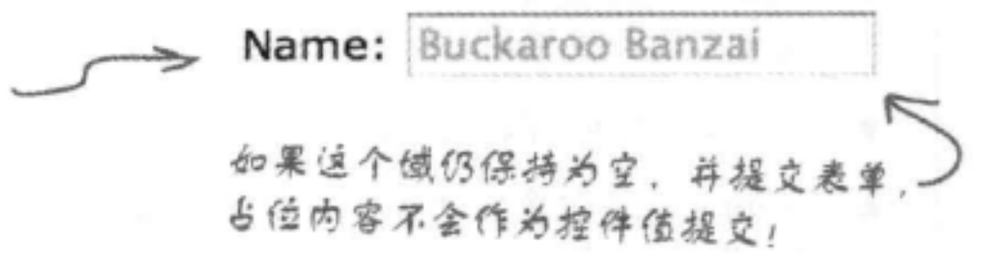



```
<select name="characters" multiple>
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny Priddy">Penny</option>
    <option value="New Jersey">Jersey</option>
    <option value="John Parker">John</option>
</select>
```

只需要增加属性multiple，就把一个单选菜单变成了多选菜单。

Placeholder

表单中大多数不同类型的<input>元素都可以使用placeholder属性，这会为填写表单的人提供一个提示，让他了解你希望这个控件中输入什么内容。例如，如果希望在一个文本域中得到名和姓，可以使用placeholder属性提供一个名和姓的示例。这个属性的值会显示在控件中，但是比增加到控件的正常内容要浅一些，一旦单击这个文本域，占位文本就会消失，所以它不会与你输入的内容混杂在一起。



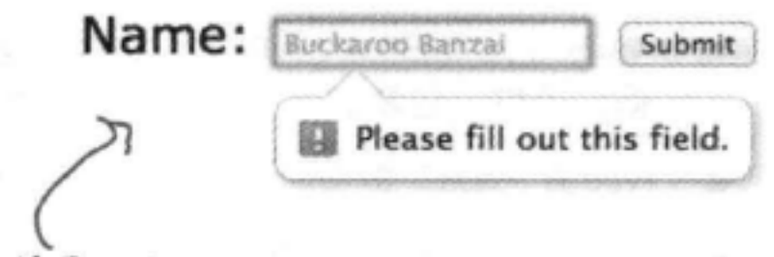

```
<input type="text" placeholder="Buckaroo Banzai">
```

placeholder属性允许你提供一个提示，使用户了解你希望表单的这一部分需要怎样的内容。

Required

这个属性可以用于任何表单控件，它指示一个域是必要的，所以，对于设置了这个属性的控件，如果没有为这些控件指定一个值，就无法正常提交表单。在支持这个属性的浏览器中，如果没有为有required属性的域指定一个值，提交表单时你会得到一个错误消息，表单不会提交到服务器。

注意，这个属性也是一个布尔属性，我们在<video>元素中已经见过。这说明，这个属性的值就是“有”或“没有”。也就是说，如果有这个属性，就说明设置了这个属性，如果没有，则未设置这个属性。所以，在这个例子中，由于有required，所以这说明已经设置这个属性，必须输入这个域，表单才能提交。



这是一个Chrome截屏图。写这本书时，并不是所有浏览器都支持required属性，不过不管怎样都可以写上这个属性。在不支持这个属性的浏览器上，表单可以提交，不过服务器脚本会抱怨你还没有填写这个域。

required是一个布尔属性，所以如果表单控件中有这个属性，说明这个域必须有一个值，表单才能正常提交。

```
<input type="text" placeholder="Buckaroo Banzai" required>
```

编辑你的“styledform.html”文件，为各个text <input>和tel <input>增加placeholder属性。要选择合适的值，使客户得到很好的提示，知道各个域希望输入什么内容。

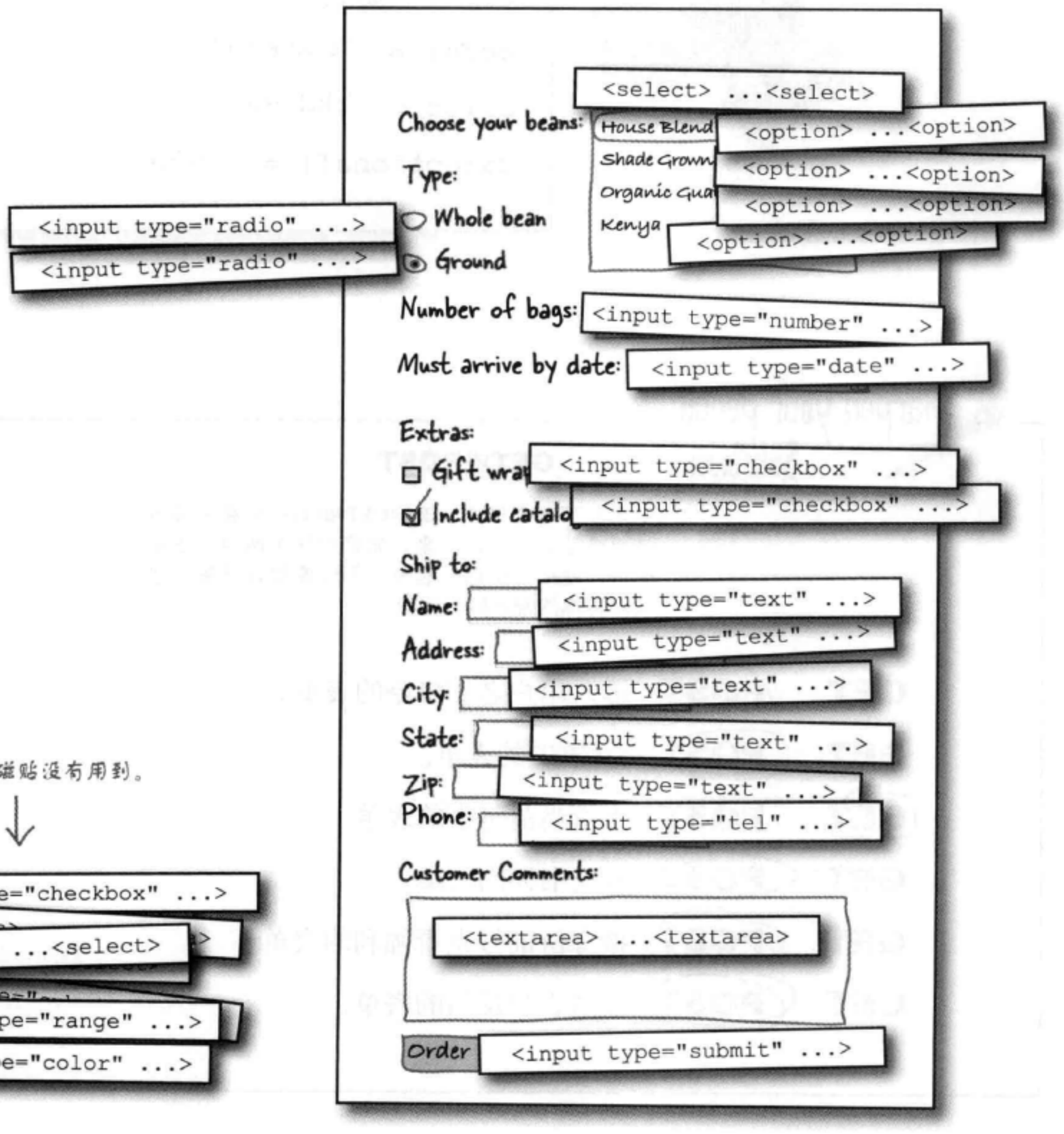
接下来，继续编辑这个文件，为Starbuzz Bean Machine页面中所有必要的表单域（所有“Ship to”域）增加required属性。既然beans和beantype都有默认值，这些域还需要required属性吗？如果从beantype删除checked属性会发生什么情况，还需要required吗？在不同的浏览器上做些实验，看看哪些浏览器支持placeholder和required。

BULLET POINTS

- <form>元素定义了表单，所有表单输入元素都嵌套在这个元素中。
- action属性包含服务器脚本的URL。
- method属性包含发送表单数据的方法，可以是POST或GET。
- POST打包表单数据，并把它作为请求的一部分发送到服务器。
- GET打包表单数据，并把数据追加到URL。
- 如果表单数据应当是私有的，或者表单数据很多，如使用了一个<textarea>或者file <input>元素，就应当使用POST。
- 对于可以加书签的请求，要使用GET。
- <input>元素在Web页面上可以作为多种不同的输入控件，这取决于它的 type”属性值。
- type为“text”时会创建一个单行文本输入框。
- type为“submit”时会创建一个提交按钮。
- type为“radio”时会创建一个单选钮。所有同名的单选钮构成一组互斥的按钮。
- type为“checkbox”时会创建一个复选框控件。通过为多个复选框指定相同的名字，可以创建一组选择。
- type为“number”时会创建一个只允许数字字符的单行文本输入控件。
- Type为“range”时会创建一个滑动条控件提供数字输入。
- “color”类型会在支持这个类型的浏览器中创建一个颜色选择器（否则只会创建一个普通的文本输入控件）。
- “date”类型会在支持这个类型的浏览器中创建一个日期选择器（否则只会创建一个普通的文本输入控件）。
- “email”、“url”和“tel”类型会创建单行文本输入，在一些移动浏览器上会出现定制键盘来方便数据输入。
- <textarea>元素会创建一个多行文本输入区。
- <select>元素会创建一个菜单，包含一个或多个<option>元素。<option>元素定义了菜单中的菜单项。
- 如果将文本放在<textarea>元素的内容中，这会成为Web页面上文本区控件中的默认文本。
- text <input>元素中的value属性可以用来为单行文本输入控件提供一个初始值。
- 在提交按钮上设置value属性可以改变按钮上显示的文本。
- 提交一个Web表单时，表单数据值与相应的数据名配对，所有名和值会发送到服务器。
- 由于表单有一个表格结构，通常会用CSS表格显示来建立表单布局。CSS还可以用来指定表单的颜色、字体风格、边框等样式。
- HTML允许用<fieldset>元素组织表单元素。
- 可以用<label>元素以一种有助于提高可访问性的方式关联标签与表单元素。
- 使用placeholder属性可以为表单用户提供一个提示，指出你希望在一个输入域中输入什么内容。
- required属性指示一个输入域是必要的，要让表单成功提交，这个输入域中必须有值。有些浏览器在你提交表单之前会强制要求在这些域中输入数据。

标记磁贴答案

你的任务是把这里的标记元素磁贴放在草图中相应的控件上。完成这个任务时，不一定所有磁贴都会用到，有些可能会剩下。下面是我们的答案。



扮演浏览器答案

```
name = "Buckaroo Banzai"
zip = "90050"
model = "convertible"
color = "chilired"
caroptions[] = "stripes"
```

Sharpen your pencil Solution

GET或POST

对于以下各个描述，确定GET和POST中哪一种方法更合适，把它圈出来。如果你认为两种方法都可行，就把两个都圈起来。不过要做好准备，你要能对你的答案做出解释……

GET **POST** 输入用户名和口令的表单。

GET **POST** 订购CD的表单。

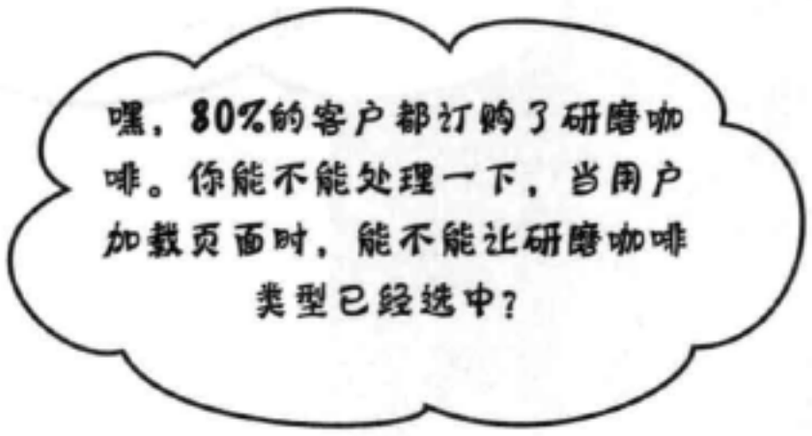
GET POST 查看当前时事的表单。

GET **POST** 提交书评的表单。

GET **POST** 按身份证号查看福利的表单。

GET **POST** 发送客户反馈的表单。

Exercise Solution

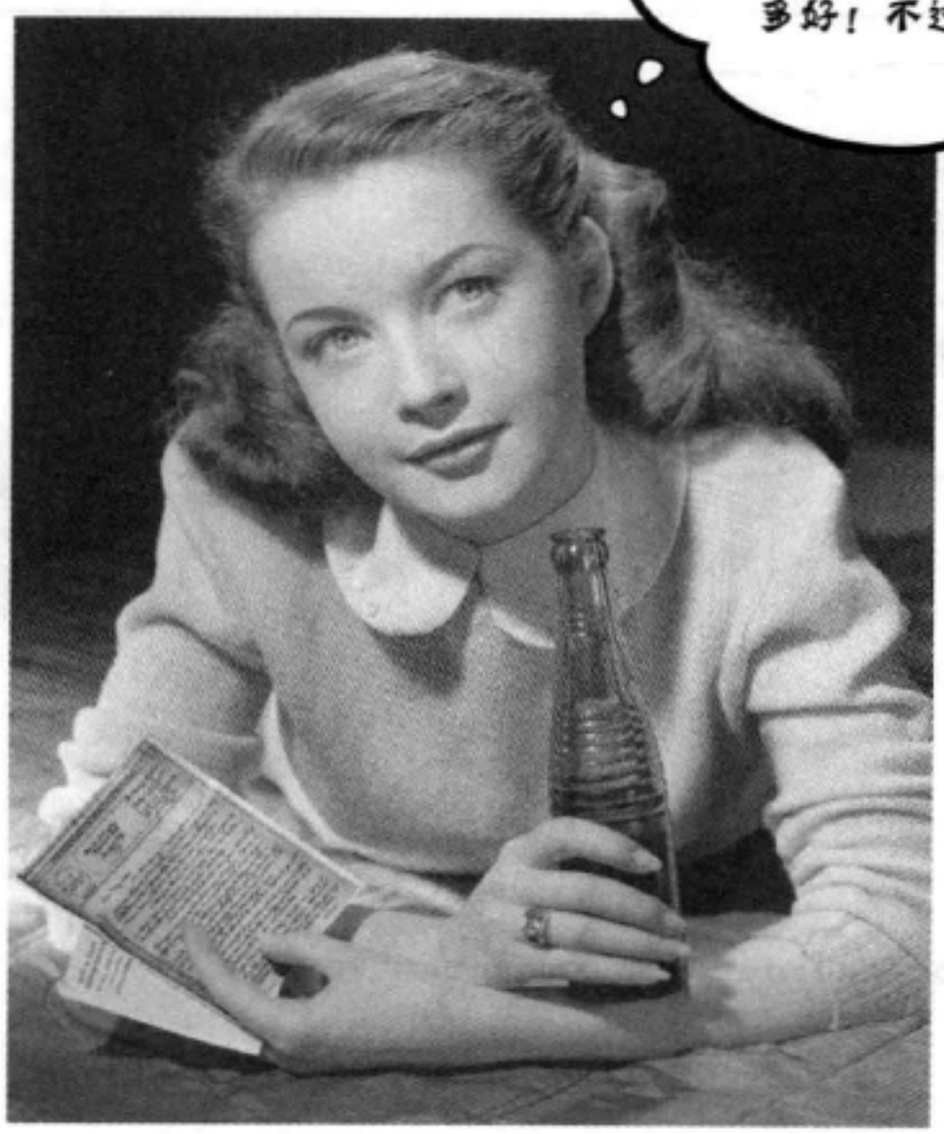


如果为单选钮输入元素增加一个布尔属性“checked”，浏览器显示表单时就会默认地选中这个元素。为“ground” radio <input>元素增加checked属性，再来测试这个页面。下面给出答案。

这只是“form.html”表单中相关的部分。

```
<form action="http://starbuzzcoffee.com/processorder.php" method="POST">
  ...
  <p>Type: <br>

    <input type="radio" name="beantype" value="whole"> Whole bean <br>
    <input type="radio" name="beantype" value="ground" checked> Ground

  </p>
  ...
</form>
```

这就是选中的“ground”单选钮的新属性。



祝贺你！

终于到终点了。

当然了，后面还有一个附录。

还有索引。

还有封底。

然后还有网站……

实际上，这是无止境的。

十大主题(我们没有谈到的)

我们已经介绍了很多，这本书也快要结束了。我们会想你的，不过在你离开之前，还要再做一点准备，否则我们实在不放心让你贸然进入这个纷乱的世界。我们不可能把你需要知道的一切都放在这样短短的一章里。实际上，我们原先确实加入了有关HTML和CSS需要了解的所有内容（其他章节中没有谈到），只不过把字体缩小到0.00004。这样才能放得下，可惜没有人能看得清。所以，我们最后还是删去了大部分内容，只把最重要的部分保留在这个“十大主题”附录中。

#1 更多CSS选择器

你已经了解了那些最常用的选择器，下面再来介绍一些你可能也想知道的选择器……

伪元素

你已经对伪类有了全面的了解，伪元素也是类似的。伪元素（Pseudo-element）可以用来选择元素的某些部分，这些部分可能不便于包围在 或中，也不方便用其他方法来选择。例如，:first-letter伪元素可以用来选择一个块元素中文本的第一个字母，这样你就能创建诸如首字母大写和首字母下沉等效果。另外可以使用:first-line伪元素选择段落的第一行。下面就使用这两个伪类来选择一个<p>元素的第一个字母和第一行：

伪元素的语法与伪类相同。

```
p:first-letter {
        font-size: 3em;
}
p:first-line {
        font-style: italic;
}
```

这里将段落的第一个字母放大，另外把第一行设置为斜体。

属性选择器

顾名思义，属性选择器就是根据属性值来选择元素。可以像这样使用：

这个选择器会选择HTML中所有包含一个width属性的图像。

```
img[width] { border: black thin solid; }
img[height="300"] { border: red thin solid; }
image[alt~="flowers"] { border: #ccc thin solid; }
```

这个选择器会选择height属性值为300的所有图像。

这个选择器会选择alt属性包含单词“flowers”的所有图像。

按兄弟选择

还可以根据兄弟元素来选择元素。例如，假设你希望只选择前面有一个<h1>元素的段落，可以使用下面这个选择器：

先写前面的元素，再写一个“+”（加号），然后是兄弟元素。

这个选择器会选择所有紧跟在一个<h1>元素后面的段落。

```
h1+p {
  font-style: italic;
}
```

结合选择器

这本书中你已经见过一些结合使用选择器的例子。例如，可以把一个类选择器用作为子孙选择器的一部分，如下：

```
.blueberry p { color: purple; }
```

这里会选择作为 blueberry 类元素的子孙的所有段落。

这里有一种模式，可以用来构造相当复杂的选择器。下面就来一步一步介绍如何应用这个模式：

❶ 首先为你想要选择的元素定义上下文，如下所示：

```
div#greentea > blockquote
```

这里我们使用了一个子孙选择器，id为“greentea”的 必须是<blockquote>的父元素。

❷ 然后给出你想选择的元素：

```
div#greentea > blockquote p
```

上下文

空格

接下来，增加<p>元素，这是我们希望在<blockquote>上下文中选择的元素。<p>元素必须是<blockquote>的一个子孙，<blockquote>则是id为“greentea”的一个 的子元素。

❸ 然后指定伪类或伪元素：

```
div#greentea > blockquote p:first-line { font-style: italic; }
```

上下文

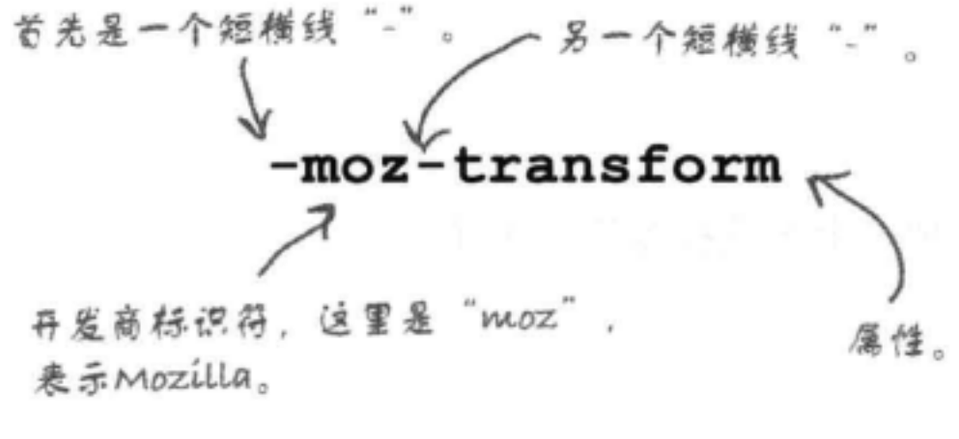
冒号

然后增加一个伪元素first-line，指定只选择这个段落的第一行。

这是一个相当复杂的选择器！你完全可以使用这种方法构造自己的选择器。

#2 开发商特定的CSS属性

浏览器制造商（换句话说，就是像Microsoft、Mozilla等开发厂商，还有WebKit的后台人员等）通常会为他们的浏览器增加新的功能来测试新特性，或者实现一直在考虑但还没有得到标准组织批准的CSS扩展。在这些情况下，开发商会创建类似这样的CSS属性：



你可以利用这些开发商特定的属性，不过它们不一定能用在交付的产品中。这个属性可能永远不会作为一个合法标准得到批准，或者开发商可能会随时改变这个属性的实现。尽管如此，我们当中很多人还是需要能够使用最新最棒的技术创建页面，只是同时必须知道你使用的属性可能随时会改变。

如果打算使用这些属性，通常会创建类似下面的CSS：


```
div {
    transform: rotate(45deg);
    -webkit-transform: rotate(45deg);
    -moz-transform: rotate(45deg);
    -o-transform: rotate(45deg);
    -ms-transform: rotate(45deg);
}
```

通常可以在各个浏览器的开发文档和发行说明中找到这些开发商特定的属性，或者可以加入与各浏览器开发过程相关的论坛，从中也可以了解到开发商特定的属性。

另外，如果你想知道这个transform属性到底做什么，可以翻开下一页，看看“#3 CSS变换和过渡”小节。

#3 CSS变换和过渡

通过使用CSS，现在可以对元素做充分的2D和3D变换。不多说了，来直接看一个例子（输入这些代码，不要怕麻烦，这绝对是值得的）。

```
<!doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>CSS Transforms and Transitions</title>
    <style>
        #box {
            position: absolute;
            top: 100px;
            left: 100px;
            width: 200px;
            height: 200px;
            background-color: red;
        }
        #box:hover {
            transform: rotate(45deg);
            -webkit-transform: rotate(45deg);
            -moz-transform: rotate(45deg);
            -o-transform: rotate(45deg);
            -ms-transform: rotate(45deg);
        }
    </style>
</head>
<body>
    <div id="box"></div>
</body>
</html>
```

这是下面的“box” 的基本样式……

这是绝对位置（你是不是很庆幸跟着我们学完了定位那一章）。

下面为这个 指定一个位置和大小……

……而且把它变成红色。

只有当 处于悬停状态时才会应用这个样式规则……没错， 也可以有悬停状态！

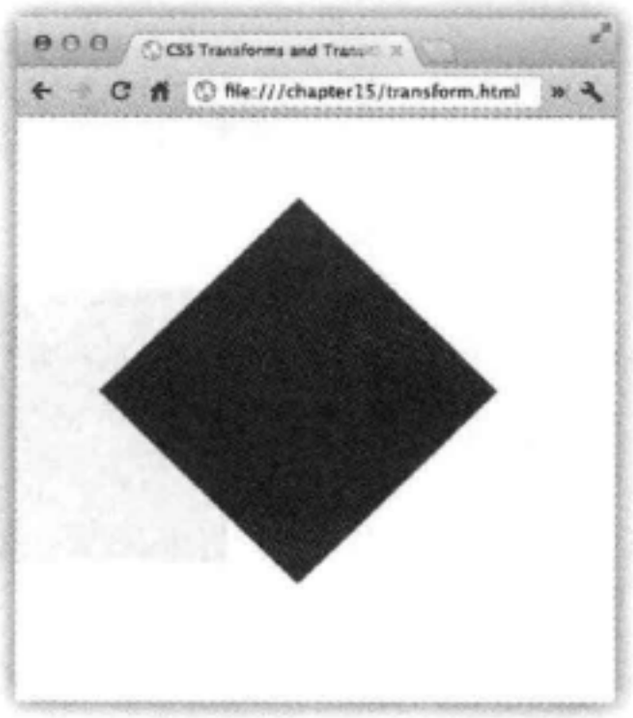
把鼠标停在 上时，会对这个元素完成变换，将它旋转45度。

这个属性只能用于IE9+。

我们还需要浏览器特定的扩展。

这是我们要变换的 。

输入这些代码，然后测试一下。鼠标经过“box” 时，应该能看到它会变换，旋转45度。如果现在想通过一个漂亮的动画平滑地完成这个变换呢？这里就要用到过渡……所以，翻开下一页。



鼠标放在 上，可以看到它旋转了！

可以为“box” 规则增加transition属性，让它在2秒内变换到它的新状态。我们是这样做的：

```
#box {
    position: absolute;
    top: 100px;
    left: 100px;
    width: 200px;
    height: 200px;
    background-color: red;
    transition: transform 2s;
    -webkit-transition: -webkit-transform 2s;
    -moz-transition: -moz-transform 2s;
    -o-transition: -o-transform 2s;
}
#box:hover {
    transform: rotate(45deg);
    -webkit-transform: rotate(45deg);
    -moz-transform: rotate(45deg);
    -o-transform: rotate(45deg);
    -ms-transform: rotate(45deg);
}
```

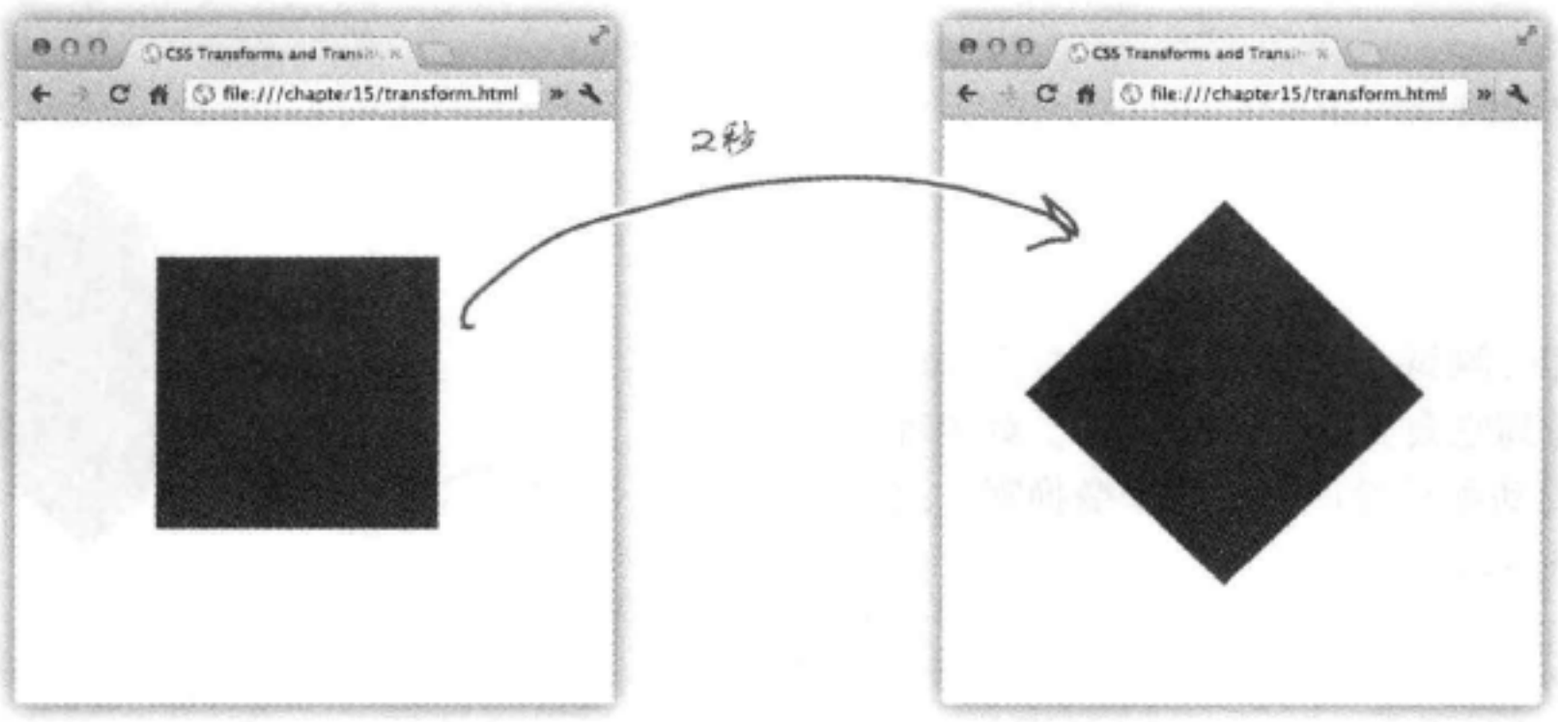
这个transition属性指出：“如果transform属性的值改变，要在指定的时间内从当前transform值过渡到新的transform值。”

transform没有默认值。也就是说，没有变换。

不过，把鼠标停在box上时，transform的值会改为45度旋转。所以从无变换到45度旋转的过渡要在2秒内完成。

transition属性的值也是一个属性，在这里，transition属性值为transform，另外还有一个持续时间（duration），即2秒。指定的属性值改变时，transition会使这个变化在指定的时间内完成，这就产生了一种动画效果。还可以对其他CSS属性完成过渡，如width或opacity。

IE目前（版本9）不支持过渡，不过版本10可能就会支持了。所以，如果你在使用IE，就看不到动画。



#4 交互性

HTML页面并不只能是被动的文档，它们完全可以有可执行的内容。可执行的内容会让页面有自己的行为。要创建可执行的内容，可以使用一种JavaScript脚本语言来编写程序或脚本。下面简单感受一下如何在页面中放入可执行的内容。

这是一个新的HTML元素<script>，允许你在HTML中放入代码。

利用JavaScript，我们使用表单的id来得到表单的一个句柄，以便对它进行处理，比如定义单击一个按钮时会发生什么。

```
<script>
    window.onload = init;
    function init() {
        var submitButton = document.getElementById("submitButton");
        submitButton.onclick = validBid;
    }
    function validBid() {
        if (document.getElementById("bid").value > 0) {
            document.getElementById("theForm").submit();
        } else {
            return false;
        }
    }
</script>
```

这里的JavaScript会检查用户的bid，确保它不会小于等于0。

如果bid大于0，则提交这个表单。否则，由于这是一个错误，所以我们不提交表单。

然后在HTML中可以创建一个表单，在提交表单之前先使用这个脚本检查bid。如果bid大于0，则提交表单。

```
<form id="theForm" method="post" action="contest.php">
    <input type="number" id="bid" value="0"><br>
    <input type="button" id="submitButton" value="Bid!"><br>
</form>
```

在JavaScript中，可以定义单击submitButton时会发生什么，并用id "bid"得到输入的值。

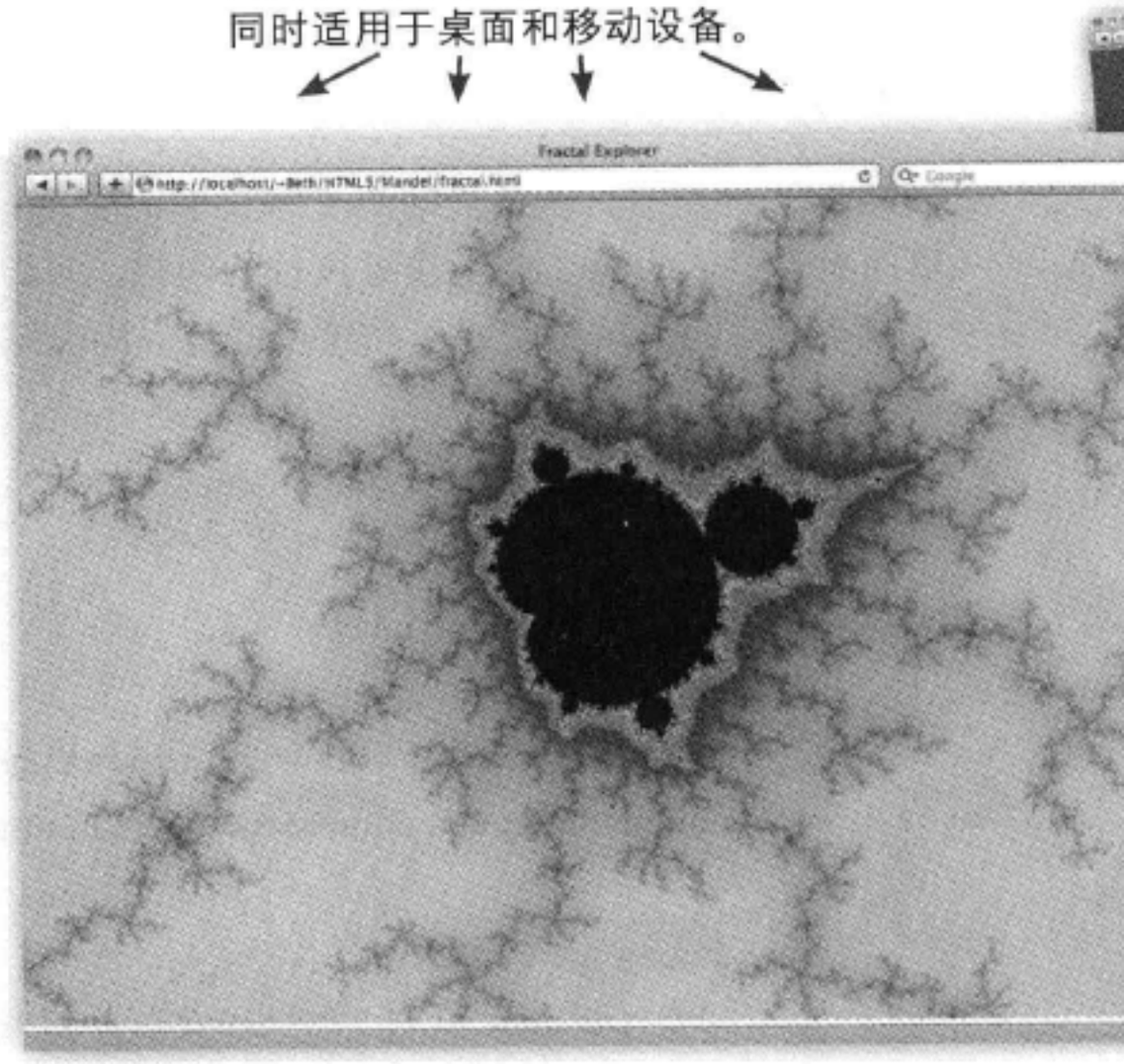
脚本还能做什么？

表单输入验证（就像前面我们所做的）是一个很常见也很有用的任务，通常用JavaScript完成（除了这个例子所示，还可以很多其他类型的验证）。不过，关于JavaScript的作用，这还仅仅是一个开头……请看下一页。

#5 HTML5 API和Web应用

除了HTML5增加的元素，HTML5还提供了一组新的应用编程接口（简写为API），可以通过JavaScript访问。这些API为你的Web页面开启了一个全新的世界，提供了丰富的描述和功能。下面来看利用JavaScript和HTML5可以做的一些事情……。

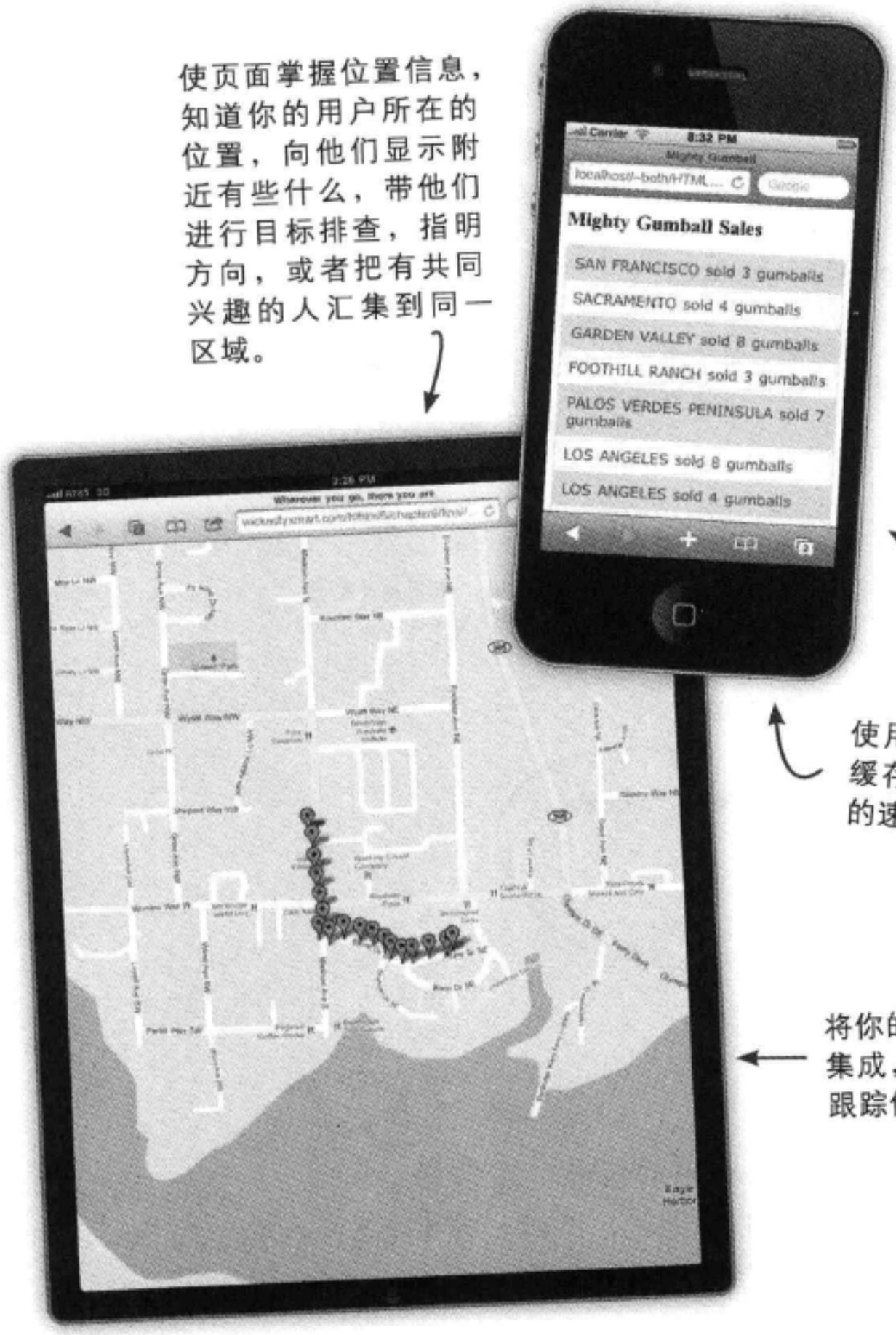
采用新方法与页面交互，这些方法同时适用于桌面和移动设备。



利用HTML5 & JavaScript，你可以直接在页面上创建一个可绘制的2D表面，根本无需插件。

使用Web工作线程可以提高JavaScript代码的效率，完成一些复杂的计算，或者使你的应用更具响应性。甚至可以更好地利用你的用户的多核处理器！

使页面掌握位置信息，知道你的用户所在的位置，向他们显示附近有些什么，带他们进行目标排查，指明方向，或者把有共同兴趣的人汇集到同一区域。



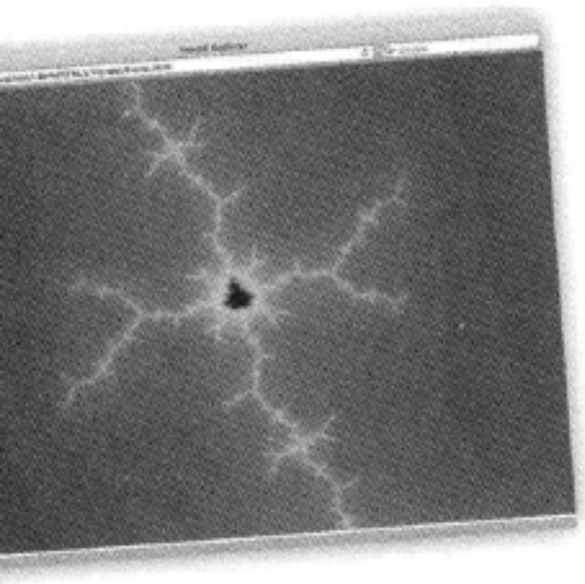
访问任何Web服务，并将数据（近实时地）传回应用。

使用浏览器存储在本地的缓存数据，提高移动应用的速度。

不再需要特殊的插件来播放视频。

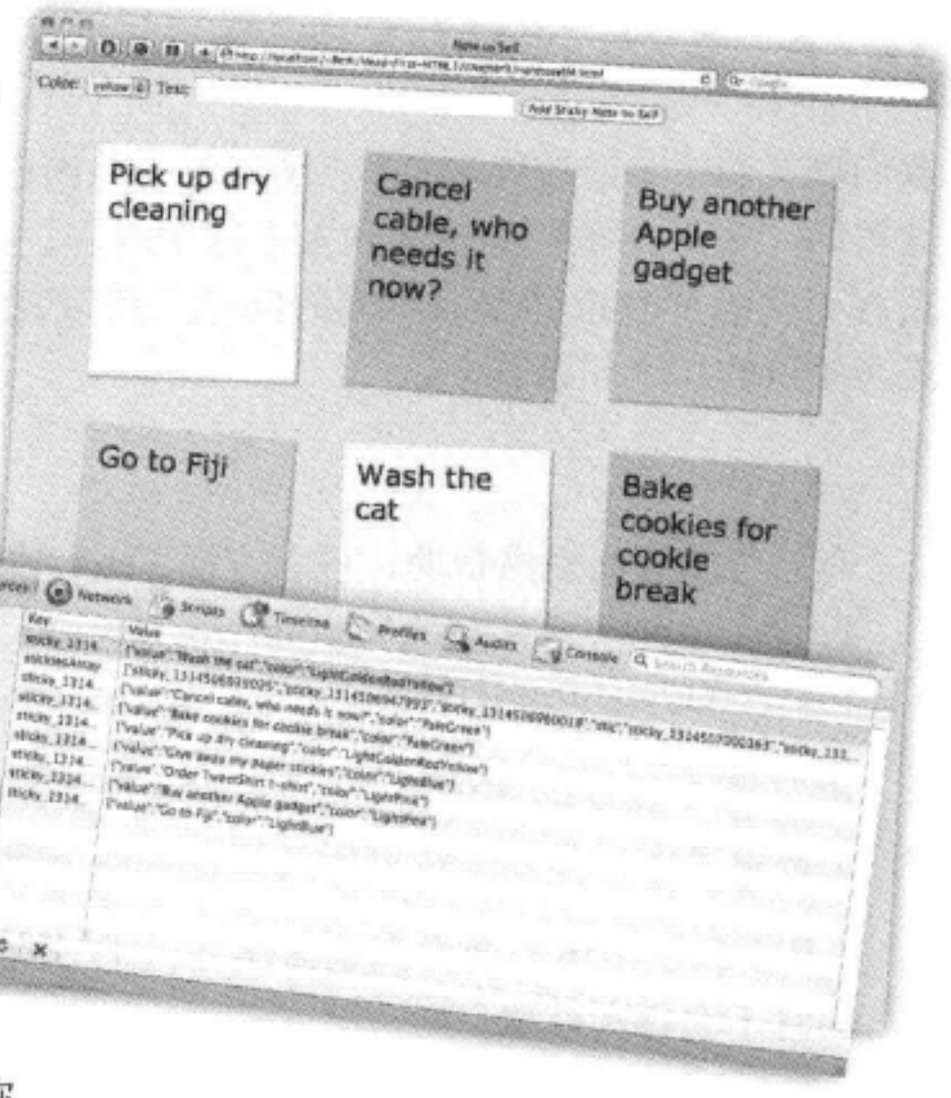
将你的页面与Google Maps集成，甚至允许用户实时地跟踪他们的移动轨迹。

使用HTML和JavaScript创建你自己的视频回放控件。

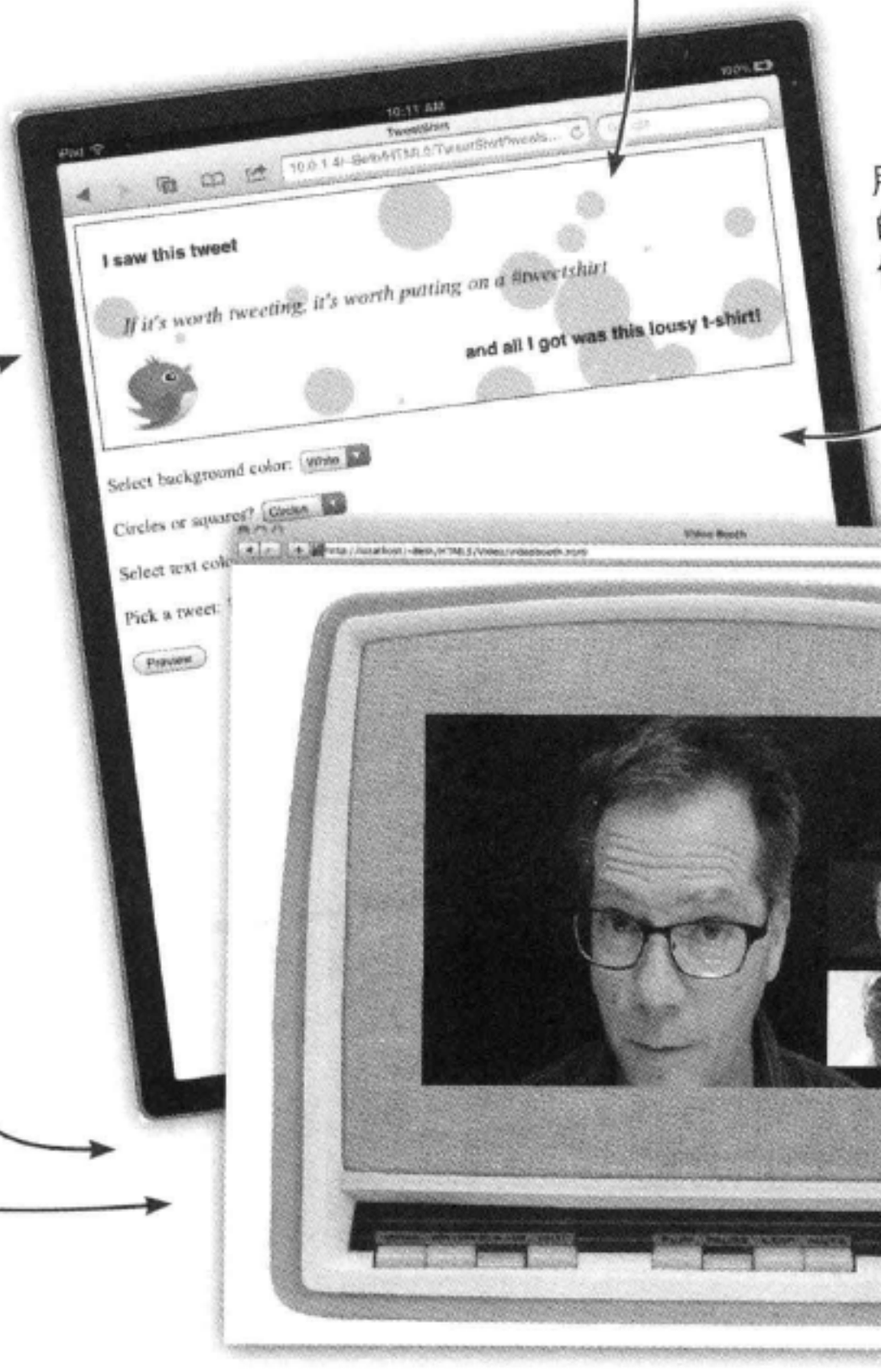


浏览器肯定不再只是应付枯燥的文档了。有了JavaScript，你可以直接在浏览器上绘制像素。

利用浏览器的本地存储。



你可以为用户在本地（浏览器中）存储大量首选项和数据，甚至可以离线访问。



用JavaScript可以让你的表单如虎添翼，提供真正的交互性。

建立完整的视频体验，可以采用新的方式加入视频。

使用JavaScript的强大功能，可以在浏览器中完成完备的视频处理。不仅能创建特效，甚至可以直接处理视频像素。

是不是很震撼？你会在我们的《Head First HTML5 Programming》一书中找到所有这些例子。

#6 再来谈谈Web字体

我们确实希望多谈谈Web字体，所以尽管前面已经介绍过Web字体，但这里的“十大主题”中还是加入了这个内容。如果你在使用Web字体，还有一些需要知道并深入研究的问题，所以我们汇总了关于Web字体应当知道的10个问题：

1. 有一些服务能提供帮助，可以方便Web字体的使用，如Google Web Fonts(http://www.google.com/webfonts)、Fonts.com(http://www.fonts.com/web-fonts)和Extensis (http://www.extensis.com/)。
2. 浏览器下载你的字体时，会有不同的表现。有些浏览器会显示一个备份字体，另外一些可能会等待字体下载完毕后才显示文本。
3. 一旦下载了字体，会由浏览器缓存，下一次遇到使用这个字体的页面时不会再次获取。
4. 所有现代浏览器(IE9+)都支持Web开放字体格式（Web Open Font Format，WOFF），这可能会成为Web字体标准。不过，Internet Explorer 8以前的版本与所有其他现代浏览器支持的字体标准有所不同（.eot），另外还有一个bug，不允许浏览器加载多个字体（所以你的@font-face规则中不能列多个字体）。如果需要在IE8及以前的版本上支持Web字体，前面提到的服务可以帮你，使你不用考虑这些跨浏览器兼容性问题。
5. 现在有很多免费的字体。可以查找“开源字体”，找出可以免费包含在你的Web页面中的字体。
6. 由于Web字体是真正的字体，可以像对传统字体一样对它们应用样式。
7. 使用Web字体可能会对你的页面性能产生一定影响，不过通常认为，与使用定制图形图像提供字体相比，这种方法会更好，往往能提供更好的性能。
8. @font-face规则中的字体应当仅限于页面中真正使用的字体。
9. 如果你有某个字体的许可证，要与你的开发商核对一下，它们可能在Web上也可以使用。
10. 与传统字体一样，一定要包含某个字体作为退路，以免你的页面的字体不可用，或者获取或者解码字体时遇到错误。

#7 创建Web页面的工具

你对HTML和CSS已经很了解了，所以现在可以确定Dreamweaver、Expression Web或Coda之类的工具是否适合你。有些应用包括更丰富的编辑器，提供了代码着色和内置预览等特性，使HTML和CSS的编辑更为容易。有些应用提供了所见即所得（WYSIWYG）的工具来创建Web页面。我们相信，根据你对HTML和浏览器支持的了解，你应该知道尽管这个目标很好，但有时也会带来问题。不过，即便如此，这些工具会提供一些很方便的特性，就算你要自己写大量HTML，这些工具也会很有帮助。这些工具提供的特性包括：

- 一个“代码”窗口，可以输入HTML和CSS，这个窗口会提供语法检查来捕获常见错误，并在你输入代码时提供常用名和属性建议。
- 一个预览发布功能，允许你将页面“上线”之前先进行测试。
- 一个网站管理器，允许你组织网站，还可以保持你的本地修改与服务器上的网站同步。需要说明，它往往会为你完成所有FTP工作。
- 有些工具还提供了内置的验证功能，使你在开发页面时能确信页面是合法的。

这些工具并非没有缺点：

- 有时这些工具在对标准的支持方面有些滞后，所以要保证你的HTML和CSS是最新的，就可能需要自己编写（或编辑）HTML。
- 并不是所有工具都要求严格地遵循标准，可能允许你的 HTML和CSS或多或少有些不规范，所以如果工具没有提供内置的验证功能，不要忘记你自己来进行验证。

要记住，可以结合使用简单的编辑器和这些比较复杂的工具。一个解决方案不一定能完全满足你的需要。所以如果需要，就可以使用一个页面创建工具。

可以考虑的一些工具

- Dreamweaver (Adobe)
- Hype (Tumult)
- Coda (Panic)
- Microsoft Expression Web
- Flux (The Escapers)
- Amaya (开源, 由W3C开发)
- Eclipse (由Eclipse Foundation支持)

最新最棒的Web编辑器总是在变化，所以一定要时刻关注Web，了解各种不同的工具。

#8 XHTML5

这本书对XHTML的介绍很少，只谈到“XHTML已经过时了”。事实上，这种说法中的XHTML只是XHTML 2，它已经从我们的视线中消失了。实际上，只要你愿意，完全可以采用XHTML风格编写HTML5。为什么会有这种想法想要这么做呢？嗯，你可能需要验证文档，或者要把你的文档转换为XML，也可能希望结合HTML支持XML技术，如SVG（你应该知道它是什么）。

下面来看一个简单的XHTML文档，然后逐步介绍重点（我们不可能涵盖这个主题所需知道的全部内容。由于涉及XML，情况很快会变得相当复杂）。

```
<!doctype html>
<html xmlns="http://www.w3.org/1999/xhtml">
   <head>
      <title>You Rock!</title>
      <meta charset="UTF-8" />
   </head>
   <body>
      <p>I'm kinda liking this XHTML!</p>
      <svg xmlns="http://www.w3.org/2000/svg">
         <rect stroke="black" fill="blue" x="45px" y="45px"
               width="200px" height="100px" stroke-width="2" />
      </svg>
   </body>
</html>
```

相同的doctype!

这是XML，我们需要增加命名空间。

所有元素都必须是良构的。需要说明，这里最后有一个/>来结束这个void元素。这是结束void标记的XML格式。

作为一个例子，我们打算使用SVG在页面上绘制一个矩形。细节并不重要。重要的是这是XML格式，仅在XML中有效，在HTML中无效。

可以把XML直接嵌在页面中！太酷了。

对于你的XHTML页面，有几个问题需要考虑：

- 页面必须是良构的XML。
- 页面应当指定application/xhtml+xml MIME类型。为此，需要确保你的服务器支持这种类型（可以阅读有关资料，或者联系你的服务器管理员）。
- 确保在<html>元素中包含XHTML命名空间（类似上面的做法）。

良构是指，要结束所有元素，用引号包围属性值，合法地嵌套元素等。

前面说过，关于XML，需要了解、需要注意的内容有很多，愿主与你同在……

#9 服务器端脚本

很多Web页面都是由服务器上运行的应用生成的。例如，考虑一个在线订购系统，你一步一步完成订购过程时，服务器会生成一系列页面，也可以考虑一个在线论坛，会有一个服务器根据存储在某处的数据库中的论坛消息来生成页面。第14章中我们就使用了一个服务器应用，来处理你为Starbuzz Bean Machine创建的表单。

很多托管公司允许你编写服务器端脚本和程序，来创建你自己的服务器应用。下面是服务器端脚本允许你做的一些事情：

- 构建一个在线商店，包括商品、购物车和一个订购系统。
- 根据用户的个人喜好，建立针对用户的个性化页面。
- 发布最新的新闻、事件和信息。
- 允许用户搜索你的网站。
- 允许用户帮助构建你的网站内容。

要创建服务器应用，需要知道一种服务器端脚本或编程语言。有很多相互竞争的Web开发语言，至于哪一种语言最好，你可能会听到人们不同的观点，这取决于你询问的对象是谁，问的人不同，得到的回答也不同。实际上，Web语言有点像汽车：你可以开着任何一辆车从Prius前往Hummer，每辆车都有自己的优点和缺点（成本、是否好用、大小、是否经济等）。

Web语言一直在不断发展：PHP、Python、Perl、Node.js、Ruby on Rails和JavaServer Pages (JSP)都很常用。如果你刚开始接触编程，PHP可能是最容易上手的语言，另外还有数百万PHP驱动的Web页面，所以你肯定不会孤军奋战。如果你有一定的编程经验，可能想试试JSP或Python。如果你更喜欢Microsoft技术，可能会选择VB.NET和ASP.NET作为服务器端方案。另外，如果JavaScript是你的最爱，可以考虑使用Node.js提供全新的方法。

#10 音频

HTML利用<audio>元素提供了一种标准方法，可以在页面中播放音频，而无需使用插件。你会发现这个元素与<video>元素非常相似：

```
<audio src="song.mp3" id="boombox" controls>
  Sorry but audio is not supported in your browser.
</audio>
```

看着很熟悉？没错，音频支持的功能与视频类似（当然要除于视频）。

同样类似于视频，每个浏览器实现的音频播放器控件都有各自不同的外观（这些控件通常包括一个进度条，以及播放、暂停和音量控制控件）。

很遗憾，同样与视频类似，音频也没有标准编码。目前有3种格式很流行：MP3、WAV和Ogg Vorbis。你会发现，不同浏览器上对这些格式的支持会有所不同（写这本书时，Chrome是唯一一个同时支持这3种格式的浏览器）。

除了简单的播放功能外，<audio>元素及其JavaScript API还提供了很多其他控制。通过结合JavaScript使用这个元素，可以隐藏音频控件，通过代码管理音频的播放，创建有趣的Web体验。利用HTML5，现在完全可以做到这一点，而不再需要使用（和学习）插件（如 Adobe Flash），并因此引入开销。

索引

符号

A

B

D

E

F

G

H

J

K

L

M

N

O

P

Q

R

S

T

U

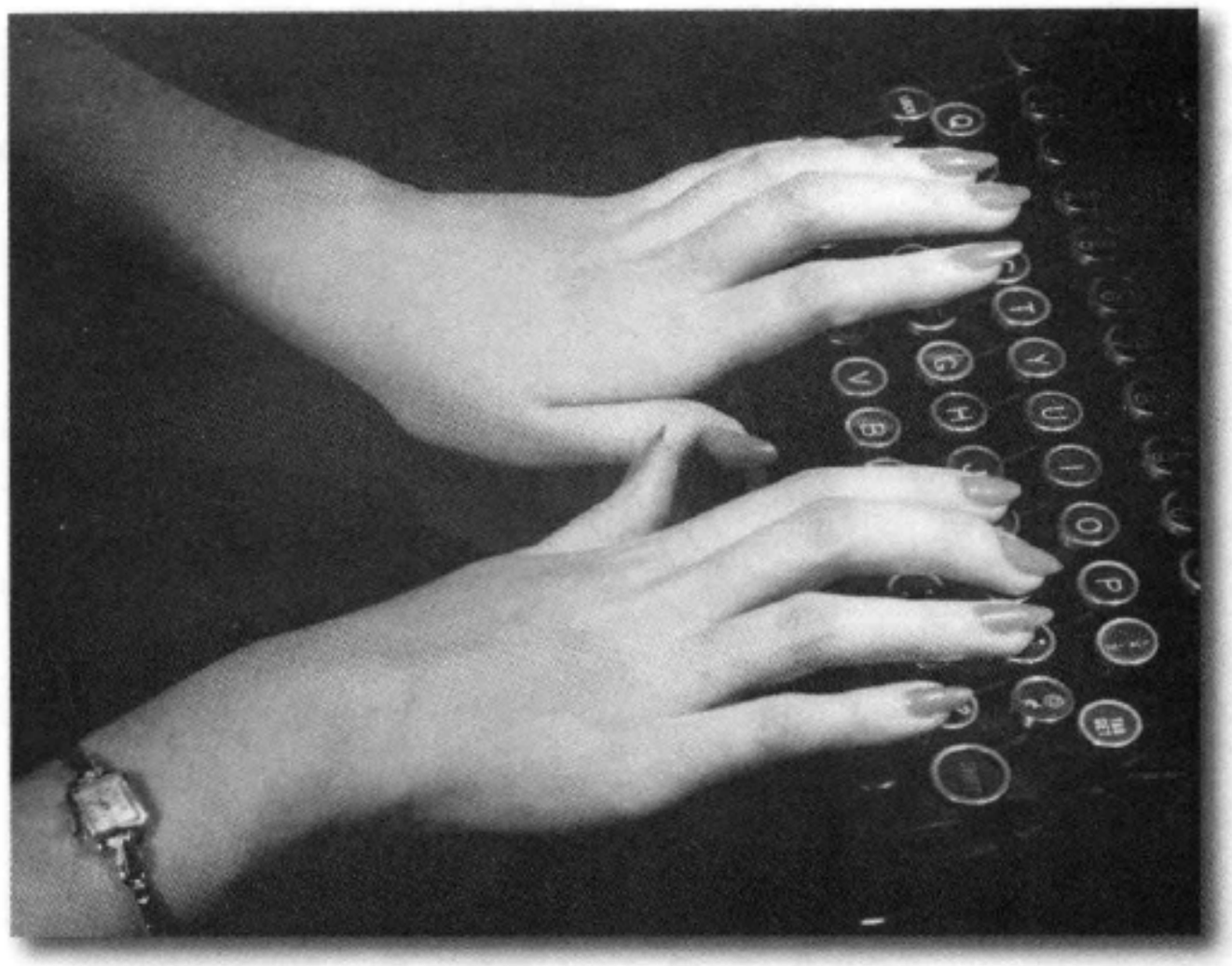
V

W

X

Z

末页



英文版所有内部版面设计都由Eric Freeman和Elisabeth Robson完成。Kathy Sierra和Bert Bates首创了Head First系列的外观。这本书使用Adobe InDesign CS和Adobe Photoshop CS完成。字体包括Uncle Stinky、Mister Frisky（要知道我们不是开玩笑）、Ann Satellite、Baskerville、Comic Sans（不相信吧）、Myriad Pro、Skippy Sharp、Savoye LET、Jokerman LET、Courier New和Woodrow。

版面设计和制版在两台Mac Pros和两台MacBook Airs上完成。

写作地点包括：Bainbridge Island, Washington; Portland, Oregon; Seaside, Florida; Lexington, Kentucky。漫长的写书过程中，我们用zero caffeine和Brew Dr. Kombucha来提神，另外离不开Foster the People, B-52s, Duran Duran, David Bowie, William Shatner, Elvis Presley, Pink Floyd, Genesis, Simple Minds, Ratt, Skid Row, Men without Hats, Men at Work, Berlin, Steve Roach, Tom Waits和Beyman Brothers的音乐，还有很多你可能不太关心的20世纪八十年代的音乐。

不是说再见

潜心加入我们的
wickedlysmart.com